ELSEVIER

Engineering Analysis with Boundary Elements 26 (2002) 133–156

ENGINEERING ANALYSIS *with* BOUNDARY ELEMENTS

www.elsevier.com/locate/enganabound

Mesh-free radial basis function network methods with domain decomposition for approximation of functions and numerical solution of Poisson's equations

Nam Mai-Duy, Thanh Tran-Cong*

Faculty of Engineering and Surveying, University of Southern Queensland, Toowoomba, QLD 4350, Australia

Received 17 January 2001; revised 6 August 2001; accepted 28 September 2001

Abstract

This paper presents the combination of new mesh-free radial basis function network (RBFN) methods and domain decomposition (DD) technique for approximating functions and solving Poisson's equations. The RBFN method allows numerical approximation of functions and solution of partial differential equations (PDEs) without the need for a traditional 'finite element'-type (FE) mesh while the combined RBFN–DD approach facilitates coarse-grained parallelisation of large problems. Effect of RBFN parameters on the quality of approximation of function and its derivatives is investigated and compared with the case of single domain. In solving Poisson's equations, an iterative procedure is employed to update unknown boundary conditions at interfaces. At each iteration, the interface boundary conditions are first estimated by using boundary integral equations (BIEs) and subdomain problems are then solved by using the RBFN method. Volume integrals in standard integral equation representation (IE), which usually require volume discretisation, are completely eliminated in order to preserve the mesh-free nature of RBFN methods. The numerical examples show that RBFN methods in conjunction with DD technique achieve not only a reduction of memory requirement but also a high accuracy of the solution.

Keywords: Mesh-free radial basis function network methods; Domain decomposition; Boundary integral equations; Function approximation; Poisson's equation

1. Introduction

Finite difference method (FDM), finite element method (FEM), boundary element method (BEM) and finite volume method (FVM) are among the most popular numerical methods for the solution of partial differential equations (PDEs) governing many problems in engineering and science. Although enormous progress has been achieved, these methods currently are still based on some discretisation of the domain of analysis into a number of 'finite elements'. Interestingly, neural networks (NNs) find applications in many disciplines [1]. Applications of NNs including multilayer perceptrons (MPs) and radial basis function networks (RBFNs) for numerical solution of PDEs are developed recently with excellent results obtained [2–11]. The advantages of the NN-based method are its ease of implementation and mesh-free feature. In the previous works [9,10], two approaches were proposed using RBFNs for approximation of function and numerical solution of differential equations (DEs) which are called direct RBFN (DRBFN) method and indirect RBFN (IRBFN) method. It is found that the IRBFN method achieves a greater accuracy than the DRBFN method. In this paper, these methods, especially the IRBFN, are considered in conjunction with a domain decomposition (DD) technique for approximation of function and solving PDEs, particularly Poisson's equations. The domain of analysis is divided into a number of subdomains. It is clear that in solving PDEs with subdomains, it is necessary to deal with subdomain interfaces due to the fact that the interfaces form part of the boundary of subdomains with unknown boundary conditions which are to be found as part of the solution process. There are some iterative non-overlapping DD methods available [12–16]. A difference between these methods is in the way to achieve an estimate of the boundary conditions at the interfaces that ensures the continuity of the solution and of its normal derivative across the interfaces. For example, in the work of Funaro et al. [12], at each iteration, at every interface between two subdomains, firstly Dirichlet boundary conditions are imposed on one side, and then Neumann boundary conditions are imposed on the

* Corresponding author. Tel.: +61-7-4631-2539; fax: +61-7-4631-2526.
E-mail address: trancong@usq.edu.au (T. Tran-Cong).

0955-7997/02/$ - see front matter
PII: S0955-7997(01)00092-3

other. On the other hand, in Yang's work [15], each iteration contains two steps. In the first step, at the interface of two subdomains, one subdomain problem requires that Dirichlet data be passed to it from the previous iteration level, while the other subdomain problem requires that Neumann data be passed to it. In the second step, the types of data passing at the interface of the two subdomains are interchanged. The numerical performances of these methods are quite good and convergence analysis can be done for general elliptic problems. Recently, we proposed a new method [16] based on boundary integral equations (BIEs) for estimation of the boundary conditions at the interfaces for problems governed by Laplace's equation. Numerical results obtained showed that this method yields a fast convergence rate. For simplicity, consider the case that consists of two subdomains and one interface. The values of boundary conditions at the interface are not taken directly from the results computed at the interface from the previous iteration. Instead the BIEs for internal points are applied to the whole original domain using the boundary data just obtained from the previous iteration to estimate the solution on the interfaces, which is used as boundary conditions of subdomain problems in the current iteration. After the new solution at the interface is estimated using BIEs, the Dirichlet data are passed to one subdomain, while the Neumann data (normal derivative) are passed to the other. In this paper, the BIE-based DD method is developed for problems governed by Poisson's equation. Unlike the homogeneous Laplace's equation, the integral transformation of Poisson's equation does not automatically lead to boundary only integral equations and therefore one of the aims of this paper is to report a new method of eliminating the volume integral in the standard integral representation of Poission's equation. The paper is organized as follows. In Section 2, the mesh-free DRBFN and IRBFN methods with DD are presented for approximation of function and its derivatives in which, firstly DRBFN and IRBFN methods are reviewed briefly, then a new feature of IRBFN method is described and finally, a 2D numerical example is carried out for the purpose of illustration. Section 3 presents a numerical approach based on IRBFN method and BIE-based DD method for solving Poisson's equations which is applied to two examples on rectangular and non-rectangular domains. The Section 4 gives some concluding remarks.

2. RBFN methods with domain decomposition for function approximation

2.1. Review of DRBFN and IRBFN methods

An RBF network which approximates a mapping $f: R^p \rightarrow R^1$ can be expressed as follows

$$f(\mathbf{x}) = \sum_{i=1}^{m} w^{(i)} g^{(i)}(\|\mathbf{x} - \mathbf{c}^{(i)}\|), \tag{1}$$

where $\mathbf{x} \in R^p$ is the input vector, m is the number of centres, $\{\mathbf{c}^{(i)}\}_{i=1}^{m}$ is the set of RBF centres, $\{g^{(i)}(\|\cdot\|\}_{i=1}^{m}$ is the set of the radial basis functions in which $\|\cdot\|$ denotes the Euclidean norm and $\{w^{(i)}\}_{i=1}^{m}$ is the set of the weights. The multiquadric function (MQ), which becomes unbounded at infinity, appears to be the best among radial basis functions in approximating a smooth input–output mapping as discussed by Powell [17]. MQ, which is employed here for building RBFN, takes the form

$$g^{(i)} = \sqrt{r^2 + a^{(i)2}}, \tag{2}$$

where $a^{(i)}$ is referred to as the width of the ith radial basis function and r is the distance between the input vector $\mathbf{x}$ and the centre $\mathbf{c}^{(i)}$. In order to keep the design of RBFN simple (i.e. using only linear algebra), the centres of the radial basis functions of the network and their widths need to be chosen in advance. The performance of RBFN can critically depend on these choices. The small or large values of RBF width make the response of neuron too peaked or too flat, respectively and therefore should be avoided. By providing a set of the input $\{\mathbf{x}^{(j)}\}_{j=1}^{n}$ where n is the number of data points and the corresponding desired output $\{y^{(j)}\}_{j=1}^{n}$, the values of the weights $\{w^{(i)}\}_{i=1}^{m}$ can be determined using linear least square method. In the following studies, the set of centres is the same as the set of data points and the width $a^{(i)}$ is determined according to the following simple relation [9]

$$a^{(i)} = \beta d^{(i)}, \tag{3}$$

where β is a factor, $\beta > 0$, and $d^{(i)}$ is the distance from the ith centre to the nearest centre. This relation (3) allows the RBF width a to be broader in area of lower data density. In the direct method (DRBFN) the closed form RBFN approximating function (1) is first obtained from a set of training points and the derivative functions are then calculated directly by differentiating such closed form RBFN. Once the weights in Eq. (1) are found, the derivatives (e.g. up to second order with respect to x_j) are calculated by

$$f_{,j}(\mathbf{x}) = \sum_{i=1}^{m} w^{(i)} h^{(i)}(\mathbf{x}), \tag{4}$$

$$f_{,jj}(\mathbf{x}) = \sum_{i=1}^{m} w^{(i)} \bar{h}^{(i)}(\mathbf{x}), \tag{5}$$

where

$$h_{(i)}(\mathbf{x}) = \frac{\partial g^{(i)}}{\partial x_j} = \frac{x_j - c_j^{(i)}}{(r^2 + a^{(i)2})^{0.5}}, \tag{6}$$

$$\bar{h}^{(i)}(\mathbf{x}) = \frac{\partial h^{(i)}}{\partial x_j} = \frac{\partial^2 g^{(i)}}{\partial x_j \, \partial x_j} = \frac{r^2 + a^{(i)2} - \left(x_j - c_j^{(i)}\right)^2}{(r^2 + a^{(i)2})^{1.5}}. \tag{7}$$

It is clear that the differentiation process is very sensitive to even a small level of noise [9]. In contrast it is

expected that on average the integration process is much less sensitive to noise. Based on this observation, in the indirect method (IRBFN), the formulation of the problem starts with the decomposition of the derivative of the function into RBFs. The derivative expression is then integrated to yield an expression for the original function. The expression for the original function is then used in the general linear least squares formulation for the unknown weights, which is then solved by singular value decomposition (SVD) method, given an appropriate set of discrete data points. For example, in the case of multivariate functions with up to second derivatives, the relevant expressions are

$$f_{,jj}(\mathbf{x}) = \sum_{i=1}^{m} w^{(i)} g^{(i)}(\mathbf{x}) = \sum_{i=1}^{m} w^{(i)} \sqrt{r^2 + a^{(i)2}}, \tag{8}$$

$$f_{,j}(\mathbf{x}) = \sum_{i=1}^{m} w^{(i)} H^{(i)}(\mathbf{x}) + C_1, \tag{9}$$

$$f_j(\mathbf{x}) = \sum_{i=1}^{m} w^{(i)} \bar{H}^{(i)}(\mathbf{x}) + C_1 x_j + C_2, \tag{10}$$

where C_1 and C_2 are functions of independent variables other than x_j and

$$H^{(i)}(\mathbf{x}) = \int g^{(i)}(\mathbf{x})\, \mathrm{d}x_j = \frac{\left(x_j - c_j^{(i)}\right)\sqrt{r^2 + a^{(i)2}}}{2} + \frac{r^2 - \left(x_j - c_j^{(i)}\right)^2 + a^{(i)2}}{2} \ln\left(\left(x_j - c_j^{(i)}\right) + \sqrt{r^2 + a^{(i)2}}\right), \tag{11}$$

$$\bar{H}^{(i)}(\mathbf{x}) = \int H^{(i)}(\mathbf{x})\, \mathrm{d}x_j = \frac{(r^2 + a^{(i)2})^{1.5}}{6} + \frac{r^2 - \left(x_j - c_j^{(i)}\right)^2 + a^{(i)2}}{2} \times \left(x_j - c_j^{(i)}\right) \ln\left(\left(x_j - c_j^{(i)}\right) + \sqrt{r^2 + a^{(i)2}}\right) - \frac{r^2 - \left(x_j - c_j^{(i)}\right)^2 + a^{(i)2}}{2} \sqrt{r^2 + a^{(i)2}}. \tag{12}$$

Since C_1 and C_2 are functions of independent variables other than x_j it is necessary to work out their functional form in those variables as well. This is achieved using the same IRBFN methodology. For example, in the case of a function of two variables x_1, x_2, and if $x_j \equiv x_1$ then $C_1 = C_1(x_2)$ is univariate and can be expressed as

$$C_1''(x_2) = \sum_{i=1}^{M} \bar{w}^{(i)} g^{(i)}(x_2) \tag{13}$$

$$C_1'(x_2) = \sum_{i=1}^{M} \bar{w}^{(i)} H^{(i)}(x_2) + \hat{C}_1, \tag{14}$$

$$C_1(x_2) = \sum_{i=1}^{M} \bar{w}^{(i)} \bar{H}^{(i)}(x_2) + \hat{C}_1 x_2 + \hat{C}_2, \tag{15}$$

where $\hat{C}_1$ and $\hat{C}_2$ are constants of integration; $\bar{w}^{(i)}$ are the corresponding weights; and M is the number of centres whose x_2 coordinates are distinct. The same can be done for $C_2 = C_2(x_2)$. The detailed implementation and accuracy of the above methods were reported previously [9].

2.2. A new feature of IRBFN method for approximation of multi-variate functions

In the previous IRBFN method [9] for approximation of the function of several variables and its derivative, each derivative $f_{,j}$ and the associated function f_j is represented by an IRBFN and trained independently resulting in relatively small systems of equations. For example, the process of building an approximating RBFN for a function of two variables $f(x_1,x_2)$ can be illustrated as follows

$$f_{,11} \rightarrow f_{,1} \rightarrow f_1 \rightarrow \{\mathbf{w}\}_1,$$

$$f_{,22} \rightarrow f_{,2} \rightarrow f_2 \rightarrow \{\mathbf{w}\}_2,$$

where $\{\mathbf{w}\}_1$ and $\{\mathbf{w}\}_2$ are the two sets of weights that allow all derivative approximations. Previous experience indicates that both sets of weights are capable of approximating the function itself within small numerical tolerance i.e. the function can be represented by either f_1 or f_2. To further confirm this experience the following function $y(x_1, x_2) = x_1 + x_2^3$, $0 \le x_1, x_2 \le 1$ is used to test the method and the results obtained show that the difference between the two approaches is insignificant. However there is no theoretical guarantee that this is the case. A better approach would be to insist that both sets must give the same approximation by

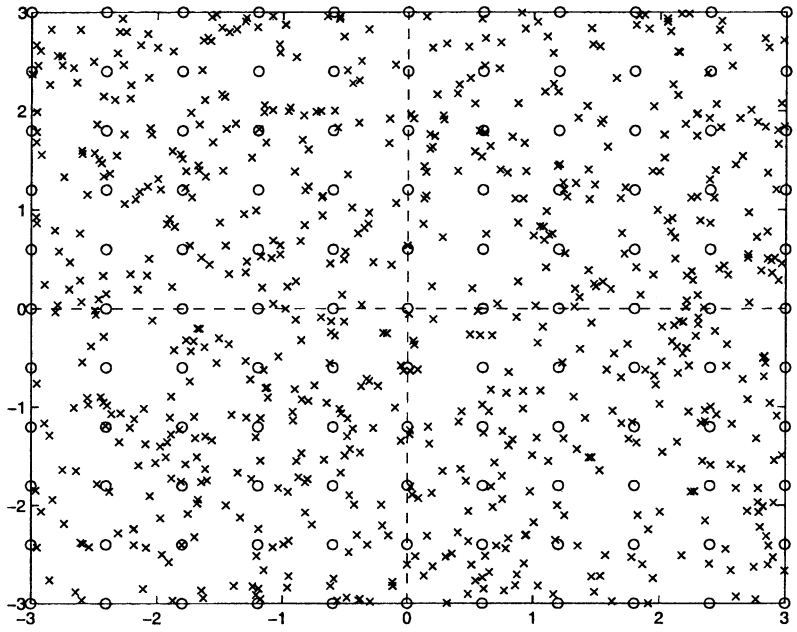

Fig. 1. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$: four subdomains of analysis and corresponding sets of training and test points. Legends ○: training point and ×: test point. The distribution is regular for the training set and random for the test set.

enforcing the condition

$$f_1 = f_2$$

and therefore $\{\mathbf{w}\}_1$ and $\{\mathbf{w}\}_2$ are solved for simultaneously with the consequence that the system of equations is larger. Fortunately, the difficulties relating to solving big matrices can be overcome by using a subregioning technique described in Section 2.3.

2.3. Subregioning of the domain

The study of approximation of function and its derivative using DRBFN and IRBFN with subregioning of the domain is necessary for the following reasons:

1. Single global domain usually requires a large number of data points resulting in
   (a) a large memory requirement as the size of the system matrix increases rapidly with increasing number of data points;
   (b) a large amount of computational time;
   (c) ill-conditioned system matrix to be inverted.
2. It is clearly much easier to find an analytical description of localised regions of a function than to find a single description of the function over its entire range, especially when the function has many locally very different behaviours.

The subregioning of domain for the purpose of function approximation is relatively simple since the data on the interfaces between subregions are also known. This is in contrast with the case of subregioning for the purpose of solving PDEs which is discussed in a later section. Each subregion is approximated by a separate RBFN and the networks can be trained independently and, if desired, in parallel. Subregioning of the domain can provide an effective means of keeping the size of the system matrices down while improving accuracy with increasing data density. Furthermore, overall speedup can be achieved by virtue of the combined effect of fast reduction in matrix inversion time for a larger number but smaller individual matrices and parallel processing.

2.4. Numerical example

We consider the following bivariate function

$$y = x_1^2 x_2 + x_2^3/3 + x_2^2/2,$$

where $-3 \le x_1 \le 3$ and $-3 \le x_2 \le 3$. This is a non-trivial example which has a complicated root structure [18] and is used here to illustrate the working of the present DD approach. The accuracy of solution produced by an approximation scheme is measured via the norm of error N_e defined as

$$N_e = \sqrt{\frac{\sum_{i=1}^{n_t} [y(\mathbf{x}^{(i)}) - f(\mathbf{x}^{(i)})]^2}{\sum_{i=1}^{n_t} y(\mathbf{x}^{(i)})^2}}, \tag{16}$$

where n_t is the number of test points, $\mathbf{x}^{(i)}$ is the ith test point, f and y are the calculated and exact function, respectively.

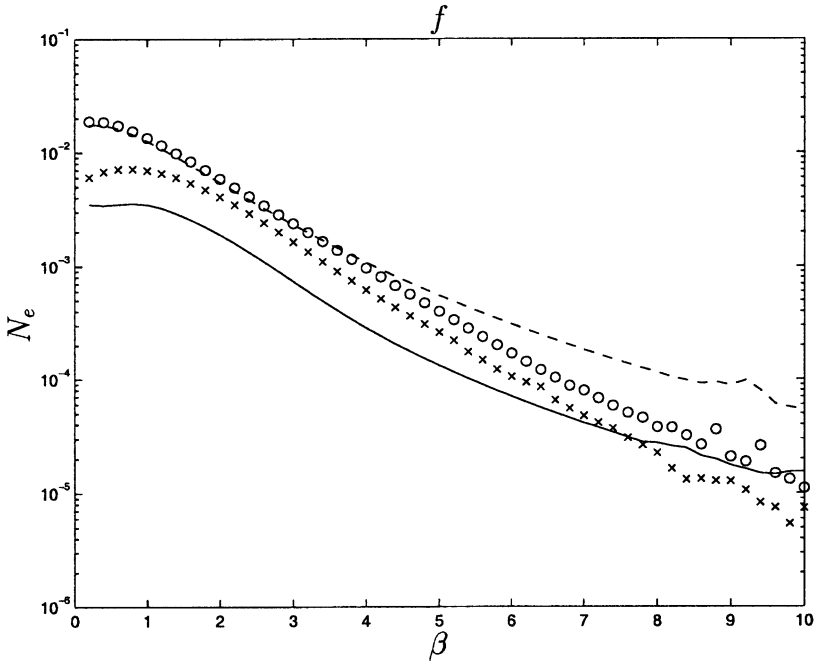

Fig. 2. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$: comparison of the quality of approximation of the original function f obtained by DRBFN and IRBFN methods versus RBF width (β). Legends dashed line: DRBFN (four subdomains), solid line: IRBFN (four subdomains), ○: DRBFN (single domain) and ×: IRBFN (single domain).

The domain now is evenly divided into four subdomains that have the same dimension of 3×3. DRBFN and IRBFN are applied to each subdomain for approximation of the function and its derivatives in which for each subdomain, 36 uniformly distributed data points (density of 6×6) and 180 random points are employed for training and testing, respectively (Fig. 1). Figs. 2–6 show that the results obtained by IRBFN on all four test sets are much

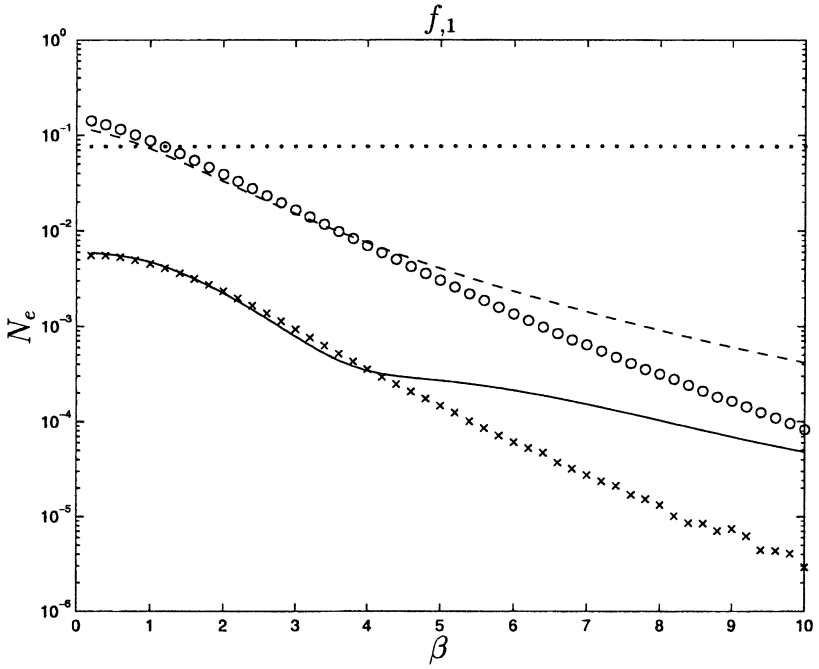

Fig. 3. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$: comparison of the quality of approximation of the first derivative $f_{,1}$ obtained by DRBFN and IRBFN methods versus RBF width (β). The result obtained by the conventional element method is also shown. Legends.: linear triangular element, dashed line: DRBFN (four subdomains), solid line: IRBFN (four subdomains), ○: DRBFN (single domain) and ×: IRBFN (single domain).

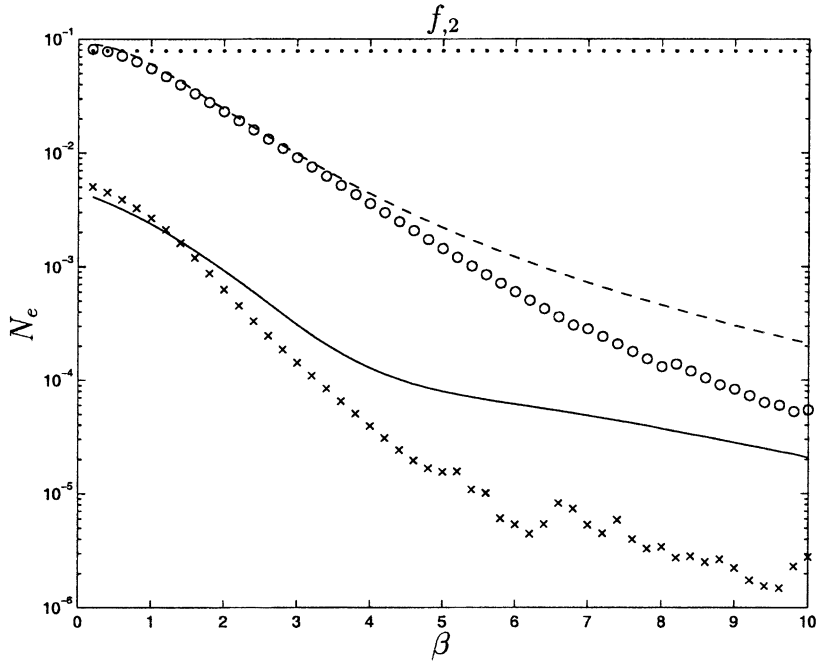

Fig. 4. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$: comparison of the quality of approximation of the first derivative $f_{,2}$ obtained by DRBFN and IRBFN methods versus RBF width (β). The result obtained by the conventional element method is also shown. Legends.: linear triangular element, dashed line: DRBFN (four subdomains), solid line: IRBFN (four subdomains), ○: DRBFN (single domain) and ×: IRBFN (single domain).

more accurate than those by DRBFN over a wide range of β starting from 0.2 to 10 with an increment of 0.2 for all functions f, $f_{,1}$, $f_{,2}$, $f_{,11}$ and $f_{,22}$. In these figures, the RBFN results obtained are also compared with the conventional element method using linear triangular elements showing the RBFN methods achieve not only the convenience of mesh-free discretisation but also a significant improvement of the accuracy of solution. The IRBFN appears to be the

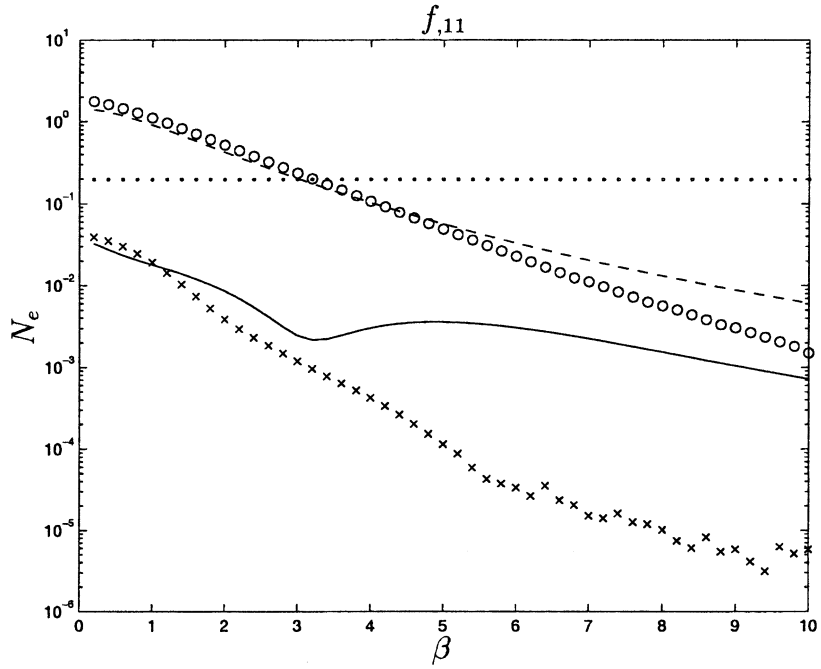

Fig. 5. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$: comparison of the quality of approximation of the second derivative $f_{,11}$ obtained by DRBFN and IRBFN method versus RBF width (β). The result obtained by the conventional element method is also shown. Legends.: linear triangular element, dashed line: DRBFN (four subdomains), solid line: IRBFN (four subdomains), ○: DRBFN (single domain) and ×: IRBFN (single domain).

Table 1
Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$ with four subdomains, density of 6×6 per subdomain and $\beta = 10$: the set of weights of the IRBFN representing functions $f_{,11}$, $f_{,1}$ and f on the subdomain $(0,0) \times (3,3)$

The set of weights associated with basic functions (centres)

Centre location		Weight	Centre location		Weight
x_1	x_2	w	x_1	x_2	w
0.00	0.00	5.403652330×10^2	1.80	0.00	9.990409779×10^2
0.00	0.60	-2.418987907×10^2	1.80	0.60	3.358710984×10^1
0.00	1.20	1.229438064×10^3	1.80	1.20	3.589643825×10^2
0.00	1.80	-1.132828097×10^3	1.80	1.80	3.152882527×10^1
0.00	2.40	3.997433966×10^2	1.80	2.40	4.472096039×10^1
0.00	3.00	-6.117795878×10^2	1.80	3.00	-9.371208654×10^2
0.60	0.00	-1.379646608×10^3	2.40	0.00	-1.325456098×10^3
0.60	0.60	-8.251069049×10^2	2.40	0.60	-8.240511121×10^2
0.60	1.20	3.570553606×10^2	2.40	1.20	3.524652740×10^2
0.60	1.80	-7.095316895×10^2	2.40	1.80	-7.596354286×10^2
0.60	2.40	6.298368428×10^2	2.40	2.40	5.196751898×10^2
0.60	3.00	1.432687508×10^3	2.40	3.00	1.381427962×10^3
1.20	0.00	1.087203622×10^3	3.00	0.00	5.245246543×10^2
1.20	0.60	7.677538772×10^1	3.00	0.60	-2.297396114×10^2
1.20	1.20	3.010431571×10^2	3.00	1.20	1.212721375×10^3
1.20	1.80	-8.411508172×10^1	3.00	1.80	-1.120122162×10^3
1.20	2.40	-7.582003733×10^1	3.00	2.40	4.418313313×10^2
1.20	3.00	-1.085719212×10^3	3.00	3.00	-6.119001935×10^2

Sets of weights for added univariate functions

Function C_1		Function C_2	
Centre location	Weight	Centre location	Weight
x_2	$\bar{w}$	x_2	$\bar{w}$
0.00	1.098542384×10^2	0.00	-7.011571984×10^2
0.60	-1.513238483×10^2	0.60	1.017144282×10^3
1.20	-8.772209189×10^1	1.20	5.039530215×10^2
1.80	8.780247456×10^1	1.80	-5.804077137×10^2
2.40	1.418346812×10^2	2.40	-8.161522249×10^2
3.00	-1.006311334×10^2	3.00	5.781986649×10^2
$\hat{C}_1$	-4.922427391×10^1	$\hat{C}_1$	-1.387643217×10^2
$\hat{C}_2$	5.301542267×10^1	$\hat{C}_2$	-1.249330530×10^3

best among methods used here and is recommended for use in solving PDEs. The next step is to study the effect of centre density on the quality of solution using IRBFN. For each subdomain, three densities of 6×6, 8×8 and 12×12 are employed. As expected, the accuracy of solution is improved when the density of centres increases (Figs. 7–11). The continuity of function and its first derivatives across interfaces is measured by error norms which are displayed in Fig. 12. Furthermore, the comparison between the single domain and multi-subdomain approximation is carried out. To ensure identical input data, the single domain is formed by exactly combining the subdomains (Fig. 1). The single domain case achieves a slightly better accuracy than the case of four subdomains when β is large (Figs. 2–6). This can be explained by the fact that at larger β values the single domain network would take into account more farfield data which are being cut off in the case of subdomain networks, which does not occur to the same extent when the β value is small. However, the overall accuracy of both methods is excellent. For example, in the case of DD the norms of error achieved are of the $O(10^{-5})$ for function and first derivatives and $O(10^{-4})$ for second derivatives (IRBFN). Furthermore, an increased data density is affordable with subregioning and the accuracy obtained with multi-subdomain solution is markedly improved as shown in Figs. 7–11. Table 1 lists the set of weights of an IRBF network representing functions $f_{,11}$, $f_{,1}$ and f on the subdomain $(0,0) \times (3,3)$.

3. IRBFN method with BIE-based domain decomposition method for Poisson's equations

The Poisson's equation governing potential problems can

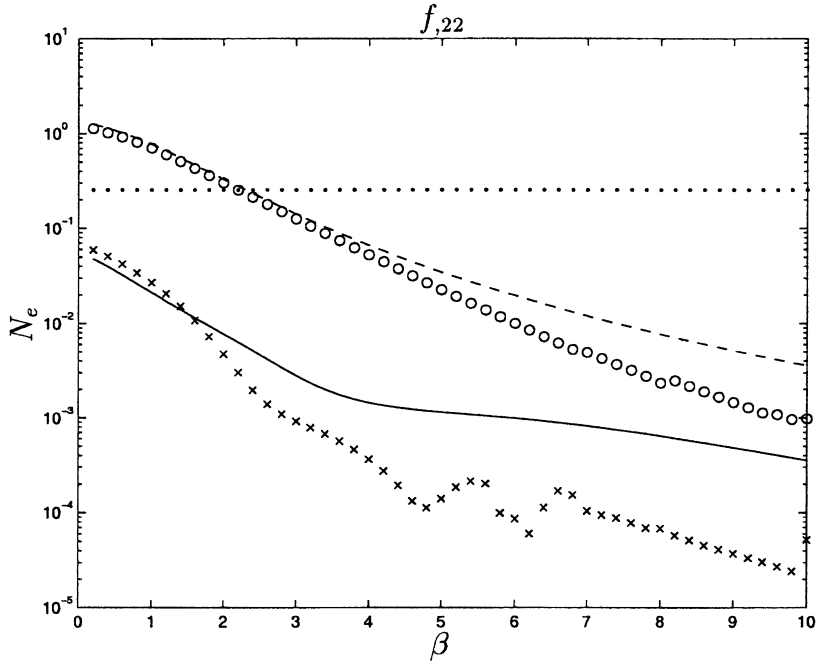

Fig. 6. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$: comparison of the quality of approximation of the second derivative $f_{,22}$ obtained by DRBFN and IRBFN method versus RBF width (β). The result obtained by the conventional element method is also shown. Legends.: linear triangular element, dashed line: DRBFN (four subdomains), solid line: IRBFN (four subdomains), ○: DRBFN (single domain) and ×: IRBFN (single domain).

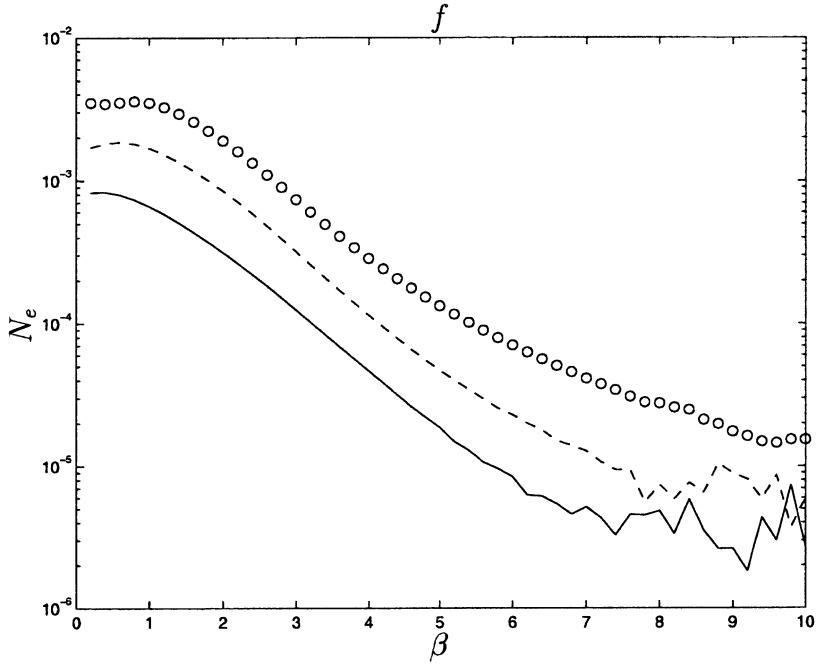

Fig. 7. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$ with four subdomains using IRBFN method: effect of centre density on the quality of approximation of the original function f. Legends ○: density of 6×6 per subdomain, dashed line: density of 8×8 per subdomain and solid line: density of 12×12 per subdomain.

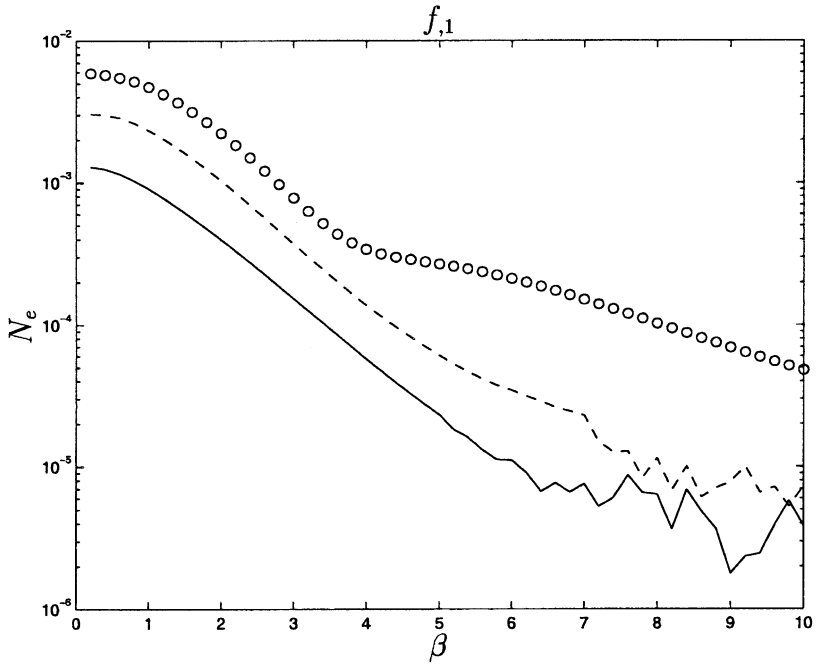

Fig. 8. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$ with four subdomains using IRBFN method: effect of centre density on the quality of approximation of the first derivative $f_{,1}$. Legends ○: density of 6×6 per subdomain, dashed line: density of 8×8 per subdomain and solid line: density of 12×12 per subdomain.

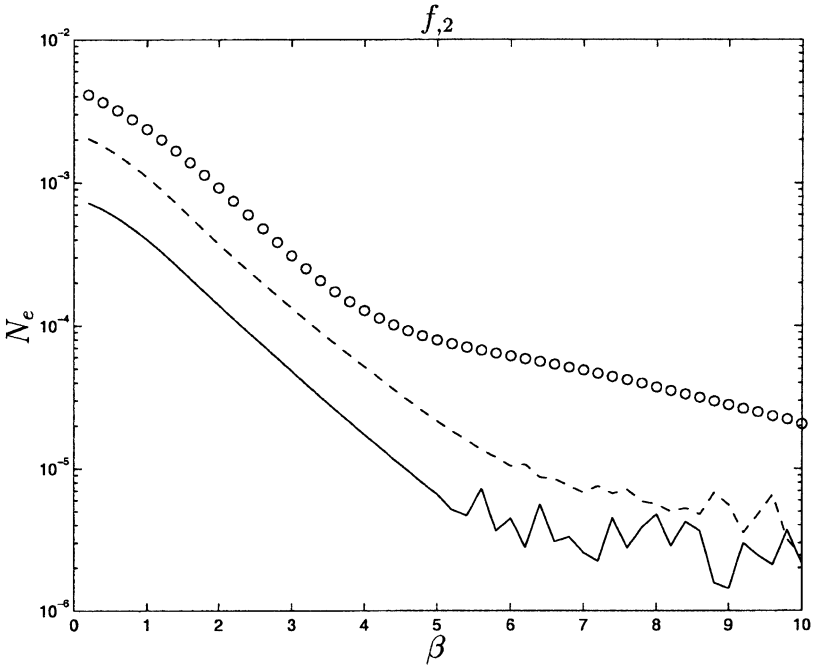

Fig. 9. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$ with four subdomains using IRBFN method: effect of centre density on the quality of approximation of the first derivative $f_{,2}$. Legends ○: density of 6×6 per subdomain, dashed line: density of 8×8 per subdomain and solid line: density of 12×12 per subdomain.

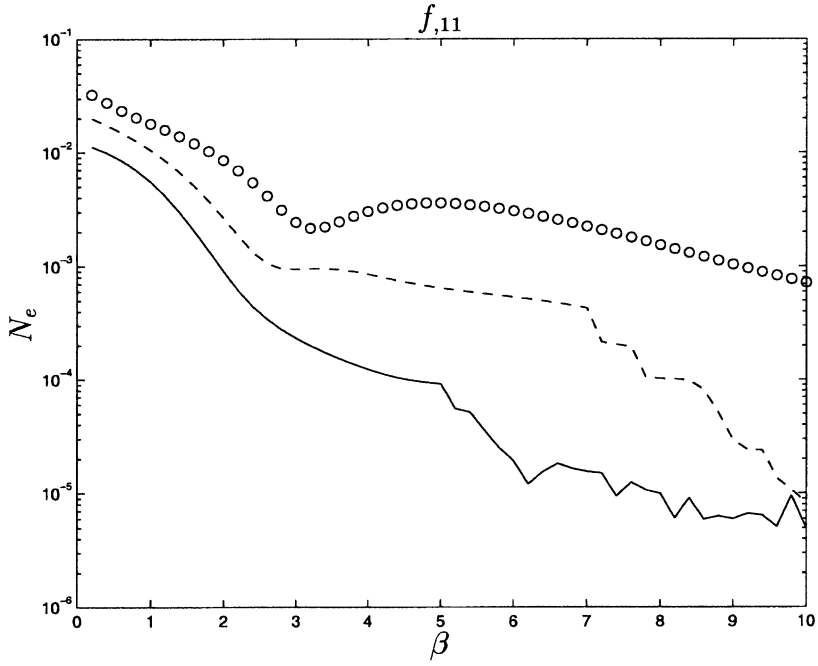

Fig. 10. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$ with four subdomains using IRBFN method: effect of centre density on the quality of approximation of the second derivative $f_{,11}$. Legends ○: density of 6×6 per subdomain, dashed line: density of 8×8 per subdomain and solid line: density of 12×12 per subdomain.

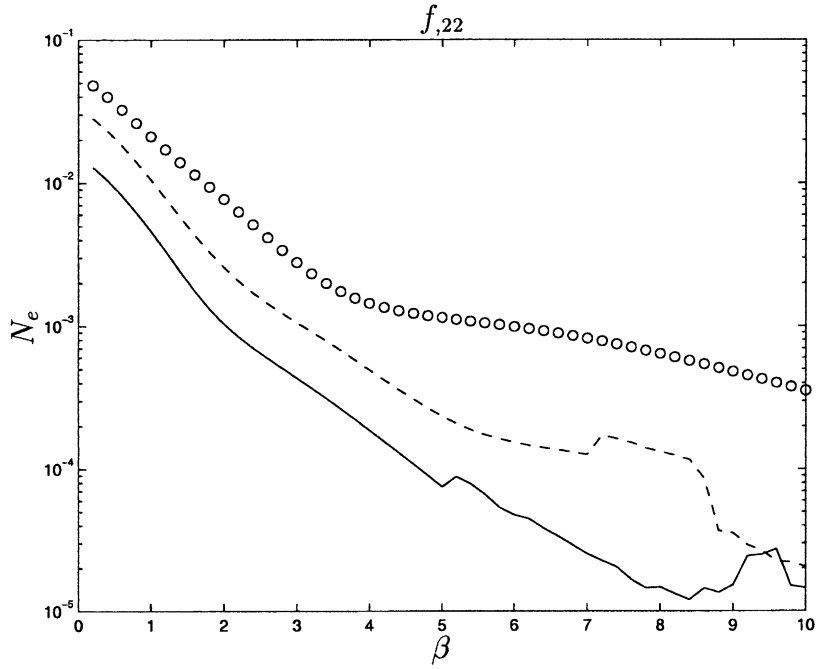

Fig. 11. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$ with four subdomains using IRBFN method: effect of centre density on the quality of approximation of the second derivative $f_{,22}$. Legends ○: density of 6×6 per subdomain, dashed line: density of 8×8 per subdomain and solid line: density of 12×12 per subdomain.

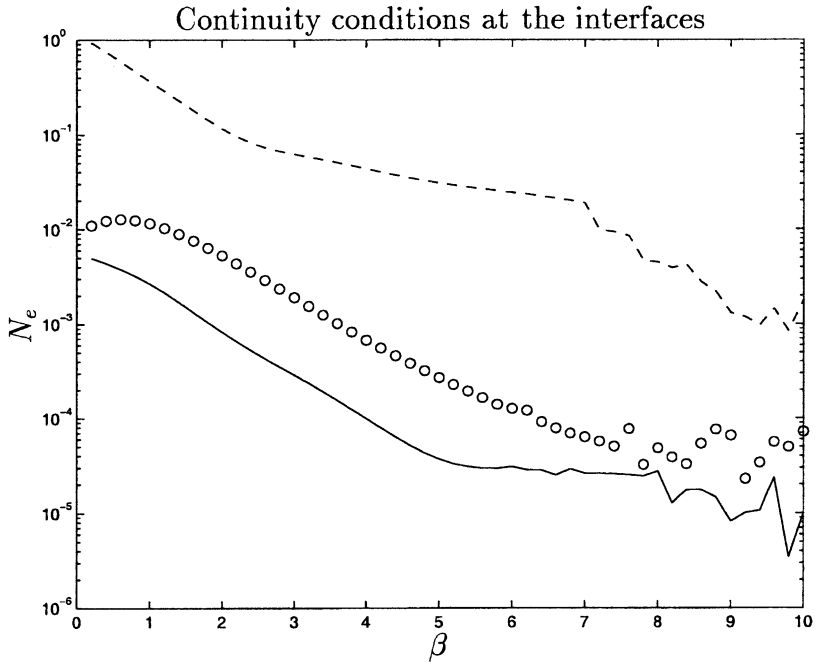

Fig. 12. Approximation of function $y(x_1, x_2) = x_1^2 x_2 + x_2^3/3 + x_2^2/2$ with four subdomains using IRBFN method: plots of error norm of the C_0 and C_1 continuity conditions at the interfaces as functions of β. Legends ○: function f, dashed line: first derivative $f_{,1}$ and solid line: first derivative $f_{,2}$. Note that since $f_{,1}$ vanishes at the interface, the absolute norm is used in this case.

be written as

$$\nabla^2 u = b, \quad \mathbf{x} \in \Omega, \tag{17}$$

$$u = \bar{u}, \quad \mathbf{x} \in \partial\Omega_u, \tag{18}$$

$$q \equiv \frac{\partial u}{\partial n} = \bar{q}, \quad \mathbf{x} \in \partial\Omega_q, \tag{19}$$

where u is the potential, q is the flux across the surface with unit normal $\mathbf{n}$, $\bar{u}$ and $\bar{q}$ are the known boundary conditions, b is known function of position and $\partial\Omega = \partial\Omega_u + \partial\Omega_q$ is the boundary of the domain Ω.

3.1. Review of IRBFN method

The sum squared error associated with Eqs. (17)–(19) is given by

$$\begin{aligned} \mathrm{SSE} = & \sum_{\mathbf{x}^{(i)} \in \Omega} [(u_{,11}(\mathbf{x}^{(i)}) + (u_{,22}(\mathbf{x}^{(i)})) - b(\mathbf{x}^{(i)})]^2 \\ & + \sum_{\mathbf{x}^{(i)} \in \Omega} [u_1(\mathbf{x}^{(i)}) - u_2(\mathbf{x}^{(i)})]^2 \\ & + \sum_{\mathbf{x}^{(i)} \in \partial\Omega_u} [u(\mathbf{x}^{(i)}) - \bar{u}(\mathbf{x}^{(i)})]^2 \\ & + \sum_{\mathbf{x}^{(i)} \in \partial\Omega_q} [(n_1 u_{,1}(\mathbf{x}^{(i)}) + n_2 u_{,2}(\mathbf{x}^{(i)})) - \bar{q}(\mathbf{x}^{(i)})]^2, \end{aligned} \tag{20}$$

where the term $u_1(\mathbf{x}^{(i)})$ is obtained by integrating $u_{,11}$ twice and $u_2(\mathbf{x}^{(i)})$ is similarly obtained via $u_{,22}$. Upon substitution of the expressions for u and its derivatives, i.e. in the form of Eqs. (8)–(10), into the above expression for SSE followed by the application of the general linear least square principle, a system of linear algebraic equations is obtained in terms of the unknown weights in the output layer of the network. The solution of the system of equations (using SVD method in this case) for the unknown weights completes the training of the network (see Ref. [10] for detail).

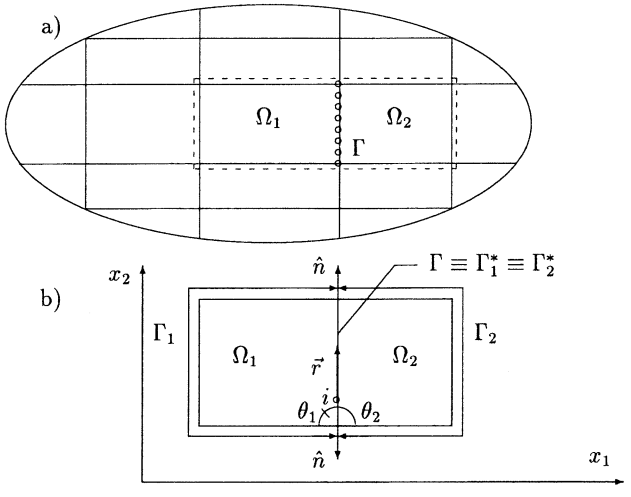

Fig. 13. Potential problem: (a) the original domain with subdomains and (b) a typical case for estimation of the interface boundary conditions. The assumption that the interface is flat is not restrictive since the philosophy of the present method is to convert the domain of analysis to one or more rectangular subdomains.

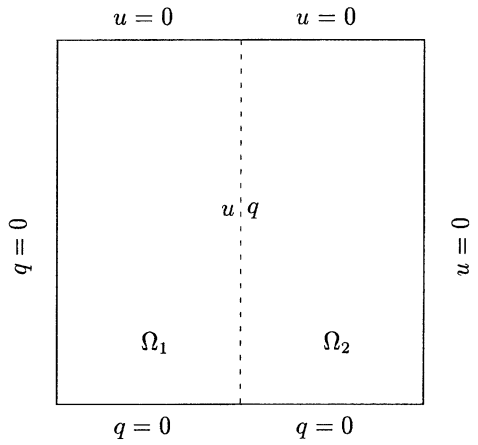

Fig. 14. Poisson's equation on a rectangular domain with two subdomains.

3.2. BIE-based domain decomposition method

IE formulations for potential problems is well documented in a number of texts [19,20] where interested readers are referred for more details. The potential problems governed by Poisson's equation can be reformulated in terms of integral equations for a given spatial point (i) as follows

$$c^{(i)}u^{(i)} + \int_{\partial\Omega} uq^* \,\mathrm{d}\Gamma + \int_{\Omega} bu^* \,\mathrm{d}\Gamma = \int_{\partial\Omega} u^* q \,\mathrm{d}\Gamma, \qquad (21)$$

where u^* is the fundamental solution to the Laplace equation, e.g. for a two dimensional isotropic domain $u^* = (1/2\pi)\ln(1/r)$ in which r is the distance from the point (i) to the current point of integration, $q^* = \partial u^*/\partial n$, $c^{(i)} = \theta/2\pi$ with θ being the internal angle of the corner in radians if (i) is a boundary point and $c^{(i)} = 1$ if (i) is an internal point. To solve large problems, the domain of analysis is divided into a number of subdomains for the same reasons as discussed earlier in Section 2. However, in the present case the boundary conditions at the interfaces are unknown, which need to be determined as part of the solution procedure. Therefore the key question that is addressed here is how to specify the interface boundary conditions. In the present work an iterative solution procedure is employed which allows the update of interface boundary conditions between iterative steps. The procedure consists of the following steps:

1. Divide the domain of analysis into a number of subdomains;
2. Guess initial boundary conditions at the interfaces for each subdomain as appropriate (according to the method explained below);
3. Solve the subproblems corresponding to subdomains using the IRBFN method;
4. Check for convergence;
5. If the procedure is not yet converged, update the boundary conditions (according to the method explained below) on all interfaces and repeat from step 3;
6. If the procedure is converged, stop.

The efficiency of the present method depends on the technique associated with step 5 (and 2 initially) of the above procedure, which is described in detail as follows. The idea of BIE-based DD method for potential problems governed by Laplace's equation [16] is further developed here for Poisson's equations. It is shown in the present work that the volume integral appearing in the standard integral

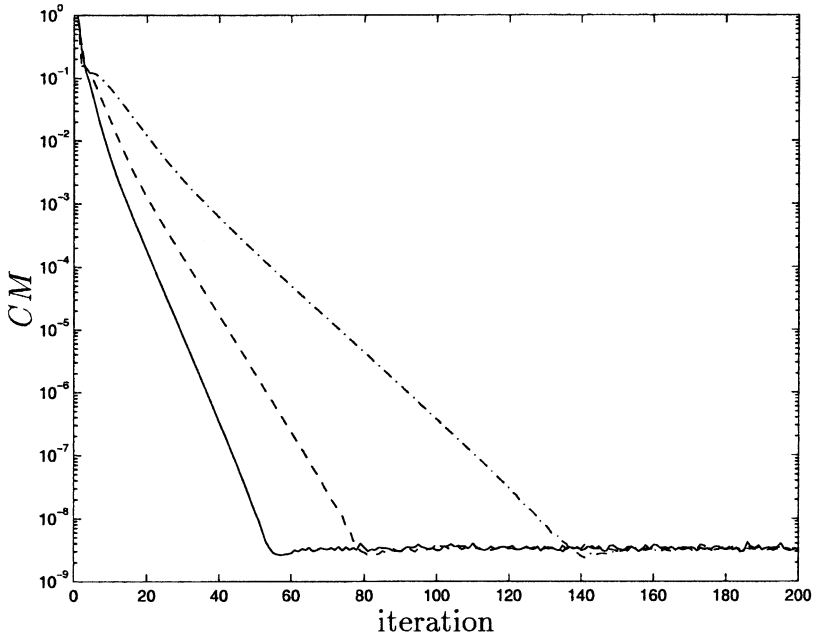

Fig. 15. Poisson's equation on a rectangular domain with two subdomains, density of 6×11 per subdomain, $\beta = 10.0$ and constant interpolation: effect of the relaxation parameter α on the convergence rate. Legends solid line: $\alpha = 0.7$, dashed line: $\alpha = 0.5$ and dashdot line: $\alpha = 0.3$.

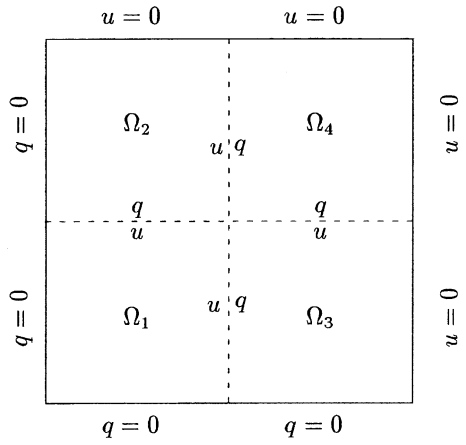

Fig. 16. Poisson's equation on a rectangular domain with four subdomains.

representation of the Poisson's equation can be effectively eliminated in the context of the IRBFN with DD. In the case of Laplace's equation, the boundary conditions at a typical point (i) of a typical interface are updated by regarding the point (i) as the internal point of the domain formed by two adjacent subdomains sharing the interface and thus BIEs can be applied for this special domain to compute a new estimate of the solution at point (i) based on the boundary data obtained in the previous iteration. A typical case of updating therefore consists of one interface and two adjacent subdomains (Fig. 13). The method is fast convergent in comparison with other methods in the literature. In the case of Poisson's equations the extra volume integral mentioned above can be treated in different ways. The simplest one is based on element by element integration schemes over a 'finite element' mesh as well as some special techniques to compute integrals when they become singular. More recently, two new approaches have been proposed where the volume integrals are approximated by boundary integrals in the first approach called the dual reciprocity method (DRM) [21] and in the second approach called the multiple reciprocity method (MRM) [22]. Despite the fact that the final integral equation involves only boundary integrals it can be argued that the approximate nature of the intermediate transformations is less than ideal. Eq. (21) for the internal points is not applied directly here for the estimation of the boundary conditions at the interfaces. Instead, an alternative equation is developed as follows.

Consider the typical case where the formulation involves the interface Γ and two adjacent subdomains Ω_1 and Ω_2 (Fig. 13). The relevant volume and boundaries are $\Omega = \Omega_1 + \Omega_2$, $\partial\Omega_1 = \Gamma_1 + \Gamma_1^*$, $\partial\Omega_2 = \Gamma_2 + \Gamma_2^*$, $\partial\Omega = \Gamma_1 + \Gamma_2$, and $\Gamma \equiv \Gamma_1^* \equiv \Gamma_2^*$. The interface values (i.e. boundary condition for the current iteration) for u are estimated by (see Appendix A)

$$u_{\text{update}}^{(i)} = \frac{1}{2}\bar{u}_1^{(i)} + \frac{1}{2}\bar{u}_2^{(i)} - \int_{\Gamma_1^*} u^*\bar{q}_1 \,\mathrm{d}\Gamma - \int_{\Gamma_2^*} u^*\bar{q}_2 \,\mathrm{d}\Gamma$$

$$= \frac{1}{2}\bar{u}_1^{(i)} + \frac{1}{2}\bar{u}_2^{(i)} - \int_{\Gamma_1^*} u^*(\bar{q}_1 + \bar{q}_2) \,\mathrm{d}\Gamma, \qquad (22)$$

where $\bar{u}_1$ and $\bar{q}_1$ denote the potential and flux obtained from solving the subproblem on Ω_1 in the previous iteration; $\bar{u}_2$ and $\bar{q}_2$ the potential and flux obtained from solving the

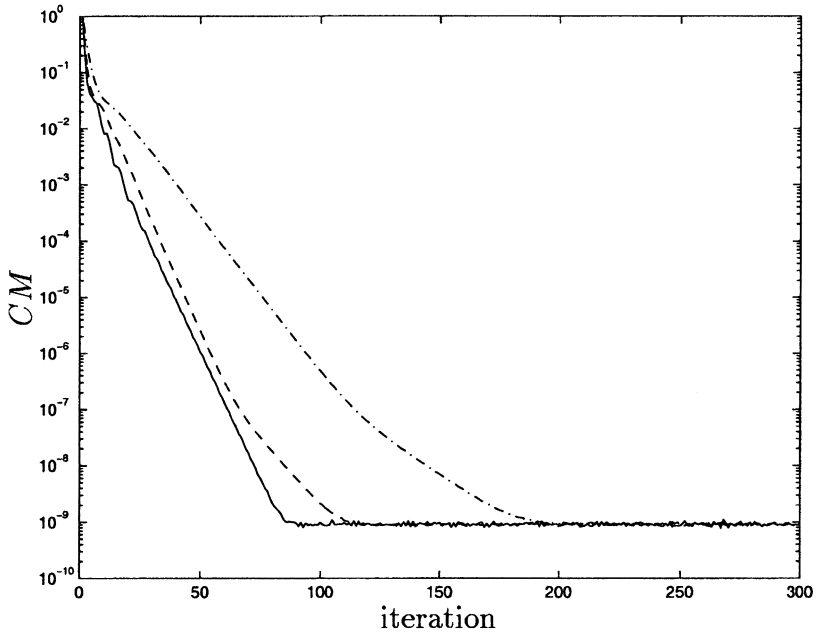

Fig. 17. Poisson's equation on a rectangular domain with four subdomains, density of 5×5 per subdomain, $\beta = 10.0$ and constant interpolation: effect of the relaxation parameter α on the convergence rate. Legends solid line: $\alpha = 1.0$, dashed line: $\alpha = 0.5$ and dashdot line: $\alpha = 0.3$.

Table 2
Poisson's equation on a rectangular domain with nine subdomains, density of 7×7 per subdomain and $\beta = 10$: the set of weights of the IRBFN representing $u_{,11}$, $u_{,1}$ and u on the middle subdomain

The set of weights associated to basic functions (centres)

Centre location		Weight	Centre location		Weight
x_1	x_2	w	x_1	x_2	w
0.67	0.67	1.693199114×10^4	0.75	0.78	2.112770124×10^4
0.67	0.69	-2.816209725×10^4	0.75	0.81	1.012951159×10^4
0.67	0.72	1.228455709×10^4	0.75	0.83	1.542473465×10^4
0.67	0.75	-1.364780532×10^4	0.78	0.67	3.804032341×10^4
0.67	0.78	3.856839495×10^4	0.78	0.69	-3.681410331×10^4
0.67	0.81	-3.759308216×10^4	0.78	0.72	1.837293513×10^4
0.67	0.83	1.093331969×10^4	0.78	0.75	3.029359776×10^4
0.69	0.67	-2.391430283×10^4	0.78	0.78	-2.742975230×10^4
0.69	0.69	1.333560446×10^4	0.78	0.81	-6.178016998×10^3
0.69	0.72	2.948646455×10^4	0.78	0.83	-1.754991203×10^4
0.69	0.75	-2.080570430×10^4	0.81	0.67	-3.369685369×10^4
0.69	0.78	-3.455831824×10^4	0.81	0.69	5.064940901×10^4
0.69	0.81	5.263324075×10^4	0.81	0.72	-4.234401926×10^4
0.69	0.83	-1.257079769×10^4	0.81	0.75	1.732534843×10^4
0.72	0.67	6.737959686×10^3	0.81	0.78	-8.557586900×10^3
0.72	0.69	3.038198282×10^4	0.81	0.81	-6.348141664×10^2
0.72	0.72	-2.450622417×10^4	0.81	0.83	1.645079836×10^4
0.72	0.75	-4.361789624×10^3	0.83	0.67	7.501093943×10^3
0.72	0.78	2.173253939×10^4	0.83	0.69	-6.210448673×10^3
0.72	0.81	-3.437141188×10^4	0.83	0.69	-6.199659754×10^3
0.72	0.83	-2.504594759×10^3	0.83	0.75	1.970736736×10^4
0.75	0.67	-1.293572781×10^4	0.83	0.78	-2.442658145×10^4
0.75	0.69	-1.579023866×10^4	0.83	0.81	2.095029960×10^4
0.75	0.72	-3.814549939×10^3	0.83	0.83	-1.094565236×10^4
0.75	0.75	-8.475572297×10^3			

Sets of weights for added univariate functions

Function C_1		Function C_2	
Centre location	Weight	Centre location	Weight
x_2	$\bar{w}$	x_2	$\bar{w}$
0.67	6.183109562×10^3	0.67	-8.038381380×10^3
0.69	-1.412741063×10^4	0.69	1.834206273×10^4
0.72	1.699781205×10^3	0.72	-2.309958341×10^3
0.75	1.263920205×10^4	0.75	-1.644634977×10^4
0.78	1.519665651×10^3	0.78	-1.773062979×10^3
0.81	-1.428871268×10^4	0.81	1.882600576×10^4
0.83	6.374766289×10^3	0.83	-8.595252093×10^3
$\hat{C}_1$	$8.746639736 \times 10^{-1}$	$\hat{C}_1$	-1.521300487
$\hat{C}_2$	-1.523691726	$\hat{C}_2$	2.404102741

subproblem on Ω_2 in the previous iteration. The flux boundary condition for the current iteration at point (i) across the interface Γ is estimated by (see Appendix A)

$$q^{(i)}_{\text{update}} = \frac{\partial u^{(i)}}{\partial x_1} - \Delta e_q = \frac{1}{2}(\bar{q}_1^{(i)} - \bar{q}_2^{(i)}) + \text{HPV}\int_{\Gamma_1^*} (\bar{u}_1 - \bar{u}_2)\frac{1}{2\pi r^2}\,\mathrm{d}\Gamma, \tag{23}$$

where HPV is the Hadamard principal value. Eqs. (22) and (23) provide a neat mechanism for the estimation of the potential u and its flux q on the interface Γ (i.e. subdomain boundary conditions for the current iteration) in terms of the solution at the previous iteration. Note that when the point (i) is at the extreme ends of the interface Γ, the right angle corner geometry (all subdomains dealt with here are rectangular as discussed in Section 3.4) requires that Eqs. (22) and (23) are replaced by

$$\frac{1}{2}u^{(i)}_{\text{update}} = \frac{1}{4}\bar{u}_1^{(i)} + \frac{1}{4}\bar{u}_2^{(i)} - \int_{\Gamma_1^*} u^*(\bar{q}_1 + \bar{q}_2)\,\mathrm{d}\Gamma \tag{24}$$

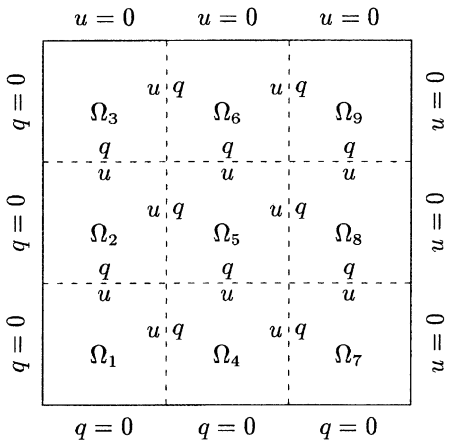

Fig. 18. Poisson's equation on a rectangular domain with nine subdomains.

and

$$\frac{1}{2}q^{(i)}_{\text{update}} = \frac{1}{4}\left(\bar{q}_1^{(i)} - \bar{q}_2^{(i)}\right) + \frac{1}{2\pi}\left(\frac{\partial \bar{u}_1^{(i)}}{\partial \hat{n}} - \frac{\partial \bar{u}_2^{(i)}}{\partial \hat{n}}\right) + \text{HPV}\int_{\Gamma_1^*}(\bar{u}_1 - \bar{u}_2)\frac{1}{2\pi r^2}\,\mathrm{d}\Gamma, \tag{25}$$

where $\hat{\mathbf{n}}$ is the unit outward normal vector at the corner point (i) as shown in Fig. 13. It is clear that by using the estimation formulae (22) and (23) for normal interface points or, formulae (24) and (25) for the extreme interface points, the volume integrals are completely eliminated and the boundary integrals are confined to the interface Γ only. This technique can be advantageously and simply combined with mesh-free RBFN methods or with other numerical methods such as BEM and FEM. Furthermore, at the start of the solution procedure an IE similar to Eq. (21) can be written for the whole original domain and then the volume integral as well as the unknown real boundary data are set to zero to make the IE a BIE with known boundary data which can be evaluated at all relevant points on all interfaces to provide an initial guess of the interface boundary conditions as in the case of Laplace's equation [16]. The above technique offers an automatic means of specifying initial guess boundary conditions on the interfaces. At a given iteration, a convergence measure (CM) is defined based on the potential u on the interfaces as:

$$\text{CM} = \sqrt{\frac{\sum_{\text{inter face}}\sum_{i=1}^{N}(\bar{u}_1(\mathbf{x}^{(i)}) - \bar{u}_2(\mathbf{x}^{(i)})}{\sum_{\text{inter face}}\sum_{i=1}^{N}(\bar{u}_1(\mathbf{x}^{(i)}))^2}}, \tag{26}$$

where the subscripts denote the two subdomains with the common inter face under consideration, (i) denotes the points on the inter face and N is the number of points on the inter face. In Sections 3.3 and 3.4, the approach based on a combination of IRBFN method with BIE-based DD method is verified with a number of example potential problems governed by Poisson's equations on rectangular and non-rectangular domains.

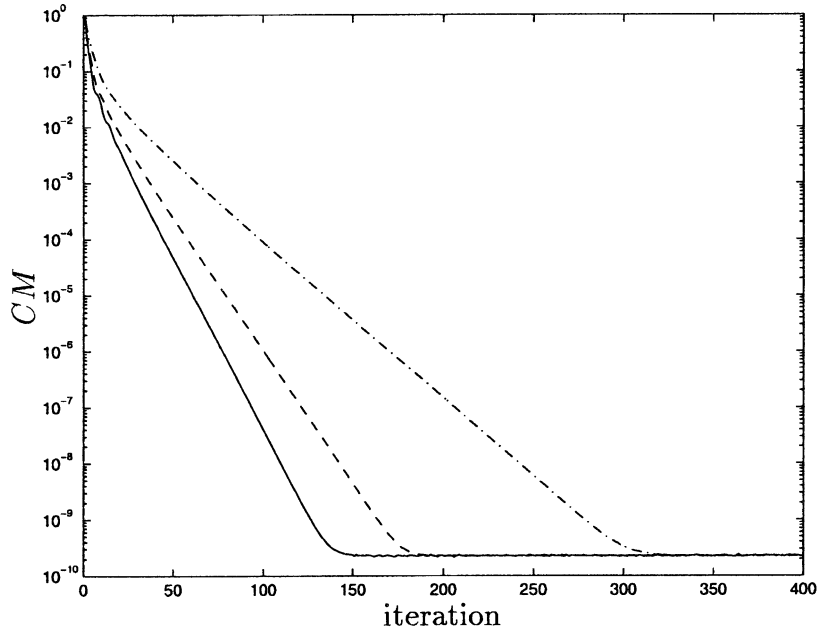

Fig. 19. Poisson's equation on a rectangular domain with nine subdomains, density of 5×5 per subdomain, $\beta = 10.0$ and constant interpolation: effect of the relaxation parameter α on the convergence rate. Legends solid line: $\alpha = 0.7$, dashed line: $\alpha = 0.5$ and dashdot line: $\alpha = 0.3$.

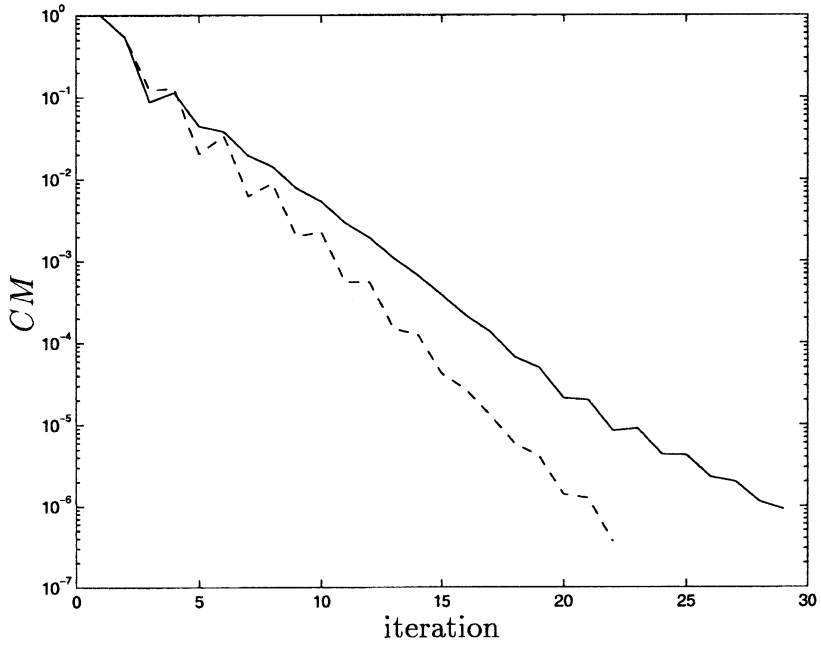

Fig. 20. Poisson's equation on a rectangular domain with two subdomains, density of 6×11 per subdomain, $\beta = 10.0$ and $\alpha = 1$: effect of interpolation type used to compute the BIEs on the convergence rate. Legends solid line: constant interpolation and dashed line: linear interpolation.

3.3. Rectangular domain

The problem here is to determine a function $u(x_1,x_2)$ satisfying the following equation

$$\nabla^2 u = \sin(\pi x_1)\sin(\pi x_2) \tag{27}$$

on the rectangle $0 \le x_1 \le 1, 0 \le x_2 \le 1$ subject to the Dirichlet condition $u = 0$ along the whole boundary of the domain. The exact solution is given by

$$u_e(x_1, x_2) = -\frac{1}{2\pi^2}\sin(\pi x_1)\sin(\pi x_2). \tag{28}$$

Owing to symmetry, only a quarter of domain with dimension being (0.5,1.0) × (0.5,1.0) is considered. Three cases of DD corresponding to two subdomains (Fig. 14), four subdomains (Fig. 16) and nine subdomains (Fig. 18) are studied in which the latter case involves a subdomain (the middle one) that is bounded by only interfaces (i.e. only artificial boundaries), providing a strong test for the BIE-based DD method. The boundary condition type at each interface is chosen as Dirichlet–Neumann type. Due to the iterative nature of the procedure, a relaxation is applied to the interface solutions between iterations to ensure convergence. The interface boundary conditions at the current iteration is estimated as follows.

$$u_k^{(i)} = \alpha u_k^{(i)} + (1-\alpha)u_{k-1}^{(i)}, \tag{29}$$

$$q_k^{(i)} = \alpha q_k^{(i)} + (1-\alpha)q_{k-1}^{(i)}, \tag{30}$$

where k denotes the current iteration and α is the relaxation parameter, $0 < \alpha \le 1$. Note that $\alpha = 1$ means no relaxation. In general, the selection of the relaxation parameter is based on experience and no well-established choice is available. The computation of boundary integrals in the estimation formulae (22)–(25) can be found in the literature (e.g. Refs. [19,20,23–27]). Figs. 15, 17 and 19 show the rate of convergence of the iterative procedure for cases of two, four and nine subdomains, respectively with the use of constant interpolation to compute the boundary integrals in Eqs. (22) and (23). For all three cases the achieved CMs, which describe the quality of the continuity of the solution u across the interfaces, are very small with the order of magnitude of about 10^{-10}–10^{-9}. With the tolerance for the CM set at 1.0×10^{-6}, it takes only 38, 50 and 75 iterations to meet the convergence criterion for two, four and nine subdomains, respectively using constant interpolation and $\alpha = 0.7$. Thus, the number of iterations required for convergence increases less than linearly with increasing number of subdomains. The convergence rate is slower when the

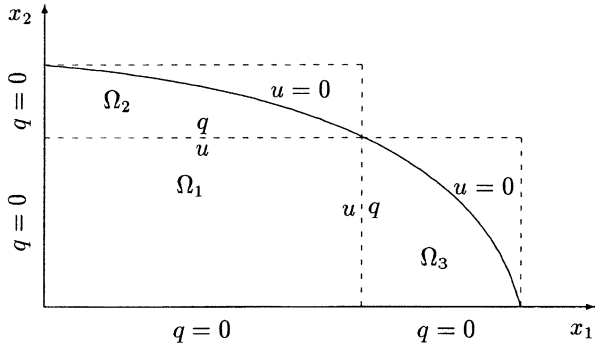

Fig. 21. Poisson's equation on a non-rectangular domain with three subdomains.

Table 3
Poisson's equation on a non-rectangular domain with three subdomains, density of 7×7 per subdomain and $\beta = 10$: the set of weights of the IRBFN representing $u_{,11}$, $u_{,1}$ and u on the subdomain $(0,0) \times (6,6.4)$

The set of weights associated to basic functions (centres)

Centre location		Weight	Centre location		Weight
x_1	x_2	w	x_1	x_2	w
0.00	0.00	$-3.273200519 \times 10^{1}$	3.00	4.27	$-1.087188369 \times 10^{2}$
0.00	1.07	-2.417528784	3.00	5.33	$1.533015567 \times 10^{2}$
0.00	2.13	$9.804610545 \times 10^{1}$	3.00	6.40	$4.851562419 \times 10^{1}$
0.00	3.20	$-1.003272271 \times 10^{2}$	4.00	0.00	$-1.489016571 \times 10^{2}$
0.00	4.27	$-4.417054190 \times 10^{1}$	4.00	1.07	$-1.267887675 \times 10^{2}$
0.00	5.33	$1.583623003 \times 10^{2}$	4.00	2.13	$1.558888831 \times 10^{2}$
0.00	6.40	$-8.442559755 \times 10^{1}$	4.00	3.20	$1.172814092 \times 10^{2}$
1.00	0.00	$1.369343111 \times 10^{2}$	4.00	4.27	$-1.532140453 \times 10^{1}$
1.00	1.07	$-8.106110291 \times 10^{1}$	4.00	5.33	$1.016357830 \times 10^{2}$
1.00	2.13	$-5.776551407 \times 10^{1}$	4.00	6.40	$-1.578114359 \times 10^{2}$
1.00	3.20	$7.976419081 \times 10^{1}$	5.00	0.00	$1.646474035 \times 10^{2}$
1.00	4.27	$2.820343572 \times 10^{1}$	5.00	1.07	$-1.855840000 \times 10^{2}$
1.00	5.33	$-2.357222003 \times 10^{2}$	5.00	2.13	$6.765236810 \times 10^{1}$
1.00	6.40	$1.616501428 \times 10^{2}$	5.00	3.20	$7.668384210 \times 10^{1}$
2.00	0.00	$-2.004233583 \times 10^{2}$	5.00	4.27	$-1.397408874 \times 10^{2}$
2.00	1.07	$3.845634059 \times 10^{1}$	5.00	5.33	$-7.678328889 \times 10^{1}$
2.00	2.13	$-3.215321182 \times 10^{1}$	5.00	6.40	$1.330324364 \times 10^{2}$
2.00	3.20	$1.148679374 \times 10^{2}$	6.00	0.00	$-8.103128221 \times 10^{1}$
2.00	4.27	$1.668440291 \times 10^{2}$	6.00	1.07	$1.749444696 \times 10^{2}$
2.00	5.33	$-8.033197245 \times 10^{1}$	6.00	2.13	$-1.371917744 \times 10^{2}$
2.00	6.40	$-7.140762573 \times 10^{1}$	6.00	3.20	$-2.726846761 \times 10^{1}$
3.00	0.00	$1.572456664 \times 10^{2}$	6.00	4.27	$1.097472024 \times 10^{2}$
3.00	1.07	$1.963528850 \times 10^{2}$	6.00	5.33	$-1.796043231 \times 10^{1}$
3.00	2.13	$-1.116318852 \times 10^{2}$	6.00	6.40	$-3.088212258 \times 10^{1}$
3.00	3.20	$-2.515371965 \times 10^{2}$			0.000000000

Sets of weights for added univariate functions

Function C_1		Function C_2	
Centre location	Weight	Centre location	Weight
x_2	$\bar{w}$	x_2	$\bar{w}$
0.00	1.321069480	0.00	$-1.379429592 \times 10^{1}$
1.07	-3.422324257	1.07	$2.981168322 \times 10^{1}$
2.13	1.081475130	2.13	2.528663931
3.20	3.092650960	3.20	$-3.913968045 \times 10^{1}$
4.27	$-2.780903157 \times 10^{-1}$	4.27	-5.126029323
5.33	-3.942437638	5.33	$5.186413245 \times 10^{1}$
6.40	2.150150746	6.40	$-2.622549517 \times 10^{1}$
$\hat{C}_1$	$8.857086521 \times 10^{-1}$	$\hat{C}_1$	$-1.228665283 \times 10^{1}$
$\hat{C}_2$	3.113289140	$\hat{C}_2$	$1.309757543 \times 10^{2}$

relaxation parameter α decreases, however it will be shown later that convergence is more stable. The accuracy of the final solution obtained is excellent with error norms of 1.67×10^{-6}, 1.85×10^{-6} and 3.66×10^{-7} for two, four and nine subdomains, respectively. Using higher order interpolation to compute boundary integrals can achieve faster convergence rate as shown in Fig. 20. It takes about 22 and 29 iterations to reach convergence for linear and constant element schemes, respectively in the case of two subdomains and $\alpha = 1$. It can be seen from Figs. 15 and 20 that the rate of convergence is more stable when the relaxation parameter α decreases. Table 2 displays the set of weights of the IRBF network representing functions $u_{,11}$, $u_{,1}$ and u on the middle subdomain.

3.4. Non-rectangular domain

Consider the following Poisson's equation

$$\nabla^2 u = -2 \tag{31}$$

on an elliptical domain with semi-major axis $a = 10$ and semi-minor axis $b = 8$ (Fig. 21). The homogeneous

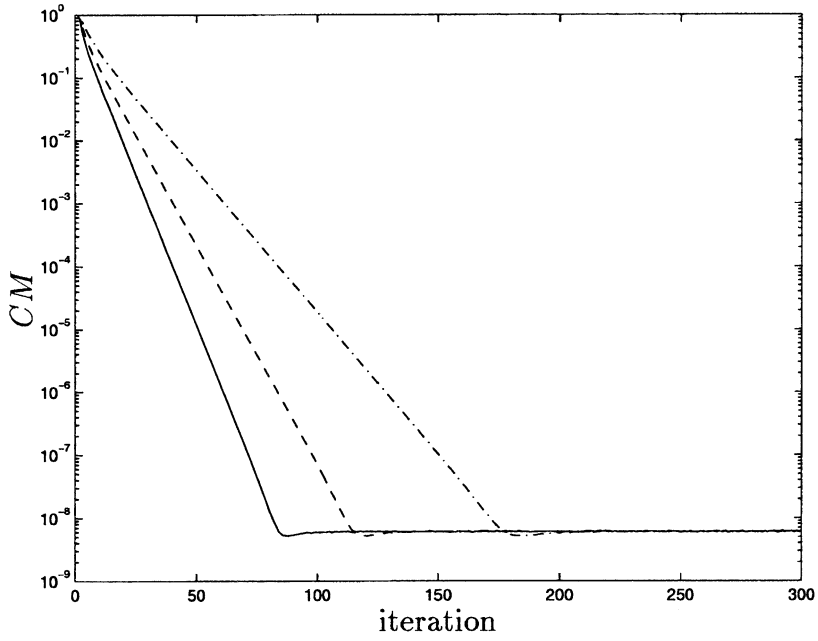

Fig. 22. Poisson's equation on a non-rectangular domain with three subdomains, density of 7×7 per subdomain, $\beta = 10.0$ and constant interpolation: effect of the relaxation parameter α on the convergence rate. Legends solid line: $\alpha = 1.0$, dashed line: $\alpha = 0.75$ and dashdot line: $\alpha = 0.5$.

condition $u = 0$ is imposed along the whole boundary. The corresponding exact solution is

$$u_e(x_1, x_2) = -\left(\frac{a^2 b^2}{a^2 + b^2}\right)\left(\frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - 1\right) \tag{32}$$

Owing to symmetry, only the first quarter of elliptical domain ($0 \le x_1 \le a$ and $0 \le x_2 \le b$) denoted by Ω is considered. The IRBF networks representing the solution u and its derivatives are trained to satisfy Poisson's equation (31) on the domain Ω and also its boundary conditions as $\{u = 0$ on the curved boundary and $q = 0$ on the lines $x_1 = 0$; $x_2 = 0\}$ denoted by $\overline{\mathrm{BC}}$. However, it can be seen that if the networks are trained to satisfy the Poisson's equation on the extended domain Ω^* ($\Omega^* \supset \Omega$) and the boundary conditions $\overline{\mathrm{BC}}$, the solution obtained must be also the solution of

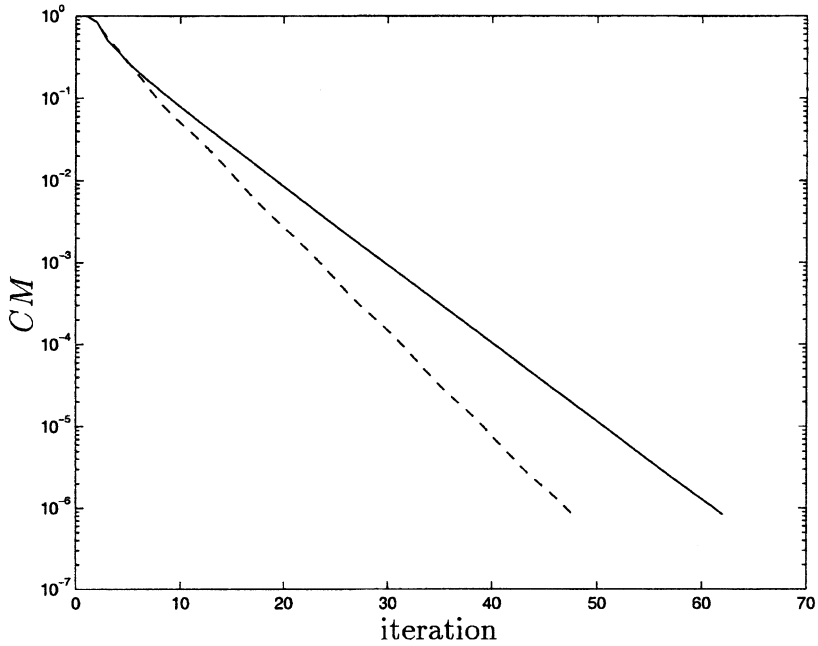

Fig. 23. Poisson's equation on a non-rectangular domain with three subdomains, density of 7×7 per subdomain, $\beta = 10.0$ and $\alpha = 1$: effect of the interpolation type used to compute the BIEs on the convergence rate. Legends solid line: constant interpolation and dashed line: quadratic interpolation.

the problem of the original domain Ω. Hence, in order to make it easier for pre-processing, the non-rectangular original domain here can be divided into a number of rectangular subdomains provided that these subdomains can cover up the original domain. Then, for each extended subdomain, the set of centres is distributed regularly as in the case of normal rectangular subdomain, while the set of collocation points consists of two parts. The first part is the same as the set of centres in order to satisfy the Poisson's equation and the second part can be chosen as a set of random points on the curved boundary (e.g. in this example, they are chosen as intersection points between the curved boundary and the rectangular gridlines through the centres) in order to satisfy the boundary conditions $\overline{\mathrm{BC}}$. Extending the non-rectangular subdomain into the rectangular subdomain allows the shape of all subdomains to be rectangle resulting in a very easy and automatic means of creating centres and collocation points. For the purpose of illustration, three subdomains $(0,0)\times(6.0,6.4)$, $(0,6.4)\times(6.0,8.0)$ and $(6.0,0)\times(10.0,6.4)$ (Fig. 21) are used to cover up the domain Ω. The effects of the relaxation parameter α and the type of interpolation used to compute boundary integrals on the convergence rate are displayed in Figs. 22 and 23, respectively. Similar remarks can be made as in the case of problems with rectangular domain discussed earlier. After the convergence is reached, the error norm of the solution is 2.35×10^{-6} and the error norm of the continuity across the interfaces is of the order of 10^{-9}. The set of weights of the IRBF network approximating functions $u_{,11}$, $u_{,1}$ and u on the subdomain $(0,0)\times(6.0,6.4)$ is shown in Table 3.

4. Concluding remarks

We have reported an approach based on mesh-free RBFN methods with DD for approximation of function and numerical solution of Poisson's equation. For function approximation, numerical results showed that IRBFN achieves greater accuracy than DRBFN over a wide range of RBF widths and therefore the dependence of the performance of the network on the RBF width can be alleviated with the adoption of the indirect approach, i.e. the IRBFN approach. In solving Poisson's equation, the BIE-based DD method for the estimation of boundary conditions at interfaces is proposed where BIE computation is confined only to the interfaces resulting in convenient incorporation of this technique into the IRBFN method. The convergence rate of the approach can be affected by the element type used to compute BIEs. The results obtained by the present approach are excellent, by the norms of error discussed earlier, on both rectangular and non-rectangular domains. The BIE-based DD method is very suitable for coarse-grained parallel processing and can be extended to problems whose governing equations can be expressed in terms of IEs such as viscous flow problems.

Acknowledgements

This work is supported by a Special USQ research Grant (Grant No. 179-310) to T. Tran-Cong. N. Mai-Duy is supported by a USQ Scholarship. This support is gratefully acknowledged. The authors would like to thank the referees for their helpful comments.

Appendix A

A.1. Estimate of the interface boundary condition for u

A mechanism for the iterative update of interface boundary u,q-values is developed here. Consider the typical case where the formulation involves the interface Γ and two adjacent rectangular subdomains Ω_1 and Ω_2 (Fig. 13). The assumption that the interface is flat is not restrictive since the philosophy of the present method is to convert the domain of analysis to one or more rectangular subdomains as explained in Section 3.4. The relevant volume and boundaries are $\Omega=\Omega_1+\Omega_2$, $\partial\Omega_1=\Gamma_1+\Gamma_1^*$, $\partial\Omega_2=\Gamma_2+\Gamma_2^*$, $\partial\Omega=\Gamma_1+\Gamma_2$ and $\Gamma\equiv\Gamma_1^*\equiv\Gamma_2^*$.

Let (u,q) be the solution to the problem corresponding to a known function b.

- Point (i) on the interface Γ can be considered as an internal point relative to the domain Ω as defined above and therefore the IE for the internal point (i) with $c^{(i)}=1$ can be expressed as

$$c^{(i)}u^{(i)}=\int_{\partial\Omega}u^*q\,\mathrm{d}\Gamma-\int_{\partial\Omega}uq^*\,\mathrm{d}\Gamma-\int_{\Omega}bu^*\,\mathrm{d}\Omega \quad \text{(A1)}$$

or,

$$u^{(i)}=\int_{\Gamma_1}u^*q\,\mathrm{d}\Gamma-\int_{\Gamma_1}uq^*\,\mathrm{d}\Gamma-\int_{\Omega_1}bu^*\,\mathrm{d}\Omega+\int_{\Gamma_2}u^*q\,\mathrm{d}\Gamma-\int_{\Gamma_2}uq^*\,\mathrm{d}\Gamma-\int_{\Omega_2}bu^*\,\mathrm{d}\Omega. \quad \text{(A2)}$$

- Point (i) on the interface Γ can also be considered as a boundary point relative to the domain Ω_1 and therefore the IE for the boundary point (i) with $c^{(i)}=1/2$ can be expressed as

$$\begin{aligned}c^{(i)}u^{(i)}&=\int_{\partial\Omega_1}u^*q\,\mathrm{d}\Gamma-\mathrm{CPV}\int_{\partial\Omega_1}uq^*\,\mathrm{d}\Gamma-\int_{\Omega_1}bu^*\,\mathrm{d}\Omega\\&=\left(\int_{\Gamma_1}u^*q\,\mathrm{d}\Gamma+\int_{\Gamma_1^*}u^*q\,\mathrm{d}\Gamma\right)-\left(\int_{\Gamma_1}uq^*\,\mathrm{d}\Gamma+\mathrm{CPV}\int_{\Gamma_1^*}uq^*\,\mathrm{d}\Gamma\right)-\int_{\Omega_1}bu^*\,\mathrm{d}\Omega\end{aligned} \quad \text{(A3)}$$

or,

$$\frac{1}{2}u^{(i)} = \int_{\Gamma_1} u^* q \,\mathrm{d}\Gamma - \int_{\Gamma_1} uq^* \,\mathrm{d}\Gamma - \int_{\Omega_1} bu^* \,\mathrm{d}\Omega + \int_{\Gamma_1^*} u^* q \,\mathrm{d}\Gamma - \mathrm{CPV}\int_{\Gamma_1^*} uq^* \,\mathrm{d}\Gamma, \tag{A4}$$

where CPV is Cauchy principal value.

- Point (i) on the interface Γ can also be considered as a boundary point relative to the domain Ω_2 and hence the IE for the boundary point (i) with $c^{(i)} = 1/2$ can be expressed as

$$\begin{aligned} c^{(i)}u^{(i)} &= \int_{\partial\Omega_2} u^* q \,\mathrm{d}\Gamma - \mathrm{CPV}\int_{\partial\Omega_2} uq^* \,\mathrm{d}\Gamma - \int_{\Omega_2} bu^* \,\mathrm{d}\Omega \\ &= \left(\int_{\Gamma_2} u^* q \,\mathrm{d}\Gamma + \int_{\Gamma_2^*} u^* q \,\mathrm{d}\Gamma\right) - \left(\int_{\Gamma_2} uq^* \,\mathrm{d}\Gamma + \mathrm{CPV}\int_{\Gamma_2^*} uq^* \,\mathrm{d}\Gamma\right) - \int_{\Omega_2} bu^* \,\mathrm{d}\Omega \end{aligned} \tag{A5}$$

or,

$$\frac{1}{2}u^{(i)} = \int_{\Gamma_2} u^* q \,\mathrm{d}\Gamma - \int_{\Gamma_2} uq^* \,\mathrm{d}\Gamma - \int_{\Omega_2} bu^* \,\mathrm{d}\Omega + \int_{\Gamma_2^*} u^* q \,\mathrm{d}\Gamma - \mathrm{CPV}\int_{\Gamma_2^*} uq^* \,\mathrm{d}\Gamma. \tag{A6}$$

At the iteration k where the convergence to the solution has not been achieved yet, the actual solutions u and q at points along $\partial\Omega$ can be expressed as

$$u = \bar{u} + \Delta\bar{u}, \quad \mathbf{x} \in \partial\Omega, \tag{A7}$$

$$q = \bar{q} + \Delta\bar{q}, \quad \mathbf{x} \in \partial\Omega, \tag{A8}$$

where $\bar{u}$ and $\bar{q}$ are current approximations obtained at the iteration $(k-1)$ by solving subdomain problems while $\Delta\bar{u}$ and $\Delta\bar{q}$ can be regarded as errors of $\bar{u}$ and $\bar{q}$, respectively. However, at points (i) along the interface Γ, there are two current approximations corresponding to the two adjacent subdomains for each variable u and q, which can be written for point (i) as

$$u^{(i)} = \bar{u}_1^{(i)} + \Delta\bar{u}_1^{(i)} = \bar{u}_2^{(i)} + \Delta\bar{u}_2^{(i)}, \quad \mathbf{x} \in \Gamma, \tag{A9}$$

$$q^{(i)} = \bar{q}_1^{(i)} + \Delta\bar{q}_1^{(i)} = \bar{q}_2^{(i)} + \Delta\bar{q}_2^{(i)}, \quad \mathbf{x} \in \Gamma, \tag{A10}$$

where subscripts are used to denote subdomains.

The substitution of expressions (A7) and (A8) into Eq. (A2) yields

$$\begin{aligned} u^{(i)} = &\int_{\Gamma_1} u^*(\bar{q} + \Delta\bar{q}) \,\mathrm{d}\Gamma - \int_{\Gamma_1} (\bar{u} + \Delta\bar{u})q^* \,\mathrm{d}\Gamma \\ &- \int_{\Omega_1} bu^* \,\mathrm{d}\Omega + \int_{\Gamma_2} u^*(\bar{q} + \Delta\bar{q}) \,\mathrm{d}\Gamma \\ &- \int_{\Gamma_2} (\bar{u} + \Delta\bar{u})q^* \,\mathrm{d}\Gamma - \int_{\Omega_2} bu^* \,\mathrm{d}\Omega \end{aligned} \tag{A11}$$

and the substitution of expressions (A7)–(A10) into Eqs. (A4) and (A6) yields, respectively,

$$\begin{aligned} \frac{1}{2}\left(\bar{u}_1^{(i)} + \Delta\bar{u}_1^{(i)}\right) = &\int_{\Gamma_1} u^*(\bar{q} + \Delta\bar{q}) \,\mathrm{d}\Gamma - \int_{\Gamma_1} (\bar{u} + \Delta\bar{u})q^* \,\mathrm{d}\Gamma \\ &- \int_{\Omega_1} bu^* \,\mathrm{d}\Omega + \int_{\Gamma_1^*} u^*(\bar{q}_1 + \Delta\bar{q}_1) \,\mathrm{d}\Gamma \\ &- \mathrm{CPV}\int_{\Gamma_1^*} (\bar{u}_1 + \Delta\bar{u}_1)q^* \,\mathrm{d}\Gamma \end{aligned} \tag{A12}$$

and

$$\begin{aligned} \frac{1}{2}\left(\bar{u}_2^{(i)} + \Delta\bar{u}_2^{(i)}\right) = &\int_{\Gamma_2} u^*(\bar{q} + \Delta\bar{q}) \,\mathrm{d}\Gamma - \int_{\Gamma_2} (\bar{u} + \Delta\bar{u})q^* \,\mathrm{d}\Gamma \\ &- \int_{\Omega_1} bu^* \,\mathrm{d}\Omega + \int_{\Gamma_2^*} u^*(\bar{q}_2 + \Delta\bar{q}_2) \,\mathrm{d}\Gamma \\ &- \mathrm{CPV}\int_{\Gamma_2^*} (\bar{u}_2 + \Delta\bar{u}_2)q^* \,\mathrm{d}\Gamma. \end{aligned} \tag{A13}$$

The subtraction of Eqs. (A12) and (A13) from Eq. (A11) yields

$$\begin{aligned} u^{(i)} = &\frac{1}{2}\left(\bar{u}_1^{(i)} + \Delta\bar{u}_1^{(i)}\right) - \int_{\Gamma_1^*} u^*(\bar{q}_1 + \Delta\bar{q}_1) \,\mathrm{d}\Gamma \\ &+ \mathrm{CPV}\int_{\Gamma_1^*} (\bar{u}_1 + \Delta\bar{u}_1)q^* \,\mathrm{d}\Gamma + \frac{1}{2}\left(\bar{u}_2^{(i)} + \Delta\bar{u}_2^{(i)}\right) \\ &- \int_{\Gamma_2^*} u^*(\bar{q}_2 + \Delta\bar{q}_2) \,\mathrm{d}\Gamma + \mathrm{CPV}\int_{\Gamma_2^*} (\bar{u}_2 + \Delta\bar{u}_2)q^* \,\mathrm{d}\Gamma. \end{aligned} \tag{A14}$$

Since point (i) is on the interface Γ ($\Gamma \equiv \Gamma_1^* \equiv \Gamma_2^*$) which is assumed flat from the beginning of this discussion, the integral CPV $\int_\Gamma uq^* \,\mathrm{d}\Gamma$ vanishes by virtue of $q^* = \partial u^*/\partial n = -(1/2\pi r)(\partial r/\partial n) = 0$.

Eq. (A14) reduces to

$$\begin{aligned} u^{(i)} = &\frac{1}{2}\left(\bar{u}_1^{(i)} + \Delta\bar{u}_1^{(i)}\right) - \int_{\Gamma_1^*} u^*(\bar{q}_1 + \Delta\bar{q}_1) \,\mathrm{d}\Gamma \\ &+ \frac{1}{2}\left(\bar{u}_2^{(i)} + \Delta\bar{u}_2^{(i)}\right) - \int_{\Gamma_2^*} u^*(\bar{q}_2 + \Delta\bar{q}_2) \,\mathrm{d}\Gamma \end{aligned} \tag{A15}$$

or,

$$u^{(i)} = \frac{1}{2}\bar{u}_1^{(i)} + \frac{1}{2}\bar{u}_2^{(i)} - \int_{\Gamma_1^*} u^*(\bar{q}_1 + \bar{q}_2)\,\mathrm{d}\Gamma + \frac{1}{2}\Delta\bar{u}_1^{(i)} + \frac{1}{2}\Delta\bar{u}_2^{(i)} - \int_{\Gamma_1^*} u^*(\Delta\bar{q}_1 + \Delta\bar{q}_2)\,\mathrm{d}\Gamma. \tag{A16}$$

The last three terms in Eq. (A16) can be regarded as the error denoted by Δe_u at the iteration $(k-1)$. Assuming a priori that the error will decrease as the iteration goes on, the updated estimate of the interface boundary condition for u at point (i), for the current iteration k, is calculated using Eq. (A16) as

$$u^{(i)}_{\text{update}} = u^{(i)} - \Delta e_u = \frac{1}{2}\bar{u}_1^{(i)} + \frac{1}{2}\bar{u}_2^{(i)} - \int_{\Gamma_1^*} \frac{1}{2\pi}\ln\left(\frac{1}{r}\right)(\bar{q}_1 + \bar{q}_2)\,\mathrm{d}\Gamma. \tag{A17}$$

When the point (i) is at a corner of the interface, the coefficients of the free term in Eqs. (A1), (A3) and (A5) are 1/2, 1/4 and 1/4, respectively and therefore the counterpart of Eq. (A17) in this case is

$$\frac{1}{2}u^{(i)}_{\text{update}} = \frac{1}{4}\bar{u}_1^{(i)} + \frac{1}{4}\bar{u}_2^{(i)} - \int_{\Gamma_1^*} \frac{1}{2\pi}\ln\left(\frac{1}{r}\right)(\bar{q}_1 + \bar{q}_2)\,\mathrm{d}\Gamma. \tag{A18}$$

The criterion for convergence is discussed in Appendix A.3 after the establishment of the mechanism for updating q in Appendix A.2.

A.2. Estimate of the interface boundary condition for q

- Point (i) on the interface Γ can be considered as an internal point relative to the domain Ω as defined above and therefore the derivative of u with respect to x_1 (Fig. 13) is calculated by

$$\frac{\partial u^{(i)}}{\partial x_1} = \int_{\partial\Omega} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \int_{\partial\Omega} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega \tag{A19}$$

or,

$$\frac{\partial u^{(i)}}{\partial x_1} = \int_{\Gamma_1} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \int_{\Gamma_1} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_1} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega + \int_{\Gamma_2} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \int_{\Gamma_2} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_2} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega. \tag{A20}$$

- Point (i) on the interface Γ can also be considered as a boundary point relative to the domain Ω_1. The flux at the point (i) in the direction x_1 (Fig. 13) is calculated by [27]

$$a_1^{\Omega_1} u^{(i)} + c_{1j}^{\Omega_1}\frac{\partial u^{(i)}}{\partial x_j} = \mathrm{CPV}\int_{\partial\Omega_1} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \mathrm{HPV}\int_{\partial\Omega_1} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_1} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega, \tag{A21}$$

where CPV and HPV are the Cauchy and the Hadamard principal values, respectively; $a_1^{\Omega_1}$ and $c_{1j}^{\Omega_1}$ are the free-term coefficients whose values depend on the local geometry of the boundary at the point (i). For smooth boundaries with continuous curvature, which is the case here, $a_1^{\Omega_1}$ and $c_{1j}^{\Omega_1}$ simply become 0 and $0.5\delta_{1j}$, respectively where δ_{hk} is the Kronecker delta (0 for $h \neq k$ and 1 for $h = k$) and therefore Eq. (A21) reduces to

$$\frac{1}{2}\delta_{1j}\frac{\partial u^{(i)}}{\partial x_j} = \mathrm{CPV}\int_{\partial\Omega_1} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \mathrm{HPV}\int_{\partial\Omega_1} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_1} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega \tag{A22}$$

or,

$$\frac{1}{2}\frac{\partial u^{(i)}}{\partial x_1} = \int_{\Gamma_1} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \int_{\Gamma_1} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_1} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega + \mathrm{CPV}\int_{\Gamma_1^*} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \mathrm{HPV}\int_{\Gamma_1^*} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma. \tag{A23}$$

This result can also be seen in Chen and Zhou [26].

- Point (i) on the interface Γ can also be considered as a boundary point relative to the subdomain Ω_2. Similarly, the flux at the point (i) in the direction x_1 (Fig. 13) is calculated by

$$c_{1_j}^{\Omega_2}\frac{\partial u^{(i)}}{\partial x_j} = \frac{1}{2}\delta_{1j}\frac{\partial u^{(i)}}{\partial x_j} = \mathrm{CPV}\int_{\partial\Omega_2} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \mathrm{HPV}\int_{\partial\Omega_2} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_1} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega \tag{A24}$$

or,

$$\frac{1}{2}\frac{\partial u^{(i)}}{\partial x_1} = \int_{\Gamma_2} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \int_{\Gamma_2} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_2} b\frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega + \mathrm{CPV}\int_{\Gamma_2^*} \frac{\partial u^*}{\partial x_1} q\,\mathrm{d}\Gamma - \mathrm{HPV}\int_{\Gamma_2^*} u\frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma. \tag{A25}$$

The substitution of Eqs. (A7) and (A8) into Eq. (A20)

yields

$$\frac{\partial u^{(i)}}{\partial x_1} = \int_{\Gamma_1} \frac{\partial u^*}{\partial x_1}(\bar{q} + \Delta\bar{q})\,\mathrm{d}\Gamma - \int_{\Gamma_1} (\bar{u} + \Delta\bar{u}) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_1} b \frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega + \int_{\Gamma_2} \frac{\partial u^*}{\partial x_1}(\bar{q} + \Delta\bar{q})\,\mathrm{d}\Gamma - \int_{\Gamma_2} (\bar{u} + \Delta\bar{u}) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_2} b \frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega. \quad \text{(A26)}$$

Let $\mathbf{n}_1$ and $\mathbf{n}_2$ denote the unit outward normal vectors at the point (i) corresponding to the subdomain Ω_1 and Ω_2, respectively. The substitution of Eqs. (A7)–(A10) into Eqs. (A23) and (A25), with $\mathbf{x}_1 \equiv \mathbf{n}_1 \equiv -\mathbf{n}_2$, yields, respectively,

$$\frac{1}{2}\left(\bar{q}_1^{(i)} + \Delta\bar{q}_1^{(i)}\right) = \int_{\Gamma_1} \frac{\partial u^*}{\partial x_1}(\bar{q} + \Delta\bar{q})\,\mathrm{d}\Gamma - \int_{\Gamma_1} (\bar{u} + \Delta\bar{u}) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_1} b \frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega + \mathrm{CPV}\int_{\Gamma_1^*} \frac{\partial u^*}{\partial x_1}(\bar{q}_1 + \Delta\bar{q}_1)\,\mathrm{d}\Gamma - \mathrm{HPV}\int_{\Gamma_1^*} (\bar{u}_1 + \Delta\bar{u}_1) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma \quad \text{(A27)}$$

and

$$-\frac{1}{2}\left(\bar{q}_2^{(i)} + \Delta\bar{q}_2^{(i)}\right) = \int_{\Gamma_2} \frac{\partial u^*}{\partial x_1}(\bar{q} + \Delta\bar{q})\,\mathrm{d}\Gamma - \int_{\Gamma_2} (\bar{u} + \Delta\bar{u}) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \int_{\Omega_2} b \frac{\partial u^*}{\partial x_1}\mathrm{d}\Omega + \mathrm{CPV}\int_{\Gamma_2^*} \frac{\partial u^*}{\partial x_1}(\bar{q}_2 + \Delta\bar{q}_2)\,\mathrm{d}\Gamma - \mathrm{HPV}\int_{\Gamma_2^*} (\bar{u}_2 + \Delta\bar{u}_2) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma. \quad \text{(A28)}$$

The subtraction of Eqs. (A27) and (A28) from Eq. (A26) yields

$$\frac{\partial u^{(i)}}{\partial x_1} = \frac{1}{2}\left(\bar{q}_1^{(i)} + \Delta\bar{q}_1^{(i)}\right) - \mathrm{CPV}\int_{\Gamma_1^*} \frac{\partial u^*}{\partial x_1}(\bar{q}_1 + \Delta\bar{q}_1)\,\mathrm{d}\Gamma + \mathrm{HPV}\int_{\Gamma_1^*} (\bar{u}_1 + \Delta\bar{u}_1) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma - \frac{1}{2}\left(\bar{q}_2^{(i)} + \Delta\bar{q}_2^{(i)}\right) - \mathrm{CPV}\int_{\Gamma_2^*} \frac{\partial u^*}{\partial x_1}(\bar{q}_2 + \Delta\bar{q}_2)\,\mathrm{d}\Gamma + \mathrm{HPV}\int_{\Gamma_2^*} (\bar{u}_2 + \Delta\bar{u}_2) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma \quad \text{(A29)}$$

or,

$$\frac{\partial u^{(i)}}{\partial x_1} = \frac{1}{2}\left(\bar{q}_1^{(i)} - \bar{q}_2^{(i)}\right) - \mathrm{CPV}\int_{\Gamma_1^*} \frac{\partial u^*}{\partial x_1}(\bar{q}_1 - \bar{q}_2)\,\mathrm{d}\Gamma + \mathrm{HPV}\int_{\Gamma_1^*} (\bar{u}_1 - \bar{u}_2) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma + \frac{1}{2}\left(\Delta\bar{q}_1^{(i)} - \Delta\bar{q}_2^{(i)}\right) - \mathrm{CPV}\int_{\Gamma_1^*} \frac{\partial u^*}{\partial x_1}(\Delta\bar{q}_1 - \Delta\bar{q}_2)\,\mathrm{d}\Gamma + \mathrm{HPV}\int_{\Gamma_1^*} (\Delta\bar{u}_1 - \Delta\bar{u}_2) \frac{\partial q^*}{\partial x_1}\mathrm{d}\Gamma. \quad \text{(A30)}$$

Owing to the fact that the interface Γ_1^* is flat, the kernels $\partial u^*/\partial x_1$ and $\partial q^*/\partial x_1$ in Eq. (A30) reduce to 0 and $1/2\pi r^2$, respectively and Eq. (A30) becomes

$$\frac{\partial u^{(i)}}{\partial x_1} = \frac{1}{2}\left(\bar{q}_1^{(i)} - \bar{q}_2^{(i)}\right) + \mathrm{HPV}\int_{\Gamma_1^*} (\bar{u}_1 - \bar{u}_2) \frac{1}{2\pi r^2}\mathrm{d}\Gamma + \frac{1}{2}\left(\Delta\bar{q}_1^{(i)} - \Delta\bar{q}_2^{(i)}\right) + \mathrm{HPV}\int_{\Gamma_1^*} (\Delta\bar{u}_1 - \Delta\bar{u}_2) \frac{1}{2\pi r^2}\mathrm{d}\Gamma. \quad \text{(A31)}$$

The second and third lines in Eq. (A31) can be regarded as the error denoted by Δe_q at the iteration $(k-1)$. Assuming a priori that the error will decrease as the iteration goes on, the updated estimate of the interface boundary condition q for subdomain Ω_1 at point (i), for the current iteration k, is calculated using Eq. (A31) as

$$q_{\mathrm{update}}^{(i)} = \frac{\partial u^{(i)}}{\partial x_1} - \Delta e_q = \frac{1}{2}\left(\bar{q}_1^{(i)} - \bar{q}_2^{(i)}\right) + \mathrm{HPV}\int_{\Gamma_1^*} (\bar{u}_1 - \bar{u}_2) \frac{1}{2\pi r^2}\mathrm{d}\Gamma. \quad \text{(A32)}$$

When the point (i) is at the corner of the interface, the counterpart of Eq. (A32) becomes

$$c_{1j}^{\Omega}\frac{\partial u^{(i)}}{\partial x_j} = c_{1j}^{\Omega}\frac{\partial u^{(i)}}{\partial x_j} - \Delta e_q = c_{1j}^{\Omega_1}\frac{\partial \bar{u}_1^{(i)}}{\partial x_j} + c_{1j}^{\Omega_2}\frac{\partial \bar{u}_2^{(i)}}{\partial x_j} + \mathrm{HPV}\int_{\Gamma_1^*}(\bar{u}_1 - \bar{u}_2)\frac{1}{2\pi r^2}\mathrm{d}\Gamma, \tag{A33}$$

where $c_{1j}^{\Omega} = 0.5\delta_{1j}$, $c_{11}^{\Omega_1} = c_{11}^{\Omega_2} = 1/4$ and $|c_{12}^{\Omega_1}| = |c_{12}^{\Omega_2}| = 1/2\pi$ [27]. Therefore, the estimate of the interface boundary condition for q at a corner point is given by

$$\frac{1}{2}q_{\mathrm{update}}^{(i)} = \frac{1}{4}\left(\bar{q}_1^{(i)} - \bar{q}_2^{(i)}\right) + \frac{1}{2\pi}\left(\frac{\partial \bar{u}_1^{(i)}}{\partial \hat{n}} - \frac{\partial \bar{u}_2^{(i)}}{\partial \hat{n}}\right) + \mathrm{HPV}\int_{\Gamma_1^*}(\bar{u}_1 - \bar{u}_2)\frac{1}{2\pi r^2}\mathrm{d}\Gamma, \tag{A34}$$

where $\hat{\mathbf{n}}$ is the unit outward normal vector at the corner point (i) as defined in Fig. 13.

A.3. Criterion for convergence

The iterative procedure converges when the solutions obtained from all subdomain problems match at the interfaces, i.e.

- $|\bar{u}_1^{(i)} - \bar{u}_2^{(i)}| < \mathrm{tol}$; and
- $|\bar{q}_1^{(i)} + \bar{q}_2^{(i)}| < \mathrm{tol}$;

where tol is a preset tolerance. When the above criterion is met, the solution obtained is continuous everywhere and the governing equation is satisfied everywhere with the given actual boundary conditions. Thus the solution obtained is the required solution. When this happens, the solution, within the specified tolerance, is

$$u^{(i)} \approx \bar{u}_1^{(i)} \approx \bar{u}_2^{(i)}, \tag{A35}$$

$$q^{(i)} \approx \bar{q}_1^{(i)} \approx \bar{q}_2^{(i)}, \tag{A36}$$

$$\Delta\bar{u}_1^{(i)}, \Delta\bar{u}_2^{(i)}, \Delta\bar{q}_1^{(i)}, \Delta\bar{q}_2^{(i)} \to 0. \tag{A37}$$

It can be seen that on convergence, Eqs. (A17), (A32), (A18) and (A34) are self-consistent.

References

[1] Haykin S. Neural networks: a comprehensive foundation. New Jersey: Prentice-Hall, 1999.

[2] Dissanayake MWMG, Phan-Thien N. Neural-network-based approximation for solving partial differential equations. Commun Numerical Meth Engng 1994;10:195–201.

[3] Takeuchi J, Kosugi Y. Neural network representation of finite element method. Neural Networks 1994;7(2):389–95.

[4] Kansa EJ. Multiquadrics — a scattered data approximation scheme with applications to computational fluid-dynamics-I. Surface approximations and partial derivative estimates. Computers Math Applic 1990;19(8/9):127–45.

[5] Kansa EJ. Multiquadrics — a scattered data approximation scheme with applications to computational fluid-dynamics-II. Solutions to parabolic, hyperbolic and elliptic partial differential equations. Computers Math Applic 1990;19(8/9):147–61.

[6] Dubal MR. Domain decomposition and local refinement for multiquadric approximations I: second-order equations in one-dimension. J Appl Sci Computation 1994;1(1):146–71.

[7] Sharan M, Kansa EJ, Gupta S. Application of the multiquadric method for numerical solution of elliptic partial differential equations. J Appl Sci Computation 1997;84:275–302.

[8] Zerroukat M, Power H, Chen CS. A numerical method for heat transfer problems using collocation and radial basis functions. Int J Numerical Meth Engng 1998;42:1263–78.

[9] Mai-Duy N, Tran-Cong T. Approximation of function and its derivatives using radial basis function networks. Submitted for publication.

[10] Mai-Duy N, Tran-Cong T. Numerical solution of differential equations using multiquadric radial basis function networks. Neural Networks 2001;14:185–99.

[11] Mai-Duy N, Tran-Cong T. Numerical solution of Navier–Stokes equations using multiquadric radial basis function networks. Int J Numerical Meth Fluids 2001;37:65–86.

[12] Funaro D, Quarteroni A, Zanolli P. An iterative procedure with interface relaxation for domain decomposition methods. SIAM J Numerical Anal 1988;25(6):1213–36.

[13] Marini LD, Quarteroni A. An iterative procedure for domain decomposition methods: a finite element approach. In: Glowinski R, Golub GH, Meurant GA, Periaux J, editors. Proceedings of First International Symposium on Domain Decomposition Methods for Partial Differential Equations, Philadelphia: SIAM, 1988. p. 129–43.

[14] Lions PL. On the Schwarz alternating method III: a variant for nonoverlapping subdomains. In: Chan TF, Glowinski R, Periaux J, Widlund OB, editors. Proceedings of Third International Symposium on Domain Decomposition Methods for Partial Differential Equations, Philadelphia: SIAM, 1990. p. 202–23.

[15] Yang D. A parallel iterative nonoverlapping domain decomposition procedure for elliptic problems. SIAM J Numerical Anal 1996;16:75–91.

[16] Mai-Duy N, Nguyen-Hong P, Tran-Cong T. A fast convergent iterative boundary element method on PVM cluster. Engng Anal Boundary Elem 1998;22:307–18.

[17] Powell MJD. Radial basis function approximations to polynomial. In: Griffiths DF, Watson GA, editors. Numerical Analysis 1987 Proceedings, Dundee: University of Dundee, 1987. p. 223–41.

[18] Chakravarthy SV, Ghosh J. Function emulation using radial basis function networks. Neural Networks 1997;10:459–78.

[19] Brebbia CA, Telles JCF, Wrobel LC. Boundary element techniques: theory and applications in engineering. Berlin: Springer, 1984.

[20] Banerjee PK, Butterfield R. Boundary element methods in engineering sciences. London: McGraw-Hill, 1981.

[21] Partridge PW, Brebbia CA, Wrobel LC. The dual reciprocity boundary element method. Southampton: Computational Mechanics Publications, 1992.

[22] Nowak AJ, Neves AC. The multiple reciprocity boundary element method. Southampton: Computational Mechanics Publications, 1994.

[23] Kutt HR. On the numerical evaluation of finite part integrals involving an algebraic singularity (report WISK 179). Pretoria: National Research Institute for Mathematical Sciences, 1975.

[24] Kutt HR. Quadrature formulae for finite part integrals (report WISK 178). Pretoria: The National Research Institute for Mathematical Sciences, 1975.

[25] Chen JT, Hong HK, Chyuan SW. Boundary element analysis and

design in seepage problems using dual integral formulation. Finite Elem Anal Des 1994;17:1–20.

[26] Chen G, Zhou J. Boundary element method. London: Academic Press, 1992.

[27] Guiggiani M. Formulation and numerical treatment of boundary integral equations with hypersingular kernels. In: Sladek V, Sladek J, editors. Singular integrals in boundary element methods. Southampton: Computational Mechanics Publications, 1998. p. 85–124.