Copyright © 2014 Tech Science Press CMES, vol.100, no.5, pp.363-397, 2014

Parallel Control-volume Method Based on Compact Local Integrated RBFs for the Solution of Fluid Flow Problems

N. Pham-Sy[1], C.-D. Tran[1] N. Mai-Duy[1] and T. Tran-Cong[1]

Abstract: In this paper, a high performance computing method based on the Integrated Radial Basis Function (IRBF), Control Volume (CV) and Domain Decomposition technique for solving Partial Differential Equations is presented. The goal is to develop an efficient parallel algorithm based on the Compact Local IRBF method using the CV approach, especially for problems with non-rectangular domain. The results showed that the goal is achieved as the computational efficiency is quite significant. For the case of square lid driven cavity problem with Renoylds number 100, super-linear speed-up is also achieved. The parallel algorithm is implemented in the Matlab environment using Parallel Computing Toolbox based on Distributed Computing Engine.

Keywords: Integrated Radial Basis Functions, Compact Local Stencils, Control Volume Method, Domain Decomposition Method, Parallel Algorithm, Superlinear Speedup.

1 Introduction

Integrated Radial Basis Function (IRBF) method was proposed by Mai-Duy and Tran-Cong (2001) as an alternative to the RBF interpolation of scattered data by Kansa (1990), which is here referred to as the differential RBF (DRBF) method. Thanks to the integration approach, IRBF method is shown to have higher order of accuracy than the DRBF method. The convergence rate was then boosted by using local/compact local schemes to form tridiagonal matrix [Hoang-Trieu, Mai-Duy, and Tran-Cong (2012); Chandhini and Sanyasiraju (2007)].

The IRBF method is further developed in combination with control-volume (CV) method [Patankar (1980)] to improve the accuracy of the solution in non-rectangular domains [Mai-Duy and Tran-Cong (2010)]. With the inherent conservation of mass, momentum and energy over control volumes, this method has been shown to

[1] Computational Engineering and Science Research Centre, Faculty of Engineering and Surveying, The University of Southern Queensland, Toowoomba, QLD 4350, Australia.

be a very effective way to deal with domains with complex boundary. In this paper, the CV method is employed in combination with two-dimensional (2D) IRBF to simulate fluid flow in the triangular cavity problem [Kohno and Bathe (2006)].

Since the scale of practical engineering problems is huge in terms of degrees of freedom, modern computational mechanics has begun to embrace parallel paradigms. With the help of parallelisation, the shortage of memory and computational power is being addressed. What remains challenging is parallel algorithms, which is the main focus of this paper. The method being considered is a domain decomposition (DD) method, which is one of the most popular methods for solving large scale problems [Quarteroni and Valli (1999)]. The DD method is used to split the computational domain into smaller sub-domains which are solved separately. Originally, sub-domains are solved sequentially one after another. In order to improve the throughput of the simulation, in this work, the sub-domains are solved in parallel. The DD method being used is the Schwarz additive overlapping DD method and the communication between parallel sub-domains are handled by Matlab-supported Message Passing Interface (MPI).

The paper is organised as follows. In Sections 2 and 3, a brief review of Compact Local IRBF (CLIRBF) and CV method is presented. The DD method is described in Section 4. Numerical results are then given and discussed in Section 5 with a conclusion in Section 6.

2 Local methods based on Integrated Radial Basis Function

In this section, a brief review of several IRBF local approaches, including one-dimension (1D) IRBF and two-dimension (2D) local 9-point IRBF stencils, is provided.

2.1 1D-IRBF method

Consider Poisson Partial Differential Equation (PDE) of a simple 2D problem as follows.

$$\nabla^2 u(\mathbf{x}) = f(\mathbf{x}), \mathbf{x} \in \Omega \tag{1}$$

where u is the field variable; $\mathbf{x}$ the position vector; Ω the considered domain and f a known function of $\mathbf{x}$.

By means of IRBF, the highest order derivatives of the PDE, second order in this case, are approximated by a weighted set of RBFs as

$$\frac{\partial^2 u(\mathbf{x})}{\partial x_j^2} = \sum_{i=1}^{N} w_i g_i = \sum_{i=1}^{N} w_i G_i(\mathbf{x}), \tag{2}$$

where x_j is the j-component of $\mathbf{x}$ ($j=1,2$); $\{w_i\}_{i=1}^N$ the set of weights and $\{g_i(\mathbf{x})\}_{i=1}^N$ the set of RBFs. The multiquadric (MQ) RBF is used in this work and given by

$$G_i(\mathbf{x}) = \sqrt{(\mathbf{x}-\mathbf{c}_i)^2+a_i^2},$$

where $\{\mathbf{c}_i\}_{i=1}^N$ is a set of centres and $\{a_i\}_{i=1}^N$ a set of MQ-RBF widths.
To obtain first-order derivatives and field variable, Eq. (2) is integrated successively with respect to x_j as follows.

$$\frac{\partial u}{\partial x_j} = \sum_{i=1}^N w_i G_i^{[1]}(\mathbf{x}) + C_1, \tag{3}$$

$$u = \sum_{i=1}^N w_i G_i^{[0]}(\mathbf{x}) + C_1 x_j + C_2, \tag{4}$$

where $G_i^{[1]}(\mathbf{x}) = \int G_i(\mathbf{x})dx_j$, $G_i^{[0]}(\mathbf{x}) = \int G_i^{[1]}(\mathbf{x})dx_j$ and C_1 and C_2 are constants of integration in the sense that $C_i = C_i(x_k)$, $k \neq i$.

Collocating equations (2) - (4) at a set of grid points $\{\mathbf{x}_i\}_{i=1}^N$ yields

$$\frac{\partial^2 \widetilde{\mathbf{u}}}{\partial x_j} = \widetilde{G}^{[2]}\widetilde{\mathbf{w}}, \tag{5}$$

$$\frac{\partial \widetilde{\mathbf{u}}}{\partial x_j} = \widetilde{G}^{[1]}\widetilde{\mathbf{w}}, \tag{6}$$

$$\widetilde{\mathbf{u}} = \widetilde{G}^{[0]}\widetilde{\mathbf{w}}, \tag{7}$$

with

$$\widetilde{\mathbf{w}} = (w_1, w_2, \cdots, w_N, C_1, C_2)^T,$$

$$\widetilde{\mathbf{u}} = (u_1, u_2, \cdots, u_N)^T,$$

$$\frac{\partial^k \widetilde{\mathbf{u}}}{\partial x_j^k} = \left(\frac{\partial^k u_1}{\partial x_j^k}, \frac{\partial^k u_2}{\partial x_j^k}, \cdots, \frac{\partial^k u_N}{\partial x_j^k}\right)^T,$$

where $u_i = u(\mathbf{x_i})$ ($i=1,2,\cdots,N$); $\widetilde{\mathbf{G}}^{[2]}, \widetilde{\mathbf{G}}^{[1]}$ and $\widetilde{\mathbf{G}}^{[0]}$ are known matrices of size $N \times (N+2)$ as presented below.

$$\widetilde{\mathbf{G}}^{[k]}(k=1,2,3)=\begin{bmatrix} G_1^{[k]}(\mathbf{x}_1) & G_2^{[k]}(\mathbf{x}_1) & \cdots & G_N^{[k]}(\mathbf{x}_1) & a_1^{[k]} & b_1^{[k]} \\ G_1^{[k]}(\mathbf{x}_2) & G_2^{[k]}(\mathbf{x}_2) & \cdots & G_N^{[2]}(\mathbf{x}_2) & a_2^{[k]} & b_2^{[k]} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ G_1^{[k]}(\mathbf{x}_N) & G_2^{[k]}(\mathbf{x}_N) & \cdots & G_N^{[k]}(\mathbf{x}_N) & a_N^{[k]} & b_N^{[k]} \end{bmatrix},$$

where

$$\left([a]^{[\mathrm{k}]}\right)^{\mathrm{T}}=\begin{cases} \begin{pmatrix} 0 & \cdots & 0 \end{pmatrix}^{\mathrm{T}}, \mathrm{k}=2 \\ \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}^{\mathrm{T}}, \mathrm{k}=1 \\ \begin{pmatrix} x_1 & \cdots & x_9 \end{pmatrix}^{\mathrm{T}}, \mathrm{k}=0 \end{cases}$$

with $(x_m, y_m)^T = \mathbf{x}_m$ and

$$\left([b]^{[\mathrm{k}]}\right)^{\mathrm{T}}=\begin{cases} \begin{pmatrix} 0 & \cdots & 0 \end{pmatrix}^{\mathrm{T}}, \mathrm{k}=1,2 \\ \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}^{\mathrm{T}}, \mathrm{k}=0 \end{cases}$$

2.2 2D IRBF local stencil scheme

In this work, a 9-point stencil scheme is employed to overcome the problem of ill-conditioned system matrix, which is an inherent problem in the global approach. According to this scheme, a local 9-point stencil for an arbitrary grid-point $\mathbf{x}_{i,j}$ $(2 \le i \le n_x - 1; 2 \le j \le n_y - 1)$ is described as follows (Fig. 1).

$$\begin{bmatrix} \mathbf{x}_{i-1,j+1} & \mathbf{x}_{i,j+1} & \mathbf{x}_{i+1,j+1} \\ \mathbf{x}_{i-1,j} & \mathbf{x}_{i,j} & \mathbf{x}_{i+1,j} \\ \mathbf{x}_{i-1,j-1} & \mathbf{x}_{i,j-1} & \mathbf{x}_{i+1,j+1} \end{bmatrix},$$

where $n_x \times n_y$ is a Cartesian grid density of the considered domain. More details can be found in Hoang-Trieu, Mai-Duy, and Tran-Cong (2012). This approximation, coupled with the control volume method, will be presented in the next sections.

3 A control volume method based on 2D IRBFs

In this approach, each grid point is surrounded by a CV and the conservative governing equations are integrated within this volume. Figure 2 shows the CV formation for a regular 2-D domain. In this figure, CVs are bounded by lines parallel to grid lines through the middle points between the reference point and its neighbours.

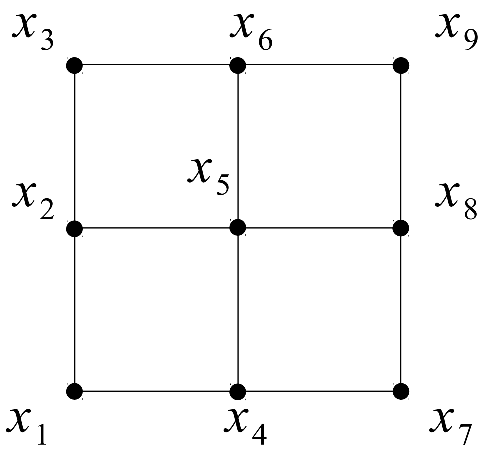


Figure 1: A 9-point local stencil.

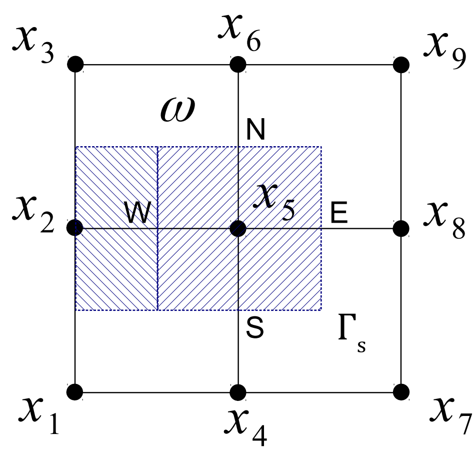


Figure 2: CV formation in 2D.

Consider the 2-D Poisson equation (1). This equation is integrated over a CV, and then by applying the divergence theorem to the resultant equation one gets

$$\oint_{\Gamma_s} \nabla u(\mathbf{x}) \cdot \hat{n} d\Gamma_s = \int_{y_b}^{y_t} \frac{\partial u(\mathbf{x})}{\partial x}\bigg|_r dy - \int_{y_b}^{y_t} \frac{\partial u(\mathbf{x})}{\partial x}\bigg|_l dy + \int_{x_l}^{x_r} \frac{\partial u(\mathbf{x})}{\partial x}\bigg|_t dx - \int_{x_l}^{x_r} \frac{\partial u(\mathbf{x})}{\partial x}\bigg|_b dx$$
$$= \int_{\Omega_s} f(\mathbf{x}) d\Omega_s, \tag{8}$$

where Ω_s and Γ_s are the CV under consideration and its surface, respectively; $\hat{n}$ the outward normal unit vector and $(.)|_{(s)}$, $(s = l, r, t, b)$ depicts integrals over the left, right, top and bottom faces of the CV respectively.

Using 5-point Gaussian quadrature scheme to discretise Eq. (8) yields

$$\frac{\Delta y}{2}\sum_{j=1}^{5}\alpha_j \frac{\partial u(y(\eta_j))}{\partial x}\bigg|_r - \frac{\Delta y}{2}\sum_{j=1}^{5}\alpha_j \frac{\partial u(y(\eta_j))}{\partial x}\bigg|_l + \frac{\Delta x}{2}\sum_{i=1}^{5}\alpha_i \frac{\partial u(x(\eta_i))}{\partial y}\bigg|_t$$
$$-\frac{\Delta x}{2}\sum_{i=1}^{5}\alpha_i \frac{\partial u(x(\eta_i))}{\partial y}\bigg|_r = \frac{\Delta x \Delta y}{4}\sum_{i=1}^{5}\sum_{j=1}^{5}\alpha_i\alpha_j f(x(\eta_i)y(\eta_j)), \tag{9}$$

where α_k and η_k $(k = i, j)$ are the weights and Gauss points, respectively.

The 2D Local IRBF approximation scheme mentioned in Section 2.2 is used to approximate the field variable and its derivatives in Eq. (9). Thus, in this approach the governing equations are forced to be satisfied locally over CVs while the boundary conditions are directly imposed using the 1D-IRBF approximant. The procedure leads to an algebraic equation system for unknown nodal values of the field variable as follows.

$$\begin{pmatrix} \widetilde{\mathbf{u}} \\ \widetilde{\mathbf{0}} \end{pmatrix} = \underbrace{\begin{bmatrix} \mathscr{G}_x^{[0]}, & \mathscr{O} \\ \mathscr{G}_x^{[0]}, & -\mathscr{G}_y^{[0]} \end{bmatrix}}_{\mathscr{C}} \begin{pmatrix} \widetilde{\mathbf{w}}_x \\ \widetilde{\mathbf{w}}_y \end{pmatrix} = \mathscr{C} \begin{pmatrix} \widetilde{\mathbf{w}}_x \\ \widetilde{\mathbf{w}}_y \end{pmatrix}, \tag{10}$$

where $\mathscr{C}$ is the conversion matrix, $\widetilde{\mathbf{w}}_x$ and $\widetilde{\mathbf{w}}_y$ the RBF weight vectors of length 15; $\widetilde{\mathbf{u}}$ the vector of length 9, and $\widetilde{\mathbf{0}}$ the zeros vector of length 9; $\mathscr{O}$ the zeros matrix of dimension 9×15, and $\mathscr{G}_x^{[0]}$ and $\mathscr{G}_y^{[0]}$ the known matrices of dimensions 9×15. Furthermore, $\widetilde{\mathbf{u}}$, $\widetilde{\mathbf{w}}_x$ and $\widetilde{\mathbf{w}}_y$ are given by

$$\widetilde{\mathbf{u}} = (u_1, \ldots, u_9)^T, \tag{11}$$

$$\widetilde{\mathbf{w}}_x = (w_{x_1}, \ldots, w_{x_9}, C_1^x(y_1), C_1^x(y_2), C_1^x(y_3), C_2^x(y_1), C_2^x(y_2), C_2^x(y_3))^T, \tag{12}$$

$$\widetilde{\mathbf{w}}_y = (w_{y_1}, \ldots, w_{y_9}, C_1^y(x_1), C_1^y(x_2), C_1^y(x_3), C_2^y(x_1), C_2^y(x_2), C_2^y(x_3))^T, \tag{13}$$

$$\mathscr{G}_x^{[0]} = \begin{bmatrix} G_{1,x}^{[0]}(\mathbf{x}_1) & \cdots & G_{9,x}^{[0]}(\mathbf{x}_1) & x_1 & 0 & 0 & 1 & 0 & 0 \\ G_{1,x}^{[0]}(\mathbf{x}_2) & \cdots & G_{9,x}^{[0]}(\mathbf{x}_2) & 0 & x_2 & 0 & 0 & 1 & 0 \\ \vdots & \ddots & \vdots & 0 & 0 & x_3 & 0 & 0 & 1 \\ \vdots & \ddots & \vdots & x_4 & 0 & 0 & 1 & 0 & 0 \\ \vdots & \ddots & \vdots & 0 & x_5 & 0 & 0 & 1 & 0 \\ \vdots & \ddots & \vdots & 0 & 0 & x_6 & 0 & 0 & 1 \\ \vdots & \ddots & \vdots & x_7 & 0 & 0 & 1 & 0 & 0 \\ \vdots & \ddots & \vdots & 0 & x_8 & 0 & 0 & 1 & 0 \\ G_{1,x}^{[0]}(\mathbf{x}_9) & \cdots & G_{9,x}^{[0]}(\mathbf{x}_9) & 0 & 0 & x_9 & 0 & 0 & 1 \end{bmatrix}, \tag{14}$$

$$\mathscr{G}_y^{[0]} = \begin{bmatrix} G_{1,y}^{[0]}(\mathbf{x}_1) & \cdots & G_{9,y}^{[0]}(\mathbf{x}_1) & y_1 & 0 & 0 & 1 & 0 & 0 \\ G_{1,y}^{[0]}(\mathbf{x}_2) & \cdots & G_{9,y}^{[0]}(\mathbf{x}_2) & y_2 & 0 & 0 & 1 & 0 & 0 \\ \vdots & \ddots & \vdots & y_3 & 0 & 0 & 1 & 0 & 0 \\ \vdots & \ddots & \vdots & 0 & y_4 & 0 & 0 & 1 & 0 \\ \vdots & \ddots & \vdots & 0 & y_5 & 0 & 0 & 1 & 0 \\ \vdots & \ddots & \vdots & 0 & y_6 & 0 & 0 & 1 & 0 \\ \vdots & \ddots & \vdots & 0 & 0 & y_7 & 0 & 0 & 1 \\ \vdots & \ddots & \vdots & 0 & 0 & y_8 & 0 & 0 & 1 \\ G_{1,y}^{[0]}(\mathbf{x}_9) & \cdots & G_{9,y}^{[0]}(\mathbf{x}_9) & 0 & 0 & y_9 & 0 & 0 & 1 \end{bmatrix}, \tag{15}$$

where $G_{i,x}^{[0]}$ and $G_{i,y}^{[0]}$ ($i = 1..9$) were defined in Section 2.1 in the x and y-directions and x_i, y_i two components of $\mathbf{x}_i$. It is noted that in (10)

$$\widetilde{\mathbf{u}} = [\mathscr{G}_x^{[0]}, \mathscr{O}] \begin{pmatrix} \widetilde{\mathbf{w}}_x \\ \widetilde{\mathbf{w}}_y \end{pmatrix}$$

is obtained by collocating the field variable over a local stencil, and

$$\widetilde{\mathbf{0}} = [\mathscr{G}_x^{[0]}, -\mathscr{G}_y^{[0]}] \begin{pmatrix} \widetilde{\mathbf{w}}_x \\ \widetilde{\mathbf{w}}_y \end{pmatrix}$$

is derived from the consistency condition $\int\int \frac{\partial^2 \mathbf{u}}{\partial x^2} dx \Big|_{\mathbf{x}_i} = \int\int \frac{\partial^2 \mathbf{u}}{\partial y^2} dy \Big|_{\mathbf{x}_i}$.

The conversion of the network-weight space into the physical space is achieved by inverting Eq. (10)

$$\begin{pmatrix} \widetilde{\mathbf{w}}_x \\ \widetilde{\mathbf{w}}_y \end{pmatrix} = \mathscr{C}^{-1} \begin{pmatrix} \widetilde{\mathbf{u}} \\ \widetilde{\mathbf{0}} \end{pmatrix}. \tag{16}$$

By substituting Eq. (16) into Eqs. (3) and (4) the first order derivatives of u with respect to x and y and the function itself over a local stencil are determined.

In the case of non-rectangular domains, the CV formation for interior points is carried out in a similar way but with some extra treatments for non-rectangular boundaries, which will be detailed in Section 5.2.

4 Parallel domain decomposition method

Domain decomposition has been successfully used to overcome the resource limitation associated with large-scale problems. Its primary objective is to split a large problem domain into small ones called sub-domains in which the problem can be solved more effectively in terms of memory and computing power (Quarteroni and Valli, 1999; Tran, Phillips, and Tran-Cong, 2009). A notable advantage of DD method in solving numerical problems is that it helps to decrease the condition number of system matrices in sub-domains. As a result, DD method helps to achieve a more stable and accurate solution. Furthermore, with the advance of parallel computing, DD technique finds itself very parallel capable. That potential for parallelisation further encourages more intensive research in DD method in recent decades.

Over the last two decades, researchers have developed parallel algorithms owing to the simplicity of grid generation to significantly increase the throughput of numerical solutions. For example, Singh and Jain (2005) used an Element-Free Galerkin method with moving least-square approximant to solve fluid flow problems. They were able to achieve high efficiency, e.g. 91.27% for a 2D problem with 8 CPUs. Shirazaki and Yagawa (1999) proposed a Mesh-Free method based parallel algorithm to solve incompressible viscous flow. They obtained a stable solution to a model with three-million degrees of freedom. However, as the speed-up was separated into two parts, namely the construction of system equations and the time integration, the efficiency of the first part was very high and even super-linear with high number of CPUs, the efficiency of the second part was not able to scale to high number of CPUs. Indeed, the efficiency droped from approximately 98% with 16 CPUs to around 50% with 64 CPUs. Ingber, Chen, and Tanski (2004) combined the method of fundamental solutions and the particular solution method to solve the transient heat conduction problems. The approach was developed using a Schwarz Neumann-Neumann DD based parallel scheme. Although the authors successfully demonstrated the accuracy of parallel algorithm in comparison with the non-parallel version, unfortunately, information regarding the efficiency of parallelisation was not given.

This work is a further development of the Schwarz Additive Overlapping DD technique (Pham-Sy, Tran, Hoang-Trieu, Mai-Duy, and Tran-Cong, 2013) using the local stencil IRBF approximants. Here, the local stencil 2D-IRBF based CV method, which is presented in the previous sections, is used to develop the parallel algorithm for solving fluid flow problems.

The additive overlapping DD method is a rather simple but effective method. It also has a high potential for parallelisation as the computation in each subdomain is independent within a time step. In this approach, the original domain is divided into several overlapping sub-domains. The function values on artificial boundaries (ABs) are initially unknown and are set to zeros (initial guess). In each iterative step, the boundary value problem is solved separately in each sub-domain. Then the function value on the artificial boundary of one sub-domain is updated by the solution from other sub-domains. This procedure is repeated until a desired tolerance is achieved. Several specific details on the use of additive overlapping DD technique will be presented briefly in the next sections.

4.1 Sub-domain formation and neighbour identification

The sub-domain formation task is straightforward with a rectangular domain as can be seen in Fig. 3. This formation has been reported in our previous work. However, it is also presented here for completeness.

The Matlab's notation will be utilised as follows.

- *lab/worker* - a computing node in distributed system.
- *lab-index* - the lab's identification in distributed system. This lab-index is used for labs to communicate with each other.

For example, in Fig. 3 the original domain is decomposed into $N_x \times N_y = 4 \times 3 = 12$ sub-domains. These sub-domains' lab-indices are enumerated from 1 to 12 and from bottom to top and left to right. Each sub-domain also has a 2D index (i, j) that determines its position in the original domain. In this example, the 2D index of sub-domain 7 is $(3,2)$. In order to determine neighbours, an arbitrary lab (i, j) simply checks the following cases

1. if $(j-1) > 0$ its left neighbour is lab $(i, j-1)$;
2. if $(j+1) \leq N_y$ its right neighbour is lab $(i, j+1)$;

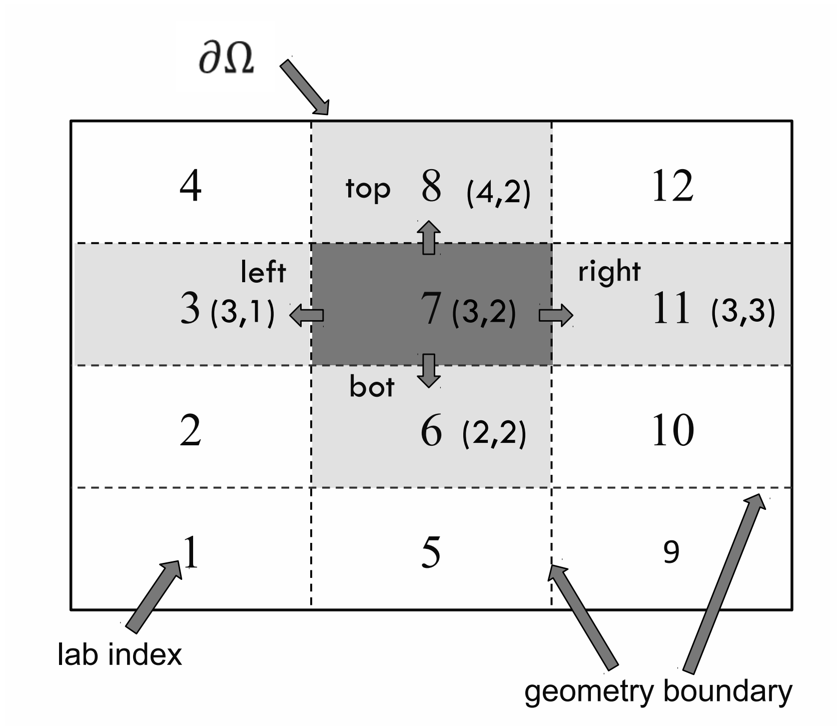


Figure 3: Enumeration in a system of $N_x \times N_y = 4 \times 3$ sub-domains of a rectangular domain.

3. if $(i+1) \leq N_x$ its top neighbour is lab $(i+1, j)$;

4. if $(i-1) > 0$ its bottom neighbour is lab $(i-1, j)$.

Unfortunately, with a non-rectangular domain (for example, Fig. 4) the above algorithm will not work because some sub-domains will lie outside the considered domain. For example, in Fig. 4 sub-domains 8 and 9 are outside of the triangular domain Ω. To overcome this situation, one first needs to create a list of sub-domains (LSD), along with their 2D indices (as usual, from bottom to top and left to right). Then inside each sub-domain the following conditions must be checked to determine a sub-domain's neighbours' lab-index. Consider a sub-domain whose 2D index is (i, j)

1. if lab $(i, j-1)$ exists in LSD then it is the left neighbour of lab (i, j);

2. if lab $(i, j+1)$ exists in LSD then it is the right neighbour of lab (i, j);

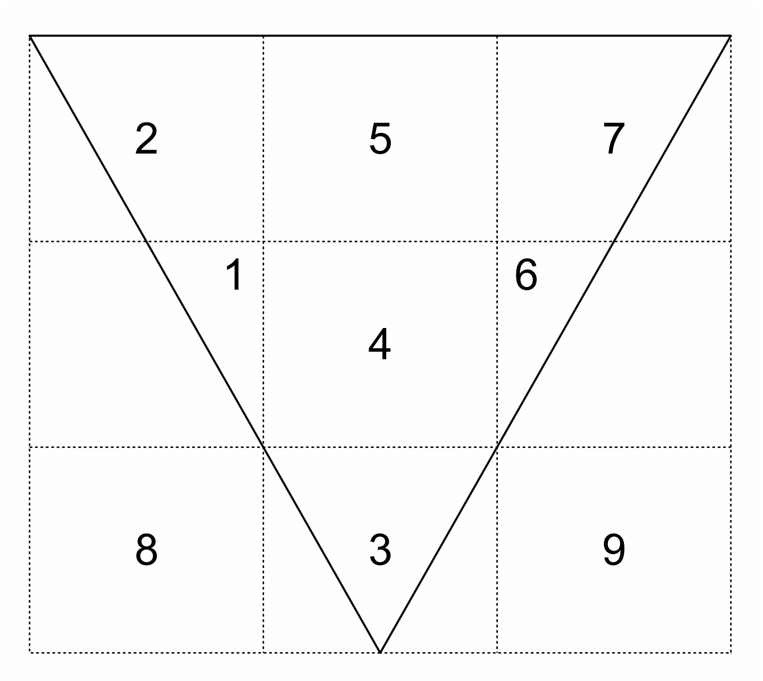


Figure 4: Enumeration in a system of 7 sub-domains of a triangular domain.

3. if lab $(i+1, j)$ exists in LSD then it is the top neighbour of lab (i, j);
4. if lab $(i-1, j)$ exists in LSD then it is the bottom neighbour of lab (i, j).

A detailed example of this process is provided in Table 1 with 7 sub-domains in a triangular domain presented in Fig. 4.

4.2 Communication and Synchronisation

In additive overlapping DD method, one of the critical tasks is the communication between the sub-domains as the function values on the artificial boundary of one sub-domain are obtained from the solution in its neighbouring sub-domains in the previous step. In the present implementation, Matlab built-in parallel communication method is utilised. Matlab communication functions allow to send an array of data to Matlab workers in a synchronized way, which means the sender must wait until the receiver fully receives a message. This mechanism itself guarantees the synchronisation between sub-domains and no extra care is needed to ensure that all sub-domains are always executing the same iterative step. More information about Matlab supported MPI implementation can be found in (MATLAB, 2012).

Table 1: Neighbour sub-domain (NB) determination for a triangular domain

labindex	2d-index	left NB	right NB	top NB	bottom NB
1	(2,1)	(2,0) $\sim$ nil	(2,2) $\sim$ 4	(3,1) $\sim$ 2	(1,1) $\sim$ nil
2	(3,1)	(3,0) $\sim$ nil	(3,2) $\sim$ 5	(4,1) $\sim$ nil	(2,1) $\sim$ 1
3	(1,2)	(1,1) $\sim$ nil	(1,3) $\sim$ nil	(2,2) $\sim$ 4	(0,2) $\sim$ nil
4	(2,2)	(2,1) $\sim$ 1	(2,3) $\sim$ 6	(3,2) $\sim$ 5	(1,2) $\sim$ 3
5	(3,2)	(3,1) $\sim$ 2	(3,3) $\sim$ 7	(4,2) $\sim$ nil	(2,2) $\sim$ 4
6	(2,3)	(2,2) $\sim$ 4	(2,4) $\sim$ nil	(3,3) $\sim$ 7	(1,3) $\sim$ nil
7	(3,3)	(3,2) $\sim$ 5	(3,4) $\sim$ nil	(4,3) $\sim$ nil	(2,3) $\sim$ 6

4.3 Termination

Since the parallel algorithm presented in this paper is a Distributed Computing Algorithm, it needs to have a termination detection process. This process has been investigated and classified into a unique class of algorithm called Distributed Termination Detection (DTD). In this paper, a bitmap DTD algorithm presented by Pham-Sy, Tran, Hoang-Trieu, Mai-Duy, and Tran-Cong (2013) is employed. This algorithm has several advantages such as symmetric detection, decentralized control and low termination detection delay, and thus ideally suits the implementation of parallel algorithm in the present work.

4.4 Algorithm of the present procedure

The present parallel method is based on the combination of the local stencil 2D-IRBF and CV, and the DD technique presented in the previous sections can now be described in an overall algorithm whose flowchart is shown in Fig. 5.

5 Numerical results

The proposed method is verified through the simulation of the lid driven cavity (LDC) fluid flow problem for two cases of rectangular and non-rectangular domains. The efficiency of the present method is analyzed.

The lid-driven cavity flow has been commonly used for verification of a numerical method owing to the availability of benchmark solutions in the literature. The problem has also been quite popular among meshless community, e.g. Lin and Atluri (2001) with meshless Local Petrov-Galerkin (MLPG) method; Shu, Ding, and Yeo (2005) with local RBF-based Differential Quadrature method; Chinchapatnam, Djidjeli, and Nair (2007) with RBF; and Kim, Kim, Jun, and Lee (2007) with Meshfree point collocation method. Therefore, in this paper the lid-driven

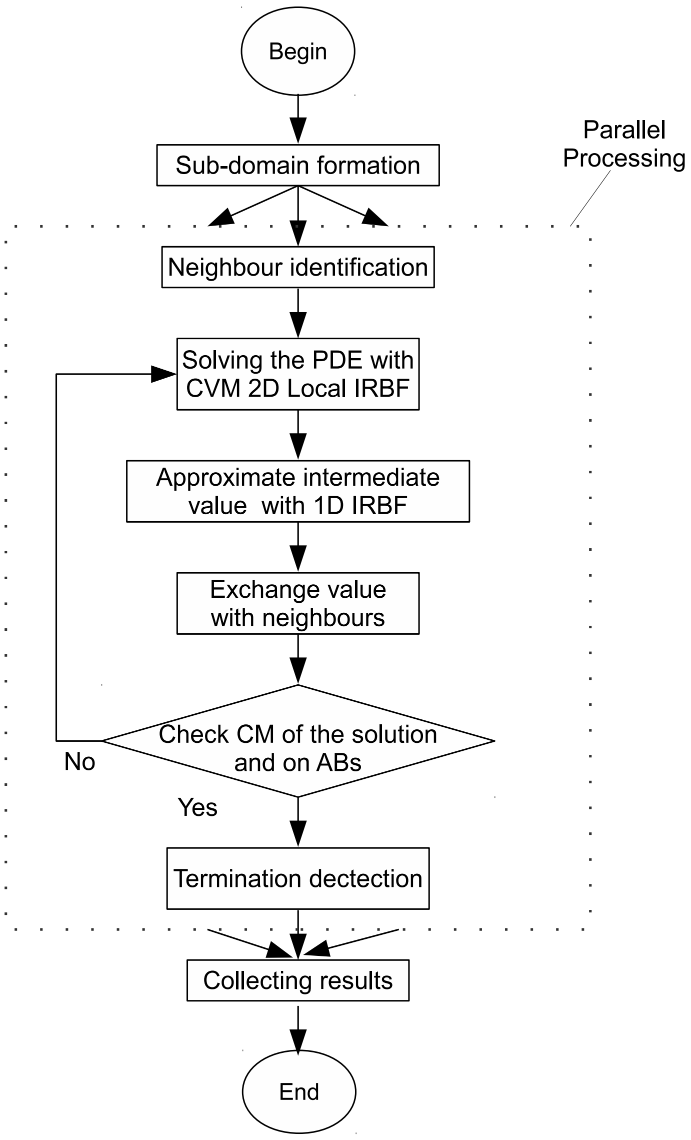

Begin
Sub-domain formation
Parallel Processing
Neighbour identification
Solving the PDE with CVM 2D Local IRBF
Approximate intermediate value with 1D IRBF
Exchange value with neighbours
Check CM of the solution and on ABs
No
Yes
Termination dectection
Collecting results
End

Figure 5: Algorithm of the Parallel domain decomposition method using the local IRBF based Control Volume approach.

cavity flow is also employed to investigate the accuracy as well as the efficiency of the present parallel scheme.

The problem is defined in the stream-function - vorticity formulation as follows.

$$\frac{\partial \omega}{\partial t}+(\frac{\partial \psi}{\partial y}\frac{\partial \omega}{\partial x}-\frac{\partial \psi}{\partial x}\frac{\partial \omega}{\partial y})=\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 \omega}{\partial x^2}+\frac{\partial^2 \omega}{\partial y^2}\right), \tag{17}$$

$$-\omega=\frac{\partial^2 \psi}{\partial x^2}+\frac{\partial^2 \psi}{\partial y^2}, \tag{18}$$

where Re is the Reynolds number, ψ the stream function; ω the vorticity and t the time. The $x-$ and $y-$ velocity components are given by $u=\dfrac{\partial \psi}{\partial y}$ and $v=-\dfrac{\partial \psi}{\partial x}$.

The problem is solved using the local 9-point stencil 2D-IRBF scheme as presented in Section 2.2 with the time derivative being discretised using a first-order Euler scheme and the diffusive terms being treated implicitly. The boundary condition for ω is computed through Eq. (18) using 1D global IRBF as described in Section 2.1.

The general procedure for solving a LDC problem is given as follows.

1. Guess the initial values of ω;
2. Solve (18) for ψ;
3. Approximate the values of ω on boundaries and the convective terms;
4. Solve (17) for ω;
5. Check convergence measure for ω.

5.1 Square cavity

For the square cavity problem, the geometry of the analysis domain with the chosen coordinate system is shown in Fig. 6. The boundary conditions are given in terms of the stream-function as.

$$\psi = 0, \quad \frac{\partial \psi}{\partial x}=0 \quad \forall (x,y)\in \Gamma_2 \cup \Gamma_3; \tag{19}$$

$$\psi = 0, \quad \frac{\partial \psi}{\partial y}=0 \quad \forall (x,y)\in \Gamma_4; \tag{20}$$

$$\psi = 0, \quad \frac{\partial \psi}{\partial y}=1 \quad \forall (x,y)\in \Gamma_1. \tag{21}$$

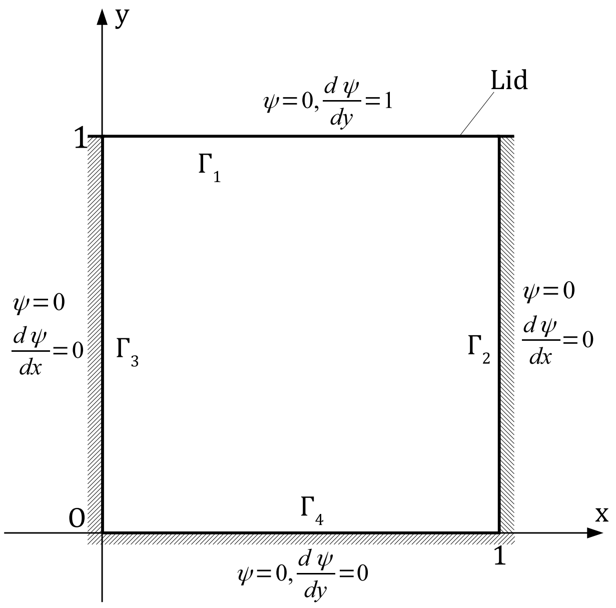


Figure 6: The square lid driven cavity fluid flow problem. Geometry and boundary conditions. No slip is assumed between the fluid and solid surfaces. The top lid is moving from left to right with a speed of 1.

The Dirichlet boundary condition on ψ is used to solve Eq. (18) in step 2, while the Neumann boundary condition on ψ is used to approximate the value of ω on boundaries. The values of ω on boundaries, in turn, are used as boundary conditions to solve Eq. (17) in step 4 above.

The iterative procedure for solving the square cavity problem with parallel DD method is as follows.

1. Divide the analysis domain into a number of sub-domains. Guess initial boundary condition on ABs;

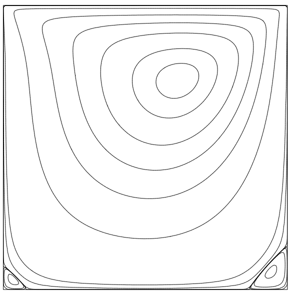
2. Solve the fluid flow problem in each and every sub-domain as described above;

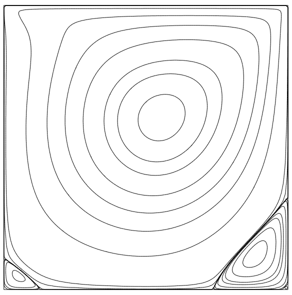
3. Exchange the values of ψ and ω at interfaces with neighbours;

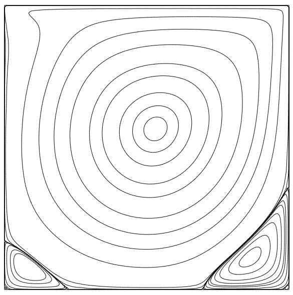
4. Calculate convergence measure on all interfaces;

5. Check for termination condition.

In this paper, the square cavity problem is simulated for a range of Reynolds numbers ($Re = 100, 400, 1000$ and 3200). Figure 7 depicts streamlines of the flow obtained by the present parallel method using a grid of 151×151 collocation points, $\Delta t = 1.E-03$, $DDTol = 1.E-06$, $CMTol = 1.E-06$ and $\beta = 2$, 4 sub-domains associated with 4 CPUs for $Re = 100, 400, 1000$ (Figs. 7(a) - 7(c)) and 2 sub-domains for $Re = 3200$ (Fig.7(d)). The results are in very good agreement with those presented in Ghia, Ghia, and Shin (1982) as well as in Botella and Peyret (1998). Similar results can be found for vorticity contours in Fig. 8. Furthermore, Fig. 9 provides the profiles of the velocities along the vertical and horizontal centrelines by the present method along with the benchmark values from Ghia, Ghia, and Shin (1982). As can be seen, the results match up very well with the benchmark solution.


(a) Re=100


(b) Re=400


(c) Re=1000

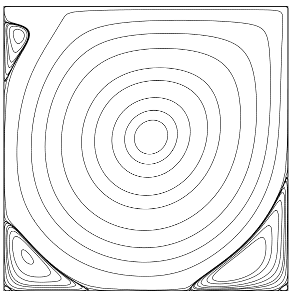
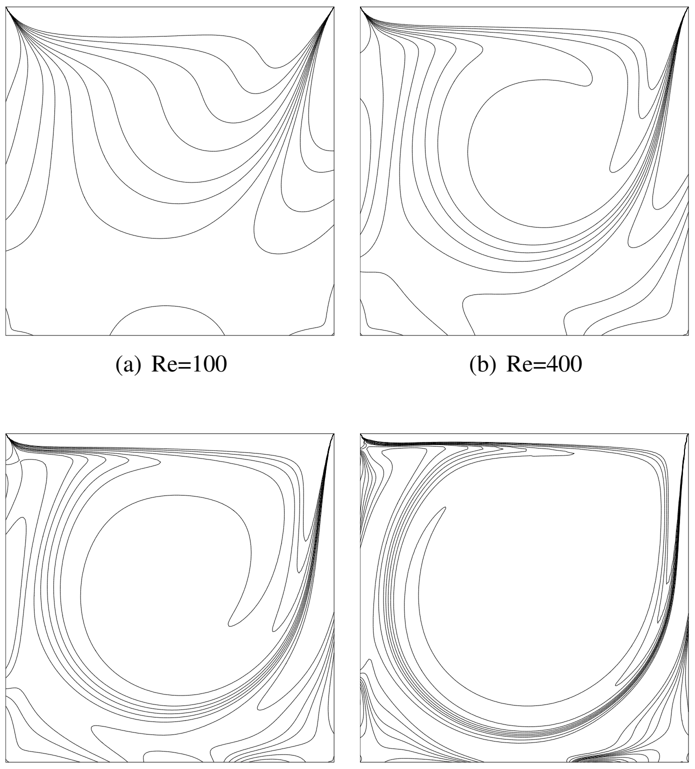

(d) Re=3200

Figure 7: The square LDC fluid flow problem. Stream-function (ψ) contours of the flow for several Reynolds numbers ($Re = 100$, 400, 1000 and 3200) by the present parallel CV method using 4 sub-domains for $Re = 100, 400, 1000$ and 2 sub-domains for $Re = 3200$ with the specifications: grid 151×151, $\Delta t = 1.E-03$, $DDTol = 1.E-06$, $CMTol = 1.E-06$ and $\beta = 2$.

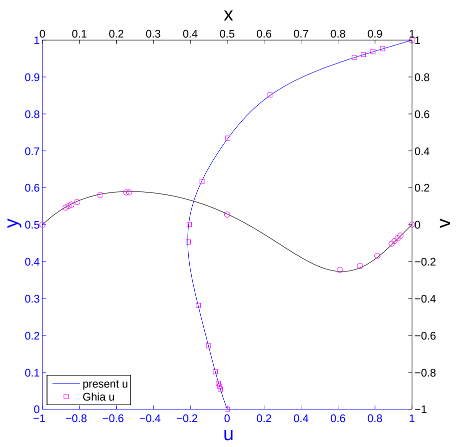


(a) Re=100 (b) Re=400

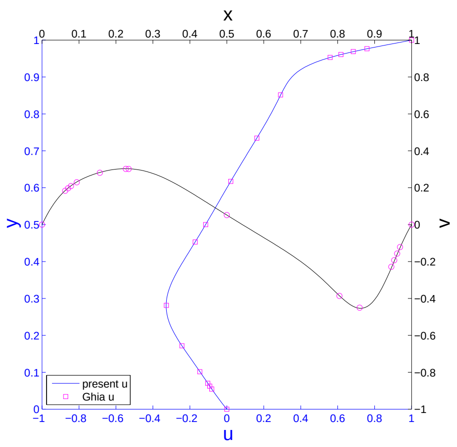
(c) Re=1000 (d) Re=3200

Figure 8: The square LDC fluid flow problem. Vorticity (ω) contours of the flow for several Reynolds numbers ($Re = 100$, 400, 1000 and 3200) by the present parallel CV method using 4 sub-domains for $Re = 100, 400, 1000$ and 2 sub-domains for $Re = 3200$. The other parameters are given in Fig. 7.

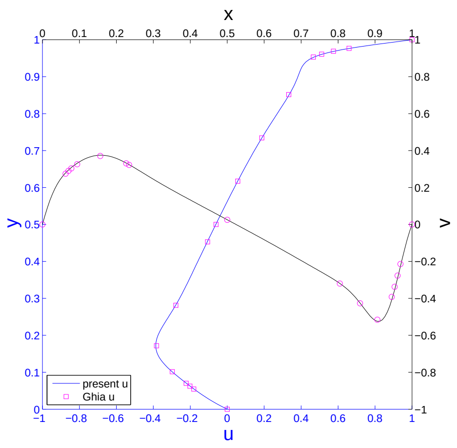
The efficiency of the present parallel method is assessed using the following criteria: the number of iterations (i), computation time (t), speed-up (spd) (ratio of the computation times using one processor and multi processors) and efficiency (eff) (the ratio of speed-up and the number of CPUs used). A fixed grid 151×151 is chosen to run the problem with various number of CPUs and the results are provided in Tables 2 - 5 for different Reynolds numbers. Results described in the left hand side of Tables 2 - 5 show that the computation time of the present parallel method (the P-CV method) for the time-dependent square lid driven cavity problem decrease tremendously as the number of sub-domains increases. An interpretation on the significant improvement of throughput can be found in Pham-Sy,



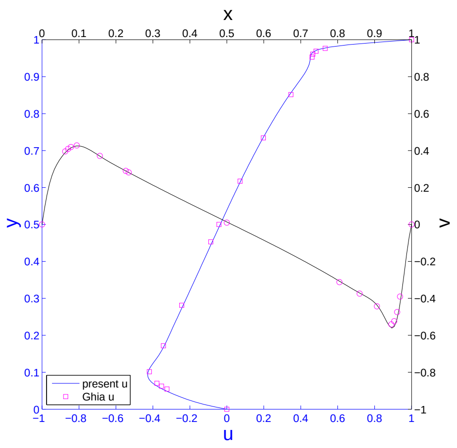
(a) Re=100



(b) Re=400



(c) Re=1000



(d) Re=3200

Figure 9: The square LDC fluid flow problem. Profiles of the u velocity along the vertical centreline and the v velocity along the horizontal centreline (solid lines) for several Reynolds numbers (Re = 100, 400, 1000 and 3200) by the present parallel CV method in comparison with the corresponding Ghia's results (□ for u velocity and ○ for v velocity). The parameters of the present method are given in Fig. 7.

Tran, Hoang-Trieu, Mai-Duy, and Tran-Cong (2013) for the parallel collocation method; a similar interpretation is applicable here. Again, there are always some thresholds (called $cpus_{opt}$) over which the increase of number of CPUs influences insignificantly on the efficiency based on all criteria (t, spd and eff). For example, the improvement of efficiency (eff) of computation is not significant anymore as the number of CPUs is more than 49, 64, 30 and 49 (Tables 2 - 5) for Re numbers 100, 400, 1000 and 3200, respectively using the grid of 151×151. Furthermore, the tendency of computational efficiency can be found similarly with the present parallel algorithm using the collocation method (named P-C method) (right hand side of the Tables 2 - 5).

Table 2: The square LDC fluid flow problem. Comparison between parallel CV (P-CV) and parallel collocation (P-C) methods with $Re = 100$, grid = 151×151, $dt = 1.E-03$, $DDTol = 1.E-06$, $CMTol = 1.E-06$, $\beta = 2$. CPUs: number of CPUs (sub-domains); i: number of iterations; $t(m)$: elapsed time (minutes); spd: speed-up; eff: efficiency. The observed super-linear speed up can be explained in terms of reduced matrix condition numbers (see main text).

	P-CV method				P-C method			
CPUs	i	t(m)	spd	eff	i	t(m)	spd	eff
1	8814	132.25	1.00	100.00	8814	130.50	1.00	100.00
2	1578	127.34	1.04	51.93	1574	126.62	1.03	51.53
4	1687	63.88	2.07	51.76	1682	63.31	2.06	51.53
6	1703	42.83	3.09	51.47	1698	44.33	2.94	49.06
9	1727	27.74	4.77	52.98	1722	27.82	4.69	52.11
12	1719	17.11	7.73	64.42	1713	16.60	7.86	65.51
16	1710	10.36	12.77	79.82	1703	10.07	12.95	80.96
20	1699	7.81	16.93	84.64	1691	7.73	16.88	84.39
25	1683	6.13	21.59	86.34	1675	5.98	21.81	87.24
30	1629	4.34	30.47	101.57	1619	4.43	29.44	98.12
36	1544	3.32	39.87	110.75	1531	3.25	40.14	111.49
42	1494	2.82	46.89	111.64	1479	2.80	46.65	111.08
49	1456	2.33	56.65	115.61	1446	2.32	56.13	114.55
56	2150	2.78	47.49	84.81	2146	2.70	48.32	86.29
64	2687	2.73	48.48	75.75	2682	2.76	47.36	74.00
72	2816	2.68	49.39	68.60	2811	2.66	48.98	68.03
81	2937	2.49	53.11	65.57	2932	2.45	53.28	65.78
90	3038	2.33	56.83	63.15	3032	2.29	56.95	63.28
100	3144	2.11	62.75	62.75	3138	2.05	63.76	63.76
110	3916	2.53	52.36	47.60	3904	2.43	53.77	48.88
121	4111	2.57	51.47	42.54	4098	2.51	52.00	42.98
132	4306	2.34	56.45	42.76	4291	2.24	58.18	44.08
144	4459	2.42	54.65	37.95	4445	2.38	54.88	38.11

Table 3: The square LDC fluid flow problem. Comparison between P-CV and P-C methods with $Re = 400$, $grid = 151 \times 151$. CPUs: number of CPUs (sub-domains); i: number of iterations; $t(m)$: elapsed time (minutes); spd: speed-up; eff: efficiency. Other parameters are given in Table 2.

	P-CV method				P-C method			
CPUs	i	t(m)	spd	eff	i	t(m)	spd	eff
1	22122	324.98	1.00	100.00	22107	338.55	1.00	100.00
2	3347	292.51	1.11	55.55	3340	240.24	1.41	70.46
4	3469	131.45	2.47	61.81	3459	130.46	2.60	64.88
6	3606	97.93	3.32	55.31	3593	90.65	3.73	62.25
9	3757	67.73	4.80	53.31	3742	58.94	5.74	63.83
12	3786	38.32	8.48	70.66	3768	34.32	9.86	82.21
16	3845	23.77	13.67	85.45	3826	23.05	14.69	91.80
20	3844	17.67	18.39	91.96	3824	17.85	18.96	94.81
25	3955	14.06	23.12	92.46	3932	13.90	24.35	97.41
30	4162	11.29	28.78	95.94	4134	10.69	31.68	105.60
36	4550	9.86	32.97	91.59	4520	9.11	37.16	103.23
42	4619	8.24	39.46	93.95	4589	8.02	42.24	100.56
49	4699	7.05	46.12	94.12	4671	7.14	47.42	96.78
56	4953	6.32	51.43	91.84	4917	6.07	55.79	99.62
64	5265	5.22	62.23	97.23	5218	5.09	66.47	103.86
72	5405	4.86	66.86	92.86	5345	4.80	70.58	98.03
81	5540	4.53	71.71	88.53	5471	4.49	75.36	93.04
90	5763	4.27	76.12	84.57	5694	4.14	81.73	90.81
100	5832	3.80	85.50	85.50	5758	3.74	90.51	90.51
110	5808	3.43	94.89	86.26	5753	3.46	97.90	89.00
121	5978	3.60	90.29	74.62	5919	3.55	95.48	78.91
132	6154	3.22	100.87	76.42	6053	3.22	105.05	79.58
144	6286	3.32	97.79	67.91	6222	3.30	102.73	71.34

The efficiency, speed-up and throughput of the present parallel method can be seen visually in Figs. 10(a), 10(c) and 10(e). These figures also depict the influence of the Reynolds number on the mentioned criteria of the present parallel algorithm with respect to the number of CPUs. For example, the efficiency of the present parallel method is higher for the lower Reynolds numbers. While the throughput increases gradually with respect to the number of sub-domains/CPUs (Fig. 10(e)),

Table 4: The square LDC fluid flow problem. Comparison between P-CV and P-C methods with $Re = 1000$, $grid = 151 \times 151$. CPUs: number of CPUs (subdomains); i: number of iterations; $t(m)$: elapsed time (minutes); spd: speed-up; eff: efficiency. Other parameters are given in Table 2.

	P-CV method				P-C method			
CPUs	i	t(m)	spd	eff	i	t(m)	spd	eff
1	30536	453.65	1.00	100.00	30442	400.02	1.00	100.00
2	5016	429.95	1.06	52.76	5081	407.46	0.98	49.09
4	4763	178.57	2.54	63.51	4824	179.80	2.22	55.62
6	4661	121.38	3.74	62.29	4655	119.21	3.36	55.93
9	4684	72.85	6.23	69.19	4671	79.54	5.03	55.88
12	5214	51.39	8.83	73.56	5129	51.00	7.84	65.36
16	6211	37.53	12.09	75.55	6139	38.45	10.40	65.02
20	6498	29.17	15.55	77.75	6428	28.54	14.02	70.09
25	6765	22.87	19.84	79.36	6699	22.90	17.47	69.87
30	7452	18.38	24.69	82.29	7397	18.76	21.32	71.07
36	8580	17.78	25.52	70.89	8517	17.02	23.50	65.28
42	8709	15.09	30.07	71.60	8660	15.25	26.24	62.48
49	8593	13.35	33.98	69.34	8543	13.17	30.37	61.97
56	9299	11.47	39.57	70.65	9207	11.21	35.69	63.74
64	10750	10.61	42.76	66.81	10628	10.28	38.91	60.80
72	11542	10.21	44.45	61.73	11410	10.15	39.42	54.76
81	11922	9.39	48.32	59.66	11789	9.19	43.51	53.72
90	12122	8.42	53.85	59.83	11959	8.56	46.75	51.95
100	11637	7.32	62.01	62.01	11475	7.20	55.53	55.53
110	12025	7.01	64.68	58.80	11863	7.08	56.48	51.34
121	12315	7.23	62.77	51.88	12121	7.10	56.36	46.58
132	13058	6.69	67.76	51.33	12874	6.69	59.77	45.28
144	13359	6.76	67.12	46.61	13159	6.81	58.78	40.82

Table 5: The square LDC fluid flow problem. Comparison between P-CV and P-C methods with $Re = 3200$, $grid = 151 \times 151$. CPUs: number of CPUs (sub-domains); i: number of iterations; $t(m)$: elapsed time (minutes); spd: speed-up; eff: efficiency. Other parameters are given in Table 2.

	P-CV method				P-C method			
CPUs	i	t(m)	spd	eff	i	t(m)	spd	eff
1	69367	1003.93	1.00	100.00	69712	999.36	1.00	100.00
2	16057	1257.83	0.80	39.91	16590	1382.11	0.72	36.15
4	17139	622.24	1.61	40.34	16500	578.98	1.73	43.15
6	17033	439.62	2.28	38.06	16755	433.14	2.31	38.45
9	17023	289.41	3.47	38.54	17386	285.37	3.50	38.91
12	15755	154.36	6.50	54.20	15437	145.97	6.85	57.05
16	16630	101.29	9.91	61.95	16835	102.68	9.73	60.83
20	17404	78.81	12.74	63.69	17283	77.67	12.87	64.33
25	16376	57.55	17.45	69.78	15759	55.19	18.11	72.43
30	15305	39.14	25.65	85.50	14071	36.78	27.17	90.56
36	19120	39.43	25.46	70.72	17046	35.32	28.29	78.59
42	16089	27.98	35.88	85.43	16799	29.59	33.77	80.40
49	17182	25.98	38.64	78.86	17667	26.87	37.19	75.90
56	22360	27.24	36.86	65.82	22660	27.31	36.60	65.35
64	26652	25.62	39.19	61.23	26790	25.65	38.96	60.87
72	27080	24.21	41.47	57.60	27573	24.45	40.88	56.77
81	28656	22.26	45.09	55.67	29246	22.39	44.64	55.11
90	31817	22.13	45.37	50.41	31127	21.51	46.46	51.62
100	32548	19.76	50.80	50.80	31885	19.68	50.77	50.77
110	33739	19.18	52.36	47.60	32418	18.82	53.11	48.28
121	26585	16.46	61.01	50.42	33365	19.68	50.78	41.97
132	34222	16.60	60.49	45.83	33334	16.86	59.27	44.90
144	35285	17.35	57.87	40.19	34716	17.44	57.29	39.79

the gradients of time curves decrease as the number of CPUs is more than around 20. This is also indicated by the efficiency curves given in Fig. 10(a). Similar trends of the efficiency, speed-up and throughput are also obtained by the present parallel algorithm using the collocation method and given in Figs. 10(b), 10(d) and 10(f). It shows that the choice of the scale for sub-domains/CPUs plays an important role

Table 6: The square LDC fluid flow problem. Condition numbers CN_ω and CN_ψ in single and parallel solutions with $Re = 100$ and grid = 151 $\times$ 151. CPUs: number of CPUs (sub-domains).

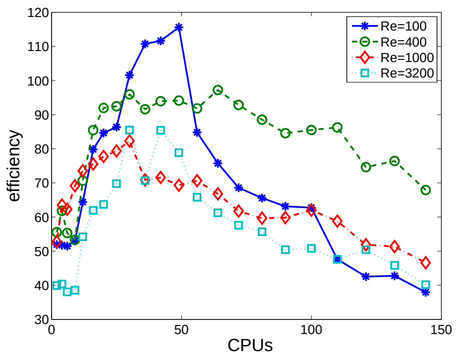
CPUs	CN_ω	CN_ψ	CPUs	CN_ω	CN_ψ
1	2.6341	1.29E+04	49	1.1581	3.02E+02
2	1.1799	5.33E+03	56	1.1581	2.44E+02
4	1.1581	3.49E+03	64	1.1581	2.06E+02
6	1.1581	2.18E+03	72	1.1581	1.83E+02
9	1.1581	1.61E+03	81	1.1581	1.64E+02
12	1.1581	1.12E+03	90	1.1581	1.44E+02
16	1.1581	8.71E+02	100	1.1567	1.28E+02
20	1.1581	6.99E+02	110	1.1581	1.19E+02
25	1.1581	5.87E+02	121	1.1581	1.12E+02
30	1.1581	4.65E+02	132	1.1567	1.03E+02
36	1.1581	3.87E+02	144	1.1567	9.56E+01
42	1.1581	3.39E+02			

in the performance of parallel computation schemes for a given problem.

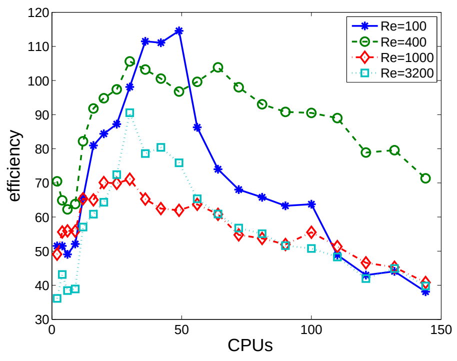
It is observed that a super-linear speed-up is achieved using a range of numbers of CPUs 30, 36, 42 and 49 with corresponding efficiencies 101%, 110%, 111% and 115% for the Reynolds number $Re = 100$ (Table 2). This is an exclusive behavior and sometimes controversial in classical parallel computing, when the speed-up is higher than the number of CPUs used in a parallel algorithm. For these cases, the super-linear speed-up is considered to be related to the decrease of condition number of each subdomain which plays a crucial role for the stability of a numerical method. Indeed, by decomposing the domain, sub-problem in each sub-domain is not only smaller in terms of degrees of freedom but also has smaller condition number (see Table 6).

The efficiency of the algorithm in large scale problems is also investigated. For testing purposes, the fluid with $Re = 1000$ is simulated using very fine grids including grid-1 = 401 $\times$ 401 and grid-2 = 601 $\times$ 601 with the following parameters $DDTol = 1.E-06$, $CMTol = 1.E-06$, $\beta = 2$ and $\Delta t = 1.E-03$ for grid-1 and $5.E-04$ for grid-2.

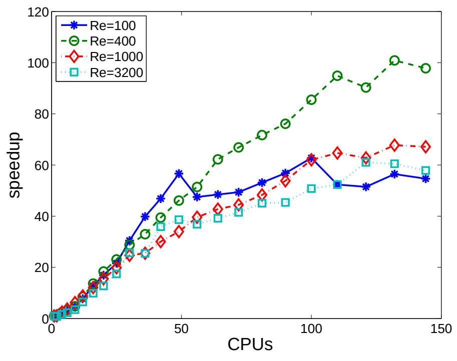
While Fig. 11(a) points out a gradual increase of throughput with respect to the number of CPUs for different scales by the present P-CV method, Fig. 11(b) depicts the influence of the grid density on the efficiency with respect to the number of CPUs. Indeed, the gradient of time curves of finer gridsize is steeper, which again indicates that the efficiency of the present parallel method will be higher for larger scale problems.



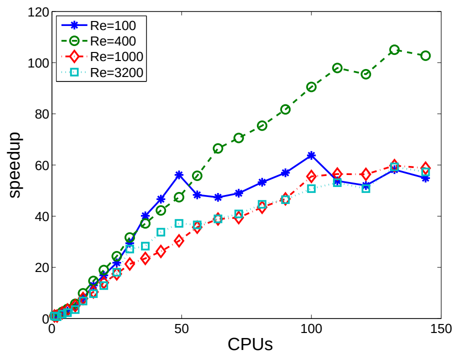
(a) P-CV method - efficiency



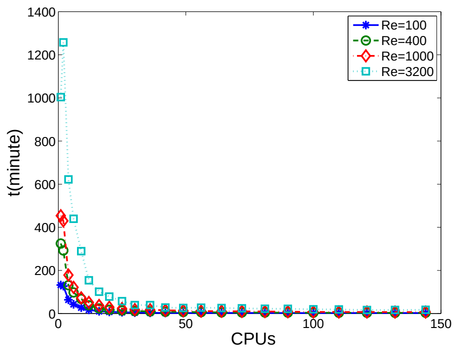
(b) P-C method - efficiency



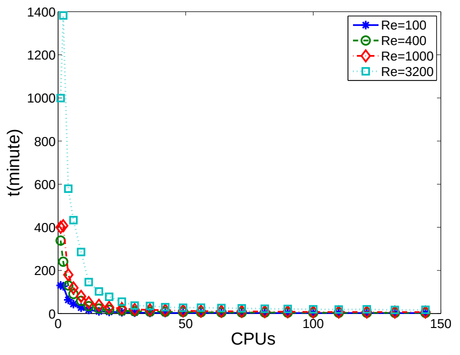
(c) P-CV method - speed-up



(d) P-C method - speed-up

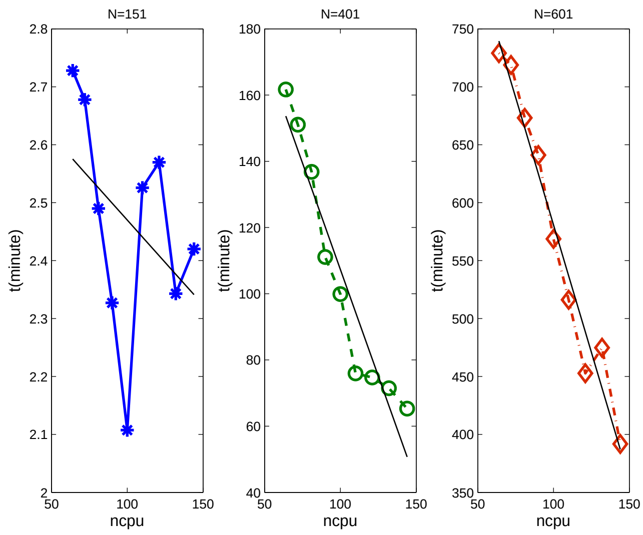


(e) P-CV method - throughput

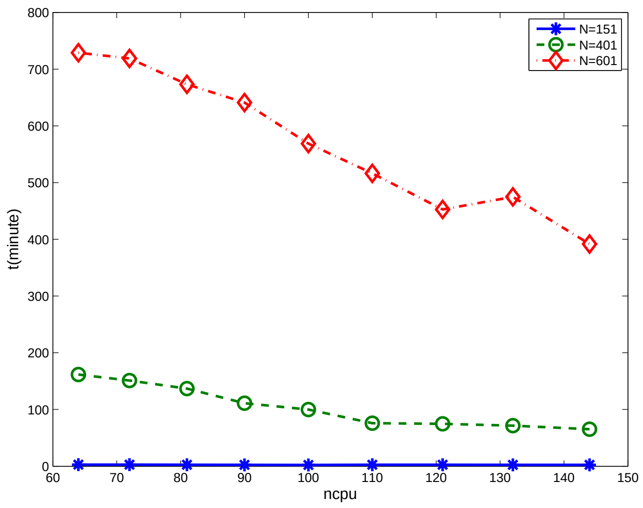


(f) P-C method - throughput

Figure 10: The square LDC fluid flow problem. Comparison between the parallel performance of the P-C and P-CV methods for several Reynolds numbers ($Re = 100$, 400, 1000 and 3200) with a grid of 151×151: the efficiency, speed-up and throughput of the two methods as a function of the number of CPUs. Other parameters are given in Tables 2 - 5.



(a)



(b)

Figure 11: The square LDC fluid flow problem. Throughput of the P-CV method with $Re = 1000$ using different grids: 151×151, 401×401 and 601×601 as a function of the number of CPUs.

5.2 *Triangular cavity*

The triangular cavity has been proposed as a test case for the numerical algorithm in the case of non-rectangular domain. The domain is an equilateral triangle with the left and right sides being fixed and the top side (also called the lid) moving at a constant velocity from left to right. The problem's geometry and boundary condition can be seen visually in Fig. 12.

It is noted that while implementing CVs with non-rectangular domains, one needs to take extra care regarding points closed to boundary to make sure that CVs do not intersect with each other nor with the boundary. Figure 13 shows an example of control volume formation for a triangular domain.

The boundary conditions are given in terms of the stream function as

$$\psi = 0, \quad \frac{\partial \psi}{\partial y} = 1 \quad \forall (x,y) \in \Gamma_1; \tag{22}$$

$$\psi = 0, \quad \frac{\partial \psi}{\partial x} = 0, \quad \frac{\partial \psi}{\partial y} = 0 \quad \forall (x,y) \in \Gamma_2 \cup \Gamma_3, \tag{23}$$

where the variables are defined before. The solving procedure remains the same as for the square cavity problem. However, when approximate the boundary value for ω two following cases must be considered.

First, for boundary points that lie on both x grid-line and y grid-line the approximation can be carried out normally by using 1D IRBF in two directions following Eq. (18).

Second, for boundary points, that lie only on x grid-line or y grid-line, its approximation, thus, is available only in one direction. In this case, equivalent formulas provided by Le-Cao, Mai-Duy, and Tran-Cong (2009) are used as follows.

$$\omega_b = -\left[1 + \left(\frac{t_x}{t_y}\right)^2\right] \frac{\partial^2 \psi_b}{\partial x^2}, \tag{24}$$

for points on x-grid line and

$$\omega_b = -\left[1 + \left(\frac{t_y}{t_x}\right)^2\right] \frac{\partial^2 \psi_b}{\partial y^2}, \tag{25}$$

for points on y-grid line, where t_x and t_y are the x- and y-components of the unit vector tangential to the boundary.

In this paper, a range of Reynolds numbers (100, 200, 500, 1000) is investigated. Again, the streamline (Fig. 14), vorticity contours (Fig. 15) and velocity profiles

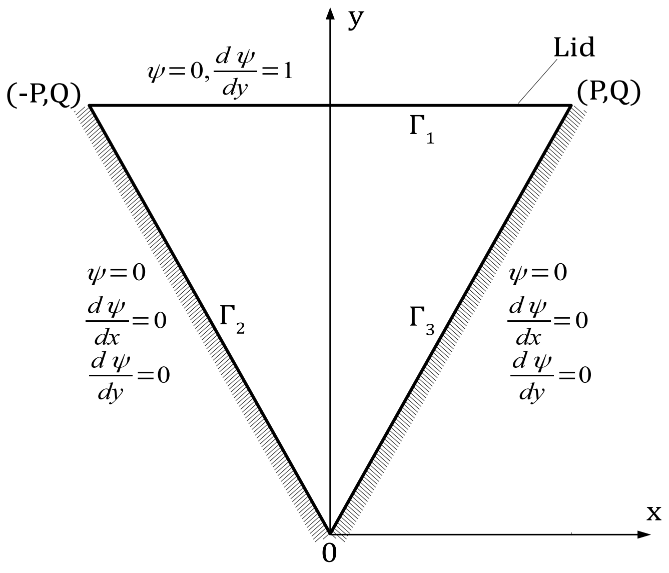


Figure 12: Triangular lid driven cavity flow problem: geometry and boundary conditions. $P=\sqrt{3}$, $Q=3$. No slip is assumed between the fluid and solid surfaces. The top lid is moving from left to right with a speed of 1.

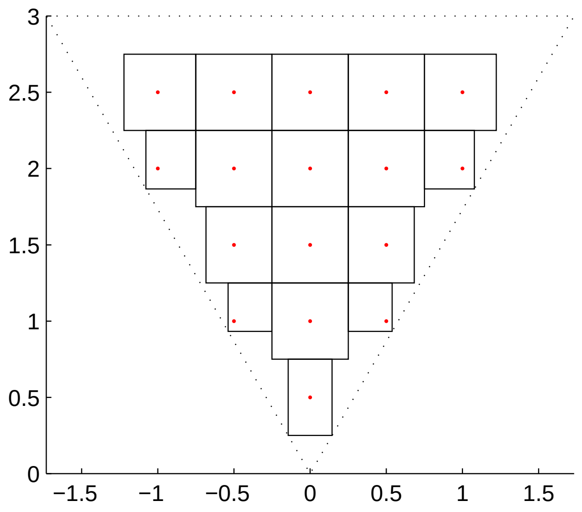
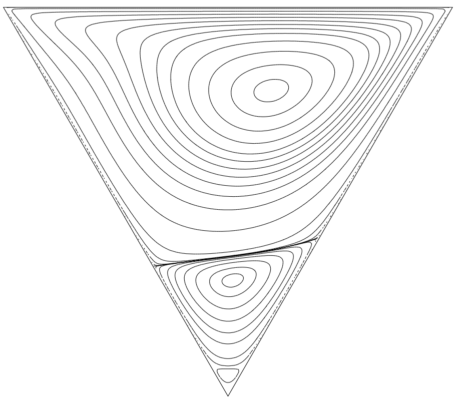
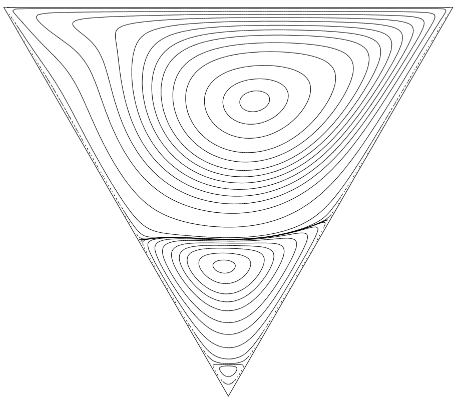
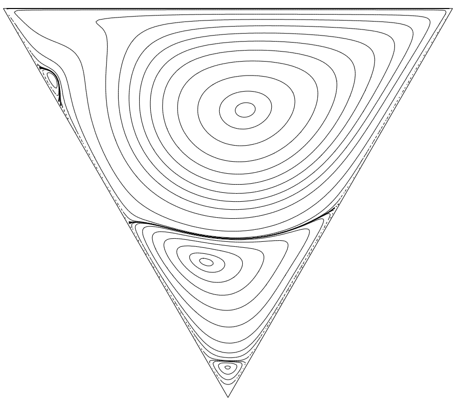


Figure 13: CV formation in 2D.

along central horizontal line $y = 2$ and vertical line $x = 0$ (Fig. 16) by the present CV method with 4 sub-domains agree very well with ones by Kohno and Bathe (2006) using flow-conditioned-based finite element method.


(a) Re=100


(b) Re=200


(c) Re=500

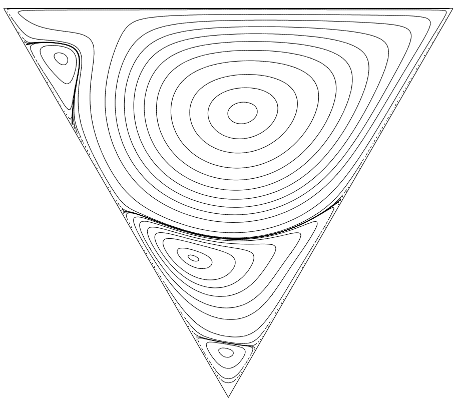
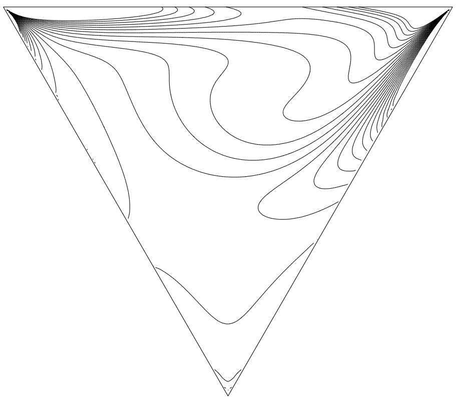
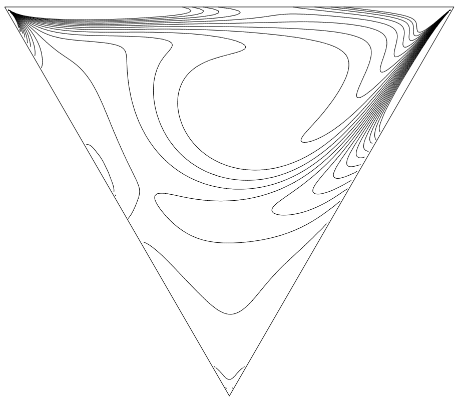

(d) Re=1000

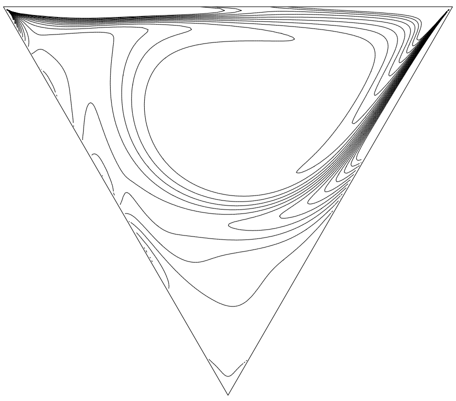
Figure 14: The triangular LDC fluid flow problem. Stream-function (ψ) contours of the flow for several Reynolds numbers by the present parallel CV method using 4 sub-domains with grid of 24697 nodes, $\Delta t = 5.E-04$, $DDTol = 1.E-06$, $CMTol = 1.E-06$ and $\beta = 1$.



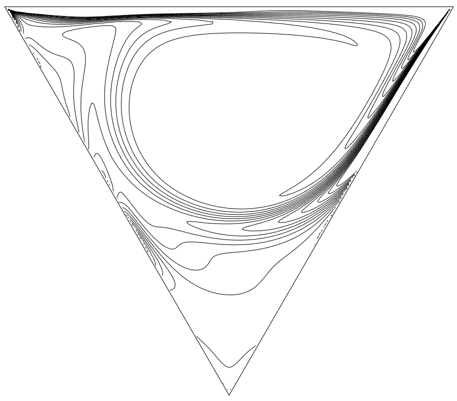
(a) Re=100



(b) Re=200



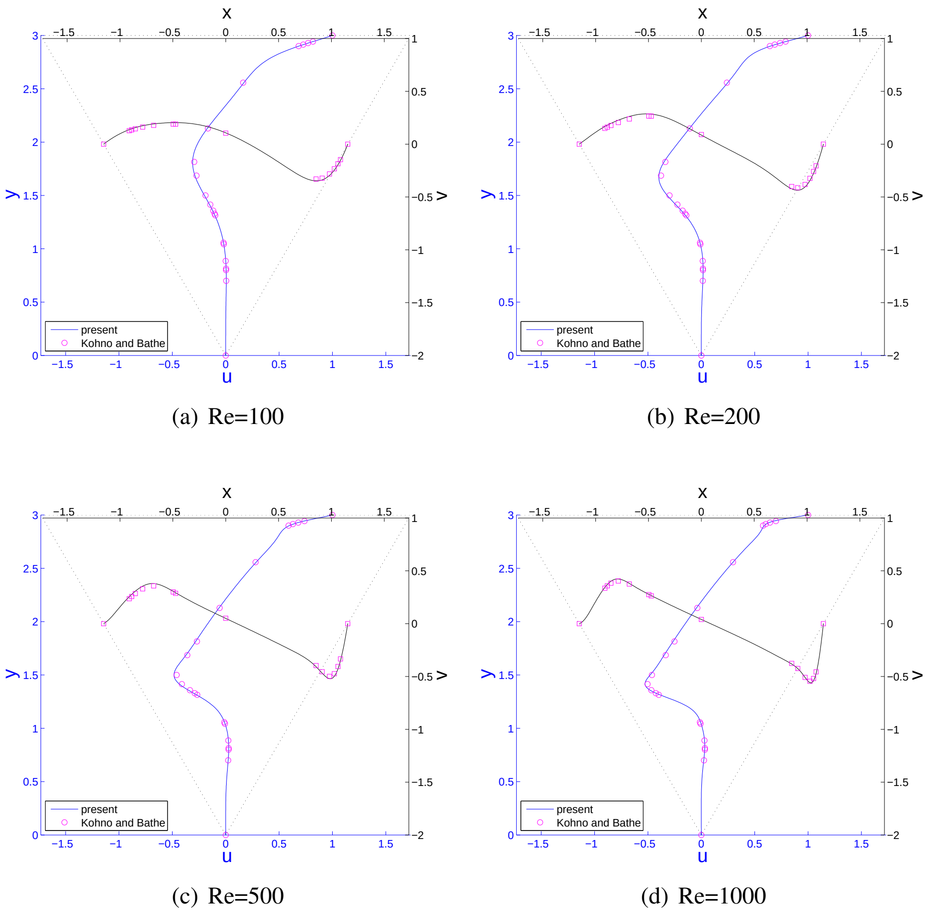
(c) Re=500



(d) Re=1000

Figure 15: The triangular LDC fluid flow problem. Vorticity (ω) contours of the flow for several Reynolds numbers by the present parallel CV method. Other parameters are given in Fig. 14.

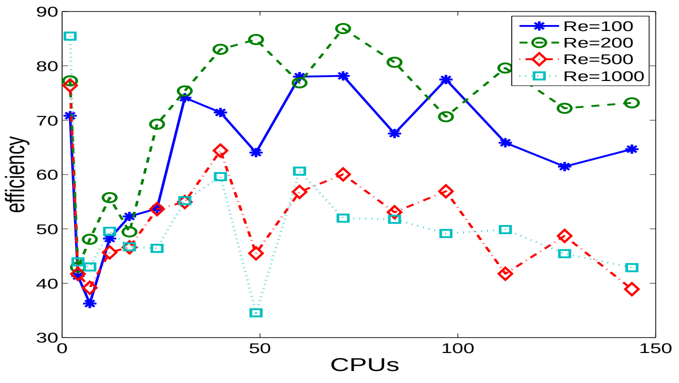
.



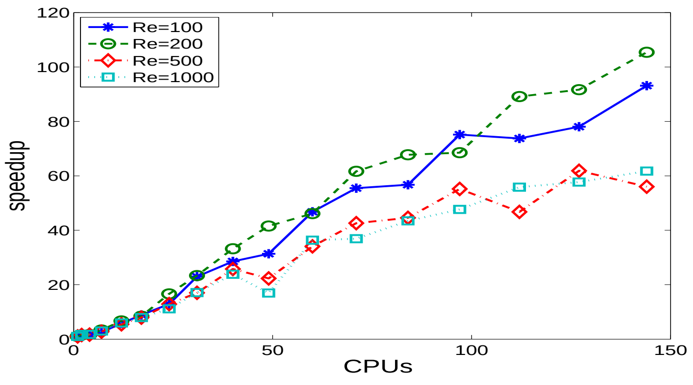
(a) Re=100 (b) Re=200

(c) Re=500 (d) Re=1000

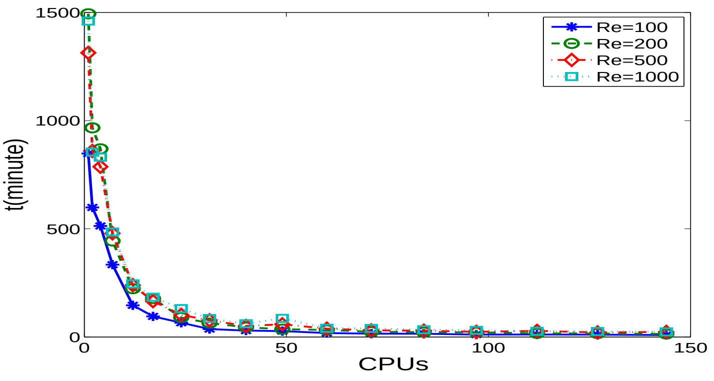
Figure 16: The triangular LDC fluid flow problem. Vorticity profiles along vertical line ($x = 0$) and horizontal line ($y = 2$) for several Reynolds numbers by the present parallel CV method in comparison with the corresponding Kohno and Bathe's results (□ for u velocity and ○ for v velocity). Other parameters of the present method are given in Fig. 14.



(a)



(b)



(c)

Figure 17: The triangular LDC fluid flow problem. Parallel performance of the P-CV methods for several Reynolds numbers using a grid of 24607 nodes: the efficiency, speed-up and throughput as a function of the number of CPUs. Other parameters are given in Table 7.

.

Table 7: The triangular LDC fluid flow problem. Results by the present P-CV method with grid of 24697 nodes, $dt = 5.E-04$, $DDTol = 1.E-06$, $CMTol = 1.E-06$, $\beta = 1$. CPUs: number of CPUs (sub-domains); i: number of iterations; $t(m)$: elapsed time (minutes); spd: speed-up; eff: efficiency.

	Re100				Re200				Re500				Re1000			
CPUs	i	t(m)	spd	eff	i	t(m)	spd	eff	i	t(m)	spd	eff	i	t(m)	spd	eff
1	47800	848.12	1.00	100	86584	1493.69	1.00	100	72673	1313.30	1.00	100	83900	1460.14	1.00	100
2	10352	598.72	1.42	70.83	15758	966.92	1.54	77.24	14362	859.59	1.53	76.39	14173	854.31	1.71	85.46
4	11643	513.13	1.65	41.32	18926	869.66	1.72	42.94	17434	786.89	1.67	41.72	18679	830.83	1.76	43.94
7	13247	334.09	2.54	36.27	18286	443.81	3.37	48.08	18898	478.99	2.74	39.17	19413	485.07	3.01	43.00
12	12594	146.53	5.79	48.23	18908	223.26	6.69	55.75	20373	239.64	5.48	45.67	20370	245.55	5.95	49.55
17	12511	95.40	8.89	52.29	23245	177.67	8.41	49.45	22050	165.74	7.92	46.61	24900	183.87	7.94	46.71
24	13729	65.77	12.90	53.73	19644	89.86	16.62	69.26	21977	102.03	12.87	53.63	28436	131.05	11.14	46.42
31	12912	36.90	22.98	74.13	21156	63.92	23.37	75.38	26608	77.11	17.03	54.94	28122	85.33	17.11	55.20
40	12695	29.69	28.57	71.42	19745	44.97	33.21	83.03	22067	50.98	25.76	64.41	26965	61.23	23.85	59.62
49	15267	27.03	31.38	64.04	20648	35.93	41.57	84.84	34223	58.90	22.30	45.51	49217	86.22	16.93	34.56
60	13267	18.12	46.80	78.01	24210	32.38	46.13	76.88	28281	38.53	34.08	56.81	29830	40.14	36.38	60.63
71	12884	15.29	55.48	78.14	20574	24.22	61.67	86.86	26316	30.81	42.62	60.03	34076	39.57	36.90	51.98
84	15687	14.95	56.73	67.54	22934	22.05	67.75	80.65	30586	29.46	44.57	53.06	35805	33.58	43.48	51.76
97	13683	11.28	75.16	77.49	25704	21.80	68.52	70.64	28596	23.79	55.21	56.91	36386	30.63	47.67	49.14
112	14760	11.50	73.75	65.85	21280	16.76	89.14	79.59	36899	28.09	46.76	41.75	33732	26.14	55.86	49.87
127	16187	10.86	78.06	61.47	24681	16.29	91.68	72.19	32695	21.23	61.87	48.71	38666	25.30	57.71	45.44
144	14810	9.11	93.13	64.68	22833	14.17	105.41	73.20	37874	23.44	56.03	38.91	38753	23.65	61.75	42.88

In terms of parallel efficiency, Table 7 gives detailed results of the parallel algorithm for several Reynolds numbers and with a grid of 205×205 (or 24697 grid points). Visual forms can be found in Fig. 17. For each Reynolds number, although results showed that the computation time decrease gradually as the number of sub-domains (CPUs) increases (Fig. 17(c)), the optimum number of CPUs $cpus_{opt}$ of the parallel method described by the efficiency (eff) for each case is not clear (Fig. 17(a)). This can be explained as the influence of the domain decomposition for a non-rectangular domain problem where the numbers of collocations/CVs in sub-domains are not equal, resulting in significant variation of the amount of work to be completed from sub-domain to sub-domain. Thus, the results show that the sub-domain formation plays an important role in parallel computation schemes.

6 Conclusion

In this paper, we proposed a DD parallel distributed method coupled with a local IRBF CV approach. The proposed method is successfully implemented to simulate the lid driven cavity flow in rectangular and non-rectangular domains. It has been shown that results produced by the method are in excellent agreement with the spectral benchmark solutions by Botella and Peyret (1998) and Ghia, Ghia, and Shin (1982) for the square domain and by Kohno and Bathe (2006) for the triangular domain. A very important achievement of this paper is the high time-efficiency of the parallel algorithm including the speed-up. It is shown that the speedup grows steadily with the increase of the number of CPUs. This indicates excellent scalability of the method. Moreover, a super-linear efficiency has been observed for several cases; this phenomenon is best explained by the decrease of condition numbers in sub-domains. The parallel algorithm performs well in both collocation and CV methods. Indeed, the trend in efficiency with increasing number of CPUs for several Reynolds numbers is consistent with results achieved by the collocation method reported in Pham-Sy, Tran, Hoang-Trieu, Mai-Duy, and Tran-Cong (2013).

Acknowledgement: The first author would like to thank the CESRC, Faculty of Health, Engineering and Sciences and University of Southern Queensland for a Ph.D. scholarship. This research was supported by the Australian Research Council. The computation was performed on the USQ high performance computing cluster supported by the Queensland Cyber Infrastructure Foundation (QCIF). The authors would like to thank the reviewers for their helpful comments.

References

Botella, O.; Peyret, R. (1998): Benchmark spectral results on the lid-driven cavity flow. *Computers & Fluids*, vol. 27, pp. 421–433.

Chandhini, G.; Sanyasiraju, Y. (2007): Local RBF-FD solutions for steady convection-diffusion problems. *International journal for numerical methods in Engineering*, vol. 72(3), pp. 352–378.

Chinchapatnam, P. P.; Djidjeli, K.; Nair, P. B. (2007): Radial basis function meshless method for the steady incompressible Navier-Stokes equations. *International Journal of Computer Mathematics*, vol. 84, no. 10, pp. 1509–1521.

Ghia, U.; Ghia, K. N.; Shin, C. T. (1982): High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *Journal of Computational Physics*, vol. 48, pp. 387–411.

Hoang-Trieu, T.-T.; Mai-Duy, N.; Tran-Cong, T. (2012): Several compact local stencils based on Integrated RBFs for fourth-order ODEs and PDEs. *Computer Modeling in Engineering & Sciences*, vol. 84(2), pp. 171–203.

Ingber, M.; Chen, C.; Tanski, J. (2004): A mesh free approach using radial basis functions and parallel domain decomposition for solving three-dimensional diffusion equations. *International Journal for Numerical Methods in Engineering*, vol. 60, pp. 2183–2201.

Kansa, E. J. (1990): Multiquadrics - A scattered data approximation scheme with applications to computational fluid-dynamics - I Surface approximations and partial derivative estimates. *Computers & Mathematics with Applications*, vol. 19(8-9), pp. 127–145.

Kim, Y.; Kim, D. W.; Jun, S.; Lee, J. H. (2007): Meshfree point collocation method for the stream-vorticity formulation of 2D incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 33-34, pp. 3095–3109.

Kohno, H.; Bathe, K.-J. (2006): A flow-condition-based interpolation finite element procedure for triangular grids. *International Journal for Numerical Methods in Fluids*, vol. 51, no. 6, pp. 673–699.

Le-Cao, K.; Mai-Duy, N.; Tran-Cong, T. (2009): An effective Intergrated-RBFN cartesian-grid discretization for the streamfunction-vorticity-temperature formulation in nonrectangular domains. *Numerical Heat Transfer, Part B: Fundamentals*, vol. 55(6), pp. 480–502.

Lin, H.; Atluri, S. (2001): The Meshless Local Petrov-Galerkin (MLPG) Method for Solving Incompressible Navier-Stokes Equations. *CMES: Computer Modeling in Engineering & Sciences*, vol. 2, no. 2, pp. 117–142.

Mai-Duy, N.; Tran-Cong, T. (2001): Numerical solution of differential equations using multiquadric radial basis function networks. *Neural Networks*, vol. 14, pp. 185–199.

Mai-Duy, N.; Tran-Cong, T. (2010): A numerical study of 2D integrated RBFNs incorporating Cartesian grids for solving 2D elliptic differential problems. *Numerical methods for Partial Differential Equation*, vol. 26, pp. 1443–1462.

MATLAB (2012): *Parallel Computing Toolbox User's Guide R2012b*. The MathWorks Inc., Natick, Massachusetts, United States, 6.1 edition.

Patankar, S. V. (1980): *Numerical heat transfer and fluid flow*. CRC Press.

Pham-Sy, N.; Tran, C.-D.; Hoang-Trieu, T.-T.; Mai-Duy, N.; Tran-Cong, T. (2013): Compact local IRBF and domain decomposition method for solving PDEs using a distributed termination detection based parallel algorithm. *CMES: Computer Modeling in Engineering & Sciences*, vol. 92, no. 1, pp. 1–31.

Quarteroni, A.; Valli, A. (1999): *Domain decomposition methods for partial differential equations*. Clarendon Press.

Shirazaki, M.; Yagawa, G. (1999): Large-scale parallel flow analysis based on free mesh method: a virtually meshless method. *Computer Methods in Applied Mechanics and Engineering*, vol. 174, no. 3-4, pp. 419–431.

Shu, C.; Ding, H.; Yeo, K. (2005): Computation of Incompressible Navier-Stokes Equations by Local RBF-based Differential Quadrature Method. *CMES: Computer Modeling in Engineering & Sciences*, vol. 7, no. 2, pp. 195–206.

Singh, I. V.; Jain, P. K. (2005): Parallel Meshless EFG Solution for Fluid Flow Problems. *Numerical Heat Transfer, Part B: Fundamentals*, vol. 48, no. 1, pp. 45–66.

Tran, C.-D.; Phillips, D. G.; Tran-Cong, T. (2009): Computation of dilute polymer solution flows using BCF-RBFN based method and domain decomposition technique. *Korea-Australia Rheology Journal*, vol. 21(1), pp. 1–12.