Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 214 (2006) 397–420

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A stencil adaptive algorithm for finite difference solution of incompressible viscous flows

H. Ding, C. Shu *

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

Received 1 October 2004; received in revised form 5 September 2005; accepted 26 September 2005
Available online 8 November 2005

Abstract

In this paper, a solution-adaptive algorithm is presented for the simulation of incompressible viscous flows. The framework of this method consists of an adaptive local stencil refinement algorithm and 3-points central difference discretization. The adaptive local stencil refinement is designed in such a manner that 5-points symmetric stencil is guaranteed at each interior node, so that conventional finite difference formula can be easily constructed everywhere in the domain. Thus, high efficiency and accuracy of central difference scheme can be ultimately enjoyed together with the solution-adaptive property. The adaptive finite difference method has been tested by three numerical examples, to examine its performance in the two-dimensional problems. The numerical examples include Poisson equation, moving interface problem and a lid-driven incompressible flow problem. It was found that the multigrid approach can be efficiently combined with solution-adaptive algorithm to speed up the convergence rate.
© 2005 Elsevier Inc. All rights reserved.

Keywords: Adaptive mesh refinement; Finite difference; Solution-adaptive; Multigrid

1. Introduction

For the numerical simulation of many practical problems in physics and engineering, it is often equivalent to solve a set of partial differential equations (PDEs), which represent the mathematical model of physical problems concerned. How to efficiently and effectively solve the PDEs is always a subject of active research in numerical analysis. In general, there are two approaches to obtain accurate solution of PDEs. One approach is to employ high-order numerical methods such as differential quadrature method (DQM) [1–3]. High-order methods can achieve a given accuracy of solution with much coarser mesh than low-order schemes. Thus, despite their larger bandwidth of the formed stiff matrix, high-order methods often prove to be more computationally efficient than low-order schemes. However, they can only yield accurate results for length scales that are larger than a few mesh cells. For problems with locally high gradient, they may

* Corresponding author. Tel.: +65 6874 6476; fax: +65 6779 1459.
E-mail address: mpeshuc@nus.edu.sg (C. Shu).

not be able to accurately capture the locally fine-scale behavior of the physical phenomenon. Therefore, the high-order methods are usually preferred for problems with smooth solution. For many types of practical problems, there exists a limit beyond which it is difficult to further improve the accuracy of the solution. Instead, we have to resort to the second approach – improving the resolution through the computational grid. Mesh refinement is desirable to improve spatial resolution. However, the uniform mesh refinement is not perfect for the applications, of which the solution may need different resolutions for different regions. That implies the waste of computational efforts. For the well-understood physical problems, a non-uniform mesh can be designed to reflect the resolution requirement of practical problems. For example, for the boundary value problems, fine resolution is typically required for regions near boundaries. But for the evolving interfacial flows, blow-up or flow field with complicated structures, *adaptive mesh refinement* (AMR) techniques are more preferred to locally increase mesh densities in the regions of interest, thus saving the computer resources. Nowadays, mesh adaptation in general plays an indispensable role in the efficient solution of industrial and scientific problems.

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 [4–18]. They all allow us to use multi-resolution approximation in the simulation, and represent the recent advances in this area. The strategies of adaptive mesh refinement can fall into two categories from the viewpoint of way of multi-resolution fulfilled. The first category includes these adaptive algorithms involved local mesh/stencil refinement. In these algorithms, either the existing mesh is split into several smaller cells or additional nodes are inserted locally, thus to obtain the *h*-refinement. This group of adaptive algorithm can be further categorized by the mesh type, i.e., hierarchical structured grid approach and unstructured mesh refinement approach. One representative of structured grid approaches is adaptive Cartesian mesh refinement proposed by Berger et al. [4,5]. 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. With regard to the unstructured mesh refinement approach, Zienkiewicz reviewed the state-of-the-art of the automatic mesh refinement strategies in the finite element community in [7], and discussed the important role of error estimation and automatic adaptation in the finite element analysis. Up to date, this group of locally adaptive refinement approaches has been widely applied to the numerical simulations in many areas, such as the compressible flow [4,5,10], incompressible flow [12–14], porous flow [19] and flow–structure interaction [20]. Unlike the local refinement algorithm, the second category of adaptive algorithms involves global mesh-redistribution. These methods move the mesh points inside the domain in order to better capture the dynamic changes of solution. Therefore, such techniques are usually referred as *moving mesh method* or *r*-refinement. This group of adaptive methods is less popular than the first group. However, it can offer some distinct features. For example, they do not need to delete/insert nodes to coarsen/refine the local mesh. The practitioners also do not need to construct and maintain a hierarchical mesh structure. Applications of the moving mesh method have been extended to many challenging problems, such as the thin flame propagating [21], drop formation [22], non-breaking free surface wave [23].

Our goal is to design a locally solution-based adaptive strategy for the finite difference method, to enhance its ability in the multi-scale incompressible viscous flow applications. The emphasis in the design is on simplicity and efficiency, which are highly appreciated in solving the practical engineering problems. These properties are also regarded as the key points for a successful adaptive algorithm. The framework of our new adaptive algorithm is based on the standard 5-point stencil and its local refinement. It is well known that the central difference scheme constructed on the 5-point stencil is a very efficient discretization method. It can achieve the second-order accuracy for the first- and second- order derivatives with least computational efforts. From the viewpoint of operation counts versus accuracy, this approximation can be considered as the most efficient one. The key idea of our adaptive stencil algorithm is to build up an adaptive hierarchy of symmetric 5-point stencils in the domain, so that the central difference can be constructed at every interior node. It is also noteworthy that the generation of the 5-point stencil for the newly inserted node is quite simple and straightforward. Moreover, at the fine/coarse stencil interfaces, the information between the nodes at different resolution levels is naturally exchanged. Therefore, it does not require any special treatment in these regions, and consequently reduces the complexity of the adaptive algorithm. We believe that the efficiency of the central difference method and simplicity of the stencil adaptation described in this paper could be a significant advantage when dealing with viscous incompressible flow problems. In this paper, the solution-adaptive finite difference scheme is validated by three two-dimensional numerical examples. They are the Poisson equation, moving interface problem and a lid-driven viscous incompressible flow problem. The numerical results show that there are significant savings of CPU running time as compared with that required by the fixed grid approach.

The paper is organized as follows. Section 2 describes the stencil adaptive algorithm in detail. Section 3 provides the spatial discretization for the differential operators encountered in the viscous incompressible flows on the two configurations of 5-point stencil. Section 4 examines the performance of present stencil adaptive algorithm by three numerical experiments. Section 5 provides a summary of this paper.

2. Stencil adaptive algorithm

In the present section, we address the stencil adaptive algorithm in a step-by-step manner so that the potential practitioners can easily understand the procedures to realize the stencil adaptive algorithm. Otherwise mentioned, all the descriptions about the stencil adaptive algorithm are only referred to stencil for the interior nodes. In this paper, we restrict ourselves to the algorithm developed for the two dimensions.

2.1. Two types of stencils and data structure

As indicated in the previous section, our new adaptive algorithm is based on the local stencil refining and coarsening. It indicates that for any interior node in the domain there is a local stencil associated with it. For the convenience of inserting and deleting nodes from the adaptive stencils, only one index is used to identify the node in the domain, i.e., global nodal index. In general, there are two types of stencils encountered in this adaptive algorithm, and they are shown in Fig. 1. The two stencils are characterized by their configurations and stencil size h (the distance between the reference node and its member node), which denotes the spatial resolution of the stencil. It is noteworthy that both of them are the 5-points symmetric stencil, so that the finite difference method can be easily implemented to approximate the derivatives at the reference node. For an arbitrary reference node i, its stencil can be symbolized as i_n^m and the positions of the nodes in the stencil are denoted by $\mathbf{x}_n^m$, where the superscript m denotes the resolution level, and the subscript $n = 0, 1, \ldots, 4$ denotes the local index of the member nodes in the stencil. Note that the local index 0 always points to the reference node itself. The symbol i_n^m can also be regarded as a reference which points to the global nodal index of the nodes in the stencil. For example, i_1^0 yields the global nodal index of the 1st member node in the resolution level 0 stencil. The structure of the adaptive stencil hierarchy and the associated data are managed in one-dimensional arrays. At one reference node, the information of its stencil (equivalent to the element in the finite element method) is stored by an array of references to its member nodes. This array provides the information of node-to-node connectivity and then allows straightforward stencil refinement and coarsening.

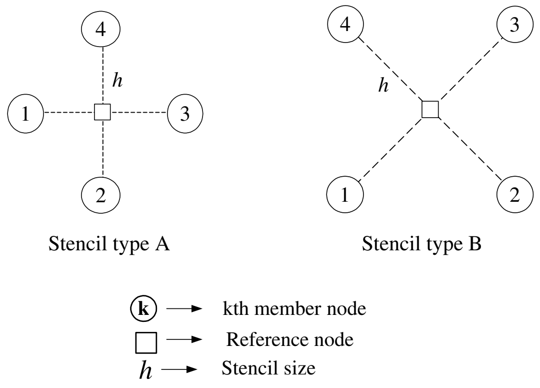


Fig. 1. Two stencil configurations.

2.2. Stencil refinement algorithm

From the viewpoint of implementation, our stencil refinement algorithm is quite simple. In brief, to locally refine a stencil for the reference node, we insert four new nodes, which are located at the midpoints of the stencil edges. These four newly inserted nodes form a refined stencil for the reference node with a refinement ratio of $\sqrt{2}$. Then, we construct the stencils for the newly inserted nodes by the existent nodes to complete this stencil-refinement process. The details are shown as follows.

2.2.1. Local stencil refinement of even resolution level

Initially, we consider a background Cartesian grid with mesh spacing as h in both the x and y directions. It is easy for an arbitrary interior node i to construct the stencil of "A" configuration, in which the four member nodes are located at $(x_i - h, y_i)$, $(x_i, y_i - h)$, $(x_i + h, y_i)$ and $(x_i, y_i + h)$ as shown in Fig. 2. If the solution at node i is considered to have an abrupt change, it is desirable to refine the stencil of node i from resolution level 0 (i_n^0) to resolution level 1 (i_n^1). Stencil refinement is achieved by the injection of extra grid points in the old stencil region to form a new stencil. In this case four new nodes are inserted. The four newly generated nodes are located at $(x_i - \frac{h}{2}, y_i - \frac{h}{2})$, $(x_i + \frac{h}{2}, y_i - \frac{h}{2})$, $(x_i - \frac{h}{2}, y_i + \frac{h}{2})$ and $(x_i + \frac{h}{2}, y_i + \frac{h}{2})$, respectively. As shown in Fig. 3, their positions are actually the midpoints of the stencil edges. More specifically, they yield

$$\mathbf{x}_1^1 = \frac{\mathbf{x}_1^0 + \mathbf{x}_2^0}{2}, \quad \mathbf{x}_2^1 = \frac{\mathbf{x}_2^0 + \mathbf{x}_3^0}{2}, \quad \mathbf{x}_3^1 = \frac{\mathbf{x}_3^0 + \mathbf{x}_4^0}{2} \quad \text{and} \quad \mathbf{x}_4^1 = \frac{\mathbf{x}_4^0 + \mathbf{x}_1^0}{2}. \tag{1}$$

It can be clearly seen that after the refinement, the reference node still keeps at the center of the stencil. Furthermore, the stencil for the node i naturally evolves from the "A" configuration of i_n^0 into "B" configuration of i_n^1.

It is very interesting to observe the stencils for the newly added nodes. We take the member node i_4^1 of node i as an example. As shown in Fig. 4, the node k represents the member node i_4^1 to avoid unnecessary confusion of expression. We can see that the stencil for node k falls into type "B" category, and it also has the same

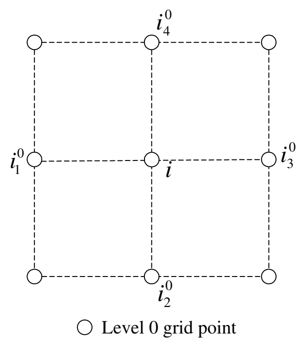


Fig. 2. Configuration of an initial stencil.

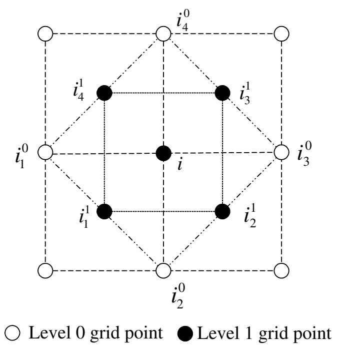


Fig. 3. Stencil refinement for the reference node from resolution level 0 to 1.

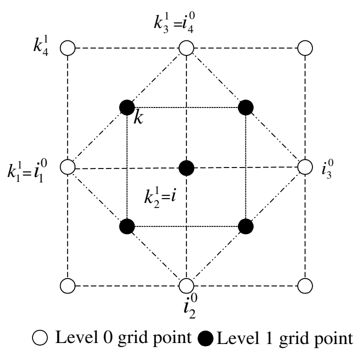


Fig. 4. Stencil construction for the newly inserted node at resolution level 1.

stencil size as the refined stencil of the node i at resolution level 1. In other words, the node k generated during the level 1 adaptation also possesses the stencil of that resolution level. This is also the case for the other newly inserted points. Actually, this is one of fundamental rules in our adaptive algorithm:

The resolution level of a new node is what it originates from.

This rule will be confirmed in the analysis of further stencil refinement.

Though the member nodes in the stencil for the node k are quite obvious from observation of Fig. 4, detailed references such as global nodal index are explicitly required and stored to maintain the stencil adaptive hierarchy. Fortunately, it is not difficult due to our stencil design. Firstly, it can be seen that three of them can be immediately determined since they are actually among the old stencil i_n^0 for the node i. The only exception is the member node k_4^1. With the fact that the reference node k represents the member node i_4^1, we find that the exceptional member node has the same local index in the stencil as the node k in the stencil i_n^1. It is emphasized that this is not a lucky coincidence. It takes place during the stencil construction for all other newly inserted nodes. This finding provides the hint to determine which member node should be paid more attention than the others. This will be of great help in the programming. In order to obtain the global nodal index of the member node k_4^1, we notice that it is also located in the stencil for the nodes i_1^0 and i_4^0 of resolution level 0. Therefore, it can be accessed by either $(i_1^0)_4^0$ or $(i_4^0)_1^0$. For the other newly inserted nodes, the stencil information can be constructed in a similar manner.

In the above demonstration, we assume that all the member nodes in the stencil i_n^0 are in the same resolution level as the reference node i. This is not true in practical cases in which some of the member nodes may have stencils of finer resolution level. However, this fact does not make our work more onerous. On the contrary, it relieves us from inserting some new nodes. For example, if the node i_1^0 has the stencil of finer level, it implies that the member nodes i_1^1 and i_4^1 in the stencil i_n^1 have been generated already during the stencil refinement of node i_1^0 from the resolution level 0 to 1. Thus, we do not need to care their position, global nodal index, or memory allocation for them. What we need to do is to access them from the side of node i_1^0, then find their global nodal index, and finally save them in the refined stencil for node i. Obviously, this condition is applicable for the stencil refinement of all resolution levels.

This example of stencil refinement from resolution level 0 to 1 for the reference node i indicates the main procedures encountered in the stencil refinement. Some necessary constraints and rules in the stencil adaptive algorithm will be introduced in the next section.

2.2.2. Local stencil refinement of odd resolution level

If further stencil refinement at node i is determined by the local resolution requirement, the stencil resolution level of node i needs to advance from present level 1 i_n^1 to level 2 i_n^2. As a consequence, four more nodes are inserted in the domain. From the viewpoint of position, they coincide to the midpoints of the stencil edges of level 1 stencil, or precisely, they yield

$$\mathbf{x}_1^2 = \frac{\mathbf{x}_4^1 + \mathbf{x}_1^1}{2}, \quad \mathbf{x}_2^2 = \frac{\mathbf{x}_1^1 + \mathbf{x}_2^1}{2}, \quad \mathbf{x}_3^2 = \frac{\mathbf{x}_2^1 + \mathbf{x}_3^1}{2} \quad \text{and} \quad \mathbf{x}_4^2 = \frac{\mathbf{x}_3^1 + \mathbf{x}_4^1}{2}. \tag{2}$$

It can be clearly seen from Fig. 5 that the stencil of level 2 has the "A" configuration. We also observe that the newly added nodes also possess level 2 stencils. This confirms the rule about stencil of inserted node in the previous section.

If we follow the same procedure and carry on the adaptation, another fundamental rule of our adaptive algorithm will be found:

The stencil of even resolution level has the "A" configuration while the stencil of odd resolution level has the "B" configuration. As the adaptation continues, the two types of stencil appear alternatively.

It should be pointed out that during the stencil refinement, only the stencils of reference node and its stencil points are affected or changed. The stencils of other nodes in the domain remain the same.

To simplify the adaptation procedure, one constraint is introduced and checked before the process of stencil refinement. That is,

- Constraint 1: The resolution levels of the stencils for the member node cannot be coarser than that of the target stencil for the reference node.

This constraint is illustrated in Fig. 6 in which the local refinement operation will not be carried out even if the local solution asks for a stencil refinement at the node k. The reason lies in that, in its present stencil, k_1^1, k_3^1 and k_4^1 are in the lower resolution level as compared to node k. This constraint guarantees that the newly generated nodes can find its stencil in the existent nodes. Note that this constraint does not affect the member node which is located on the boundary. If one prefers more smooth variation of local density of grid points, another constraint can be introduced as a preliminary condition for stencil refinement. That is,

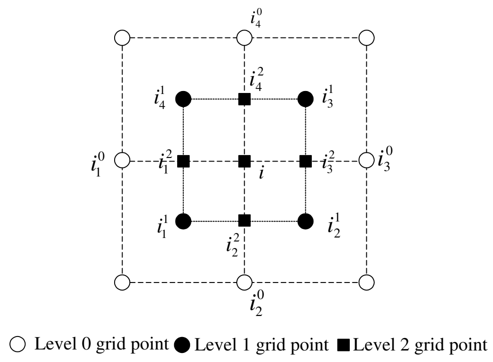


Fig. 5. Stencil refinement for the reference node from resolution level 1 to 2.

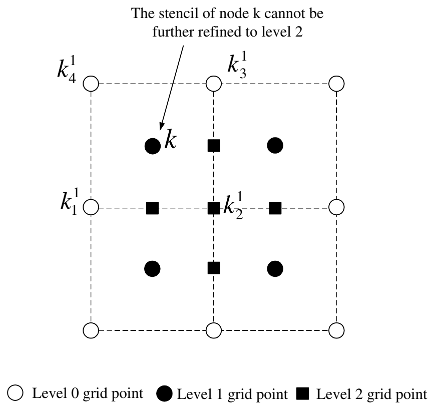

The stencil of node k cannot be further refined to level 2
k_4^1
k_3^1
k
k_1^1
k_2^1
○ Level 0 grid point ● Level 1 grid point ■ Level 2 grid point

Fig. 6. Illustration of adaptation constraint 1.

- Constraint 2: The maximum difference of resolution level in one stencil cannot be greater than a certain value, for example, one or two.

This constraint can avoid too rapid variation of local node density. For example, without violating the constraint 1 the stencil for the node *i* in Fig. 5 can be refined continuously, though the stencils of other nodes remain the coarsest resolution level. It will generate a set of clustered nodes around the node *i*, which may not be appreciated. The drawback of this constraint is that more nodes will be inserted into the domain though some of them may not be necessary, and therefore computational efficiency is impaired. In consideration of this, we treat this constraint as an optimal option.

The stencil construction for the newly inserted nodes of even resolution level has a very similar situation as that discussed in the stencil refinement from resolution level 0 to 1, i.e., three immediate member nodes and one undetermined. The determination of the exceptional member node also follows the same accessing strategy described in the previous section.

2.2.3. Stencil refinement around the boundaries

In general, local stencils can also be constructed for the nodes on the boundaries. Due to the fact that only boundary conditions are discretized at these nodes, we do not need to maintain a complete 5-points stencil. Instead, the stencils at the boundary node are maintained so as to sufficiently discretize the boundary conditions (especially for Neumann boundary conditions since Dirichlet boundary conditions do not need discretization). It is emphasized that all the stencil refinement processes in this study are only carried out on the stencils for the interior nodes despite of the existence of the stencils for the boundary nodes. Then, one may ask how the present adaptive algorithm inserts new nodes on the boundaries if the stencils at the boundaries are not refined. The answer is that the stencil adaptive algorithm can automatically insert nodes on the boundaries if such a request occurs. This can be shown by the illustration of the stencil refinement near the boundaries in Fig. 7. It can be seen in Fig. 7 that the stencil refinement for the interior node k will insert a boundary node k_1^2 in the domain. That is, the insertion of new boundary nodes is performed when the stencils for interior node near the boundary are refined. Consequently, another question may arise – how we know that the newly inserted node k_1^2 is a boundary node and which boundary it locates at. The answer to this question is related to the position determination of the node k_1^2. From Eq. (2), we know that it is determined by the position of nodes which are located on the same boundary as k_1^2. Then, another rule for the adaptive algorithm can be derived from this observation:

A newly inserted node is identified as a boundary node if and only if the two nodes it stems from are located on the same boundary. The boundary node is also located on the same boundary.

2.3. Local stencil coarsening

The use of adaptive mesh refinement in the numerical simulations is motivated by that the node density should reflect the features of the solutions. Thus, fine grids are only restricted to those regions where fine

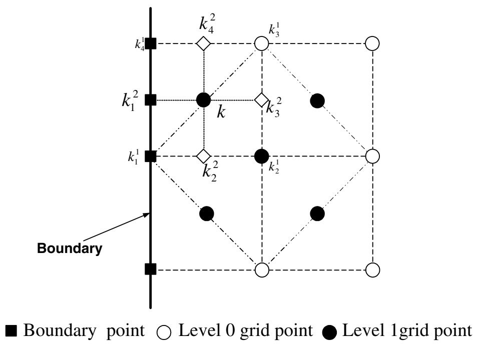


Fig. 7. Stencil refinement around the boundary.

resolution is needed. To track a dynamic problem in which the solution evolves in time, only using stencil refinement is far from sufficient to produce a satisfactory node distribution. In this regard, it is very essential to use stencil coarsening since it can remove the unnecessary fine stencil from the domain and help to achieve maximum computational efficiency.

If a node is determined to carry out stencil coarsening, its stencil will be recovered to the configuration of one level coarser. As compared with the stencil refinement, the process of stencil coarsening is much easier since the adaptation information at every interior node has been recorded during previous stencil refinement. The stencil coarsening is also restricted by some constraints. They are listed as follows:

- Constraint 3: When resolution level of the target stencil reaches the original level where the reference node is generated, the stencil coarsening stops.
- Constraint 4: The resolution levels of the stencils for the member nodes cannot be finer than the resolution level of the target stencil.

The constraint 3 is an obvious one, and the constraint 4 is helpful in avoiding too rapid variation of local node density.

2.4. Life of a node

For a dynamically adaptive grid, there are a lot of changes at the nodes and stencils in response to the solution variation. The process of stencil refinement introduces new nodes into the domain while stencil coarsening removes the fine stencils from the domain. It should be noted that stencil removing does not mean the deletion of nodes. A node is deleted from the domain only when it is considered "dead" or it is not used by any stencil except itself. It is difficult to determine whether a node is still useful or "living" since it is possibly shared by many stencils at the same time. To solve this problem, a concept of "life of node" is introduced. The life of a node equals to the number of stencils which have it as a member node (not the reference node). Thus, a "living" node is a node which still exists in at least one stencil other than the one attached to itself. During the computation, we only solve the partial differential equations at the "living" nodes. The "dead" nodes will be deleted from the sequence of existent nodes and the memory occupied is also released then. For the stencil refinement and coarsening, the lives of the related nodes are computed as following:

- Stencil refinement: The lives of all the member nodes in the coarse stencil will be subtracted by 1. At the same time, the lives of all the member nodes in the refined stencil nodes are added by 1.
- Stencil coarsening: The lives of all the member nodes in the fine stencil will be subtracted by 1. At the same time, the lives of all the member nodes in the coarsened stencil are added by 1.
- Node deletion: Once a node is "dead" (life equals zero) and deleted consequently, the lives of all the member nodes in its stencil are subtracted by 1.

From the definition above, we can see that the life of a newly inserted node should be initialized by 1. The nodes on background or the coarsest grid will never be "dead" till the end of the computation. They are "eternal" in this sense.

2.5. Action indicator

In this stencil adaptive algorithm, an action indicator is adopted to monitor and simultaneously control the resolution levels of the solution. The role it plays is equivalent to the "error indicator" in the other AMR algorithms, i.e., it monitors the variation of the parameter of interest, and sends the commands to perform the corresponding process. This parameter of interest in the viscous incompressible flow can be pressure or vorticity. In this work two monitor parameters are used to measure the local variation of the solution. They are defined as follows:

1. Absolute difference, which is defined by $\mathit{\Delta}_1 = \max(u_i) - \min(u_i)$.
2. Relative difference, which is defined by $\mathit{\Delta}_2 = \frac{\max(u_i)-\min(u_i)}{\max|u_i|}$,

where u is the parameter of interest, and the subscript $i = 0, 1, \ldots, 4$ denotes the local index of nodes in one stencil.

During the computation, the monitor parameter will be repeatedly computed at every interior node. Generally, there is no "perfect" monitor parameter. It usually depends on the practical purpose of the practitioners, and the difference in interests may lead to respective monitor parameters. The action indicator takes the monitor parameter of the local stencil as a critical parameter for the decision whether a stencil refinement or coarsening is needed. The action indicator is usually constructed by constraints and thresholds. In this study there are two thresholds in the action indicator. One represents the upper bound $\theta_{\max}$, and the other represents the lower bound $\theta_{\min}$. For the monitor parameter at the interior nodes, its value must satisfy

$$\theta_{\min} \quad \Delta \quad \theta_{\max}. \tag{3}$$

Otherwise, a stencil refinement command will be sent from action indicator if $\Delta > \theta_{\max}$ unless the finest resolution level allowed is achieved or the target stencil violates the constraints 1 or 2. Similarly, a stencil coarsening command will be sent if $\mathit{\Delta} < \theta_{\min}$ unless the target stencil violates the constraints 3 or 4. It can be anticipated that the action indicator will restrict the magnitude of local variation with respect to the parameter of interest to a certain range.

2.6. Initial value of dependent variables for the newly inserted nodes

During the stencil adaptation, new nodes are generated and deleted continually as the solution evolves. It is very critical to give an accurate value of dependent variables at the newly generated nodes, especially for the time-dependent problems. An inaccurate interpolation process not only introduces additional numerical error, but may also spur instability to yield unphysical solution. Fortunately, an interpolation scheme of $O(h^4)$ accuracy can be easily constructed in our adaptation. From the above descriptions of stencil refinement algorithm, we can see that all the member nodes in the stencil for the newly inserted node already exist in the domain. Therefore, we are sure that the value of the dependent variables is known at these member nodes. Based on this fact, an interpolation process at the newly generated node can be established with a leading error term of second-order,

$$(\cdot)_i = \frac{1}{4}\sum_{k=1}^{4}(\cdot)_{i_k^m} - \frac{h^2}{4}\Delta(\cdot)_i + O(h^4), \tag{4}$$

where $(\cdot)_i$ denotes the value of any dependent variables at newly inserted node i. $(\cdot)_{i_k^m}$ denotes the value of dependent variables at the member node in the stencil for node i at resolution level m. The symbol Δ denotes the Laplacian operator.

For the interpolation of one dependent variable u at the newly inserted node, we substitute the variable u and Δu into Eq. (4), which yields

$$u_i = \frac{1}{4}\sum_{k=1}^{4}u_{i_k^m} - \frac{h^2}{4}\Delta u_i + O(h^4), \tag{5}$$

$$(\Delta u)_i = \frac{1}{4}\sum_{k=1}^{4}(\Delta u)_{i_k^m} - \frac{h^2}{4}\Delta(\Delta u)_i + O(h^4). \tag{6}$$

Substituting (6) into (5), we have

$$u_i = \frac{1}{4}\sum_{k=1}^{4}u_{i_k^m} - \frac{h^2}{16}\sum_{k=1}^{4}(\Delta u)_{i_k^m} + O(h^4). \tag{7}$$

The central difference scheme is employed to approximate the Laplacian operator. For the present 5-point stencil, it gives

$$\Delta u_j = \frac{u_{j_1}+u_{j_2}+u_{j_3}+u_{j_4}-4u_j}{H_j^2} + \mathrm{O}(H_j^2), \tag{8}$$

where the subscript $j_1,\ldots,j_4$ denotes the member node in the stencil for the node j and H_j the mesh size. Taking consideration of constraint 1, we are aware that $\mathrm{Max}_{j=1,\ldots,4}(H_j) \leqslant \sqrt{2}h$.

It gives the final interpolation scheme for the dependent variable u at the newly inserted node by substituting (8) into (9). It can be clearly seen that the leading error of this interpolation scheme has an order of four with respect to the mesh size h.

3. Spatial discretization and stencil-adaptive multigrid Poisson solver

3.1. Spatial discretization on the local stencils

As discussed in Section 1, the central difference scheme is used to approximate the spatial discretization of partial differential operators. In this section, we will show how to construct the discretization for the first-order derivatives and the Laplacian operator, which is often encountered in the simulation of incompressible viscous flows.

For the interior nodes with a stencil of "A" configuration, the discretization of spatial derivative is quite straightforward. The formulation is exactly the same as the conventional finite difference scheme on a standard 5-point stencil. At the node i as shown in Fig. 1, the first-order derivative with respect to x and the Laplacian operator can be approximated with the second-order of accuracy by

$$\begin{aligned} \left(\frac{\partial u}{\partial x}\right)_i &= \frac{u_{i_3^k}-u_{i_1^k}}{2h_k} + \mathrm{O}((h_k)^2), \\ \left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)_i &= \frac{u_{i_1^k}+u_{i_2^k}+u_{i_3^k}+u_{i_4^k}-4u_i}{(h_k)^2} + \mathrm{O}((h_k)^2) \end{aligned} \tag{9}$$

where h_k denotes the mesh size of the stencil of kth resolution level. In this formulation, for the "A" configuration stencil, k is an even number. For the spatial discretization on the stencil of "B" configuration, the derivative discretization for the first-order derivatives needs a bit special treatment. The above two differential operators can be discretized as follows:

$$\begin{aligned} \left(\frac{\partial u}{\partial x}\right)_i &= \frac{u_{i_2^k}+u_{i_4^k}-u_{i_1^k}-u_{i_3^k}}{2\sqrt{2}h_k} + \mathrm{O}((h_k)^2), \\ \left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)_i &= \frac{u_{i_1^k}+u_{i_2^k}+u_{i_3^k}+u_{i_4^k}-4u_i}{(h_k)^2} + \mathrm{O}((h_k)^2), \end{aligned} \tag{10}$$

where the resolution level k is an odd number. Our analysis based on the two-dimensional Taylor series expansion shows that this formulation yields a scheme of second-order accuracy.

The mesh size h_k of stencils of the resolution level k is proportional to the coarsest stencil width h_0 by a constant factor. The factor r has the form of

$$r = \frac{h_0}{h_k} = 2^{\frac{k}{2}} \quad \text{or} \quad h_k = \left(\frac{\sqrt{2}}{2}\right)^k h_0. \tag{11}$$

The hierarchy of the mesh size of the stencil also indicates the resolution improvement between two adjacent adaptive levels, i.e., a refinement ratio of $\sqrt{2}$.

3.2. Stencil-adaptive multigrid Poisson solver

To accelerate the convergence speed for solving elliptic equations, a stencil-adaptive multigrid technique is employed in this study. In the following, we will describe how to combine the stencil-adaptive algorithm with the multigrid technique. It is well known that in the non-adaptive multigrid technique, a hierarchy of grids

$\tilde{\Omega}_0, \tilde{\Omega}_1, \ldots, \tilde{\Omega}_i$ are used, in which the subscript i indicates the resolution level of the grid with the larger i meaning the finer grid. In the stencil-adaptive multigrid approach, the domain is considered to be covered by a hierarchy of stencil groups instead of grids. The stencil groups can be categorized into the global one and local one. The global stencil groups denoted by $\Omega_0, \Omega_1, \ldots, \Omega_i$ are equivalent to the grids $\tilde{\Omega}_0, \tilde{\Omega}_1, \ldots, \tilde{\Omega}_i$ and cover the whole domain, respectively. It is noteworthy that they are only designed for the purpose of multigrid acceleration. Therefore, they are already generated before starting the computation and not involved in the solution-adaptive process. With this understanding, the stencil group Ω_i actually represents the background stencils in the solution-adaptive computation. For simplicity, we call these stencil levels indexed by i "multigrid levels". The local stencil groups $(\Omega_{i+1}, \ldots, \Omega_{i+j})$ are introduced during the solution process and thus dynamically changed by the solution-adaptive algorithm. In order to distinguish them from the global groups of stencils, their resolution levels are indexed by $(i+j)$, in which the index j represents the relative resolution level to the background stencils in Ω_i. In general, the local stencil group Ω_{i+j} is restricted to smaller and smaller sub-domains with the increasing of relative level j. For simplicity, we call these stencil levels generated during the solution-adaptive process as "AMR levels". Finally, we maintain such a hierarchy of stencils in the stencil-adaptive multigrid acceleration,

$$\Omega_0, \Omega_1, \ldots, \Omega_i, \Omega_{i+1}, \ldots, \Omega_{i+j},$$

and the sum of the stencils in the domain Ω yields $\Omega = \Omega_0 \cup \Omega_1 \cup, \ldots, \Omega_i \cup \Omega_{i+1} \cup, \ldots, \Omega_{i+j}$. The conception of stencil groups is illustrated in Fig. 8, in which the nodes attached with stencils of corresponding resolution levels are plotted.

From the observation of the stencil hierarchy, we can see that the stencil-adaptive multigrid technique differs from the standard multigrid technique in the sense of existence of the local stencil groups, i.e., $\Omega_{i+1}, \ldots, \Omega_{i+j}$. This is also the critical problem in the stencil-adaptive multigrid technique, i.e., how to carry out smoothing processes on these local stencil groups, especially at the interface nodes between the coarse and fine stencil groups. In general, we follow the idea of Brandt [24] to keep the approximated solution at the interface nodes constant. In other words, in the coarse-fine region, the approximation values at the nodes with coarse stencil are interpreted as Dirichlet boundary condition for the nodes among the fine local stencil groups.

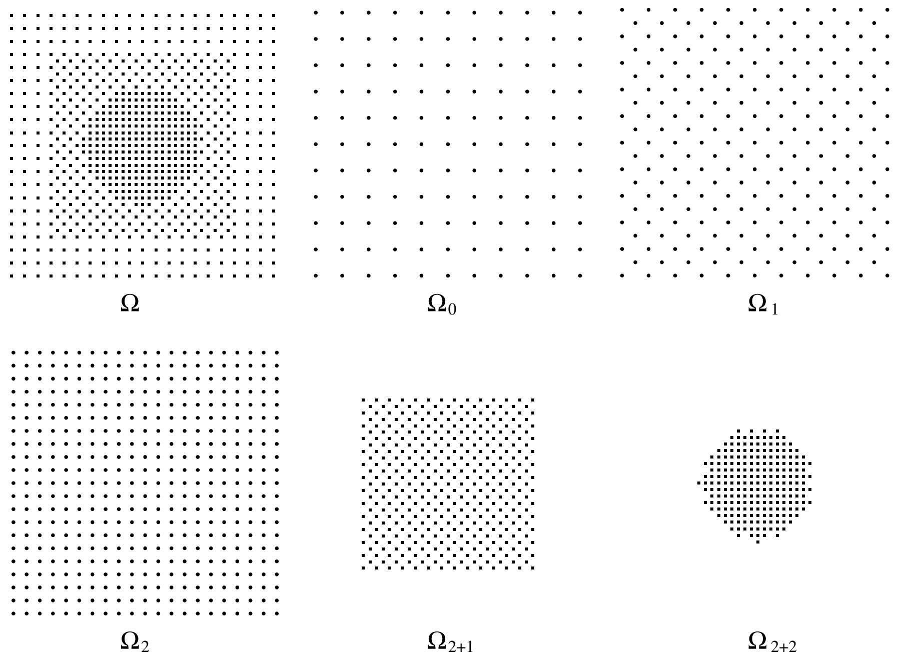


Fig. 8. Node distribution corresponding to the hierarchy of the stencil groups.

A full approximation scheme (FAS) multigrid V-cycle for the linear system $L_k u_k = f_k$ has the following form:

```
$u_k^{n+1} = FAS - V(k, u_k^n, L_k, f_k, \upsilon_1, \upsilon_2)$

Begin $\bar{u}_k^n = Smoothing(u_k^n, L_k, f_k, \upsilon_1)$
    $\bar{u}_{k-1}^n = \begin{cases} \overset{k-1}{I_k} u_k^n & \text{on } \Omega_k \cap \Omega_{k-1}, \\ u_{k-1}^n & \text{on the remaining part of } \Omega_{k-1}. \end{cases}$
    $f_{k-1} = \begin{cases} L_{k-1}\bar{u}_{k-1}^n + I_k^{k-1}(f_k - L_k\bar{u}_k^n) & \text{on } \Omega_k \cap \Omega_{k-1}, \\ f^\Omega & \text{on the remaining part of } \Omega_{k-1}. \end{cases}$

    If (k equals 1) then $Solve(\bar{u}_{k-1}^n, L_{k-1}, f_{k-1})$
    Else $w_{k-1}^n = FAS - V(k-1, \bar{u}_{k-1}^n, L_{k-1}, f_{k-1}, \upsilon_1, \upsilon_2)$
    $\bar{u}_k^n = \bar{u}_k^n + I_{k-1}^k(w_{k-1}^n - \bar{u}_{k-1}^n)$
    $u_k^{n+1} = Smoothing(\bar{u}_k^n, L_k, f_k, \upsilon_2)$

End $FAS - V(k, u_k^n, L_k, f_k, \upsilon_1, \upsilon_2)$
```

In the V-cycle described above, the subscript k denotes the stencil resolution level and superscript n the number of cycles. The "*Smoothing*" operation consists of two iterations (υ_1, υ_2) of successive over-relaxation (SOR) while the "*Solve*" operation solves the linear equations to convergence. The restriction and interpolation operator (I_k^{k-1} and I_{k-1}^k) are defined according to the features of our stencil adaptive algorithm as following:

- Restriction operator

$$I_k^{k-1} : \frac{1}{8}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}_k^{k-1}$$

for the restriction from A-type stencil to B-type stencil, and

$$\frac{1}{8}\begin{bmatrix} 1 & 0 & 1 \\ 0 & 4 & 0 \\ 1 & 0 & 1 \end{bmatrix}_k^{k-1}$$

for the restriction from B-type stencil to A-type stencil.

- Restriction operator

$$\overset{k-1}{I_k} : \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}_k^{k-1}.$$

- Interpolation operator

$$I_{k-1}^k : \left]\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}\right[_k^{k-1}$$

for the interpolation at the nodes which have both the kth and $(k-1)$th level of stencils. For the rest of nodes which have only the kth level of stencil, the interpolation operator is

$$\frac{1}{4}\left]\begin{matrix}1 & 0 & 1\\ 0 & 0 & 0\\ 1 & 0 & 1\end{matrix}\right[_{k}^{k-1}$$

from A-type stencil to B-type stencil, and

$$\frac{1}{4}\left]\begin{matrix}0 & 1 & 0\\ 1 & 0 & 1\\ 0 & 1 & 0\end{matrix}\right[_{k-1}^{k}$$

from B-type stencil to A-type stencil.

An example of stencil-adaptive multigrid V-cycle algorithm is illustrated in Fig. 9.

4. Numerical experiments

In this section, three examples are presented to examine the performance of the proposed adaptive refinement procedure.

4.1. Poisson equation

The first test case is the solution of Poisson equation. It is used to illustrate the second-order convergence of this stencil adaptive finite difference scheme and examine the acceleration performance of the stencil-adaptive multigrid Poisson solver. The modeling equation is the Poisson equation in a unit square with Dirichlet boundary conditions on all sides:

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=f(x,y). \tag{12}$$

The source function $f(x,y)$ on the right side of Eq. (12) is determined in such a manner that the exact solution u of Poisson equation is the given one. In this case, the solution $u(x,y)=\sin(2\pi x)\sin(2\pi y)$ yields

$$f(x,y)=-8\pi^2\sin(2\pi x)\sin(2\pi y). \tag{13}$$

In this study, there are two pre-refined node distributions used as shown in Fig. 9. To evaluate the convergence rate of present adaptive finite difference scheme, we select the "A" configuration of node distribution (as shown in Fig. 10) to do the test. The node distribution is generated in the following manner:

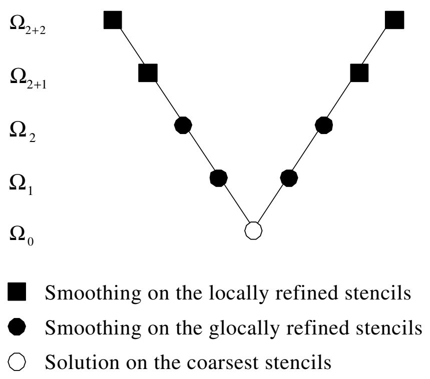

Ω_{2+2}
Ω_{2+1}
Ω_2
Ω_1
Ω_0
■ Smoothing on the locally refined stencils
● Smoothing on the glocally refined stencils
○ Solution on the coarsest stencils

Fig. 9. A V-cycle in the stencil multigrid technique.

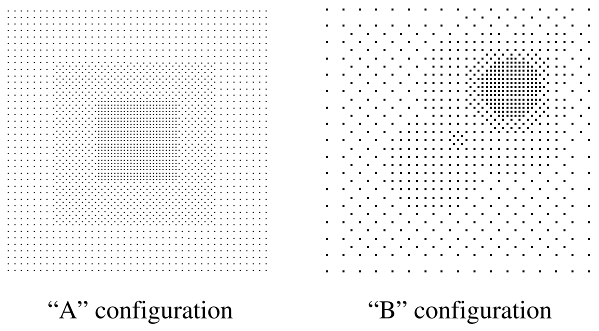


Fig. 10. Predefined stencil refinement for the Poisson equation.

- The global stencil group of resolution level 0 (Ω_0) is initially generated from a Cartesian grid.
- Then mesh refinement process is performed globally four times to generate the set of global stencil groups (Ω_1, Ω_2 and Ω_3) for multigrid acceleration and the background stencils (Ω_4).
- Finally, a local stencil group of resolution level 5 (Ω_{4+1}) is patched on the background stencils in the region ($0.2 \leqslant x \leqslant 0.8, 0.2 \leqslant y \leqslant 0.8$) and one more local stencil group (Ω_{4+2}) covers the region ($0.35 \leqslant x \leqslant 0.65, 0.35 \leqslant y \leqslant 0.65$).

To solve the Poisson equation numerically, the Laplacian operator in Eq. (12) is discretized at the stencils in the domain by the central finite difference scheme in Eqs. (9) and (10). The stencil-adaptive multigrid method is then employed to solve the resultant algebraic equations. The dependent variable u at the interior nodes is initially set to zero, and the convergence criterion is set to 10^{-12} which is considered small enough to obtain a converged solution. When the solution is converged, its accuracy is measured by the average L_2 norm of relative error, which is defined as,

$$\text{Average } L_2 \text{ norm of relative error}: \sqrt{\frac{1}{N_m}\sum_{i=1}^{N_m}\left|\frac{u_{\text{num}} - u_{\text{exact}}}{u_{\text{exact}}}\right|^2},$$

where N_m denotes the total number of nodes attached with a stencil of resolution level m. In order to study the convergence rate of the adaptive finite difference scheme, three groups of background stencils are employed, and they are equivalent to the Cartesian grids of 21×21, 41×41 and 81×81, respectively. The numerical results in terms of average L_2 norm of relative error are quantitatively shown in Table 1. It can be clearly observed that the solution is converging to the exact solution with second-order of accuracy in the L_2 norms for all the stencil resolution levels. This indicates the second-order convergence of this stencil adaptive finite difference scheme. However, due to the fixed stencils in the domain, the resolution level of the stencils in this case does not reflect the features of the solution. It can be observed from Table 1 that the solution accuracy at the stencils Ω_{4+1} is actually always lower than those at the stencils Ω_4 and Ω_{4+2}. This fact implies that it may be more appropriate to improve the stencil resolution level in the regions where the stencils of resolution level 1 occupy. Therefore, the solution adaptive algorithm should be included. That is what will be done in the second test case.

Table 1
Numerical results of the adaptive Poisson solver

Size of background stencil	Average L_2 norm of relative error		
	Ω_4	Ω_{4+1}	Ω_{4+2}
0.05	0.00718	0.00784	0.00659
0.025	0.00185	0.00197	0.00168
0.00125	0.000464	0.000494	0.000422
Average convergence rate	1.978	1.994	1.982

The performance of the stencil-adaptive multigrid Poisson solver is investigated regarding the convergence factor and its sensitivity to the number of stencil levels, the number of nodes on the coarsest level and the configuration of overall node distribution. The convergence factor is defined by

$$q^{(k)} = \frac{\left\|\text{residual}^{k-1}\right\|_\infty}{\left\|\text{residual}^{k}\right\|_\infty},$$

where $\|\text{residual}\|_\infty$ represents the infinite norm of the residuals at the finest stencils, and the superscript k denotes the iteration number.

We carried out three cases to examine the performance of the adaptive multigrid Poisson solver. The first test case has the "A" configuration of pre-refined nodes (as shown in Fig. 10), 2 AMR levels and 5×5 Cartesian grid as the coarsest level (Ω_0). The second test case also has the "A" configuration of pre-refined nodes and 2 AMR levels, but uses 11×11 Cartesian grid as the coarsest level (Ω_0). The third test case has the "B" configuration (as shown in Fig. 10) of pre-refined nodes, 4 AMR levels and 5×5 Cartesian grid as the coarsest level (Ω_0). In all cases, there are 5 multigrid levels employed, respectively. The convergence histories of these three cases obtained by the stencil-adaptive multigrid Poisson solver are presented in Fig. 11, with respect to the residual norm and convergence factor. Based on the observation of Fig. 11, we find that the convergence histories of the first and third cases are very similar. Their convergence lines with respect to residual norm are almost parallel and their average convergence factors are around 6.5. This observation indicates that the adaptive multigrid Poisson solver is not sensitive to the number of AMR levels and the configuration of node distribution. Comparatively, the second case has a lower average value of convergence factor (around 3.5). It implies that the size of coarsest stencils has some effects on the acceleration of convergence speed. Overall, in all cases, the residual reduces at least by 10 orders of magnitude within 15 V-cycles. This fact shows that the stencil adaptive algorithm can be combined with the multigrid technique. It is noted that the average convergence factor achieved in our study is relatively low as compared with that obtained by the standard multigrid technique, i.e., 15–20. The main reason may be due to the fact that the linear difference operator on the coarse stencils does not satisfy the so-called *Galerkin coarser grid operator* defined by the difference operator on the fine stencils, restriction operator and interpolation operator:

$$L_{k-1} \neq I_k^{k-1} L_k I_{k-1}^k.$$

As a result, the convergence speed is hampered. Comparatively, the standard multigrid technique using the full-weighting restriction operator does satisfy this relationship. On the other hand, from the numerical experiments, we can see that the convergence speed is not the best but is still satisfactory.

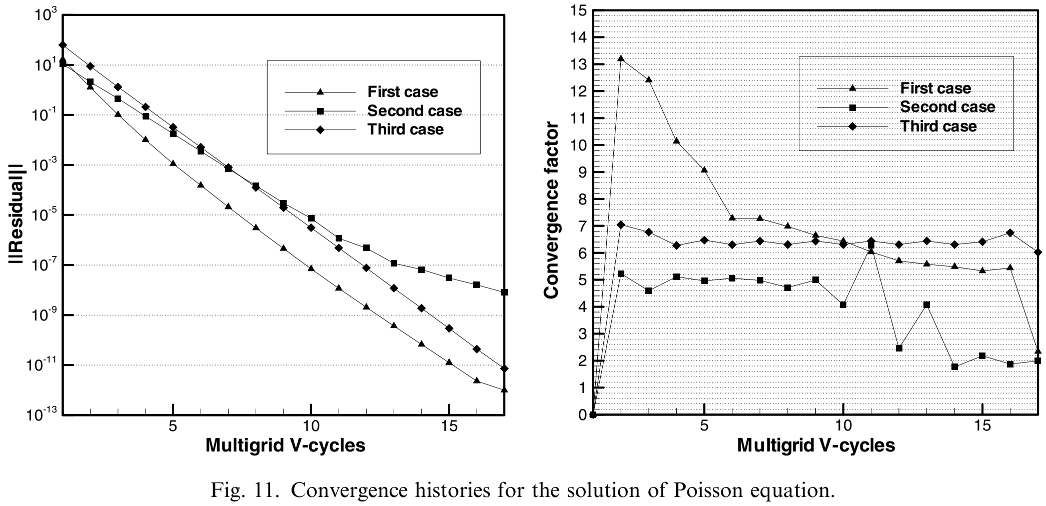


Fig. 11. Convergence histories for the solution of Poisson equation.

4.2. A combustion problem

From the viewpoint of computational efficiency, the use of adaptive algorithm is more desirable for the moving front/interface problems, in which the use of grid with single resolution is very inefficient. The second test case is such an example. It is a reaction–diffusion problem, in which the evolution of the moving interface between two chemical materials represents the main characteristics of the problem. The physical phenomenon of this problem can be described as follows: initially a moving steep layer is located at the center of the square due to the interaction between diffusion and reaction. Then, the steep layer moves quickly towards the boundaries. More details of this problem can be seen in [25]. To model the single one-step reaction of a mixture with two chemicals, the simplified modeling equation is given by

$$\frac{\partial f}{\partial t} = \varepsilon\left(\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}\right) + D(1+\beta - f)\mathrm{e}^{-\delta f}, \tag{14}$$

where δ is the activation energy, β is the heat release, and D is the Damkohler number defined by

$$D = \frac{R\mathrm{e}^{\delta}}{\beta\delta}.$$

Here R is the reaction rate. In this study, the values of these parameters are chosen as $\varepsilon = 0.1$, $R = 5$, $\beta = 1$ and $\delta = 20$. The boundary condition for this problem is given by $f|_{\partial\Omega} = 1$, and the initial condition is set to $f|_{t=0} = 1 + \sin^{50}(\pi x)\sin^{50}(\pi y)$, which contains a very sharp transition region.

In this case, the relative difference is chosen as the monitor parameter, and the upper and lower thresholds of action indicator are set to 0.1 and 0.01, respectively. The maximum resolution level of the stencils is restricted to 4, which implies that the mesh size of the finest stencil is as much as 1/4 of the coarsest one. To advance the solution in time, an explicit four-stage Runge–Kutta method is employed. The time step is set to 0.00001. The numerical results with a background/initial Cartesian grid of 81×81 are shown in Figs. 12.1(a)–12.3(a), which represent the time-dependent solution at simultaneous time = 0.01, 0.02 and 0.03, respectively. The corresponding node distributions are shown in Figs. 12.1(b)–12.3(b). As illustrated in Figs. 12.1(b)–12.3(b), with the multilevel stencil refinement, one can achieve a very reasonable node distribution for tracking the moving interface in the domain. Furthermore, it can be observed that the hierarchy resolution levels excellently reflect the sharp transition of solution around the moving interface. It demonstrates the ability of present adaptive algorithm in solving the moving interface problems.

However, we still need to evaluate the accuracy of the solution. Since there is no exact solution for this case, we adopt the solution at the time $t = 0.3$ computed on a 1281×1281 uniform grid as a reference one. In our studies, the solution-adaptive simulations start with three coarsest stencil sizes of 0.04, 0.025 and 0.0125, respectively, which correspond to the Cartesian grids of 21×21, 41×41 and 81×81. At the same time, the simulation is also carried out on the uniform grids of 81×81, 161×161 and 321×321, in which the mesh sizes represent the finest resolution level as those in the adaptive cases. Thus, we can have a quantitative accuracy comparison between the solution on the uniform mesh and that on the adaptive stencils. In order to obtain a fair comparison, the relative errors are only computed at the nodes in the region of $0.2 \leqslant \sqrt{(x-0.5)^2 + (y-0.5)^2} \leqslant 0.27$, in which the interface at $t = 0.3$ is located and so do the stencils of finest resolution. The numerical results in terms of average L_2 norm of relative error are listed in Table 2. It can be clearly seen that the second-order convergence is also achieved in this unsteady moving front case. More importantly, we can see that the present stencil adaptive algorithm not only allows us to place the fine stencils according to the feature of the solution, but also recovers the accuracy achieved on the equivalent fine grid.

4.3. Driven flow in a square cavity

In this test, a steady incompressible lid-driven flow problem in a square cavity is solved by using the adaptive finite difference method. The governing equations are the two-dimensional steady incompressible Navier–Stokes equations in the vorticity-stream function form, which can be non-dimensionalized as

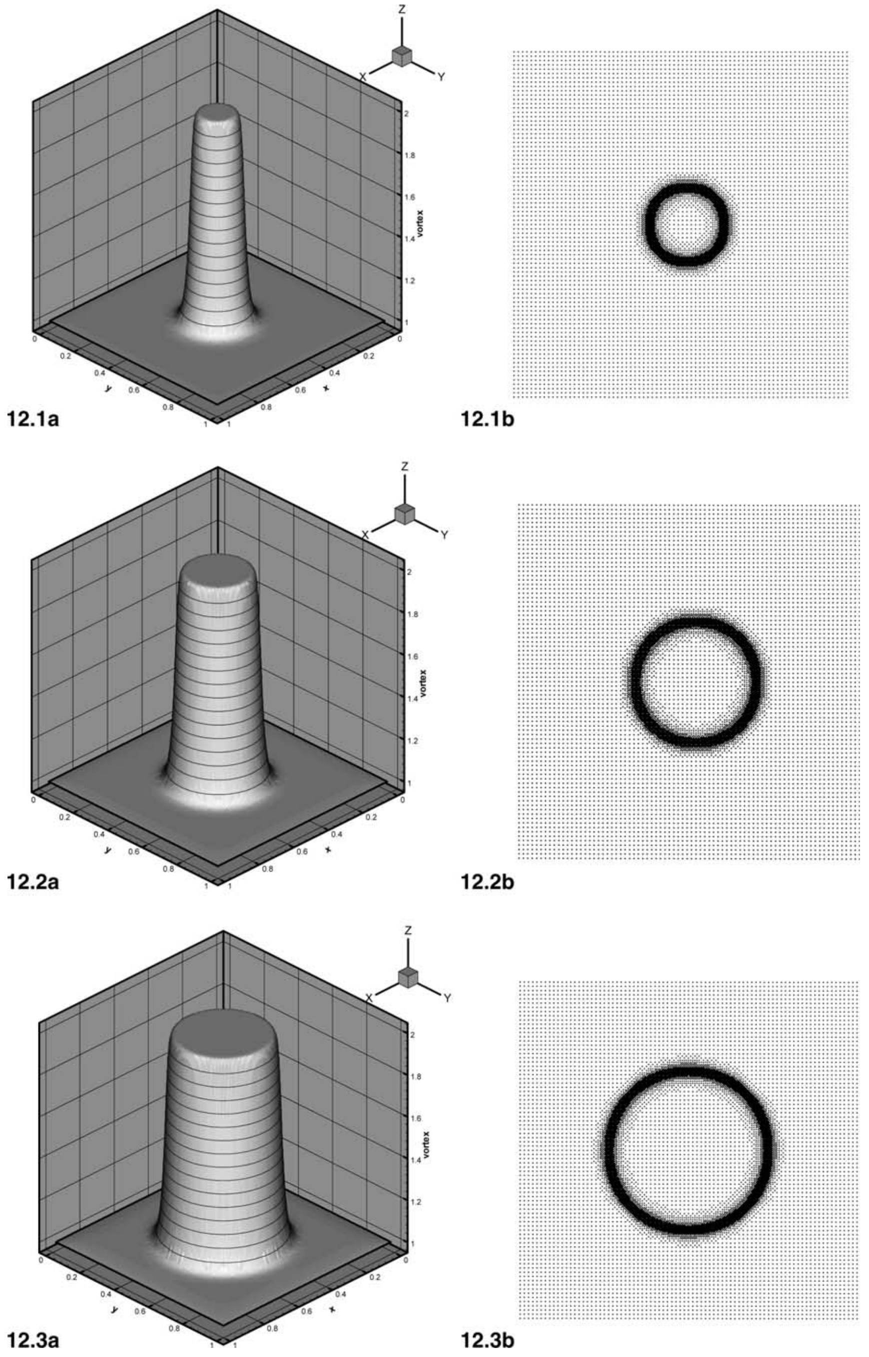


Fig. 12. Evolution histories of the solution and node distribution for the moving front case: (12.1a) solution at $t = 0.01$, (12.1b) node distribution at $t = 0.01$, (12.2a) solution at $t = 0.02$, (12.2b) node distribution at $t = 0.02$, (12.3a) solution at $t = 0.03$, (12.3b) node distribution at $t = 0.03$.

$$\begin{aligned}
&u\frac{\partial\omega}{\partial x} + v\frac{\partial\omega}{\partial y} = \frac{1}{Re}\left(\frac{\partial^2\omega}{\partial x^2} + \frac{\partial^2\omega}{\partial y^2}\right),\\
&\frac{\partial^2\psi}{\partial x^2} + \frac{\partial^2\psi}{\partial y^2} = \omega,
\end{aligned} \tag{15}$$

Table 2
Numerical results for the moving front problem

Uniform		Stencil adaptive at resolution level 4	
Background grid	Relative error	Background grid	Relative error
81×81	0.0630	21×21	0.0603
161×161	0.0137	41×41	0.0118
321×321	0.00268	81×81	0.00264

where u, v denote the components of velocity in the x and y directions, which can be calculated from the stream function

$$u=\frac{\partial\psi}{\partial y},\quad v=-\frac{\partial\psi}{\partial x},$$

Re is the Reynolds number. For the incompressible flow problem, the vorticity is considered as a very important flow parameter. A careful analysis of vorticity distribution can have a good understanding of the flow field. Therefore, the vorticity is chosen as the parameter of interest in this case. The two-dimensional lid-driven flow problem at Reynolds number of $Re = 1000$ is considered. The solution of this case by the finite difference method on the uniform mesh of 161×161 is shown in Fig. 13 in terms of the vorticity contour in which the magnitude of vorticity is demonstrated by color variation. All the differential operators are discretized by finite difference on the local stencils as shown in Eqs. (9) and (10). The set of coupled algebraic equations are then solved by the successive over-relaxation iterative method. All the computations are carried out on a personal computer with Pentium 4 (2.2G) and Fortran 90 compiler.

In the first test we use the absolute difference in the action indicator since it can provide a measure of the local gradient of the vorticity. The simulation is started on a grid with a 41×41 Cartesian mesh as the resolution level 0. The upper and lower thresholds are firstly set to 1.0 and 0.1, and the finest adaptive level is set to 2 and 4, respectively. The resultant node distributions at the final stage of computation are presented in Figs. 14(a) and (b). They contain a total number of 3589 and 7635 nodes, respectively. As compared with the vorticity contour in Fig. 13, it can be clearly seen that the node distribution in Fig. 14 is quite good in the sense that the stencil resolution levels reflect the local magnitude of the vorticity variation. It is reasonable to think that these node distributions are more suitable for the numerical simulation than the uniform node distribution in terms of efficiency. The numerical results with respect to the velocity component u at the vertical centerline of $x = 0.5$ and v at the horizontal centerline of $y = 0.5$ are plotted in Figs. 15(a) and 16(a). Since there is no analytical solution for this problem, the Ghia's result [26] is adopted as the benchmark data to validate the numerical results. In general, the numerical results achieved by the stencil adaptive algorithm have similar accuracy as those obtained on the uniform fine grids. Figs. 15(b) and (c), and 16(b) and (c) present the

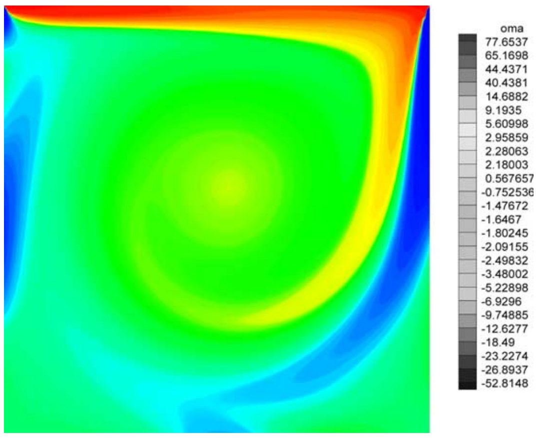
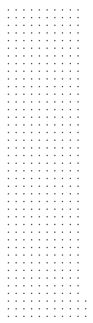


Fig. 13. Vorticity contour of lid-driven cavity flow at $Re = 1000$.



enlarged view of the velocity component u around $y = 0.2$ and 0.9 and v around $x = 0.2$ and 0.9, where the biggest differences occur. We can see that the maximum difference between the solution of u component on adaptive stencils and uniform grids is about 0.02 in Fig. 15(b), and less than 0.01 in Fig. 15(c). The reason

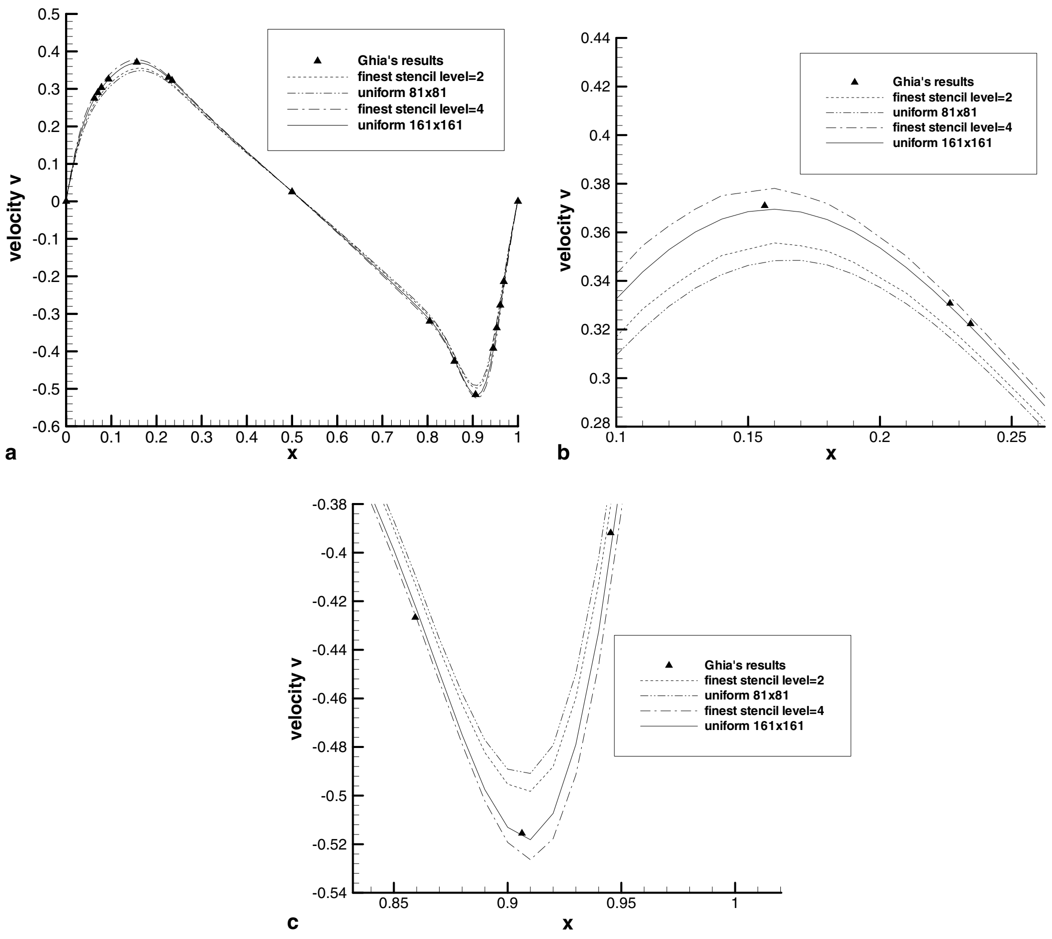


Fig. 16. Velocity component v in horizontal centerline for the flow in a square cavity at $Re = 1000$ using absolute difference as monitor parameter: (a) velocity component v in horizontal centerline, (b) enlarged view around $x = 0.2$, (c) enlarged view around $x = 0.9$.

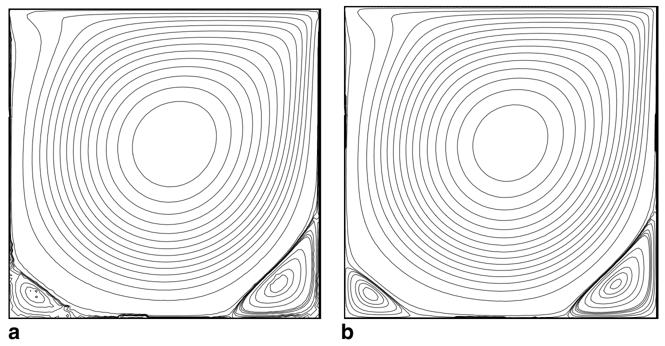
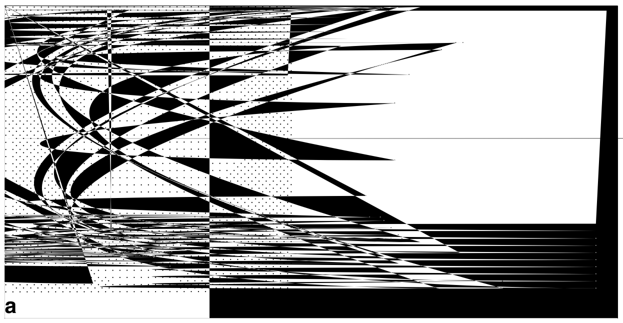


Fig. 17. Streamlines for the flow in a square cavity at $Re = 1000$: (a) using absolute difference, (b) using relative difference.

may be due to the fact that the stencils around $y = 0.2$ at the vertical centerline have a comparatively coarser resolution level than those around $y = 0.9$. However, though we use very coarse stencils around the square center (see Fig. 14), Figs. 15(a) and 16(a) show that there is little difference in the solution in terms of velocity components u and v. This observation indicates that the adaptive finite difference scheme works very effectively.



On the other hand, for the driven cavity problem, one may be more interested in the accurate capturing of the corner eddy located at the left and right lower corners. For the adaptive finite difference scheme with absolute difference as the monitor parameter, it is very difficult since the magnitude of the vorticity variation at the corner area is very small. As a result, the stencil cannot be refined there, so the numerical results at the lower corners are not very satisfactory as illustrated by the streamlines in Fig. 17(a). One immediate way to improve this situation is the use of another monitor parameter in the action indicator, for example, relative difference. We set the upper and lower thresholds for relative difference in the action indicator to 0.3 and 0.1, and the finest resolution level also to 2 and 4, respectively. The generated node distributions with relative difference as the monitor parameter are shown in Fig. 18, in which a total number of 4776 and 10,884 nodes are included, respectively. It can be seen in Fig. 18 that stencils of fine resolution are now placed at the lower corners, and the streamlines in these regions are consequently improved, as shown in Fig. 17(b). The numerical comparison as shown in Figs. 19(a) and 20(a) confirms that the results produced by adaptive finite difference method are very accurate. Figs. 19(b) and (c), and 20(b) and (c) present the enlarged view of the velocity component u around $y = 0.2$ and 0.9 and v around $x = 0.2$ and 0.9. In general, they are essentially equal to those obtained by central-difference on the uniform meshes with finest mesh size.

From the observation of the node distributions generated by both absolute and relative differences (as shown in Figs. 14 and 18), it is obvious that our stencil adaptive algorithm can appropriately adjust the resolution level of the stencils to reflect the important features of the flow. Thus, it is reasonable to expect a favorable return of improved computational efficiency. This anticipation is confirmed by the efficiency comparison presented in Table 3. As compared with the fixed grid method (also central difference), our adaptive algorithm requires much less total number of nodes to achieve an equivalently accurate solution. Consequently, the running time is also significantly reduced. It is noteworthy that the running time listed in Table 3 includes the overhead of the stencil adaptation and stencil manipulation.

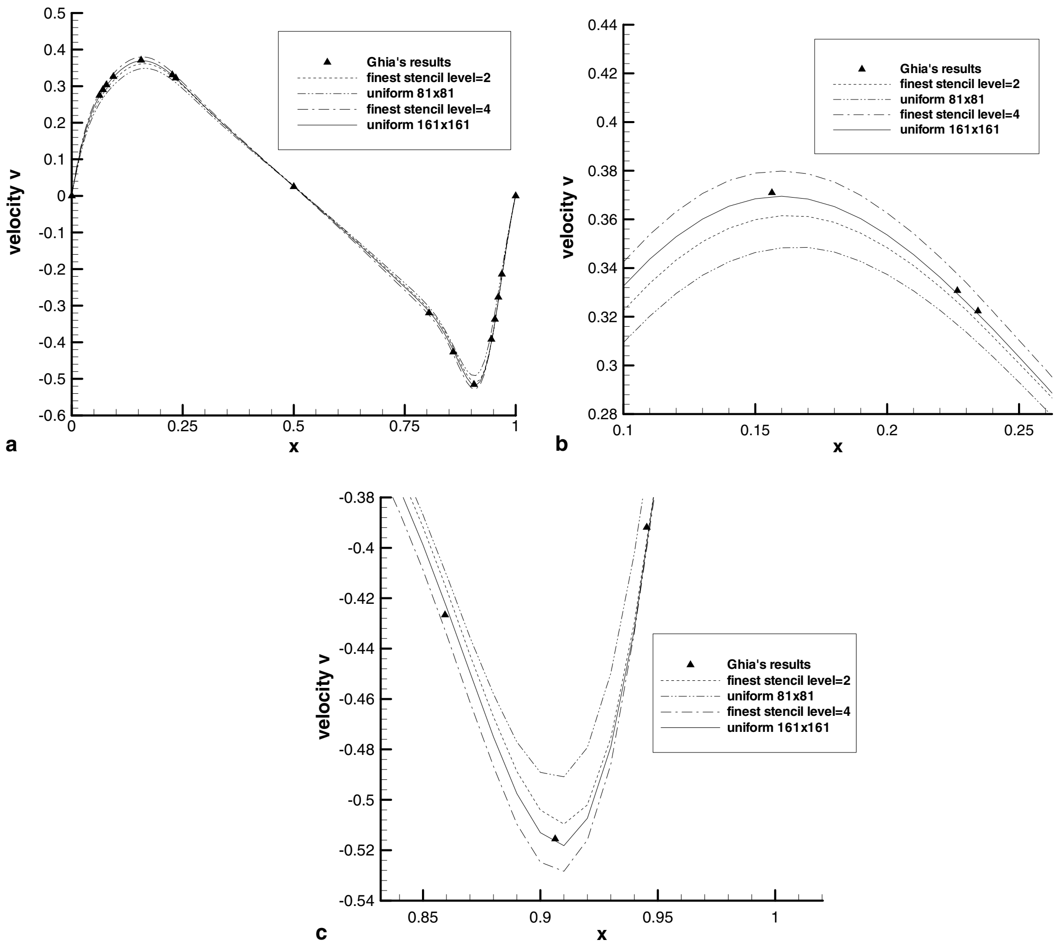


Fig. 20. Velocity component v in horizontal centerline for the flow in a square cavity at $Re = 1000$ using relative difference as monitor parameter: (a) velocity component v in horizontal centerline, (b) enlarged view around $x = 0.2$, (c) enlarged view around $x = 0.9$.

Table 3
Efficiency comparison between adaptive algorithm and uniform grid method

	Monitor parameter	Finest resolution level	Number of nodes	Iteration number	Running time (s)
Stencil adaptive	Absolute difference	2	3589	7275	11.80
		4	7635	15,008	53.82
	Relative difference	2	4776	9108	20.68
		4	10,884	12,216	61.45
Uniform grid		81×81	6561	12,367	49.85
		161×161	25,921	47,088	817.77

In summary, it can be observed from the numerical tests that our stencil adaptive algorithm can help finite difference scheme to achieve an equivalent accuracy as it obtains on the fine uniform mesh, and at the same time greatly save the CPU time. In other words, it achieves "fine-grid accuracy for coarse-grid cost". The studies in this paper are only restricted to the two-dimensional problems with simple geometries. The extension of this stencil-adaptive algorithm to the three-dimensional problem is still in progress. The possible difficulty in the three-dimensional case is how to provide appropriate constraints to guarantee the smooth stencil refinement and coarsening.

5. Conclusions

An efficient and fully solution-adaptive finite difference procedure for two-dimensional incompressible viscous flows is presented in this paper. The stencil adaptive algorithm is simple, but effectively exploits the high efficiency of the central difference scheme. It is able to automatically adjust the local stencil to reflect the transient behavior of the solution. Three numerical experiments have been carried out to examine the performance of present adaptive algorithm. Numerical results show that the second-order convergence has been achieved by the present adaptive finite difference method. They also indicate that the method can achieve a solution of comparable accuracy to that obtained by traditional finite difference on the uniform fine grid at the equivalent finest resolution, but significantly reduce the running time. In this study, we only concentrate on problems in the regular domain, and the multigrid-acceleration based on the adaptive stencils is still relatively low as compared with the standard multigrid technique. The problems with complex geometry and how to further improve the multigrid acceleration are regarded as the future works.

References

[1] C. Shu, Differential Quadrature and its Application in Engineering, Springer, London, 2000.
[2] R.E. Bellman, J. Casti, Differential quadrature and long-term integration, J. Math. Anal. Appl. 34 (1971) 235–238.
[3] R.E. Bellman, B.G. Kashef, J. Casti, Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations, J. Comput. Phys. 10 (1972) 40–52.
[4] M.J. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, J. Comput. Phys. 53 (1984) 484–512.
[5] M.J. Berger, R. LeVeque, Adaptive mesh refinement for two-dimensional hyperbolic systems and the AMRCLAW software, SIAM J. Numer. Anal. 35 (1998) 2298–2316.
[6] A.M. Khokhlov, Fully threaded tree algorithms for adaptive refinement fluid dynamics simulations, J. Comput. Phys. 143 (1998) 519–543.
[7] O.C. Zienkiewicz, J.Z. Zhu, The three R's engineering analysis and error estimation and adaptivity, Comput. Meth. Appl. Mech. Eng. 82 (1990) 95–113.
[8] J.Z. Zhu, O.C. Zienkiewicz, Adaptive techniques in the finite element method, Commun. Appl. Numer. Meth. 4 (1988) 197–204.
[9] P.A. Durbin, G. Iaccarino, An approach to local refinement of structured grids, J. Comput. Phys. 181 (2002) 639–653.
[10] R.B. Pember, J.B. Bell, P. Colella, W.Y. Curtchfield, M.L. Welcome, An adaptive Cartesian grid method for unsteady compressible flow in irregular regions, J. Comput. Phys. 120 (1995) 278–304.
[11] S. Popinet, Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries, J. Comput. Phys. 190 (2003) 572–600.
[12] L.H. Howell, J.B. Bell, An adaptive mesh projection method for viscous incompressible flow, SIAM J. Sci. Comput. 18 (4) (1997) 996–1013.
[13] M.L. Minion, A projection method for locally refined grids, J. Comput. Phys. 127 (1996) 158–178.
[14] A.S. Almgren, J.B. Bell, P. Colella, L.H. Howell, M.L. Welcome, A conservative adaptive projection method for the variable density incompressible Navier–Stokes equations, J. Comput. Phys. 142 (1998) 1–46.

[15] M. Gerritsen, P. Olsson, Design and efficient solution strategy for fluid flows II. Stable high-order central finite difference schemes on composite adaptive grids with sharp shock resolution, J. Comput. Phys. 147 (1998) 293–317.
[16] S. Adjerid, J.E. Flaherty, A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations, SIAM J. Numer. Anal. 23 (1986) 778–796.
[17] W.M. Cao, W.Z. Huang, R.D. Russell, An *r*-adaptive finite element method based upon moving mesh PDEs, J. Comput. Phys. 149 (1999) 221–244.
[18] R. Li, T. Tang, P. Zhang, Moving mesh methods in multiple dimensions based on harmonic maps, J. Comput. Phys. 170 (2001) 562–588.
[19] R.D. Hornung, J.A. Trangenstein, Adaptive mesh refinement and multilevel iteration for flow in porous media, J. Comput. Phys. 136 (1997) 522–545.
[20] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (1999) 509–534.
[21] T. Basebi, R.M. Thomas, A study of moving mesh methods applied to a thin flame propagating in a detonator delay element, Comput. Math. Appl. 45 (2003) 131–163.
[22] E.D. Wilkes, S.D. Phillips, O.A. Basaran, Computational and experimental analysis of dynamics of drop formation, Phys. Fluids 11 (12) (1999) 3577–3598.
[23] D.M Greaves, A.G.L. Borthwick, G.X. Wu, R.E. Taylor, A moving boundary finite element method for fully non-linear wave simulations, J. Ship Res. 41 (1997) 181–194.
[24] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Math. Comput. 31 (1977) 333–390.
[25] A.K. Kapila, Asymptotic Treatment of Chemically Reacting Systems, Pitman, Boston, 1983.
[26] U. Ghia, K.N. Ghia, C.T. Shin, High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method, J. Comput. Phys. 48 (1982) 387–411.