INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2006; **51**:897–912
Published online 28 December 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1161

Adaptive mesh refinement-enhanced local DFD method and its application to solve Navier–Stokes equations

C. Shu*,† and Y. L. Wu

Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 117576, Singapore

SUMMARY

Recently, the domain-free discretization (DFD) method was presented to efficiently solve problems with complex geometries without introducing the coordinate transformation. In order to exploit the high performance of the DFD method, in this paper, the local DFD method with the use of Cartesian mesh is presented, where the physical domain is covered by a Cartesian mesh and the local DFD method is applied for numerical discretization. In order to further improve the efficiency of the solver, the newly developed solution-based adaptive mesh refinement (AMR) technique is also introduced. The proposed methods are then applied to the simulation of natural convection in concentric annuli between a square outer cylinder and a circular inner cylinder. Numerical experiments show that the present numerical results agree very well with available data in the literature, and AMR-enhanced local DFD method is an effective tool for the computation of flow problems. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: DFD method; Cartesian mesh; adaptive refinement; incompressible flow; natural convection; eccentric annuli

1. INTRODUCTION

Most of flow problems can be simulated by using conventional numerical approaches such as finite difference (FD) method [1–3] and differential quadrature (DQ) method [4–8]. Usually, these methods confine the numerical discretization of governing equations in the physical domain, which is regular. Due to this feature, if irregular geometry is involved, the coordinate transformation and body-fitted grid generation have to be introduced. This process not only brings more complexity into the computation, but also introduces additional error into the simulation.

Recently, the domain-free discretization (DFD) method was presented by Shu and his co-workers [9, 10] to solve partial differential equations (PDEs) on an irregular domain without

*Correspondence to: Chang Shu, Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 117576, Singapore.
†E-mail: mpeshuc@nus.edu.sg

Received 30 August 2005
Revised 15 November 2005
Accepted 16 November 2005

introducing the coordinate transformation. The basic idea of DFD is inspired from the analytical method. That is, the discrete form of given PDEs can be irrelevant of the solution domain. It may involve some points which may not be the mesh node and can be inside or outside the solution domain. The functional values at those points can be evaluated by approximate forms of solution along a line. In the work of Shu and Fan [9], the PDEs on the singly connected domain are solved in the Cartesian coordinate system, and the approximate forms of solution are obtained along each vertical or horizontal line. In the work of Shu and Wu [10], the PDEs on the doubly connected domain are solved in the cylindrical coordinate system, and the approximate forms of solution are obtained along each radial line. Since approximate form of solution along each line is obtained by using all the information of mesh nodes at that line, the DFD method is named the global DFD approach. In the earlier applications [9, 10], this approach has demonstrated its high efficiency and accuracy in solving PDEs such as Navier–Stokes equations on irregular domains. However, the global DFD encounters two major difficulties. One is for extrapolation. When the outside point is far away from the boundary, the global form of solution along the respective line, an example being the Lagrange interpolated polynomial, would generate large extrapolation coefficients, causing large numerical errors. Another difficulty is for multiply connected (more that two) domain problems. For this case, it is difficult to get approximate form of solution along a line where more than two boundary points are involved. To remove these difficulties and make the method be more general, the local DFD approach is developed in this work. In the present approach, all numerical work including discretization of derivatives and approximate form of solution for interpolation/extrapolation is made locally by using low-order polynomials. Note that in the local DFD approach, any problem is solved in the Cartesian coordinate system. So, in this sense, the local DFD is a kind of Cartesian mesh solver.

In the present DFD-Cartesian mesh solver, the low-order FD schemes are adopted for numerical discretization. Since the FD schemes have simple forms on the uniform mesh, they are usually applied on the uniform mesh. Obviously, the use of uniform mesh in the whole domain is not efficient for a flow problem since there is a thin boundary layer near solid walls which need finer mesh than other regions. We have to afford huge computational cost to capture the thin boundary layer if a single uniform mesh is used. From the computational point of view, the use of adaptive mesh refinement (AMR) technique is the best remedy for the efficiency improvement. In the past decades, intensive research and efforts have been devoted to the development of adaptive refinement procedures. As a result, a large number of adaptive algorithms have been proposed [11–15]. One representative of structured grid approaches is adaptive Cartesian mesh refinement proposed by Berger and Oliger [11] and Berger and LeVeque [12]. Their approach is established on regular Cartesian meshes, but arranged hierarchically with different resolutions. At the fine/coarse cell interfaces, special treatment is required for the communications between the meshes at different levels. Recently, Ding and Shu [16] proposed a new AMR strategy—stencil adaptive algorithm, which is easy for implementation, and has been successfully applied to simulate the incompressible flow with simple geometry.

In this work, the local DFD approach is combined with the stencil adaptive algorithm [16] to solve problems with curved boundary. The proposed approach is validated by its application to simulate natural convection in a concentric annulus between an inner circular cylinder and an outer square cylinder. The obtained numerical results agree very well with available data in the literature.

2. AMR-ENHANCED LOCAL DFD METHOD

2.1. Local DFD method

The details of the global DFD method can be found in the work of Shu and Fan [9] and Shu and Wu [10]. The procedure of the local DFD method on the Cartesian mesh (the present method) is described here, which is illustrated in Figure 1.

For simplicity, we assume that the mesh is uniform, and the mesh spacing in both the x and y directions is the same. Suppose that the solution of a problem is expressed by $u(x, y)$ in the Cartesian coordinate system. In Figure 1, the mesh nodes inside the physical domain (interior mesh nodes) are represented by the solid circle, while the mesh nodes outside the domain are represented by open circles. The open squares on the boundary are points intersected by the mesh lines and boundary curve.

It should be noticed that the discretization of the PDE is always made at the interior mesh nodes (i.e. the solid circle in Figure 1). Therefore, the functional values at the nodes outside the domain (i.e. the open circle in Figure 1) are not defined. Since the numerical discretization only involves the functional values at the reference node and its neighbours, for the nodes which are not near the boundary such as F, there is no problem to discretize derivatives in both the x and y directions by using the central-difference scheme on the uniform mesh. For numerical discretization of derivatives at the nodes near the boundary such as B1, the local DFD method is applied. According to the DFD method, the discrete form of PDEs can involve some nodes outside the solution domain, and the functional values at these outside nodes such as node E can be computed by using local low-order approximate form of solution. Therefore, the central-difference scheme on the uniform mesh can still be used by the local DFD method.

For example, the discrete form of the first and second derivatives, respectively, in the x direction at the position B1 can be approximated by

$$\frac{\partial^2 u_{\mathrm{B1}}}{\partial x^2} = \frac{1}{h^2}[u_{\mathrm{E}} - 2u_{\mathrm{B1}} + u_{\mathrm{C1}}] + o(h^2) \tag{1}$$

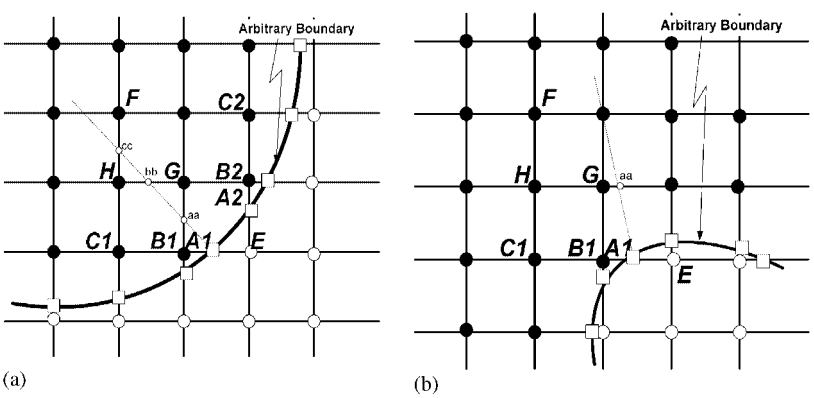

Figure 1. Configuration of the DFD-Cartesian mesh method with mesh and extrapolation points.

$$\frac{\partial u_{B1}}{\partial x} = \frac{1}{2h}[u_E - u_{C1}] + o(h^2) \tag{2}$$

where h is the mesh spacing on the uniform mesh in the x direction, $O(h^2)$ denotes the second order of the schemes.

The functional value at node E can be calculated by extrapolation through the local approximate form of the solution. In this paper, we will use the three local points (A1, B1, C1) to constitute a quadratic polynomial to do extrapolation at the node E. Nodes B1, C1 are the nodes inside the domain where the functional values are given by the governing equation, while point A1 is the intersection point of the mesh line and the boundary, where the functional value can be given by the boundary condition. The respective formulation for the extrapolation at node E can be written as

$$u_E = \frac{(x_E - x_{B1})(x_E - x_{C1})}{(x_{A1} - x_{B1})(x_{A1} - x_{C1})}u_{A1} + \frac{(x_E - x_{C1})(x_E - x_{A1})}{(x_{B1} - x_{C1})(x_{B1} - x_{A1})}u_{B1} + \frac{(x_E - x_{A1})(x_E - x_{B1})}{(x_{C1} - x_{A1})(x_{C1} - x_{B1})}u_{C1} \tag{3}$$

Substituting Equation (3) into Equations (1) and (2), the discrete form of the x derivatives at node B1 can be given, which only involves information at interior and boundary points.

Similarly, the discrete form of the first- and second-order derivatives in the y direction at the position B2 is,

$$\frac{\partial^2 u_{B2}}{\partial y^2} = \frac{1}{h^2}[u_{C2} - 2u_{B2} + u_E] + o(h^2) \tag{4}$$

$$\frac{\partial u_{B2}}{\partial y} = \frac{1}{2h}[u_{C2} - u_E] + o(h^2) \tag{5}$$

where

$$u_E = \frac{(y_E - y_{B2})(y_E - y_{C2})}{(y_{A2} - y_{B2})(y_{A2} - y_{C2})}u_{A2} + \frac{(y_E - y_{C2})(y_E - y_{A2})}{(y_{B2} - y_{C2})(y_{B2} - y_{A2})}u_{B2} + \frac{(y_E - y_{A2})(y_E - y_{B2})}{(y_{C2} - y_{A2})(y_{C2} - y_{B2})}u_{C2} \tag{6}$$

It should be noted that two distinct values are obtained at the node E. One is used for the discretization of x derivatives at node B1, while the other is used for the discretization of y derivatives at node B2.

2.2. Implementation of boundary conditions

From the previous section, it is known that the boundary conditions should be implemented at points A1 and A2, and the functional values calculated from the boundary conditions have to be substituted into the discrete form of the PDEs through the process of extrapolation (Equations (3) and (6)). The implementation of the boundary conditions at points A1 and A2 can influence the computation greatly. Basically, there are two types of boundary conditions:

Dirichlet type and Neumann type. The implementation of these two boundary conditions is shown below.

2.2.1. Dirichlet-type boundary condition. If the boundary conditions at points A1 and A2 are the Dirichlet type, i.e. the variable values are known at points A1 and A2, the implementation is straightforward. That is, the functional values can be substituted directly into Equations (3) and (6).

2.2.2. Neumann-type boundary condition. If the gradient in a particular direction (usually normal to the boundary) is given as the boundary condition, it may be necessary to use some close-to-boundary points to approximate the derivatives at boundary. Here, we take the point A1 as an example. Suppose that the first-order derivative is given at the boundary point A1. Then the following second-order one-sided approximation form can be used,

$$\left(\frac{\partial u}{\partial x}\right)_{\mathrm{A1}} \approx \frac{u_{\mathrm{B1}}(x_{\mathrm{C1}}-x_{\mathrm{A1}})^2 - u_{\mathrm{C1}}(x_{\mathrm{B1}}-x_{\mathrm{A1}})^2 - u_{\mathrm{A1}}[(x_{\mathrm{C1}}-x_{\mathrm{A1}})^2-(x_{\mathrm{B1}}-x_{\mathrm{A1}})^2]}{(x_{\mathrm{C1}}-x_{\mathrm{A1}})(x_{\mathrm{B1}}-x_{\mathrm{A1}})(x_{\mathrm{C1}}-x_{\mathrm{B1}})} \tag{7}$$

When zero gradient in the x direction is prescribed and the first-order approximation is adopted, we have

$$\left(\frac{\partial u}{\partial x}\right)_{\mathrm{A1}} = 0 \Rightarrow \frac{u_{\mathrm{B1}}-u_{\mathrm{A1}}}{x_{\mathrm{B1}}-x_{\mathrm{A1}}} = 0 \Rightarrow u_{\mathrm{A1}} = u_{\mathrm{B1}} \tag{8}$$

If the Neumann boundary condition at point A1 is given in the normal direction to the boundary $\bar{n}$, say $\partial u/\partial \bar{n}, \partial^2 u/\partial \bar{n}^2$, the points aa, bb, cc as shown in Figure 1, which are the intersection points of the perpendicular line through A1 and the grid lines, are used in the approximation formulae similar to Equations (7) and (8) except that the perpendicular distances $|\mathrm{aa}-\mathrm{A1}|, |\mathrm{bb}-\mathrm{A1}|, |\mathrm{bb}-\mathrm{aa}|$ instead of the distances in the x direction are used. The functional values at the points aa, bb, cc can be calculated by interpolation between nodes B1 and G (for point aa), nodes H and G (for point bb), and nodes H and F (for point cc).

2.3. Identifying the status of mesh nodes

In the local DFD method, an important step is to identify which mesh node is the interior node where the governing equation should be discretized, and which mesh node is outside of the physical domain but near the boundary where extrapolation should be applied. In other words, we should know the status of the mesh nodes. In this work, the status for interior node is represented by 0 (the solid circle in Figure 1), such as

$$\mathrm{STATUS(node\ B1)} = 0$$

and the status for the extrapolation node is represented by 1 (the open circle in Figure 1), i.e.

$$\mathrm{STATUS(node\ E)} = 1$$

In the local DFD method, the discretization of the PDE is only made at mesh nodes where $\mathrm{STATUS}(i,j) = 0$.

It is obvious that the status of mesh node should be identified before the local DFD method is applied. In this work, we propose the so-called 'odd/even parity method' inspired from the

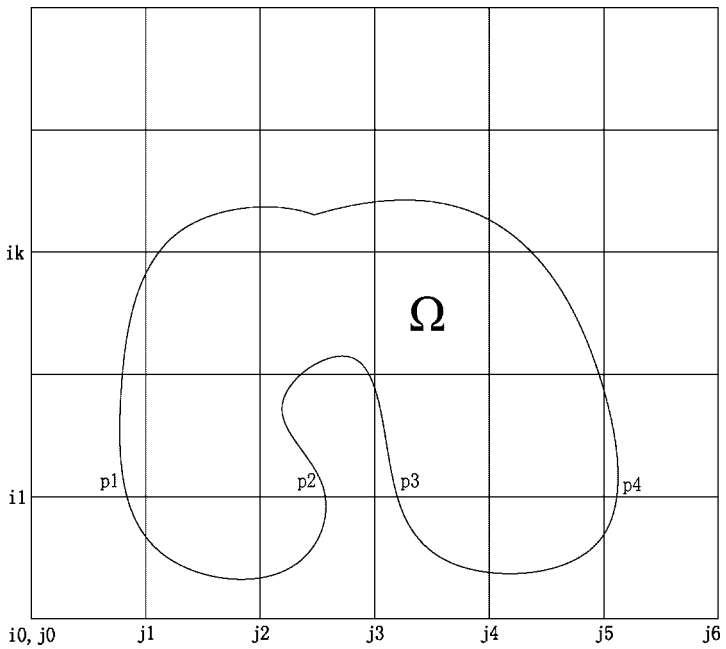

Figure 2. Illustration of 'odd/even parity method'.

scan-line polygon fill algorithm in computer graphics. The 'odd/even parity method' can be illustrated in Figure 2.

Suppose that there is a scan-line $i1$ in the horizontal direction. This line has four intersection points with solid body (denoted by Ω), which are noted as p1, p2, p3, p4. It was found that the odd index of intersection point (such as p1, p3) is always the point where the scan-line moves in the body while the even index of the intersection point (such as p2, p4) is the point where the scan-line moves out of the body. Therefore, we can number the series of intersection points in forward sequence along the scan direction, and mark the mesh nodes between every odd/even parity as STATUS $= 1$, i.e. STATUS$(i1, j1–j2) = 1$ and STATUS$(i1, j4–j5) = 1$ in Figure 2.

We found that this method can determine the status of nodes in the whole domain very quickly.

2.4. *Adaptive mesh refinement*

The adaptive stencil refinement approach proposed by Ding and Shu [16] is applied in this work to enhance the efficiency of local DFD method. Some details of the approach are described below.

2.4.1. Two types of stencil and numerical discretization. As shown in Figure 3, there are two types of stencil for a reference node A, where nodes marked $1, 2, 3, 4$ are the four supporting

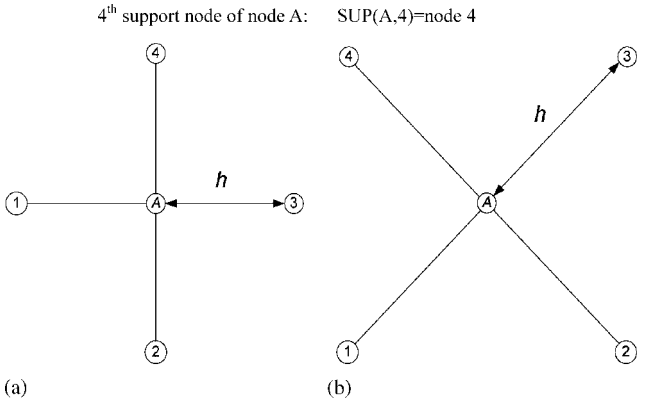

Figure 3. Two types of stencil on uniform Cartesian mesh: (a) Type I; and (b) Type II.

nodes. They are recorded as

$$\begin{aligned}
&\text{1st support node of node A:} && \text{SUP(A, 1)} = \text{node 1}\\
&\text{2nd support node of node A:} && \text{SUP(A, 2)} = \text{node 2}\\
&\text{3rd support node of node A:} && \text{SUP(A, 3)} = \text{node 3}\\
&\text{4th support node of node A:} && \text{SUP(A, 4)} = \text{node 4}
\end{aligned}$$

The four supporting nodes and node A form the stencil. Numerical discretization of the first-order derivative at node A for the two types of stencil can be written as

$$\text{Type I}: \left(\frac{\partial u}{\partial x}\right)_{\mathrm{A}} = \frac{u_3 - u_1}{2h}, \left(\frac{\partial u}{\partial y}\right)_{\mathrm{A}} = \frac{u_4 - u_2}{2h} \tag{9a}$$

$$\text{Type II}: \left(\frac{\partial u}{\partial x}\right)_{\mathrm{A}} = \frac{(u_2 + u_3) - (u_1 + u_4)}{2\sqrt{2}h}, \left(\frac{\partial u}{\partial y}\right)_{\mathrm{A}} = \frac{(u_3 + u_4) - (u_1 + u_2)}{2\sqrt{2}h} \tag{9b}$$

where h is the mesh spacing. On the other hand, the Laplacian operator on the two types of stencil (Types I and II) can be written as

$$(\nabla^2 u)_{\mathrm{A}} = \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)_{\mathrm{A}} = \frac{u_1 + u_2 + u_3 + u_4 - u_{\mathrm{A}}}{h^2} \tag{10}$$

Equation (10) is very useful in the discretization of N–S equation since its diffusive operator is Laplacian operator.

2.4.2. Stencil refinement. As shown in Figure 4, when refinement is needed, four new nodes $1', 2', 3', 4'$ are added by simply taking the midpoint of the edge $\overline{12}$, $\overline{23}$, $\overline{34}$, $\overline{41}$ for Type I, or by taking the midpoint of the edge $\overline{41}$, $\overline{12}$, $\overline{23}$, $\overline{34}$ for Type II.

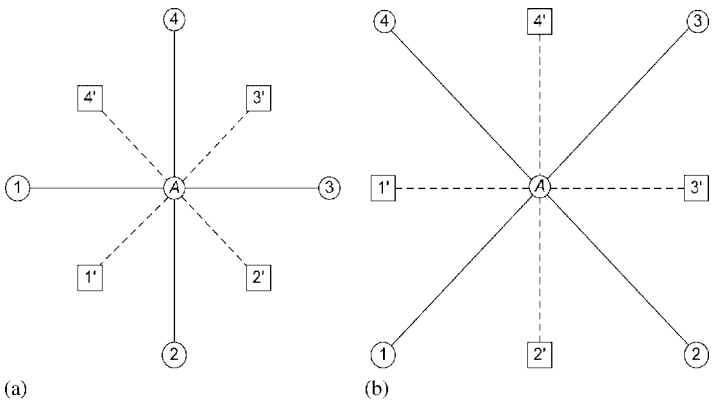

Figure 4. Transformation of two types of stencil when refinement is performed: (a) Type I $\rightarrow$ Type II; and (b) Type II $\rightarrow$ Type I.

The new supporting nodes for node A (nodes $1', 2', 3', 4'$) can be recorded as

$$\mathrm{SUP}_{\mathrm{new}}(\mathrm{A}, 1) = \text{node } 1'$$

$$\mathrm{SUP}_{\mathrm{new}}(\mathrm{A}, 2) = \text{node } 2'$$

$$\mathrm{SUP}_{\mathrm{new}}(\mathrm{A}, 3) = \text{node } 3'$$

$$\mathrm{SUP}_{\mathrm{new}}(\mathrm{A}, 4) = \text{node } 4'$$

which form another type of stencil. As shown in Figure 4, it is obvious that in the above refinement process, Type I stencil will change to Type II stencil while Type II stencil will change to Type I stencil. And for each case, the mesh spacing is reduced by the rate of $\frac{\sqrt{2}}{2}$. As the refinement is carried out at node A, level-by-level, the stencil type appears alternately.

For the new added nodes such as $1', 2', 3', 4'$, it is necessary to find their supporting nodes and determine their functional values. We found that node A and its new added nodes $1', 2', 3', 4'$ have the same stencil type. Take node $1'$ in Figure 4 as an example. Its stencil is shown in Figure 5:

The first supporting node of node $1'$ is located at node B, which can be either the second supporting node of node 1 or the first supporting node of node 2. The supporting nodes for node $1'$ can then be determined as follows:

$$\mathrm{SUP}(1', 1) = \mathrm{SUP}(1, 2) \text{ or } \mathrm{SUP}(2, 1), \text{whichever being located at node B}$$

$$\mathrm{SUP}(1', 2) = \text{node } 2$$

$$\mathrm{SUP}(1', 3) = \text{node A}$$

$$\mathrm{SUP}(1', 4) = \text{node } 1$$

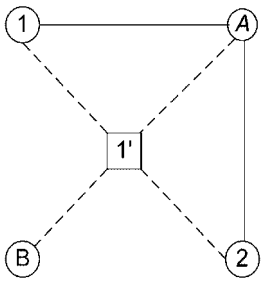

Figure 5. The stencil type of node $1'$.

The functional value at node $1'$ can be obtained by taking the average of its four supporting nodes. The supporting nodes and the functional values of nodes $2', 3', 4'$ can be determined in a similar way. After the refinement, the governing equations are discretized at the original mesh nodes and the new refined mesh nodes to get the numerical solution.

It should be noted that the discretization of the derivatives at a mesh node is always based on its support nodes. Therefore, it is necessary to record the connection between the support nodes and the mesh node at every level of refinement. This can make mesh refinement/coarsening very easy.

2.5. Solution-based mesh refinement or coarsening

In this paper, the refinement and coarsening are solution based. That is, the adaptive mesh is refined or coarsened according to the characteristics of the flow. Refinement or coarsening takes place only after a solution is sufficiently converged. In our study, if the difference of certain variable (such as temperature) between the node A and its support nodes is larger than a user-specified maximum value, the node A is flagged to be refined. On the contrary, if the difference is less than a predefined minimum value, the node A is flagged to be coarsened. It is obvious that the total number of the mesh nodes in the adaptive mesh depends on these predefined maximum and minimum values.

3. SIMULATION OF NATURAL CONVECTION IN CONCENTRIC ANNULUS BETWEEN INNER CIRCULAR CYLINDER AND OUTER SQUARE CYLINDER

Natural convection in a concentric annulus is the most commonly encountered problem in practical applications. In this work, it is solved by the AMR-enhanced local DFD method.

3.1. Governing equations and numerial discretization

The Navier–Stokes equations in the vorticity stream function formulation are taken as the governing equations for the problem, which are written as

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = \omega \tag{11}$$

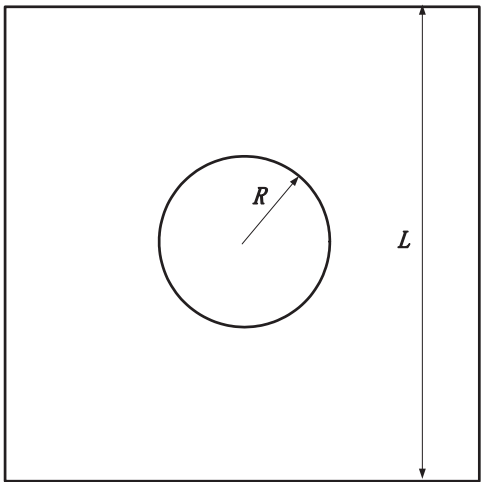

Figure 6. Schematic of the natural convection problem.

$$\frac{\partial \omega}{\partial t} + u\frac{\partial \omega}{\partial x} + v\frac{\partial \omega}{\partial y} = Pr\left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right) - Pr \cdot Ra \cdot \frac{\partial T}{\partial x} \tag{12}$$

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \tag{13}$$

where ω, ψ, T, Pr, and Ra are the vorticity, stream function, temperature, Prandtl number and Rayleigh number, u, v are the components of velocity in the x and y directions, which can be calculated by

$$u = \frac{\partial \psi}{\partial y}, \qquad v = -\frac{\partial \psi}{\partial x} \tag{14}$$

The geometry of the problem is given in Figure 6. It is a concentric annulus between a square outer cylinders and a circular inner cylinder. The reference length is taken as the side length of square L. The radius ratio is defined as $\mathrm{rr} = L/2R$, where R is the radius of the inner circular cylinder. It is clear that this is a problem with irregular geometry in the Cartesian coordinate system.

The boundary conditions on two impermeable isothermal walls are given by

$$\psi = u = v = 0, \qquad \omega = \frac{\partial^2 \psi}{\partial n^2}, \qquad T = 1 \tag{15}$$

on the inner cylinder and

$$\psi = u = v = 0, \qquad \omega = \frac{\partial^2 \psi}{\partial n^2}, \qquad T = 0 \tag{16}$$

on the outer cylinder. It should be noted that according to Equation (11), the implementation of vorticity at the boundary is actually the approximation of the second-order derivative of stream-function. As shown in Figure 1, the vorticity at the wall ω_{A1} can be given by the following second-order scheme as

$$\omega_{A1} = \frac{3}{l^2}(\psi_{aa} - \psi_{A1}) - \frac{1}{2}\omega_{aa} \tag{17}$$

where l is the perpendicular distance between aa and A1. Point aa is the first intersection point between the line through A1 along the direction of vector $\bar{n}$ and the grid line.

In the AMR-enhanced local DFD method, the second-order central difference scheme is applied in both the x and y directions to approximate the spatial derivatives. Equations (11)–(13) can be discretized by the present method at a mesh node (x_i, y_j) as

$$\frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta x^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta y^2} = \omega_{i,j} \tag{18}$$

$$\begin{aligned} &\frac{\mathrm{d}\omega_{i,j}}{\mathrm{d}t} + u_{i,j}\frac{\omega_{i+1,j} - \omega_{i-1,j}}{2\Delta x} + v_{i,j}\frac{\omega_{i,j+1} - \omega_{i,j-1}}{2\Delta y} \\ &= Pr \cdot \left(\frac{\omega_{i+1,j} - 2\omega_{i,j} + \omega_{i-1,j}}{(\Delta x)^2} + \frac{\omega_{i,j+1} - 2\omega_{i,j} + \omega_{i,j-1}}{(\Delta y)^2} \right) \\ &- Pr \cdot Ra \cdot \frac{T_{i+1,j} - T_{i-1,j}}{2\Delta x} \end{aligned} \tag{19}$$

$$\begin{aligned} &\frac{\mathrm{d}T_{i,j}}{\mathrm{d}t} + u_{i,j}\frac{T_{i+1,j} - T_{i-1,j}}{2\Delta x} + v_{i,j}\frac{T_{i,j+1} - T_{i,j-1}}{2\Delta y} \\ &= \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{(\Delta x)^2} + \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{(\Delta y)^2} \end{aligned} \tag{20}$$

In Equations (18)–(20), if one or more of nodes $(i-1,j),(i+1,j),(i,j-1),(i,j+1)$ (i.e. the supporting nodes of mesh node (x_i, y_j)) are not located in the physical domain, (STATUS = 1), then the functional values at these nodes can be evaluated by using Equations (3) and (6). The time derivatives in Equations (18)–(20) are approximated by Euler implicit scheme. The resultant algebraic equations are then solved by SOR method.

3.2. Numerical results and discussion

At first, the present method is validated by its application to simulate natural convection in the concentric annulus between inner circular and outer square cylinders. In this study, the Prandtl number is fixed at 0.71. The cases with the aspect ratios of $\mathrm{rr} = 1.67, 2.5, 5.0$ and Rayleigh numbers of $10^4, 10^5$, and 10^6 were studied. The mesh size of uniform mesh adopted in our computation is 71×71.

Table I. Comparison of results for natural convection in concentric annuli between an inner circular cylinder and an outer square cylinder ($Pr = 0.71$).

		ψ_{max}			$\bar{N}u_i$		
Ra	rr	Present work	Reference [18]	Reference [17]	Present work	Reference [18]	Reference [17]
	5.0	1.76	1.71	1.73	2.065	2.082	2.071
10^4	2.5	1.01	0.97	1.02	3.222	3.245	3.331
	1.67	0.50	0.49	0.50	5.353	5.395	5.826
	5.0	10.10	9.93	10.15	3.781	3.786	3.825
10^5	2.5	8.31	8.10	8.38	4.902	4.861	5.080
	1.67	5.08	5.10	5.10	6.184	6.214	6.212
	5.0	20.93	20.98	25.35	6.362	6.106	6.107
10^6	2.5	24.44	24.13	24.07	9.181	8.898	9.374
	1.67	19.86	20.46	21.30	12.30	12.00	11.62

The local heat transfer rate on the inner cylinder can be computed by

$$q = h(T_i - T_o) = -k\frac{\partial T}{\partial n} \tag{21}$$

where h represents the local heat transfer coefficient, k is the thermal diffusivity. Because $T_i - T_o = 1$, from Equation (21), we can obtain

$$h = -k\frac{\partial T}{\partial n} \tag{22}$$

Then the average heat transfer coefficient $\bar{h}$ can be computed as

$$\bar{h} = \frac{1}{2\pi}\int_0^{2\pi} h\,\mathrm{d}\theta \tag{23}$$

And the average Nusselt numbers for the outer and inner boundaries are, respectively, determined by

$$\bar{N}u_i = \frac{\bar{h}_i S_i}{k}, \bar{N}u_o = \frac{\bar{h}_o S_o}{k} \tag{24}$$

where S_i and S_o are defined in the same way as in the work of Moukalled and Acharya [17]. In their work, the computational domain was taken as half of the physical domain due to the symmetry, so S_i and S_o are taken as half of the circumferential lengths of the inner and outer cylinder surfaces, respectively. Since at steady state, the Nusselt numbers along the inner and outer walls are the same, there is no need to pay separate attention to $\bar{N}u_i$ and $\bar{N}u_o$. Thus in this study, we only show the value of $\bar{N}u_i$.

The maximum stream function value ψ_{max} and the average Nusselt number $\bar{N}u_i$ obtained by the present method are compared with those of Moukalled and Acharya [17] as well as those of Shu *et al.* [18] in Table I. It should be noted that due to different ways of non-dimensionalization between the work of Moukalled and Acharya [17] and the present study,

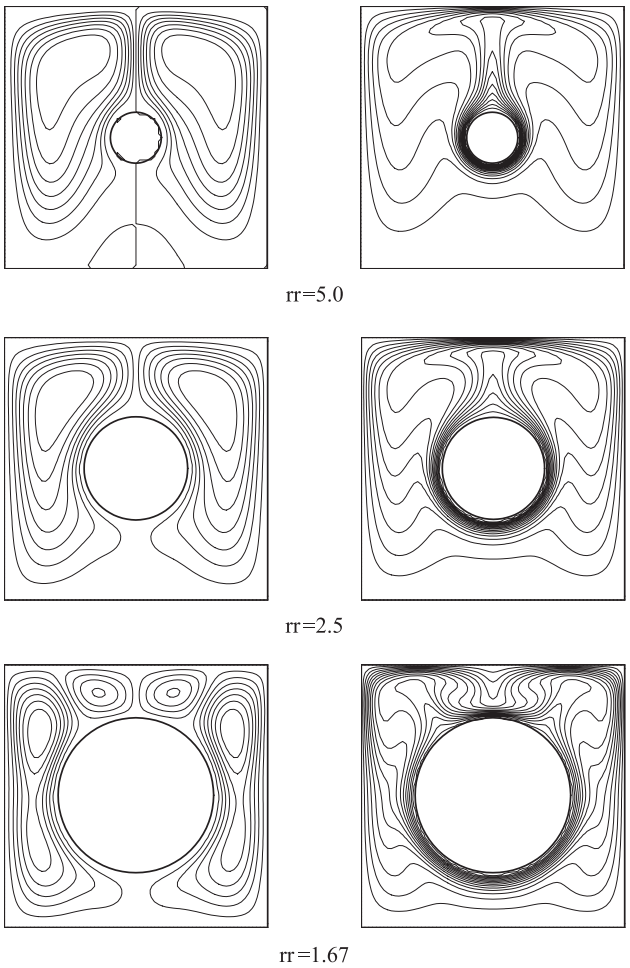

Figure 7. Streamlines and isotherms in concentric annulus between an inner circular cylinder and an outer square cylinder ($Ra = 10^6; Pr = 0.71; \mathrm{rr} = 5.0, 2.5, 1.67$).

the equivalent $\psi_{\max}$ in Table I is the one given from Moukalled and Acharya [17] multiplying by the Prandtl number. It is noted that the reference length used in the Rayleigh number is the side length of square L. From Table I, it can be seen that the present results agree very well with available data in the literature. The streamlines and isotherms for $Ra = 10^6$ are shown in Figure 7.

To study the efficiency of the AMR-enhanced local DFD method, cases with aspect ratio of $\mathrm{rr} = 2.5$ and Rayleigh numbers of $10^4, 10^5$ and 10^6 are considered. In this paper, the temperature is adopted as the indicator of the solution-based mesh refinement/coarsening. In the study, the maximum and minimum values of the temperature difference between a certain node and its support nodes $\Delta T_{\max}, \Delta T_{\min}$, which are used as the flag to implement refinement/coarsening, are given as $\Delta T_{\max} = 0.4,\ \Delta T_{\min} = 0.03$.

Table II. Comparison of the number of nodes and running time needed to achieve the similar accuracy by the DFD-Cartesian mesh solver with and without adaptive refinement ($Ra = 10^4, 10^5, 10^6$; $Pr = 0.71$; rr $= 2.5$).

Ra	Method	No. of nodes (before refinement)	No. of nodes (after refinement)	Running time	$\psi_{\max}$	$\bar{N}u_i$
10^4	Method 1	$13 \times 13 = 289$	383	0.828	1.05	3.327
	Method 2	$35 \times 35 = 1225$	1225	3.016	1.05	3.368
	Reference [17]	—	—	—	1.02	3.331
10^5	Method 1	$21 \times 21 = 441$	981	5.422	8.35	5.101
	Method 2	$65 \times 65 = 4225$	4225	79.67	8.29	4.904
	Reference [17]	—	—	—	8.38	5.080
10^6	Method 1	$41 \times 41 = 1681$	2379	42.25	24.74	9.321
	Method 2	$71 \times 71 = 5041$	5041	117.41	24.44	9.181
	Reference [17]	—	—	—	24.07	9.374

Method 1: DFD-Cartesian mesh solver with AMR; Method 2: DFD-Cartesian mesh solver without AMR.

It is found that, to achieve the acceptable accuracy, the mesh must be fine enough to capture geometric features of the complex boundary as well as the gradient of the variables. In fact, the mesh refinement is mainly needed in the region near the boundary to capture the thin boundary layer. In other regions, relatively coarse mesh can be used. The combination of fine and coarse meshes can greatly save the computational effort while keeping the accuracy of solution. Fortunately, this can be made by using the AMR-enhanced local DFD method.

Table II compares the efficiency and accuracy of local DFD method with and without AMR, where Method 1 denotes the DFD-Cartesian mesh solver with adaptive refinement and Method 2 denotes the solver without adaptive refinement. It should be noted that the relaxation factors of SOR iteration for ψ, ω, T and time interval Δt in the Euler implicit scheme are fine-tuned to their optimal values for both methods.

It can be seen from Table II that, as Rayleigh number increases, more and more mesh points are needed to keep accuracy of numerical solution. For any case, to reach the same order of accuracy, the final number of field nodes needed by Method 1 is far less than that of Method 2, and the running time for Method 1 is also less than Method 2. This indicates that the efficiency of the DFD-Cartesian mesh method is improved greatly by implementing the adaptive refinement technique. Figure 8 shows the adaptive mesh according to the temperature field. Clearly, very fine mesh points are distributed in the region where the temperature gradient is high.

4. CONCLUSIONS

In this paper, the AMR-enhanced local DFD method is presented and applied to solve natural convection in concentric annulus between an inner circular cylinder and an outer square cylinder. The obtained numerical results compare well with available data in the literature. It has been demonstrated that with present method, the body-fitted grid generation and coordinate

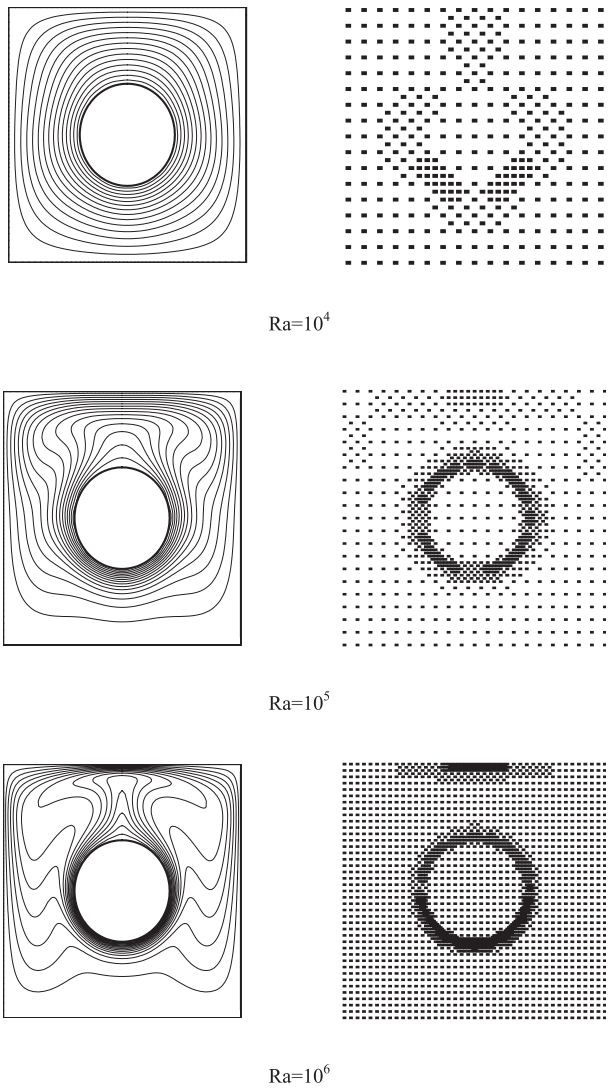

Figure 8. Isotherms and adaptive refined meshes based on temperature solution ($Ra = 10^4, 10^5, 10^6$; $Pr = 0.71$; rr = 2.5).

transformation, which are necessary for the traditional numerical methods such as FD and DQ method for the problem with irregular geometry, are totally avoided.

With the AMR technique introduced in this paper, the local DFD method can make the numerical computation much more effective than its original version. Good efficiency and accuracy indicate that the present method with the AMR has potential to solve practical flow problems.

REFERENCES

1. Hildebrand FB. *Finite-difference Equations and Simulations*. Prentice-Hall: New York, 1968.
2. Ozisik MN. *Finite Difference Methods in Heat Transfer*. CRC Press: Boca Raton, FL, 1994.
3. Thomas JW. *Numerical Partial Differential Equations*: *Finite Difference Methods*. Springer: Berlin, 1995.
4. Bellman R, Kashef BG, Casti J. Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations. *Journal of Computational Physics* 1972; **10**:40–52.
5. Shu C. *Differential Quadrature and its Application in Engineering*. Springer: London, 2000.
6. Shu C, Richards BE. Application of generalised differential quadrature to solve two-dimension incompressible Navier–Stokes equations. *International Journal for Numerical Methods in Fluids* 1992; **15**:791–798.
7. Du H, Lim MK, Lin RM. Application of generalized differential quadrature method to structural problems. *International Journal for Numerical Methods in Engineering* 1994; **37**:1881–1896.
8. Bert CW, Malik M. Differential quadrature method in computational mechanics: a review. *Applied Mechanics Review* 1996; **49**:1–28.
9. Shu C, Fan LF. A new discretization method and its application to solve incompressible Navier–Stokes equation. *Computational Mechanics* 2001; **27**:292–301.
10. Shu C, Wu YL. Domain-free discretization method for doubly connected domain and its application to simulate natural convection in eccentric annuli. *Computer Methods in Applied Mechanics and Engineering* 2002; **191**:1827–1841.
11. Berger MJ, Oliger J. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics* 1984; **53**:484–512.
12. Berger MJ, LeVeque R. Adaptive mesh refinement for two-dimensional hyperbolic systems and the AMRCLAW software. *SIAM Journal on Numerical Analysis* 1998; **35**:2298–2316.
13. Khokhlov AM. Fully threaded tree algorithms for adaptive refinement fluid dynamics simulations. *Journal of Computational Physics* 1998; **143**:519–543.
14. Durbin PA, Iaccarino G. An approach to local refinement of structured grids. *Journal of Computational Physics* 2002; **181**:639–653.
15. Pember RB, Bell JB, Colella P, Curtchfield WY, Welcome ML. An adaptive Cartesian grid method for unsteady compressible flow in irregular regions. *Journal of Computational Physics* 1995; **120**:278–304.
16. Ding H, Shu C. A stencil adaptive algorithm for finite difference solution of incompressible viscous flows. *Journal of Computational Physics*, 2005, in press.
17. Moukalled F, Acharya S. Natural convection in the annulus between concentric horizontal circular and square cylinders. *Journal of Thermophysics and Heat Transfer* 1996; **10**:524–531.
18. Shu C, Xue H, Zhu YD. Numerical study of natural convection in an eccentric annulus between a square outer cylinder and a circular inner cylinder using DQ method. *International Journal of Heat and Mass Transfer* 2001; **44**:3321–3333.

