A Newton-type Method for Fluid Computation

LI AIDAN

NATIONAL UNIVERSITY OF SINGAPORE

2004

A Newton-type Method for Fluid Computation

BY

LI AIDAN
(B. Eng.)

DEPARTMENT OF MECHANICAL ENGINEERING

A THESIS SUBMITTED FOR THE DEGREE OF MASTER OF ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE

2004

ACKNOWLEDGEMENTS

The author would like to express her sincere gratitude to her supervisor, Associate Professor Yeo Khoon Seng, for supervising this project. The author has learnt much from his expertise in the field of computational fluid dynamics. His insight and guidance have ensured the project ran smoothly.

The author would also like to sincerely thank her co-supervisor, Professor Nhan Phan-Thien, for his supervision and assistance for this project.

Her sincere thanks also extend to Wu Long, Qu Kun, Chen Pengfei, Li Yangfang, Zhao Xijing, Pen kun, Liao Wei, Ding Hang for their valuable discussion and help.

Acknowledgements are also extended to all technicians in the Fluid Mechanics Laboratory at Workshop II, NUS, who provided help throughout this project.

The author would like to especially express sincere thanks to her beloved families, for their endless support and encouragement without which the project could not have been finished smoothly.

TABLE OF CONTENTS

SUMMARY

Today, computational fluid dynamics (CFD) plays an indispensable role in fluid and aerodynamic design. Accuracy and efficiency are two important factors in the success of a numerical method. The monotonic residual error reduction procedure proposed by Liu (2002) is further developed in this study with the incorporation of a multi-grid scheme to accelerate convergence and improve overall computational efficiency. The various components of the multigrid procedure including the smoothing method, coarsening method, restriction operator, the prolongation operator, and the effects of the various multi-grid parameters have been studied to optimize computational performance. A fourth-order refinement scheme has also been developed which allows highly accurate solutions to be derived with only relatively minimal increase in computational effort compared to the basic second-order scheme. Consistent with the fourth-order discretization of the operator, the fourth-order accurate pressure boundary conditions are used. Comparisons are made between the present method and the conventional Newton's method in both single-grid and multi-grid implementations.

The prototypical two-dimensional driven cavity flow problem is set as the basic test problem. Two kinds of correction functions (CF) have been designed to compare with the performance of Newton's method. It is demonstrated that correction function 3 shows slightly better performance than correction function 2. The conventional Newton's method is very sensitive to the initial guessed solution and the rate of successful convergence is fairly poor in many applications. The present scheme has a much higher rate of successful convergence. This is especially so for multi-grid implementation. The proposed method can lead to nearly monotonic decrease in the residual errors no matter whether single-grid or multi-grid method has been used in

small Reynolds number problems. The multi-grid method is able to maintain a rapid rate of the residual error decay throughout the computation, leading to large savings in computational effort especially for high Reynolds number flows. The use of full weighting is found to be slightly superior to optimal weighting. The fourth-order refinement scheme offers important gains over the standard second-order scheme. The fourth-order refinement scheme typically shows a higher computational efficiency for a given mesh than the second-order scheme because its application seems to promote a more rapid rate of convergence. Furthermore, it preserves or even enhances the accuracy of the solutions using far fewer mesh points than that of corresponding second-order scheme. The employment of fourth-order refinement does not incur a large CPU-time penalty for the accuracy gain even though it requires the mesh to be sufficiently fine to achieve convergence for high Reynolds number flows. Hence, it is a useful variation of the present method for problems that require high accuracy solutions.

LIST OF TABLES

LIST OF FIGURES

Chapter 1: Introduction

1.1 Background

Since the emergence of computational fluid dynamics (CFD) in the 1950s, it has revolutionized the world of aerodynamics (Anderson *et al*. 1996). Today, CFD has become an indispensable tool of aerodynamic design. Increasingly, experimental tests are reserved for confirmation of predicted performance. The steady increase in the speed of computers and the concomitant developments in numerical algorithms have made it possible now to simulate complex flow problems to a high level of accuracy. Incompressible fluid flows are commonly found in a wide range of industrial applications. The incompressible Navier-Stokes equations (INSE), which govern the behaviour of many industrial flows, are very difficult to be solved analytically on the basis of known principles. The effort to simulate these flows has been under way since the beginning of CFD. Computational Fluid Dynamics offers the only realistic hope for solving practical problems encountered in industry.

In the realm of computation, accuracy and efficiency are the two most important factors in the success of a numerical method. As a result, it has been the goals of researchers to devise schemes that solve the INSE efficiently and with accuracy. In our research group, a new monotonic approximation numerical method for steady incompressible Navier-Strokes equations (SINSE) was recently proposed by Liu (2002). The method devised by Liu analyses the effect of the nonlinear terms of the SINSE on the residual error, and suggests a correction that enables the residual error to be reduce monotonically. However, the use of simple iterative techniques by Liu

resulted in a slow rate of convergence to the solution. In addition, using only a first-order pressure boundary condition, the accuracy of his solutions is another important aspect that could be improved. Thus, the present thesis seeks to overcome some of the original inadequacies of Liu (2002), and to further develop the proposed scheme in terms of improved convergence, accuracy and efficiency.

1.2 Literature review

1.2.1 Approaches to the Incompressible Navier-Stokes Equations (INSE)

Efficient solution of the two dimensional INSE has been an important problem in computational science since its inception:

$$\frac{\partial u_2}{\partial x_2} + \frac{\partial u_3}{\partial x_3} = 0, \tag{1.1}$$

$$\frac{\partial u_1}{\partial x_2} + \frac{\partial (u_2)^2}{\partial x_2} + \frac{\partial (u_2 u_3)}{\partial x_3} - \frac{1}{\mathrm{Re}}\left[\frac{\partial^2 u_2}{\partial (x_2)^2} + \frac{\partial^2 u_2}{\partial (x_3)^2}\right] = 0, \tag{1.2}$$

$$\frac{\partial u_1}{\partial x_3} + \frac{\partial (u_3)^2}{\partial x_3} + \frac{\partial (u_2 u_3)}{\partial x_2} - \frac{1}{\mathrm{Re}}\left[\frac{\partial^2 u_3}{\partial (x_2)^2} + \frac{\partial^2 u_3}{\partial (x_3)^2}\right] = 0. \tag{1.3}$$

Here, (u_1, u_2, u_3) corresponds to (p, u, v) where p is the pressure, and u and v are the components of the velocity field in the x- and y- coordinate directions. The x_2 and x_3 in (1.1-1.3) correspond to the x- and y- spatial coordinates respectively. The quantities of equations (1.1-1.3) are assumed to have been non-dimensionalized by a characteristic velocity and length scales. Re denotes the Reynolds number. The restriction to incompressible flow introduces the computational difficulty that there is no obvious link between the velocity components and the pressure in the continuity equation (1.1). Over the last three decades, various methods have been developed to solve these

equations. The methods may be broadly classified under two categories: the primitive variable formulations and formulations based on derived variables.

A popular derived variables approach is the Vorticity-Stream-function method, in which the explicit treatment of the continuity equation is avoided by replacing the velocity components with the vorticity and stream-function. The pressure does not appear in the resultant equations. There are only two partial differential equations are solved instead of three for the Navier-Stokes equations. The reduction in the number of dependent variables and equations makes this approach attractive. However, a problem with this approach lies in the boundary conditions, especially in complex geometries. The values at the boundaries of the dependent variables are not so straightforward to specify, and some special treatments are needed. Another important drawback of this approach is the difficulty of extending this formulation to three space dimensions since a three-dimensional stream function cannot be defined. In three dimension, vorticity-related formulations lead to more dependent variables, typically six, which can be seen in the vorticity-vector potential formulation used by Mallinson and Davis (1977). As a result, three-dimensional vorticity-related formulations have not been used very often.

Another popular approach is the primitive variables formulation. The primitive variables, namely the pressure and the velocities, can easily be defined in real geometry compared to the derived quantities such as the stream-function and vorticity. Consequently, the extension to three spatial dimensions creates no additional difficulty. Primitive variables approach can be further grouped into two main categories.

The first category is the method based on compressible flow algorithm, namely the artificial compressibility method, which was first proposed by Chorin (1967) almost four decades ago. In this method the solutions to the steady INSE are sought by applying a pseudo-transient formulation to the unsteady momentum equations with an artificial time derivative of pressure in the continuity equation. At the same time, an artificial compressibility parameter is applied. With this artificial term, the time-dependent equation system is symmetric hyperbolic-parabolic type and the strict requirement of satisfying mass conservation in each step is relaxed. This allows efficient numerical schemes developed for compressible flows to be used for incompressible flows. Chang and Kwak (1984) suggested some guidelines for choosing the artificial compressibility parameters. Various applications have been reported for obtaining steady-state solutions (Kwak *et al.* 1986; Chang *et al*. 1988). Turkel (1987) extended this concept with more sophisticated preconditioners than those originally proposed by Chorin (1967) to allow for faster convergence. To obtain time-dependent solutions using this method, a dual-time stepping technique is used. The physical time derivative terms are treated implicitly as source terms. An iterative procedure can then be applied in each physical time step such that the continuity equation is satisfied (Rogers *et al*. 1991). Several variations of the artificial compressibility approach can be found in the literature (Rizzi and Eriksson, 1985).

The other category is the method based on projection or pressure correction. In 1965, Harlow and Welch (1965) published one of the earliest, and most widely used pressure-based primitive variables method called the Marker-and-Cell (MAC) method. The method is characterized by the use of staggered grid and a Poisson equation for pressure. However, the strict requirement of obtaining the correct pressure for a

divergence free velocity field in each step may significantly slow down the overall computation. Ever since its introduction, numerous variations of the MAC method have been devised. And the best known method to solve the steady INSE is the SIMPLE-family developed by Patankar and Spalding (1972), in which the correct pressure field is desired only when the solution is converged. The acronym, SIMPLE, stands for the Semi-Implicit Method for Pressure-Linked Equations and describes the iterative procedure by which the solution to the discretised equation is obtained. The unique feature of this method is the simple way of estimating the velocity and the pressure correction. Patankar (1980) introduced a revised algorithm SIMPLER which converges faster than SIMPLE. Doormaal and Raithby (1984) developed a more efficient algorithm as a consistent SIMPLE algorithm called SIMPLEC. And they have made a systematic comparison of these three SIMPLE-type algorithms and concluded that SIMPLEC and SIMPLER are more efficient than SIMPLE, with SIMPLEC to be preferred. Also the SIMPLE-type algorithms on staggered grids have been generalised to collocated grid, which are being increasing applied in recent studies (Melaaen, 1991; Coelho and Pereira, 1992). Nevertheless, there are certainly some critical issues such as checkerboard problem that requires attention when collocated grid is used. Since the inception of SIMPLE-type algorithms, methods that incorporate acceleration technique have been a favourable choice for INSE computation.

Tamamids *et al*. (1996) carried out a comparison of accuracy, grid independence, convergence behaviour, and computational efficiency of the two representative methods, pressure-based and artificial compressibility, for calculating three-dimensional steady incompressibility laminar flows. They concluded that both

methods have merits and demerits. For accuracy, the results from pressure-based method are slightly favourable even though both methods produce reasonable results compared with experimental data and are grid independent. Artificial compressibility method converges faster with suitable parameter selection; however, it requires much more memory in the computation.

1.2.2 The techniques of Newton-like method

Newton's method is the master method for solving non-linear equations $F(x)=0$. It linearizes the function about an estimated value of x using the first two terms of the Taylor series:

$$F(x) \approx F(x_0) + F'(x_0)(x - x_0). \tag{1.3}$$

We assume throughout that F is continuously differentiable. At the k^{th} step, the Newton step s_k can be determined by solving the linear Newton equation:

$$F'(x_k)s_k = -F(x_k), \tag{1.4}$$

where x_k is the current approximate solution and F' is the Jacobian matrix of the system. Then the current approximation is updated via:

$$x_{k+1} = x_k + s_k. \tag{1.5}$$

A satisfactory solution is found by iterating this process until $\|F(x_k)\|$ or $\|s_k\|$ (or both) is sufficiently small. This method is attractive because it converges quadratically when the estimate is close enough to the root (Ortega and Rheinboldt, 1970). That means the error at iteration $k+1$ is proportional to the square of the error at step k. As a result, only a few iterations are needed with sufficiently good initial guess.

However, for large systems, the rapid convergence is more than offset by its principal disadvantage. Because the Jacobian has to be evaluated at each iteration in this method, it induces high computational and storage cost. Nevertheless, Newton's method will still converge even if equation (1.3) is not solved exactly. And under some circumstances computing the exact solution may not be justified, because the linearization of $F(x) = 0$ around x_k is valid only in a neighbourhood of x_k. Then, if the solution of (1.3) produces a s_k that is too large, there may be poor agreement between F and its local linear model. Therefore, it seems reasonable to use an iterative method and compute some approximate solution. The Newton-iterative methods (Ortega and Rheinboldt, 1970) provide a trade-off between the accuracy with which the Newton equations are solved and the amount of work per iteration.

Dembo *et al*. (1982) proposed a class of inexact Newton methods which compute an approximate solution to the Newton equations in some unspecified manner such that

$$\left\| F'(x_k)s_k + F(x_k) \right\| \le \eta_k \left\| F(x_k) \right\|. \tag{1.6}$$

A forcing term η_k was introduced to control the level of accuracy. The optimal choice of η_k is critical to the efficiency of the method and is problem-specific (Shadid *et al*., 1997). Eisenstat and walker (1996) outlined the forcing term choices that result in desirably fast local convergence and also tend to avoid over-solving the Newton equations. In addition, they concluded that very small forcing terms might converge more rapidly but with less accuracy in the iterative linear solver compared with the larger forcing terms chosen.

In computing equations (1.3) and (1.6), the convergence is only local. That means the iterations may not converge if the initial estimate is far from the solution. Eisenstat

and walker (1994) established and analysed globally convergent inexact Newton methods, incorporating features designed to improve convergence from arbitrary starting points. They proposed that if s_k satisfies not only the equation (1.6), but also the following conditions, where $t \in (0, 1)$,

$$\left\| F(x_k + s_k) \right\| \leq [1 - t(1 - \eta_k)] \left\| F(x_k) \right\|, \tag{1.7}$$

then, a decrease in $\|F\|$ at each iteration could make convergence to a solution from a poor initial approximation. Some other globalized Newton-like algorithms were established based on this algorithm, such as backtracking methods and equality curve methods.

Newton-Krylov methods

There are many ways to compute an inexact Newton step s_k that satisfies equation (1.6) and the efficiency of the inexact Newton method is strongly affected by this choice. The Krylov subspace method (Freund *et al.*, 1992) is well suited for this purpose because it only requires the matrix-vector product $F'(x_k)\upsilon$ with

$$F'(x_k)\upsilon \approx \frac{F(x_k + \varepsilon\upsilon) - F(x_k)}{\varepsilon}. \tag{1.8}$$

So the Jacobian F' never needs to be explicitly formed. This further specialization of inexact Newton methods leads to the class of methods referred to as Newton-Krylov methods, which are actively applied in a large variety of problems recently such as radiation-diffusion problems (Mousseau *et al.*, 2000) and incompressible flow problems (Knoll and Mousseau, 2000; Mchugh and Knoll 1994). Among Krylov subspace methods, GMRES (Saad and Schultz, 1986) is generally preferred, since it

minimizes the residual at every iteration. However, the work and storage requirements per iteration grow linearly with the number of iterations so that it is expensive to use. Alternatives such as Bi-Conjugate gradient stabilized method (Vorst, 1990) and Orthogonal Residuals method (Edwards *et al*., 1994) can be considered.

1.2.3 Finite difference and artificial viscosity discretization schemes

When standard central differences are used to discretize the steady incompressible Navier-Stokes equations (SINSE), an instability problem arises because of the singular perturbation character of the momentum equations at high Reynolds numbers. The two dimensional linearized model of the momentum equations, which are the key problem of this thesis, is described as an example:

$$Lu = -\varepsilon\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + a\frac{\partial u}{\partial x} + b\frac{\partial u}{\partial y} = f^{\Omega}(x, y) \quad (\Omega = (0,1)^2). \tag{1.9}$$

Here, a and b are constants and the parameter ε represents inverse of Reynolds number. In the unit square $\Omega=(0, 1)^2$, it is singular perturbation problem because in the limiting case as $\varepsilon \to 0$, equation (1.9) is no longer elliptic but hyperbolic. The standard central five-point finite differences discretization for equation (1.9) reads

$$\begin{aligned} L_h &= \frac{\varepsilon}{h^2}[4u(i, j) - u(i, j+1) - u(i, j-1) - u(i+1, j) - u(i-1, j)] \\ &+ \frac{a}{2h}u(i+1, j) - u(i-1, j) + \frac{b}{2h}(i, j+1) - u(i, j-1) \\ &= \frac{4\varepsilon}{h^2}u(i, j) + \frac{bh - 2\varepsilon}{2h^2}u(i, j+1) - \frac{bh + 2\varepsilon}{2h^2}u(i, j-1) \\ &+ \frac{ah - 2\varepsilon}{2h^2}u(i+1, j) - \frac{ah + 2\varepsilon}{2h^2}u(i-1, j). \end{aligned} \tag{1.10}$$

where h is the mesh size for both the x direction and the y direction on the Cartesian grid. In this discretization scheme, non-physical oscillations develop in the solution if

the viscous term is small enough compared with the convective term because of the symmetric three points differencing of the convective term. For function (1.10), the stability condition is:

$$\frac{h}{\varepsilon}\max(|a|,|b|) \le 2 \tag{1.11}$$

The left-hand side of equation (1.11) is called the mesh-Péclet number Pe (Trottenberg *et al*., 2001). If the Péclet condition (1.1) is fulfilled, L_h from equation (1.10) gives a reasonable and stable $O(h^2)$ approximation for L. If the Péclet condition is not fulfilled, some off-diagonal elements of the matrix become positive. As a consequence, the matrix is no longer an M-matrix (A matrix A is said to be an M-matrix if and only if $a_{i,j} \le 0,\ i \ne j = 1(1)n,\ A$ is non-singular and $A^{-1} \ge 0$.). Thus, the L_h obtained from central differencing becomes unstable.

When ε is very small, an extremely fine grid must be used to ensure the numerical stability of central difference schemes. This will result in large memory and CPU time requirements that are clearly undesirable. However, the numerical instability at small ε can be alleviated by the use of upwind discretizations. With regard to equation (1.10), the first-order upwind can be described as:

$$\begin{aligned} L &= \frac{\varepsilon}{h}[4u(i,j)-u(i,j+1)-u(i,j-1)-u(i+1,j)-u(i-1,j)] \\ &+ \frac{a-|a|}{2h}[u(i+1,j)-u(i-1,j)]+\frac{b-|b|}{2h}[(i,j+1)-u(i,j-1)]+\frac{|a|+|b|}{h}u(i,j) \\ &= \frac{4\varepsilon+h(|a|+|b|)}{h}u(i,j)+\frac{h(b-|b|)-2\varepsilon}{2h}u(i,j+1)-\frac{h(b+|b|)+2\varepsilon}{2h}u(i,j-1) \\ &+ \frac{(a-|a|)h-2\varepsilon}{2h}u(i+1,j)-\frac{(a+|a|)h+2\varepsilon}{2h}u(i-1,j) \end{aligned} \tag{1.12}$$

Equation (1.12) leads to a stable problem and the corresponding matrix is an M-matrix. However, the first-order upwind scheme is only O(h) in accuracy, which is not satisfactory for more critical applications. This leads to the development of higher-order upwind schemes. Upwind schemes have been used extensively to control numerical instability. Recent examples include Kopteva (2003); Li (2000); Kupferman and Tadmor (1997).

An alternative way to control numerical instability is the use of artificial diffusion. The artificial diffusion terms can smooth out non-physical discontinuities in the flow. And, sometimes these terms can also counteract the dispersion error in the numerical scheme. In fact, the first-order upwind discretization can be regarded as a special case of the artificial diffusion approach in the central difference discretizations since, e.g. for $a > 0$,

$$a\left(\frac{\partial u}{\partial x}\right)_{h,upwind} = \frac{a}{h}\left(u_i - u_{i-1}\right) = a\frac{u_{i+1} - u_{i-1}}{2h} - \frac{ah}{2}\frac{\left(u_{i+1} - 2u_i + u_{i-1}\right)}{h^2} \tag{1.13}$$

Equation (1.13) shows that the first-order upwind scheme is a combination of a central difference term and an extra dissipative term, and it can lead to a stable discretization scheme. Similarly with the upwind schemes, first-order and higher-order artificial dissipation terms are preferred in different cases. In Liu *et al*. (1998), fourth-order artificial dissipation terms were added to their systems to suppress spurious numerical oscillations when the grid size was not small enough to render the physical viscosity effective. In this thesis, a generalized artificial diffusion scheme is applied in computing the linear operator. And, when the residual error converges to a very small value tending to zero, a highly accurate solution can be obtained.

The customary central difference schemes are second-order in accuracy. If high accuracy is required, then the higher-order difference schemes are preferred. In this thesis, besides the second order schemes, the fourth-order difference scheme will also be considered. If fourth-order central finite difference expressions are substituted for the derivatives in equation (1.9), then the following algorithm is obtained:

$$\begin{aligned} L_h u = -\varepsilon & \left(\frac{16u_{i+1,j} - 30u_{i,j} + 16u_{i-1,j} - u_{i-2,j} - u_{i+2,j}}{12h^2} \right. \\ & \left. + \frac{16u_{i,j+1} - 30u_{i,j} + 16u_{i,j-1} - u_{i,j-2} - u_{i,j+2}}{12h^2} \right) \\ & + a\frac{-u_{i+2,j} + 8u_{i+1,j} - 8u_{i-1,j} + u_{i-2,j}}{12h} + b\frac{-u_{i,j+2} + 8u_{i,j+1} - 8u_{i,j-1} + u_{i,j-2}}{12h}. \end{aligned} \tag{1.20}$$

1.2.4 Iterative methods

After suitable discretization, most partial differential equations are eventually transformed into linear algebraic equations that can be represented symbolically as:

$$LU(X) = f(X) \quad (X \in \Omega) \tag{1.14}$$

where the size of the matrix L is equal to the number of unknowns representing the discretized solution of the original equations. U is the vector of dependent variables and $X = (x_1, x_2, ...x_d)$ are the d independent variables of the d-dimensional problem. f is a known function on domain Ω. Any system of discretized algebraic equations can be solved by direct methods such as the Gauss elimination or LU decomposition. However, considering the numerical error and the expensive computational cost, it is often undesirable to solve large equation systems exactly using direct methods. On the other hand, the iterative methods are often effective in solving large linear systems as long as the convergence is guaranteed.

Classic iterative method

To derive the classical iterative method, matrix L can be written as $L = D - L^{-} - L^{+}$, where $D = diag(L)$, assuming $\det(D) \neq 0$, and L^{-} is the strictly lower and L^{+} the strictly upper triangular matrices, respectively. Thus, the Jacobi method is defined as:

$$U^{(n+1)} = D^{-1}(L^{-} + L^{+})U^{(n)} + D^{-1} f\,, \tag{1.15}$$

where $U^{(n)}$ is the approximate solution after n iterations. The Gauss-Seidel iterative method is defined as:

$$U^{(n+1)} = (D - L^{-})^{-1} L^{+} U^{(n)} + (D - L^{-})^{-1} f\,. \tag{1.16}$$

and the Successive Over-Relaxation (SOR) iterative method is represented by:

$$\begin{gathered} U^{(n+1)} = \chi_{\omega} U^{(n)} + c_{\omega}, \quad \chi_{\omega} := (D - \omega L^{-})^{-1}[(1-\omega)D + \omega L^{+}]. \\ c_{\omega} := \omega (D - \omega L^{-})^{-1} f \end{gathered} \tag{1.17}$$

where ω is the over-relaxation factor.

Rapid convergence of an iterative method is the key for its success. It turns out that the properties of the matrix L have important impact on the convergence of the linear systems. In the simplest method, the Jacobi method, it converges when matrix L is irreducible and weakly diagonally dominant. However, the Jacobi method is expensive because it requires a number of iterations proportional to the square of the number of grid points in one direction. The Gauss-Seidel method converges twice as fast as the Jacobi method. However, the rate of convergence is still very slow for large problems.

The SOR method provides a significant improvement over the Gauss-Seidel by evaluating $U^{(n+1)}$ from the values of $U^{(n)}$ and $(U^{(n+1)})_{GS}$, which can be seen in

equation (1.13). A necessary condition for the SOR method to converge is the restriction on $\omega \in (0, 2)$. And the number of iterations for convergence is sensitive to the choice of ω. Generally, the finer the grid, the larger the optimum ω will be. For values of ω less than the optimum, the convergence is monotonic and the rate of convergence increases as ω increases. Otherwise, the convergence rate deteriorates and the convergence is oscillatory when ω exceeds the optimum. An optimum choice is given by Hageman and Young (1981).

$$\omega = \frac{2}{1+(1-\mu^2)^{1/2}}. \tag{1.18}$$

where μ is the largest eigenvalue of $I - D^{-1}L$. However, finding μ explicitly can be as expensive as the original problem. Hence, a preferred practice is to estimate a μ value as the iteration proceeds. Equation (1.18) then provides an improved value for μ. Hadjidimos (2000) summarized some different choices of ω in cases where the matrix L possesses additional properties, such as positive definiteness, L-, M-, H-matrix property and p-cyclic consistently ordered property etc. When the optimum over-relaxation factor is used, the number of iterations for a certain amount of error reduction is proportional to the number of grid points in one direction. Therefore, it is adopted in the present work.

The SOR method can be changed to the symmetric successive over-relaxation method (SSOR) through making a small modification. First, the SOR scheme is applied to the unknowns in a certain order. This is followed by applying the same scheme to the unknowns in the reverse order, using the same ω. Usually, the SSOR method is less efficient than the SOR method, unless acceleration techniques such as Chebyshev and

conjugate gradient are also included. For comparison, the SSOR method is also applied in this project.

Multigrid method

Multigrid methods are one of the fastest numerical methods for many types of partial differential equations (Trottenberg *et al.*, 2001). It has been used widely since it was introduced in the 1970s by Brandt. With grid spacing h as a subscript, the linear algebraic equation (1.14) can be represented as:

$$L_h U_h = f_h \,. \tag{1.19}$$

Here, L_h is discrete operator and L_h^{-1} is assumed to exist. A conventional iterative technique for solving Eq. (1.19) consists of repeated sweeps of some relaxation scheme until convergence is achieved. However, it is often experienced that the convergence slows down after few iterations. This phenomenon can be explained by a local Fourier analysis of the error, which is probably the most powerful tool for the quantitative analysis and the design of efficient multigrid methods (Trottenberg *et al*., 2001). Using local Fourier analysis, the smoothing factor that refers to the error reduction in one iteration step measured in an appropriate norm can easily be calculated for many smoothers. It is observed that SOR produces a good smoothing rate for those error components whose wavelength is comparable to the size of the mesh. For those error components with longer wavelength, the smoothing rate is poorer.

A wavelength, which is long relative to a fine mesh, is shorter relative to a coarser mesh. The multigrid is created on a basis of this feature. It consists of two basic

ingredients: smoothing and coarsening grid correction. Firstly, the classic iterative method is used as smoother with appropriate iterations on a given fine mesh to eliminate the high frequency error components. Then the multigrid switches to a coarser mesh with double or more step size H, where the error components with wavelength comparable to H are rapidly annihilated. Then the fine-grid solution determined in first step need to be corrected to reflect appropriately the removal of the H-wavelength error components. One step of such an iterative two-grid cycle proceeds as following:

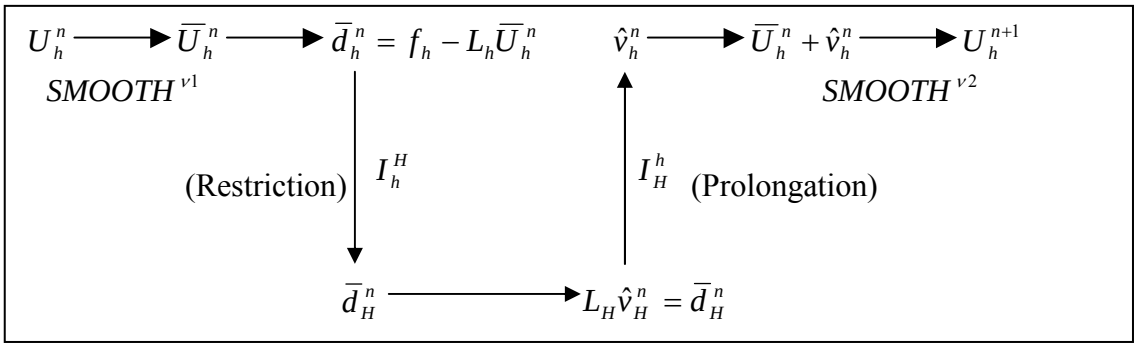


Fig 1.1 Structure of two grid cycle for linear equations

Here, a pre-smoothing symbol $SMOOTH^{v1}$ means computing $\bar{U}_h^n$ by applying $v1$ steps of a given smoothing procedure to U_h^n and the post-smoothing symbol $SMOOTH^{v2}$ means applying $v2$ steps of the given smoothing procedure to obtain U_h^{n+1}. The superscript n means the number of multigrid cycles and the subscript is representative of the grid size. Since the median value $\bar{U}_h^n$ is calculated, it is easy to calculate the residual $\bar{d}_h^n = f_h - L_h\bar{U}_h^n$. Because all the values are firstly calculated in the fine mesh, the values obtained must be transferred to the coarser grids. This operation of transfer of grid values from fine to coarse grids is termed *restriction*. In the figure above, I_h^H is the restriction operator. Consequently, the correction $\hat{v}_H^n$ obtained in the coarse

mesh must be transferred back to the fine mesh. This procedure is named prolongation with operator symbol I_H^h. When the corrected approximation $U_h^{n,afterCGC} = \bar{U}_h^n + \hat{v}_h^n$ is computed, the second basic process of multigrid, coarse grid correction (CGC), is completed.

Multigrid methods are obtained when this process is repeated over a sequence of fine to coarse grids. Multigrid methods have been created that have different grid cycling patterns: V-cycle, W-cycle and F-cycle and so on. The V-cycle and W-cycle are particularly popular. In this project, only V-cycle is considered.

In the two-grid cycle process, the choice of the six individual components, the smoothing procedure, the numbers *v*1 and *v2* of smoothing steps, the coarse grid Ω_H, the fine-to-coarse restriction operator I_h^H, the coarse grid operator and the coarse-to-fine prolongation operator I_h^H, may have a strong influence on the efficiency of the resulting algorithm. To construct the optimal algorithms for a practical problem, it is important to select the proper components.

The simplest form of restriction operator is injection, which identifies grid functions at coarse grid points with the corresponding grid functions at fine grid points. In general, the restriction operator may be formulated in terms of the weighted averages of neighbouring fine-grid values. Full weighting and half weighting are two choices that have different features. The full weighting provides better stability and convergence properties while the half weighting is computationally more efficient. Obviously, the full weighting involves all eight points adjacent to a given point (*i, j*). When the standard coarsening is employed, the full weighting scheme reads,

$$
\begin{aligned}
u_{2h(i+1,j+1)} &= (I_h^{2h} u_h)_{i+1,j+1} = \frac{1}{4} u_{h(2i+1,2j+1)} \\
&+ \frac{1}{8}[u_{h(2i+2,2j+1)} + u_{h(2i+1,2j+2)} + u_{h(2i,2j+1)} + u_{h(2i+1,2j)}] \\
&+ \frac{1}{16}[u_{h(2i+2,2j+2)} + u_{h(2i,2j+2)} + u_{h(2i+2,2j)} + u_{h(2i,2j)}].
\end{aligned} \tag{1.20}
$$

The half weighting is five points restriction operator derived from Eq. (1.20) by eliminating the influence of the four corner points and doubling the centre point influence.

For the prolongation operator, the simplest form is derived using linear interpolation. A very frequently used interpolation method is bilinear interpolation which is correspondent with Eq. (1.20). Nine points are involved so that the value at the cell centre can be obtained as the arithmetic mean of the four corner points as following:

$$
\begin{aligned}
u_{h(2i+1,2j+1)} &= (I_{2h}^{h} u_{2h})_{(2i+1,2j+1)} = u_{2h(i+1,j+1)} \\
u_{h(2i+2,2j+1)} &= (I_{2h}^{h} u_{2h})_{(2i+2,2j+1)} = \frac{1}{2}[u_{2h(i+1,j+1)} + u_{2h(i+2,j+1)}] \\
u_{h(2i+1,2j+2)} &= (I_{2h}^{h} u_{2h})_{(2i+1,2j+2)} = \frac{1}{2}[u_{2h(i+1,j+1)} + u_{2h(i+1,j+2)}] \\
u_{h(2i+2,2j+2)} &= (I_{2h}^{h} u_{2h})_{(2i+2,2j+2)} = \frac{1}{4}[u_{2h(i+1,j+1)} + u_{2h(i+2,j+1)} \\
&\qquad + u_{2h(i+1,j+2)} + u_{2h(i+2,j+2)}]
\end{aligned} \tag{1.21}
$$

Prolongation by this form introduces no ambiguity near boundaries even if the boundary conditions have been eliminated.

While multigrid methods are highly efficient solvers in their own right, they also serve as excellent preconditioners, and their use in this context makes the performance and robustness of the multigrid method less sensitive to the selection of components such at inter-grid transfers and coarse grid solvers. It is useful especially to combine

multigrid with acceleration technique for a large class of complicated real-life applications. Recently, multigrid procedures have been applied as preconditioners in Newton-Krylov methods; see Liu *et al*. (1998), Knoll and Mousseau (2000) and Pernice and Tocci (2000) for incompressible flows, Rider *et al*. (1999) for equilibrium radiation diffusion, Mousseau *et al*. (2000) for non-equilibrium radiation diffusion, Tidriri (1997) for compressible flows, and Chacón *et al*. (2000) for 2D Fokker-Planck algorithm. Oosterlee and Washio (1998) reported a comparison of multigrid as a solver and a preconditioner for singularly perturbed problems.

1.3 Objectives and Scope

A new approximate numerical method is designed based on the analysis of the effect of nonlinear terms in SINSE so that the residual error is reduced monotonically. Consequently, the first objective of this project is to investigate the monotonic convergent property. In this method SINSE are decomposed into two parts: linear part and nonlinear part. And the effort mainly focuses on the linear part. Even though the solution convergence rate can be affected by the problem parameters such as the Reynolds number, the mesh size, the numerical process also plays an important role to decide the efficiency of the computation. The use of simple iterative techniques of the linear part leads to a rather slow convergence rate for the solutions. Then one of the purposes of this thesis is to find an efficient linear solver to improve the overall computational efficiency of this method. Other than the efficiency, the accuracy is another essential aspect that must be concerned. Hence, the third target of this project is to improve the accuracy to higher-order.

Multigrid method is implemented in the current computational method. The various components of the multigrid procedure including the smoothing method, coarsening method, restriction operator and the prolongation operator, and the effect of the parameters will be investigated to optimize the combination.

In order to describe the convergent performance, this method will be compared with Newton's method with both the single grid and multigrid implementation.

A fourth-order difference scheme will be developed to compare with the basic second-order scheme. Consistent with the fourth-order discretization of the operator, the fourth-order accurate pressure boundary conditions are used.

Chapter 2 Algorithms and principles

This chapter describes the basic algorithms of the monotonic approximation method and provides some mathematical principles for solving the conservative incompressible Navier-Stokes equations. Both the second-order and fourth-order accurate discretization schemes are discussed. The generalized artificial dissipation is introduced to overcome the instability of the system. And some parameters that play important roles in the numerical procedure, such as diffusion coefficient, are presented here. The application of a multigrid procedure is also explained in some details here.

2.1 Algorithms of the monotonic approximate method

In solving the conservative INSE (1.1-1.3) on domain Ω, the nonlinear terms of the momentum equation introduce another challenge other than the difficulty of coping with the velocity-pressure coupling. Generally, Newton's method can be used to reduce the nonlinear equation to its local linear form (1.4). However, computing the Jacobian matrix F' is expensive most of the time. This monotonic approximation method was proposed by Liu (2002) based on an analysis of the nonlinear terms.

First, the INSE (1.1, 1.2) are discretized by using the second-order central difference scheme on a uniform mesh. In fact, with a multigrid solution technique, non-uniform mesh or grid-clustering coordinate transformations are not essential since local mesh refinement may be achieved by simply defining progressively finer grids in designated subdomains of the computation region if it is deemed necessary. The discretization of (1.1-1.3) leads to:

$$\frac{u_2^n(i+1,j)-u_2^n(i-1,j)}{2\Delta x_2}+\frac{u_3^n(i,j+1)-u_3^n(i,j-1)}{2\Delta x_3}=h_1^n, \tag{2.1}$$

$$\begin{aligned}&\frac{u_1^n(i+1,j)-u_1^n(i-1,j)}{2\Delta x_2}+\frac{u_2^n(i,j+1)u_3^n(i,j+1)-u_2^n(i,j-1)u_3^n(i,j-1)}{2\Delta x_3}\\&+\frac{\left[u_2^n(i+1,j)\right]^2-\left[u_2^n(i-1,j)\right]^2}{2\Delta x_2}+\frac{u_2^n(i+1,j)-2u_2^n(i,j)+u_2^n(i-1,j)}{\mathrm{Re}(\Delta x_2)^2}\\&+\frac{u_2^n(i,j+1)-2u_2^n(i,j)+u_2^n(i,j-1)}{\mathrm{Re}(\Delta x_3)^2}=h_2^n,\end{aligned} \tag{2.2}$$

$$\begin{aligned}&\frac{u(i,j+1)-u(i,j-1)}{2\Delta x}+\frac{u(i+1,j)u(i+1,j)-u(i-1,j)u(i-1,j)}{2\Delta x}\\&+\frac{\left[u(i,j+1)\right]-\left[u(i,j-1)\right]}{2\Delta x}+\frac{u(i+1,j)-2u(i,j)+u(i-1,j)}{\mathrm{Re}(\Delta x)}\\&+\frac{u(i,j+1)-2u(i,j)+u(i,j-1)}{\mathrm{Re}(\Delta x)}=h,\end{aligned} \tag{2.3}$$

where the superscript n represents the cycle step and h_1^n, h_2^n, h_3^n are the residual errors of the three equations in the nth numerical cycle. When the residual errors between consecutive iterations obey:

$$\left\|h_i^{n+1}\right\|_\Omega < \left\|h_i^n\right\|_\Omega \qquad \text{(for all } n\text{)} \tag{2.4}$$

where $\left\|h_m^n\right\|_\Omega = \max_\Omega\{\left|h_m^n\right|\}$ $(m=1,2,3)$ denotes the maximum norm of the residual errors h_m^n over the computational domain Ω, the numerical scheme is said to be monotonically decaying. Convergence is defined to occur when the norm of the residual error, $\|h\|_\Omega$, $\|h\|_\Omega = \max_{\Omega,m}\left\{\left\|h_m^n\right\|_\Omega\right\}$ $(m=1,2,3)$, which is the maximum error over the whole domain and equation, is less than the specified tolerance TOL.

$$\|h\|_\Omega < \mathrm{TOL} \tag{2.5a}$$

And a relative convergent criterion based on

$$\|h\|_\Omega < TOL\left\|h_m^1\right\|. \tag{2.5b}$$

where $\left\| h_m^1 \right\|$ is the maximum residual error in the first iteration and a TOL=10^{-8} is frequently implemented.

To rewrite the Eqs. (2.1-2.3) in incremental form, the sequences of pressure and velocities e_m^n are defined as

$$e_m^n = u_m^{n+1} - u_m^n \quad (m=1, 2,3) \tag{2.6}$$

Then, subtracting the equations (2.1-2.3) of the nth step from those of the n+1th step, the perturbation SINSE is obtained.

$$\frac{e_2^n(i+1,j)-e_2^n(i-1,j)}{2\Delta x_2}+\frac{e_3^n(i,j+1)-e_3^n(i,j-1)}{2\Delta x_3}=h_1^{n+1}-h_1^n \tag{2.7}$$

$$\begin{aligned}
&\frac{e_1^n(i+1,j)-e_1^n(i-1,j)}{2\Delta x_2}+\frac{u_2^n(i+1,j)e_2^n(i+1,j)-u_2^n(i-1,j)e_2^n(i-1,j)}{\Delta x_2}\\
&+\frac{u_2^n(i,j+1)e_3^n(i,j+1)-u_2^n(i,j-1)e_3^n(i,j-1)}{2\Delta x_3}+\frac{e_2^n(i,j+1)u_3^n(i,j+1)-e_2^n(i,j-1)u_3^n(i,j-1)}{2\Delta x_3}\\
&+\frac{e_2^n(i+1,j)-2e_2^n(i,j)+e_2^n(i-1,j)}{\mathrm{Re}(\Delta x_2)^2}+\frac{e_2^n(i,j+1)-2e_2^n(i,j)+e_2^n(i,j-1)}{\mathrm{Re}(\Delta x_3)^2}\\
&=h_2^{n+1}-h_2^n-(\frac{e_2^n(i,j+1)e_3^n(i,j+1)-e_2^n(i,j-1)e_3^n(i,j-1)}{2\Delta x_3}+\frac{\left[e_2^n(i+1,j)\right]^2-\left[e_2^n(i-1,j)\right]^2}{2\Delta x_2})
\end{aligned} \tag{2.8}$$

$$\begin{aligned}
&\frac{e_1^n(i,j+1)-e_1^n(i,j-1)}{2\Delta x_3}+\frac{u_2^n(i+1,j)e_3^n(i+1,j)-u_2^n(i-1,j)e_3^n(i-1,j)}{2\Delta x_2}+\\
&\frac{e_3^n(i,j+1)u_3^n(i,j+1)-e_3^n(i,j-1)u_3^n(i,j-1)}{\Delta x_3}+\frac{e_2^n(i+1,j)u_3^n(i+1,j)-e_2^n(i-1,j)u_3^n(i-1,j)}{2\Delta x_2}\\
&+\frac{e_3^n(i+1,j)-2e_3^n(i,j)+e_3^n(i-1,j)}{\mathrm{Re}(\Delta x_2)^2}+\frac{e_3^n(i,j+1)-2e_3^n(i,j)+e_3^n(i,j-1)}{\mathrm{Re}(\Delta x_3)^2}\\
&=h_3^{n+1}-h_3^n-(\frac{e_2^n(i+1,j)e_3^n(i+1,j)-e_2^n(i-1,j)e_3^n(i-1,j)}{2\Delta x_2}+\frac{\left[e_3^n(i,j+1)\right]^2-\left[e_3^n(i,j-1)\right]^2}{2\Delta x_3})
\end{aligned} \tag{2.9}$$

In Eqs. (2.8) and (2.9), it is obvious that the nonlinear convection terms on the right-hand side are homogeneous of order 2. As a result, when the residual errors h_m^n approach zero and the sequences e_m^n decrease correspondingly, the nonlinear terms will decay faster than the linear terms. This means the effect of these nonlinear terms will

become unimportant as the solution approaches convergence. However, the effects of the nonlinear terms could be very large when the approximation is far from the solution. Consequently, to find the solution u_m^n, the discretized Eqs. (2.8-2.9) are decomposed into two parts: linear and nonlinear part. The first or linear part is used to generate a correction e_m^n for u_m^n. The second part or nonlinear part is then used to determine a scaling constant *s*, such that the

$$u_m^{n+1} = u_m^n + se_m^n \tag{2.10}$$

possesses a smaller overall residual error. In developing the method, we define the following linear and nonlinear components of equations (2.7-2.9) as well as the additional function f_m^n, which are termed correction functions:

$$L_1^n e_m^n = \frac{e_2^n(i+1,j)-e_2^n(i-1,j)}{2\Delta x_2} + \frac{e_3^n(i,j+1)-e_3^n(i,j-1)}{2\Delta x_3} = f_1^n \tag{2.11}$$

$$\begin{aligned} L_2^n e_m^n = {} & \frac{e_1^n(i+1,j)-e_1^n(i-1,j)}{2\Delta x_2} + \frac{u_2^n(i+1,j)e_2^n(i+1,j)-u_2^n(i-1,j)e_2^n(i-1,j)}{\Delta x_2} \\ & + \frac{u_2^n(i,j+1)e_3^n(i,j+1)-u_2^n(i,j-1)e_3^n(i,j-1)}{2\Delta x_3} + \frac{e_2^n(i,j+1)u_3^n(i,j+1)-e_2^n(i,j-1)u_3^n(i,j-1)}{2\Delta x_3} \\ & + \frac{e_2^n(i+1,j)-2e_2^n(i,j)+e_2^n(i-1,j)}{\mathrm{Re}(\Delta x_2)^2} + \frac{e_2^n(i,j+1)-2e_2^n(i,j)+e_2^n(i,j-1)}{\mathrm{Re}(\Delta x_3)^2} = f_2^n \end{aligned} \tag{2.12}$$

$$\begin{aligned} L_3^n e_m^n = {} & \frac{e_1^n(i,j+1)-e_1^n(i,j-1)}{2\Delta x_3} + \frac{u_2^n(i+1,j)e_3^n(i+1,j)-u_2^n(i-1,j)e_3^n(i-1,j)}{2\Delta x_2} \\ & + \frac{e_3^n(i,j+1)u_3^n(i,j+1)-e_3^n(i,j-1)u_3^n(i,j-1)}{\Delta x_3} + \frac{e_2^n(i+1,j)u_3^n(i+1,j)-e_2^n(i-1,j)u_3^n(i-1,j)}{2\Delta x_2} \\ & + \frac{e_3^n(i+1,j)-2e_3^n(i,j)+e_3^n(i-1,j)}{\mathrm{Re}(\Delta x_2)^2} + \frac{e_3^n(i,j+1)-2e_3^n(i,j)+e_3^n(i,j-1)}{\mathrm{Re}(\Delta x_3)^2} = f_3^n \end{aligned} \tag{2.13}$$

$$g_2^n = \frac{e_2^n(i,j+1)e_3^n(i,j+1)-e_2^n(i,j-1)e_3^n(i,j-1)}{2\Delta x_3} + \frac{\left[e_2^n(i+1,j)\right]^2-\left[e_2^n(i-1,j)\right]^2}{2\Delta x_2} \tag{2.14}$$

$$g_3^n = \frac{e_2^n(i+1,j)e_3^n(i+1,j)-e_2^n(i-1,j)e_3^n(i-1,j)}{2\Delta x_2} + \frac{\left[e_3^n(i,j+1)\right]^2-\left[e_3^n(i,j-1)\right]^2}{2\Delta x_3} \tag{2.15}$$

Consequently, the incremental discretized Eqs. (2.8 and 2.9) can be rewritten as:

$$L_k^n e_m^n + g_k^n + h_k^n = h_k^{n+1},\ \text{(k=2, 3)} \tag{2.16}$$

while, Eq. (2.7) can be written as:

$$L_1^n e_m^n + h_1^n = h_1^{n+1} \tag{2.17}$$

In order to keep the scheme converging monotonically, the correction functions must be correctly posed. According to Liu (2002), the correction functions have to meet the following criteria.

For $x_k \in \Omega_1$ where $\left\|h_m^n\right\|_\Omega \geq \left|h_m^n\right| > \varepsilon \left\|h_m^n\right\|_\Omega \quad (0 < \varepsilon < 1)$, f_m^n satisfies:

$$\begin{aligned} &sign(f_m^n) = -sign(h_m^n) \\ &\varepsilon_1 \left\|h_m^n\right\|_\Omega \geq \left|f_m^n\right| > \varepsilon_2 \left\|h_m^n\right\|_\Omega \quad (0 < \varepsilon_2 < \varepsilon_1 < 1) \end{aligned} \tag{2.18}$$

And for $x_k \in \Omega_2$ where $\left|h_m^n\right| \leq \varepsilon \left\|h_m^n\right\|_\Omega$, f_m^n satisfies:

$$\left\|h_m^n\right\|_\Omega > \varepsilon_3 \left\|h_m^n\right\|_\Omega \geq \left|f_m^n\right| \quad (0 < \varepsilon_3 < 1) \tag{2.19}$$

Here, the domain decomposition is convenient for mathematical verification and provides more scope to handle the convergent problem. But in practice, the only key requirement for the correction function is that the signs of f_m^n are different from those of residual errors h_m^n, i.e. $sign(f_m^n) = -sign(h_m^n)$. Three kinds of correction functions, numbered 1 to 3, satisfying the requirements are used in this study.

CF 1: $f_m^n = -h_m^n$ (2.20)

CF 2: $f_m^n = -bh_m^n \qquad 0 < b < 1$ (2.21)

CF 3: $$\begin{cases} f_m^n = -\left\|h_m^n\right\|_\Omega * sign(h_m^n) * \alpha & \left(\beta \left\|h_m^n\right\| < \left|h_m^n\right|\right) \\ f_m^n = -h_m^n & \left(\beta \left\|h_m^n\right\| > \left|h_m^n\right|\right) \end{cases} \tag{2.22}$$

CF is the abbreviation for correction function. The parameters, b, $0.5 < \alpha < 1$ and $0 < \beta < 1$, may be turned to find the optimal/good convergence behaviour.

Besides the correction function, another important parameter to control the monotonical property is the scale factor s, which was introduced earlier to modify the increment e_m^n. The scale factor s is not a constant. It changes with n as computation proceeds. In the first case (2.20), the value of s is always equal to one, which results in the Eqs. (2.11-2.13) being just the Newton equations. For CFs 2 and 3, the choice of s is governed by the nonlinear terms. Then, with a suitable value of s ($0 < s \le 1$) to correct the increment e_m^n, the function (2.16) can be written as:

$$L_k^n s e_m^n + g_k^n s^2 + h_k^n = h_k^{n+1} \tag{2.23}$$

From Eq. (2.18), as long as the nonlinear terms g_k^n are bounded, there exists a constant C such that $L_k^n s e_m^n + g_k^n s^2$ has the same sign as $L_k^n s e_m^n$ when $0 < s < C$. And considering that the signs of f_m^n are different from that of the residual errors h_m^n, it is obvious that the absolute value of h_m^{n+1} will be smaller than that of h_m^n. If the second case, $f_m^n = -bh_m^n, 0 < b < 1$, is taken as an example, the Eq. (2.23) is turned into:

$$\begin{aligned} h_k^{n+1} &= L_k^n s e_m^n + g_k^n s^2 + h_k^n \\ &= s \cdot f_k^n + s^2 \cdot g_k^n + h_k^n \\ &= -bsh_k^n + s^2 \cdot g_k^n + h_k^n. \end{aligned}$$

By the Cauchy-Schwarz inequality,

$$\begin{aligned}\left\|h_k^{n+1}\right\|_\Omega &= \left\|s\cdot f_k^n + s^2\cdot g_k^n + h_k^n\right\|_\Omega \\ &\le \left\|s\cdot f_k^n + h_k^n\right\|_\Omega + s^2\cdot\left\|g_k^n\right\|_\Omega \\ &= \left|1-bs\right|\cdot\left\|h_k^n\right\|_\Omega + s^2\cdot\left\|g_k^n\right\|_\Omega \\ &= (\left|1-bs\right| + s^2\cdot\frac{\left\|g_k^n\right\|_\Omega}{\left\|h_k^n\right\|_\Omega})\cdot\left\|h_k^n\right\|_\Omega .\end{aligned}$$

For all n, to get $\left\|h_i^{n+1}\right\|_\Omega < \left\|h_i^n\right\|_\Omega$, it requires that

$$\left|1-bs\right| + s^2\cdot\frac{\left\|g_k^n\right\|_\Omega}{\left\|h_k^n\right\|_\Omega} < 1.$$

This implies that for $0 < s \le 1$,

$$s < b\frac{\left\|h_k^n\right\|_\Omega}{\left\|g_k^n\right\|_\Omega}.$$

Hence, the constant *s* can be set to

$$s = \varepsilon_2 \min(1.0, \frac{b\left\|h_k^n\right\|_\Omega}{\left\|g_k^n\right\|_\Omega}) \tag{2.24}$$

where $\left\|g_k^n\right\|_\Omega$ is the maximum norm of the values g_k^n and $0 < \varepsilon_2 < 1$ is an additional parameter that may be involved to control the value of *s*. It is obvious that the constant *s* is a critical parameter to keep the residual error converging monotonically.

2.2 Generalized dissipation scheme

The flow equations are discretized by using central difference scheme and the algorithm introduced in last paragraph offers a novel way to deal with the nonlinear terms in the momentum equations. However, it is still a big challenge to solve the

linear perturbation Eqs. (2.11-2.13) because the correction of pressure e_1^n is uncoupled between the continuity perturbation Eq. (2.11) and the linear momentum operator Eqs. (2.12 and 2.13). A generalized dissipation scheme is applied here, which serves both to connect the correction pressure e_1^n to the perturbation velocities (e_2^n, e_3^n) as well as to produce artificial dissipation to control stability of numerical procedure.

$$L_1^n e_m^n + CUI\nabla^2 e_1^n = \overline{f_1^n} \tag{2.25}$$

$$L_k^n e_m^n + MCUI\nabla^2 e_k^n = \overline{f_k^n} \tag{2.26}$$

Here, *CUI* and *MCUI* are the damping factors to control the artificial dissipation terms. For momentum Eq. (2.26), those artificial diffusion terms can provide additional dissipation to suppress numerical spurious oscillation. Meanwhile, this ensures diagonal dominance for the resulting algebraic equations, thus lending the necessary stability property to the evolving solutions. And for the discretized continuity Eq. (2.1), the artificial pressure diffusion term $\nabla^2 e_1^n$ provides the needed coupling of the pressure correction field $e_1^n(x_2, x_3)$ to the velocity correction field (e_2^n, e_3^n). Hence, the pressure correction e_1^n can be obtained by solving the resultant Poisson equation if the velocity is treated explicitly. Besides, the artificial diffusion scheme will not contaminate the final physical solutions because the artificial diffusion terms are added in incremental forms, which means they are going to approach zero when the residual errors h_m^n and the increments e_m^n decrease to less than the accepted tolerance of convergence.

In the present algorithm, the artificial dissipation terms play an important role in the stabilization of discretization and they do not affect the accuracy which is determined here by satisfaction of residual condition to required tolerance. However, they may

affect the operation of the original correction function f_k^n, resulting in potential loss of monotonic convergence property.

$$f_1^n = \overline{f_1^n} - CUI\nabla^2 e_1^n \tag{2.27}$$

$$f_k^n = \overline{f_k^n} - MCUI\nabla^2 e_k^n \tag{2.28}$$

Hence the selection of the damping parameters *MCUI* and *CUI* must be carefully done. They have to be chosen small enough to maintain a good convergent behaviour but large enough so that the discrete system becomes sufficiently stable. For high Reynolds numbers, the unstable influence of the dominating advection terms have to be suppressed by increasing *MCUI* and *CUI*.

2.3 Fourth-order refinement

The algorithms introduced above are all based on second-order accurate central difference. It is well known that the performance of iterative methods is sensitive to the number of equations to be solved, the type of boundary conditions applied and other factors. In particular, if the number of equations or the Reynolds number increases, the rate of convergence of an iterative procedure often deteriorates. Hence, an increase in the number of equations to be solved is associated with a higher cost per iteration, thereby limiting the practical size of the problem that can be solved. Applying a higher-order method which decreases the number of equations while preserving high accuracy can partially alleviate this problem. Hence, a fourth-order discretized difference scheme rather than a second-order scheme can be used to reduce the number of equations significantly. A unique feature of the present fourth-order scheme is that only the residual errors h_m^n as given by (2.1-2.3) are computed to

fourth-order while the rest of the operators are maintained at second-order. It does not matter that the other parts of the scheme are retained at second-order, since the final solution will be fourth-order if convergence is governed by the fourth-order residual errors meeting the set tolerance. The key is that the convergence is actually attained. Consequently, we have also termed the present higher-order procedure a fourth-order refinement. The refinement scheme helps to reduce the computational work to obtain high accuracy solutions since much of the work is done via lower-order operators. The fourth-order refinement incorporates full fourth-order boundary conditions.

It is straightforward to refine the second to fourth order accuracy without major changes to the second-order scheme. The fourth-order central difference for all the derivatives may be written as:

$$\frac{\partial^2 u}{\partial x^2} = \frac{-u_{i+2} + 16u_{i+1} - 30u_i + 16u_{i-1} - u_{i-2}}{12\Delta x^2}$$
$$\frac{\partial u}{\partial x} = \frac{-u_{i+2} + 8u_{i+1} - 8u_{i-1} + u_{i-2}}{12\Delta x} \tag{2.29}$$

The fourth-order residual errors h_m^n will be obtained if all the derivatives in three governing equations and the boundary conditions are replaced by the appropriate fourth-order approximations. During the procedure of computation, accuracy and efficiency are the two most important issues. Truncation error, as well as the round-off errors, combines to compromise the accuracy of the results. And the truncation error is the main consideration for accuracy when the tolerance for convergence is set to be very low. Compared with second-order discretization, fourth-order accurate discretization requires slightly more work in one iterative cycle; however, it keeps a high accuracy with less number of equations. The comparison between second-order

and fourth-order finite difference is made in subsequent numerical studies. The fourth-order refinement is particularly useful if we are seeking for highly accurate solutions.

2.4 Boundary condition

The INSE requires no a pri-boundary conditions on the pressure. Only the velocity boundary conditions are sufficient for the determination of both velocity and pressure (Gresho and Sani, 1987; Koh, 2000). The algorithm presented in this thesis, the Poisson equation for the pressure correction in Eq. (2.25), splits the computation of the pressure from the computation of velocity. Then, a boundary condition for the pressure must be specified. And the numerical computations indicate that the accuracy and stability are also affected by the specific choice of pressure boundary condition.

The simplest boundary condition for the pressure field p is:

$$\frac{\partial u_1}{\partial x} = 0. \tag{2.30}$$

which was used by Liu (2002). Condition (2.30) provides a low-order approximation of the pressure boundary condition, which is not satisfactory in many applications.

According to Gresho and Sani (1987), the alternative choice is the Neumann boundary condition obtained by applying the normal component of the momentum equations on the boundary. Thus, the necessary pressure boundary condition can be derived from the normal and tangential components of velocities.

If the x-momentum equation (1.2) is applied on the y-boundary, the conditions on the left and right boundary walls are obtained as,

$$\left.\frac{\partial u_1}{\partial x_2}\right|_{wall} = -\frac{\partial (u_2)^2}{\partial x_2} + \frac{1}{\mathrm{Re}}\frac{\partial^2 u_2}{\partial (x_2)^2}, \tag{2.31}$$

Similarly, the Neumann boundary conditions on the top and bottom boundary walls may be derived from the *y*-momentum equation (1.3) as,

$$\left.\frac{\partial u_1}{\partial x_3}\right|_{wall} = -\frac{\partial (u_3)^2}{\partial x_3} + \frac{1}{\mathrm{Re}}\frac{\partial^2 u_3}{\partial (x_3)^2}. \tag{2.32}$$

where we note that u_2=u_3=0 at a stationary wall, but $\partial\left(u_2\right)^2/\partial x_2$ and $\partial\left(u_3\right)^2/\partial x_3$ could not be zero. To discretize those Eqs. (2.31 and 2.32), second-order and fourth-order finite difference schemes corresponding to the required accuracy of the residual errors are employed. The second-order discretization of the pressure boundary condition at the left wall is obtained as follows. The pressure on the left wall is given by Taylor expansion as:

$$u_1(1, j) = \frac{4u_1(2, j) - u_1(3, j)}{3} - \frac{2h}{3}\left.\frac{\partial u_1}{\partial x_2}\right|_{1, j} \tag{2.33}$$

The terms of $\left.\frac{\partial u_1}{\partial x_2}\right|_{(1, j)}$ given by (2.31) are then evaluated by second-order differences as:

$$\left.\frac{\partial (u_2)^2}{\partial x_2}\right|_{1, j} = \frac{-2[u_2(3, j)]^2 + 4[u_2(2, j)]^2 - 3[u_2(1, j)]^2}{2h}, \tag{2.34}$$

$$\left.\frac{\partial^2 u_2}{\partial (x_2)^2}\right|_{1, j} = \frac{2u_2(1, j) - 5u_2(2, j) + 4u_2(3, j) - u_2(4, j)}{h^2}. \tag{2.35}$$

Then, the second-order pressure boundary on the left wall is obtained through reorganizing the Eqs. (2.31, 2.33-2.35).

The fourth-order pressure boundary on the left wall can be similarly obtained. The pressure on the left wall is obtained if Eq. (2.31) is given by:

$$u_1(1,j) = \frac{48u_1(2,j) - 36u_1(3,j) + 16u_1(4,j) - 3u_1(5,j)}{25} - 12h\left.\frac{\partial u_1}{\partial x_2}\right|_{1,j} \tag{2.36}$$

where, the terms of $\left.\frac{\partial u_1}{\partial x_2}\right|_{(1,j)}$ given by (2.31) are evaluated based on fourth-order differences as:

$$\left.\frac{\partial (u_2)^2}{\partial x_2}\right|_{1,j} = \frac{-25[u_2(1,j)]^2 + 48[u_2(2,j)]^2 - 36[u_2(3,j)]^2 + 16[u_2(4,j)]^2 - 3[u_2(5,j)]^2}{12h}, \tag{2.37}$$

$$\left.\frac{\partial^2 u_2}{\partial (x_2)^2}\right|_{1,j} = \frac{45u_2(1,j) - 154u_2(2,j) + 214u_2(3,j) - 156u_2(4,j) + 61u_2(5,j) - 10u_2(5,j)}{12h^2}. \tag{2.38}$$

For fourth-order scheme, a separate set of equation is also needed for the first interior node next to a wall.

2.5 Relaxation scheme and multigrid procedure

In the current application, the operator algorithm produces a set of linear, elliptic, scalar equations (see Eqs. 2.25 and 2.26) that need to be solved. Because the multigrid technique is the fastest numerical method to solve elliptic equations, it is adopted as the first choice in this project to improve the convergence rate.

In the multigrid method, the role of the iterative method is not so much to reduce the error as to smooth it. Hence, the most important criterion to choose the iterative relaxation scheme is its ability to eliminate the high-frequency error components. The SOR method gives rapid reduction of the corresponding high frequency components so that it is suitable for error smoothing. And the smoothing properties will turn out to be dependent on the right choice of relaxation parameters. The solution on the each grid is obtained with a fixed number of sweeps of the smoother except possibly for the

coarsest grid. On the coarsest grid any method, including direct solution, may be used if only this method has sufficiently good convergence properties.

In this project, the multigrid uses a simple "V" cycle (see Figure 2.1) and incorporates SOR as a smoother.

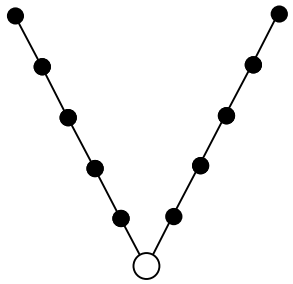


Fig 2.1 Structure of a six level multigrid V cycle

In Fig. 2.1, every black point represents one grid where the solutions are obtained after some suitable numbers of SOR iteration. The circle means some other smoothing procedure may be used on the coarsest grid. SSOR and direct methods are investigated here. However, there are no significant improvement in the computational cost using these two methods at the coarsest grid in this study. In a multigrid process, local mesh refinement may be achieved by defining progressively finer grids in designated subdomains of the computation region. Then uniform mesh with standard 2h coarsening is applied in the main bulk of our study, 4h coarsening is applied in selected case studies (see Fig. 2.2):

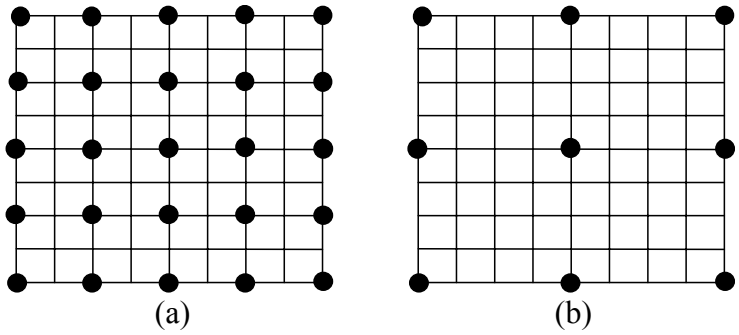


Fig 2.2 (a) Standard 2h and (b) 4h-coarsening of a uniform mesh

Although these choices may not be optimal for the multigrid as a solver, they turn out to be acceptable approaches. For the restriction operator, the variables are always restricted using the optimal weighting method. But for the residual errors, both optimal and full weighting are applied to compare their effects on the convergence performance. For the prolongation operator, bilinear operator given in the Eq. (1.22) is employed throughout the multigrid procedure.

In this solution technique, a kind of inexact Newton method handles the nonlinear terms. The artificial pressure diffusion is added to couple the system and make the numerical procedure stable. The multigrid algorithm solves the linear elliptic incremental equations. Implementing more sophisticated multigrid strategies and measuring their effects on CPU time and convergence are areas for future research.

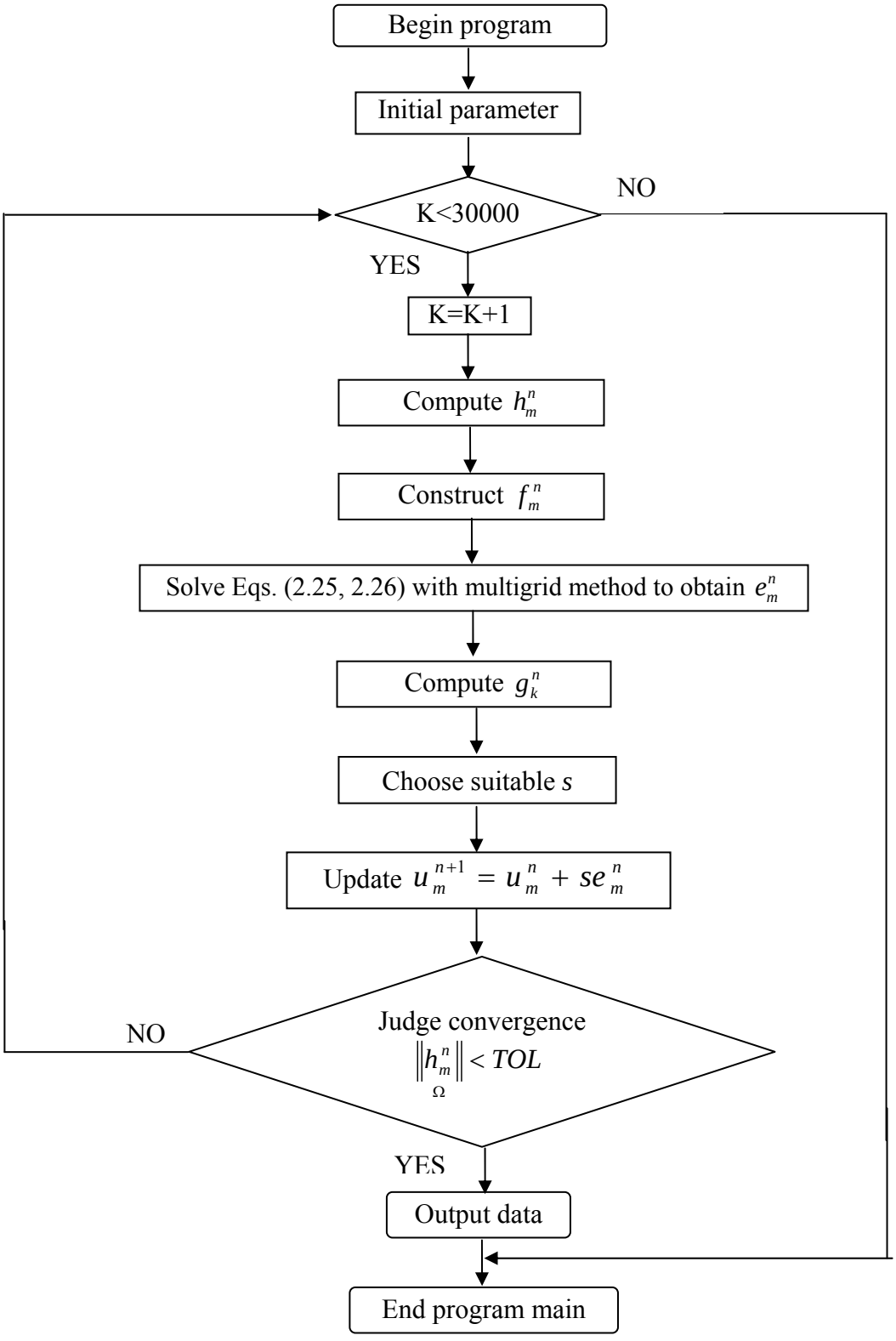
2.6 Numerical procedure

Based on the aforementioned description of the principle, the computational sequence of each iteration cycle is listed below:

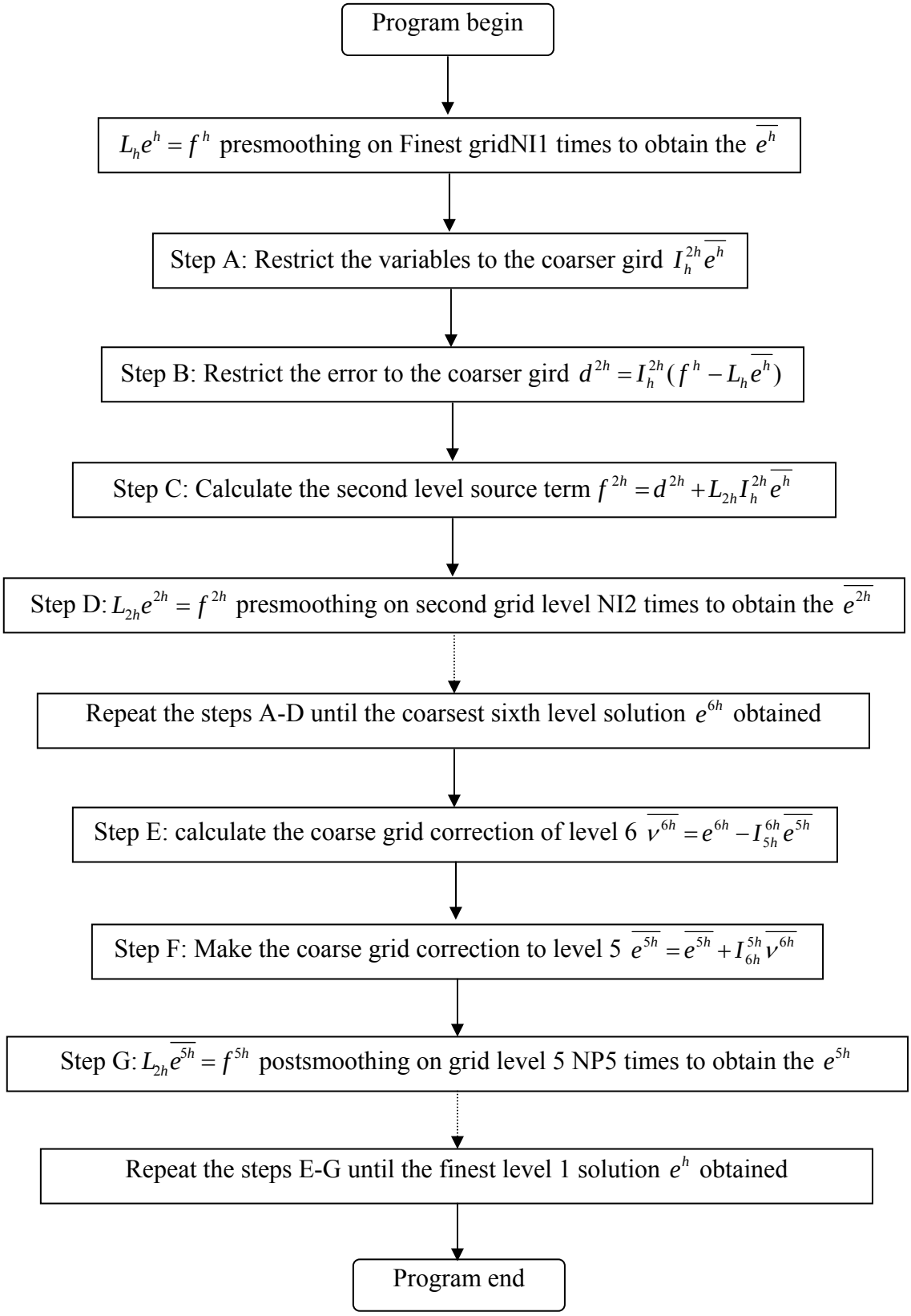
1. Input initial data and set up initial flow field;

2. Compute the residual errors h_m^n from the Eqs. (2.1 and 2.3) (h_m^n is computed to fourth-order for the fourth-order requirement scheme.);
3. Construct the correction function f_m^n subject to the specifications (2.18) and (2.19);
4. Solve the Eq. (2.25) to obtain e_1^n using multigrid method;
5. Solve the Eq. (2.26) to obtain e_2^n and e_3^n using multigrid method;
6. Compute g_k^n from Eqs. (2.14 and 2.15);
7. Compute the value of $\left\| h_m^n \right\|_\Omega$ and $\left\| g_k^n \right\|_\Omega$, according the Eq. (2.24) to choose a proper constant *s*;
8. Compute $u_m^{n+1} = u_m^n + se_m^n$ to obtain the n+1th step variables;
9. Check for convergence and return to step 2 if necessary until the convergent criteria are satisfied.

The following flow chart summarizes the above computational steps as one outer-loop iteration:



The computational flowchart of one multigrid cycle to solve $L_h e^h = f^h$ is shown as:



Chapter 3: Numerical evaluations

In this chapter, we will carry out a systematic numerical evaluation of the monotonic approximation scheme and the various extensions that we have described in Chapter 2. These will be done in the context of the standard test problem of two-dimensional incompressible fluid flows in a driven square cavity, whose geometry is depicted schematically in Figure 3.1.

Numerical issues and performance of the method and its extensions in terms of the following are presented and discussed:

- Monotonicity
- Stability and convergence
- Rate of convergence and convergence history
- Accuracy
- Multigrid acceleration
- Parameter optimization
- Fourth-order refinement

Comparisons are made with other published results where possible.

3.1 Physical and numerical parameters of the test problems

The monotonic approximation numerical method is applied to the two-dimensional driven cavity flow. The driven cavity is a classic fluid dynamics benchmark that is widely used as standard test cases for evaluating the stability and accuracy of numerical methods for incompressible flow problems. Figure 3.1 shows the geometry

and the boundary conditions for the flow in a driven cavity together with the appropriate nomenclature. The programs are implemented on 3.06 GHz Pentium IV Xeon CPUs of NUS computer centre. As the machines are running under shared environment, the CPU time that we presented later should be taken as indicative value than absolute.

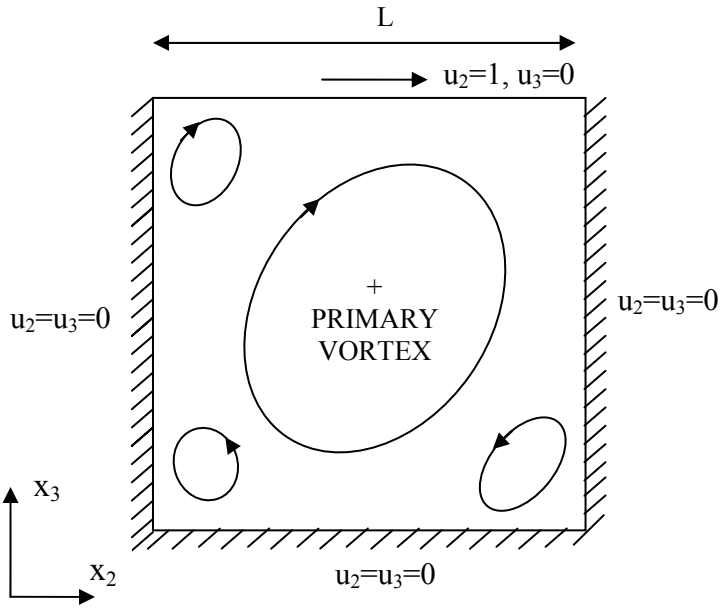


Fig. 3.1 Geometry of the driven cavity flow

The parameters appearing in this program are defined as follows.

CUI=Diffusion coefficient for the continuity equation,

MCUI= Diffusion coefficient for the momentum equations,

W= overrelaxation parameter in the SOR method,

Re=Reynolds number,

NI1, NI2, NI3, NI4, NI5, NI6 are the numbers of sweeps of the smoother at the 1^{st}, 2^{nd},…, 6^{th} multigrid level respectively, with 1^{st} representing the finest grid level,

N=mesh size,

3.2 Monotonic scheme on a single grid

From the design of algorithm described in Chapter 2, it is clear that this approximation method may produce residual error $\|h\|_\Omega$ that converges monotonically if suitable parameters are selected. This property is approximately demonstrated in Figure 3.2. As can be seen, the convergence behaviour is monotonic except for slight oscillations at the beginning. There are two causes for the slight oscillations. One is the incomplete convergence of the linear part. In order to save the computational cost, the linear part iteration employs only 100 SOR sweeps for the single-grid cases and one 6 level V-cycle for multigrid cases. The other reason is the addition of the generalized artificial dissipation terms in equations (2.27) and (2.28), which are not part of the original monotonic theory. In these figures, the CFs 1 to 3 refer to the three correction functions.

When the CF 1 is used, s is set to be one throughout the computation. Then the equations (2.12) and (2.13) are reduced to the Newton equations. Correspondingly, the other two correction functions (2 and 3) can be thought as inexact or modified Newton methods. The history of scale factor s is included in Fig 3.3. It can be seen that s tends to 1.0 fairly quickly, typically in less than 30 steps. In order to save the computation cost, s is set to be 1.0 after a set number of outer iterations instead of being obtained through equation (2.24). As reviewed in Chapter 1, Newton's method has very good convergence property when the initial guess is sufficiently close to the solution. Otherwise, the computation may fail. Fortunately, this restriction can be improved by using the present error reduction scheme with the correction functions 2 and 3. As shown in Figure 3.4, the flow at Re =1000 is calculated using a uniform mesh with 129×129 grid points and standard initial condition, which is

$(u_1, u_2, u_3) \equiv (0,0,0)$ for all $x=(x_2,x_3) \in \Omega$ at the beginning. It is evident that the convergent behaviours for correction functions 2 and 3 are much less oscillatory than that of for correction function 1 (with constant s=1) which corresponds to Newton's method. At Re=5000 with standard initialization and grid size of 129×129, Newton's method (correction function 1 and s=1) can not converge, but the present error reduction scheme converges with correction function 2 and 3. The term CF 1 will be used synonymously with Newton's method.

In order to further compare the convergence property of the proposed scheme with Newton's method, flow at Re=5000 is simulated in the uniform mesh 129×129 with initial condition (u_1, u_2, u_3) randomly generated between (-0.1, 0.1). The convergence history is shown in Figure 3.4. Only two lines are shown here because Newton's method also fails to converge. The random initial condition makes the convergence history line slightly more oscillatory than those initialized with 0. However, the total iteration to convergence is almost the same. The higher stability of the present method than the Newton's method can be verified from the Tables 3.1-3.2. It can also be seen in Figures 3.2-3.4 that correction function 3 exhibits better performance than correction function 2 in single grid computation. As shown in Tables 3.1 and 3.2, at Re=1000 with mesh size 65×65, the total outer loop iteration number is 246 for CF 2 while for CF 3 the outer loop iteration number is 221, about 10% better.

It is obvious from Figure 3.5 that the larger the value of *b* in the correction function 2, the better the convergence behaviour will be. This property can also be found in the Table 3.3. The Newton's method converges fastest with iterative number of 899 and CPU time of 156.7 s followed by the CF 2 with *b* value equal to 0.9. When *b* is equal

to 0.7, the iterative number is almost one and half times more than that of the Newton's method. However, Newton's method is most seriously oscillatory in these 4 CFs as can be seen in Figure 3.5. Indeed for relative convergence tolerance of 10^{-6}, Newton's method is slower in convergence than CF with b=0.7, 0.8 and 0.9. With the value *b* becoming larger, it is more like the Newton's method, which means the convergence performance is more sensitive to the parameters and fails easily when multigrid is applied in solving the linear equations. Thus, it is important to select a suitable value of *b* to guarantee the convergence and good performance. In this project, the value of *b* is set at 0.7 unless otherwise indication.

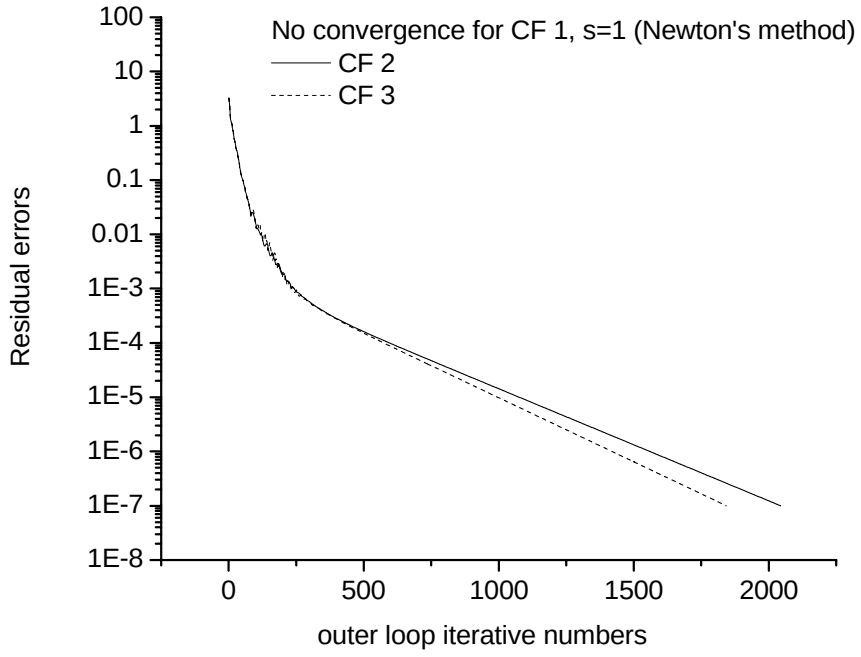


Fig. 3.2 Comparison of convergence history for the three correction functions using single grid. (mesh size of 129×129, W=0.23, α=0.5, β=0.9, CUI=MCUI=0.12, Re=5000)

At Re=1000, the results for the grid of 129×129 and 531×531 are presented in Figures 3.6 and 3.7, respectively. It can be observed that they are almost identical and also indistinguishable with the results of Schreiber and Keller (1983), meaning that the finest grid of 129×129 is adequate for Re=1000. And among all the parameters, the

finest mesh size is the most important parameter. With CUI=0.08, if the mesh size is larger than 45×45 at Re=1000, the results remain satisfactory.

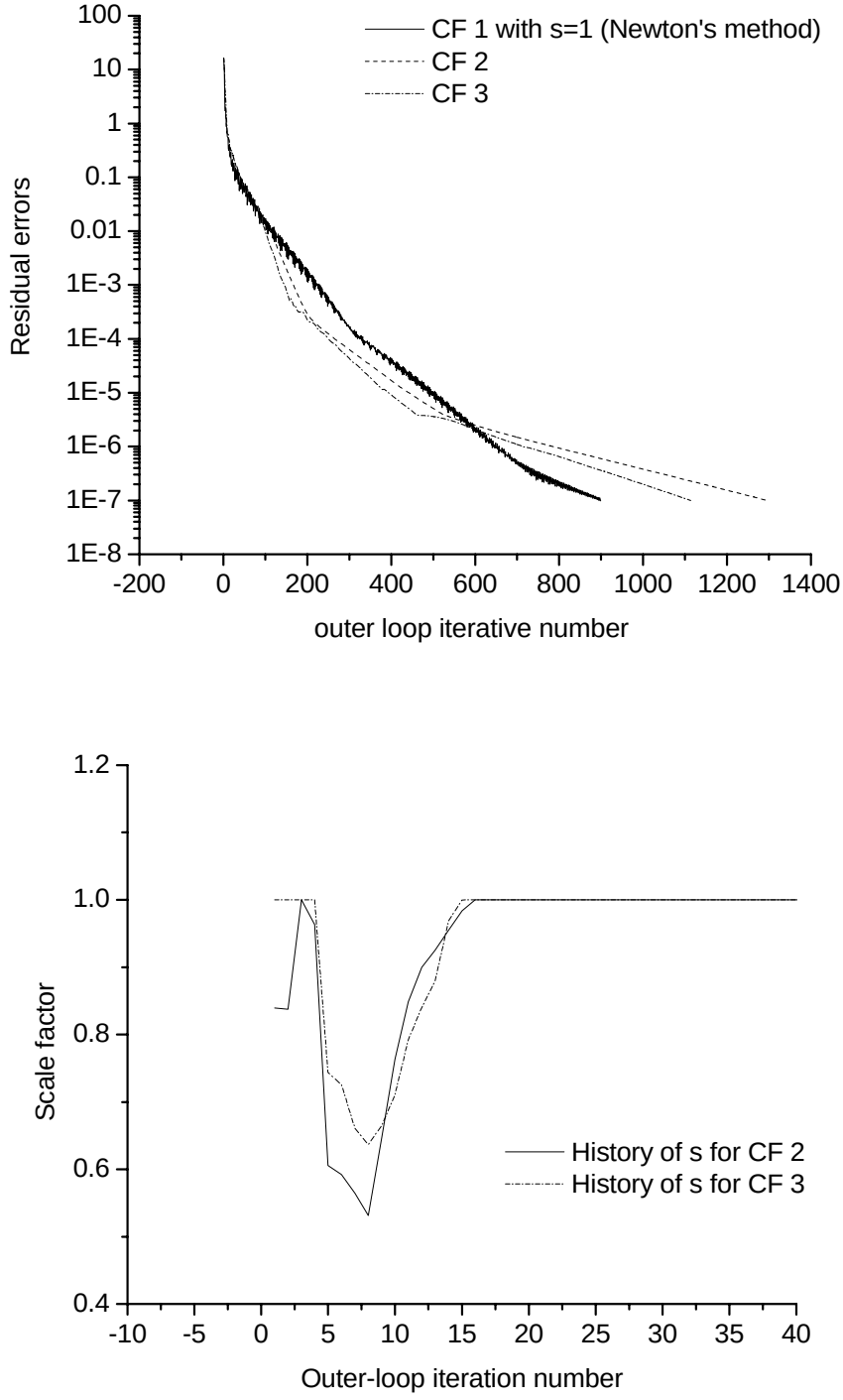


Fig. 3.3 Comparison of convergence history and the history of scale factor for the three correction functions using single grid. (mesh size of 129×129, W=0.23, CUI=MCUI=0.10, Re=1000)

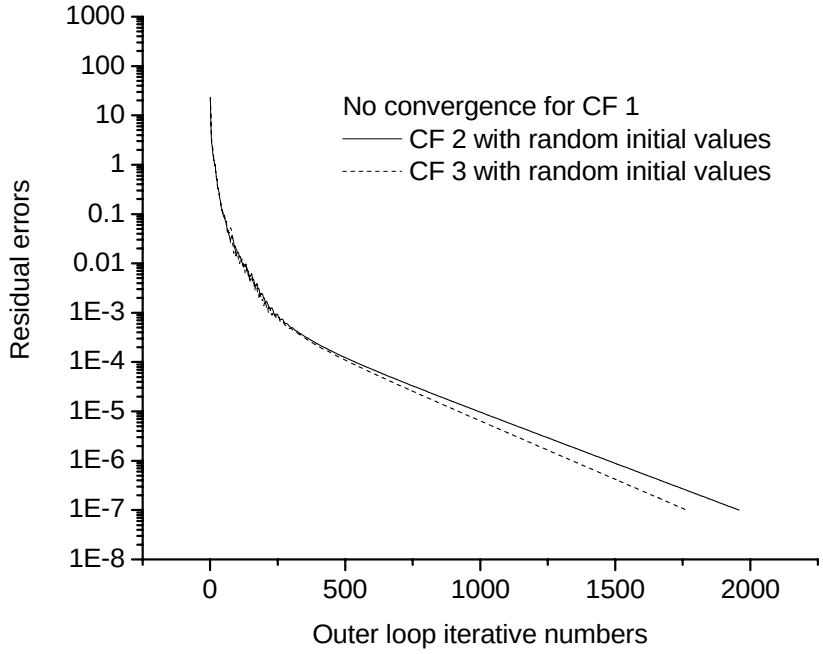


Fig. 3.4 Comparison of convergence history for the three correction functions with random initial values using single grid. (mesh size of 129×129, W=0.23, α=0.5, β=0.9, CUI=MCUI=0.12, Re=5000)

Table 3.1 Comparison of three correction functions for single grid at Re=1000, mesh size of 65×65, CUI=MCUI=0.07

Re=1000 Mesh size: 65×65 CUI=MCUI=0.07	CF 1	CF 2		CF 3	
		No. of Iteration	CPU time (s)	No. of Iteration	CPU time (s)
Single grid	F	246	13.4	221	13.1

Table 3.2 Comparison of CPU time for the three correction functions of multigrid (2D finest mesh size of 129×129, W=0.23, CUI=MCUI=0.12, Re=5000)

	CF 1	CF 2	CF 3
Multigrid 6L	F	43 (s)	39.5 (s)
Single grid	F	391 (s)	314 (s)

Table 3.3 Comparison with different values of *b* and Newton's method (mesh size of 129×129, W=0.23, CUI=MCUI=0.10, Re=1000)

		b=0.7		*b*=0.8		*b*=0.9		Newton's method	
		Iterative number	CPU time (s)	Iterative number	CPU time (s)	Iterative number	CPU time (s)	Iterative number	CPU time (s)
Single gird		1295	225.8	1137	193.9	1014	174.0	899	156.7
Multigrid	1	1293	84.1	F		F		F	
	2	1368	66.4						

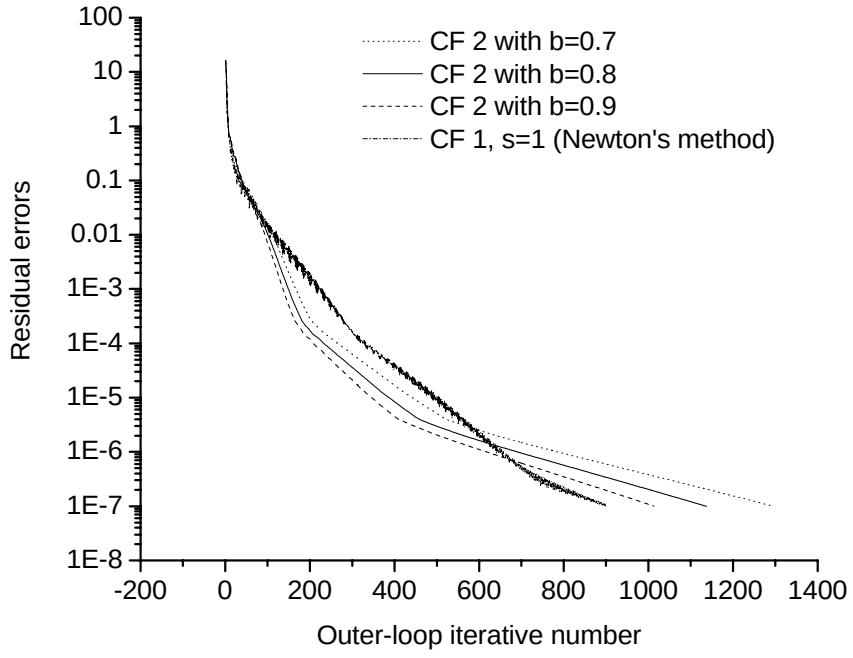


Fig 3.5 Comparison of convergence behaviour with different values of *b* and Newton's method (mesh size of 129×129, W=0.23, CUI=MCUI=0.10, Re=1000)

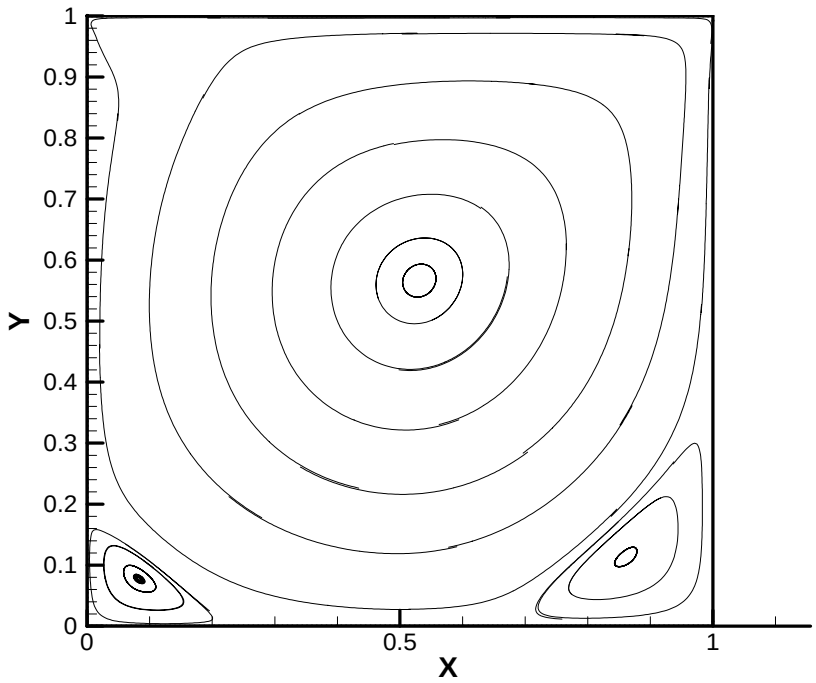


Fig. 3.6 Streamline pattern (Re=1000, Finest grid 129×129, W=0.23, CF 2, CUI=MCUI=0.10)

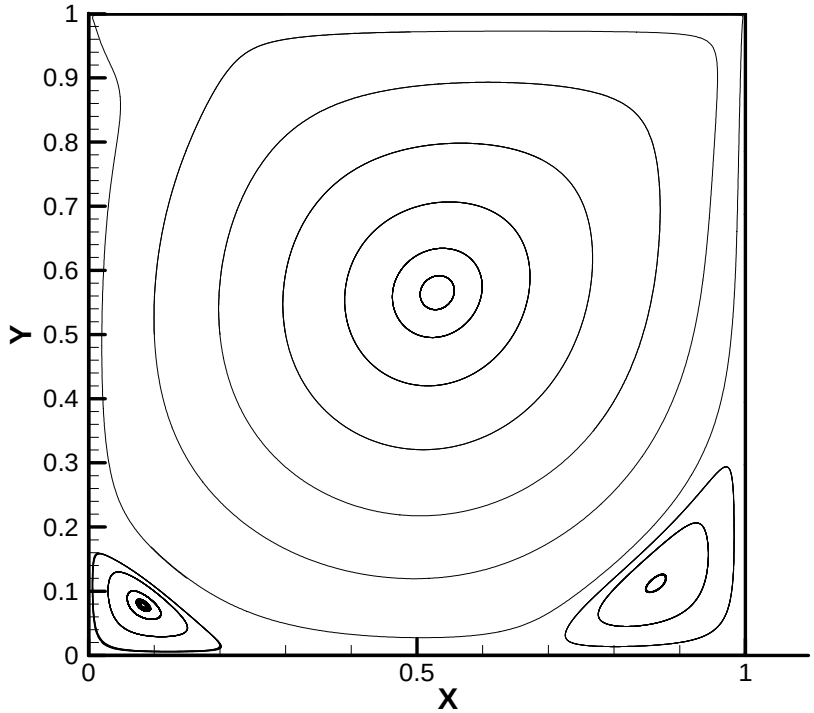


Fig. 3.7 Streamline pattern (Re=1000, Finest grid 531×531, W=0.23, CF 2, CUI=MCUI=0.10)

3.3 Multigrid acceleration of the monotonic scheme

In this section, we will investigate the improvements that multigrid has over the single grid with second-order accuracy. To determine the overall accuracy of the scheme, Re=1000 is taken as an example. Five calculations are performed in a series of mesh sizes with identical parameters. Because it is impossible to find the exact solution of U in the domain, the solution at a relatively much large mesh size, e.g. 481×481, is used as a reference. The variation of the maximum error in U with mesh refinement is plotted in Figure 3.8. Here, $\mathrm{EU} = \max_{\Omega} \{ \left| U^{\mathrm{N}} - U \right| \}$ and N is the mesh sizes. From this figure, we can see that the scheme is indeed of second-order accuracy with a very light correlation coefficient of nearly 1.0. The small deviation from the 2.0 slope may be due to the fact that the reference solution is not the exact analytical solution but a numerical solution obtained from the much finer mesh.

Figure 3.9 shows the distribution of vertical flow velocity *v* along the horizontal line passing through the centre of the cavity at (0.5, 0.5) for Re=5000. A magnified view of right minima is also given. The distribution of horizontal flow velocity *u* along a vertical line passing through the geometric centre of the box and a magnified view of left minima corner are shown in Fig. 3.10. In both figures, the results from Ghia *et al*. (1982) are taken as the reference, in which the local maxima and minima of *v*-velocity along the horizontal line passing through the geometric centre of cavity are 0.43648 and -0.55408 for Re=5000 respectively. It is evident that the present results are very close to theirs. In our case with 129×129, the local maxima and minima of *v*-velocity along the horizontal line passing through the geometric centre of cavity are 0.41833 and -0.51901 respectively. Furthermore, the present maxima and minima values also

occur at the same spatial points as in Ghia et al. Especially, for the mesh size of 257×257, the results are in excellent agreement with theirs.

Table 3.4 Maximum error for various mesh sizes (Re=1000, CUI=MCUI=0.10, W=0.23, CF 2)

	Mesh sizes	$\left\|U - U^{ref}\right\|_{\Omega}$
Reference	418	—
1	161	0.00663
2	121	0.01205
3	97	0.01865
4	81	0.02629
5	61	0.04449

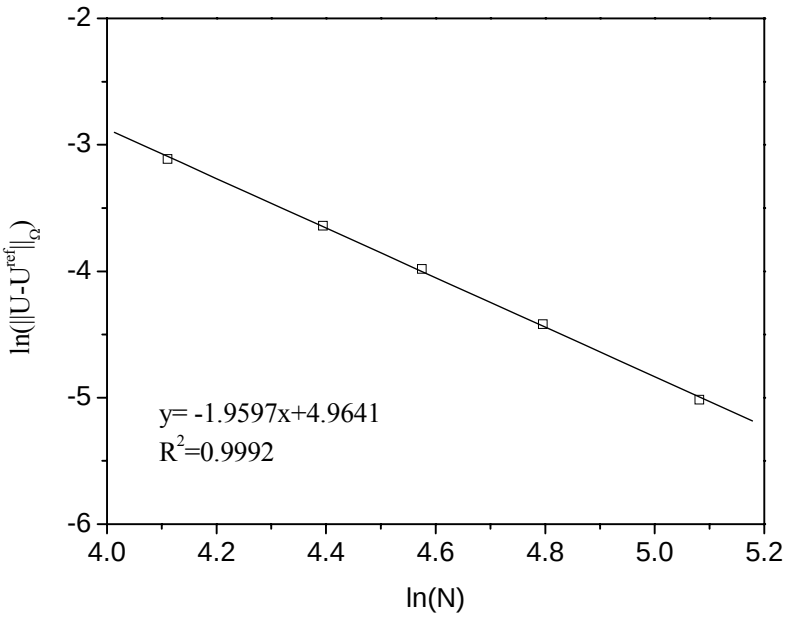


Fig 3.8 Maximum error as a function of mesh size

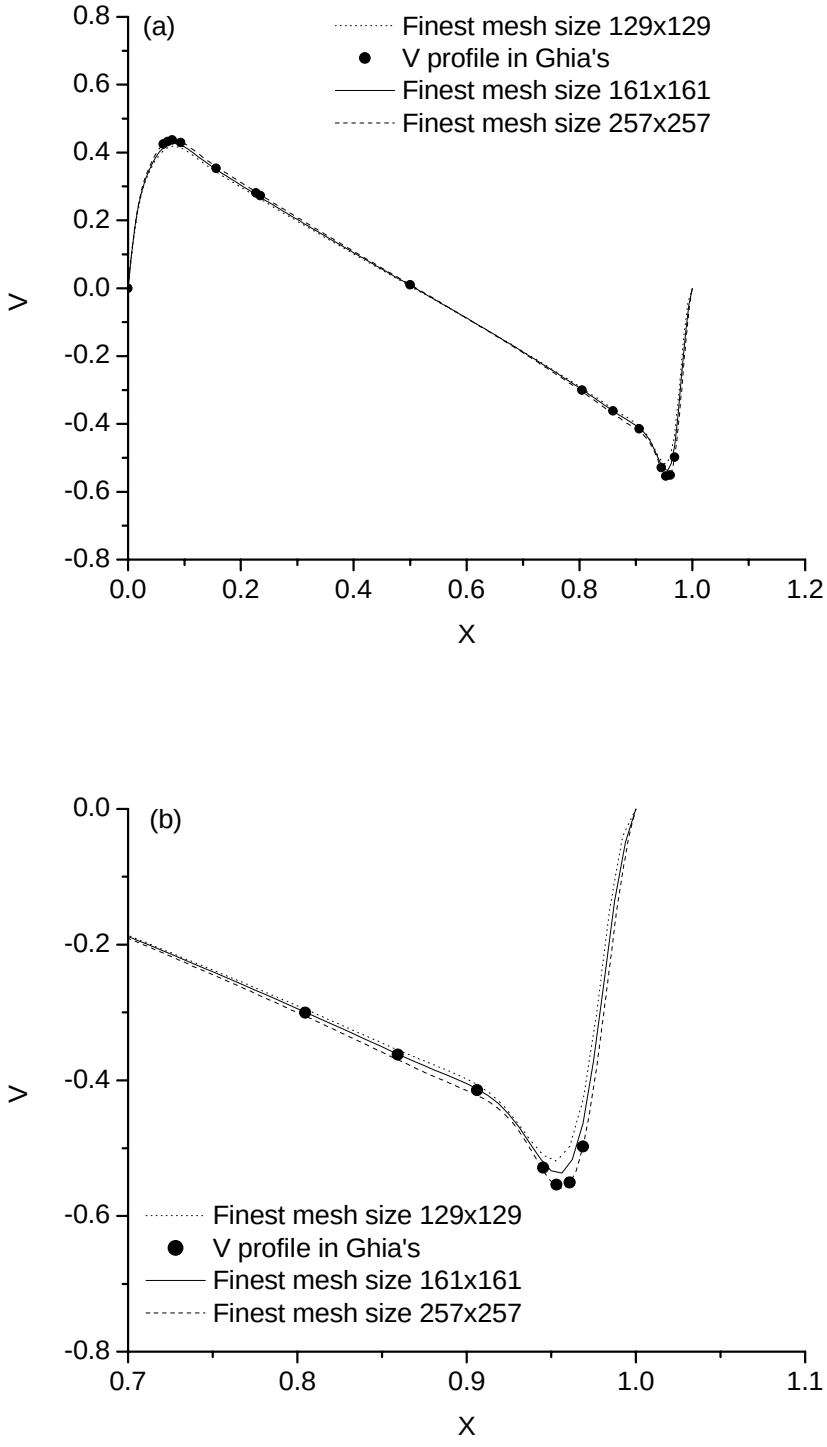


Fig. 3.9 (a) the comparison of V velocity profiles along the horizontal line through geometric center of the box for different mesh sizes with results from Ghia et al.'s (1982) (Re=5000, CF 2) and (b) the magnified view of right-hand minimum point.

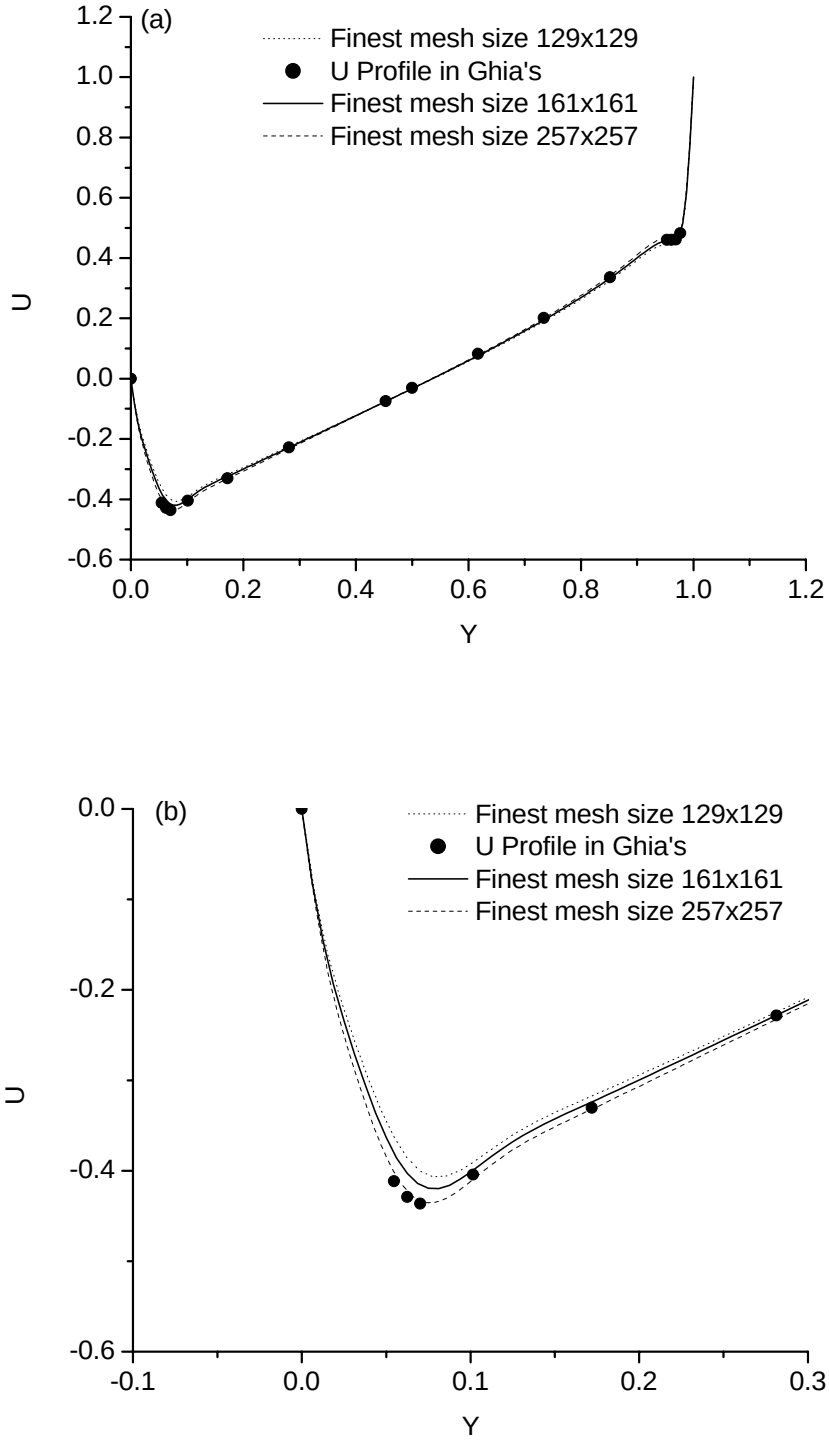


Fig. 3.10 (a) the comparison of U velocity profiles along the vertical line through geometric centre of the box for different mesh sizes with results from Ghia et al.'s (1982) (Re=5000, CF 2) and (b) the magnified view of left-hand minimum point.

In order to appreciate what multigrid could achieve in terms of more rapid convergence to solutions and improvements in computation times, a limited parametric study of the multigrid cycle parameters and components was carried out in this study. It is found that the finest mesh width is the most important parameter, especially for high Re. As Re increases, very coarse grids could not be included in the procedure to avoid divergence. Hence, the choice of the finest mesh size will determine whether the multigrid program will be convergent or not. However, the other parameters also have great effects on the convergence performance.

Table 3.5 lists the effects of different numbers of sweep of the smoother in each grid on the convergence performance for a case with Re=5000 and a finest level grid of 129×129. This table shows that the Newton's method fails in all the 20 cases considered in which a 6 level multigrid was used. The present error reduction procedure with correction function 2 succeeds in all but 2 cases. Efforts have been taken to minimize the outer-loop iterative number and the CPU running time. Cases No. 4 and No. 18 give the two best outer-loop iterative number and CPU running time among the studied cases. Usually, the more sweeps are applied to the finest grid, the lesser the number of outer-loop needed. However, higher sweep number at the finest grid will result in increased CPU running time in one outer loop, leading to higher total computational cost. Conclusively, the sweeps in the finest grid should be minimized whenever possible to save the computational cost while the sweeps in the coarser grids should be relatively increased to maximize error reduction in each outer loop. But, sometimes, too many smoothing sweeps in the coarse grids could make the convergence behavior oscillatory and lead to higher outer-loop iterative number and CPU running time. This phenomenon can be seen in cases No. 17 and No. 18, in

which the No. 18 sweep sequence of (10, 5, 5, 4, 4, and 2) converges faster than case No. 17 which has (10, 5, 5, 6, 5, and 2). For this case with Re=5000 and a finest mesh of 129×129 (MCUI=CUI=0.12), the No. 18 sweep combination appears to have the shortest CPU running time of 42.5 s with 874 outer iterations.

The minimum iterative number occurs for case No. 4 in which 4h coarsening was applied, but it has a significantly higher CPU time of 87.9 s. It is interesting that sometimes the 4h coarsening has better convergence performance than the standard coarsening. Figure 3.11 shows a comparison of convergence histories of cases No. 4 and No. 18.

With regard to the residual error restriction operators, the use of 9-point restriction, or full-weighting, is found to be superior to 5-point restriction, or optimal-weighting as shown in Table 3.6. Both full-weighting and optimal-weighting are examined for the flow with Re=10000. It is found that the full-weighting operator results in a much more stable multigrid scheme. However, as shown in Figure 3.12, once both operators are convergent, there is little difference in their convergence histories.

The computational advantage gained by use of the multigrid procedure is best illustrated in terms of the behaviour of the maximum value of the residual error in the finest grid. Figures 3.13-3.17 show the maximum residual errors obtained during a single grid computation as well as a 6 level multigrid calculation with the other parameters remaining the same. In these figures, the solid lines denote the behaviour of the single grid calculations with 100 SOR iterations and the dash lines are for the multigrid computations with multigrid sweeping sequence shown in the legend. Flow

configurations with Re=5000 and Re=10000 have been examined for different finest mesh sizes and different correction functions. In all cases, the single-grid calculations exhibit a rapid initial decay of the residuals during first 200 iterations. Thereafter, the solid curves show a marked decrease in their slope. The multigrid process seems to retain the high initial decay rate almost during the overall computation. Furthermore, compared with multigrid method, the residual decay histories for the single-grid computations show a lot more oscillations; especially for Re=10000. That may be because 100 SOR iterations are not enough to obtain a good approximate solutions of the intermediate linear systems for the finer mesh of 257×257.

Other than the convergence rate, the efficiency of the computation is also significantly improved by employing the multigrid method in solving the linear equations. Tables 3.7-3.8 summarize the convergence performance of three flow configurations with three correction functions 2 and 3; the Newton's method using CF 1 fails in all cases. It is evident that multigrid procedure becomes more efficient at higher mesh sizes. As shown in Table 3.7, for Re=10000 (257×257 mesh), the single-grid method needs 10533 outer-loop iterations while 3628 is enough for the multigrid method. In terms of the elapsed CPU time, there is also a substantial improvement from 15607s for single grid method to 920s for multigrid method. It is about 2.9 times lesser in iterative numbers and 16.9 times lesser in the running time. However, for Re=1000 with mesh size of 65×65, even though the CPU running time in multigrid process is 2 times faster than the single grid method, the total iterative numbers of the single grid method is almost 200 less than that of 6-level multigrid computation. This is because the running time for one outer loop iteration is greatly decreased by applying the

multigrid procedure, even though the residual error reduction of one outer iteration in the single-grid scheme is larger than for the multigrid scheme.

Table 3.5 Different results with different parameter using multigrid for CF 2 (Finest Mesh size of 129×129, Re=5000, MCUI=CUI=0.12)

W	No.	NI1	NI2	NI3	NI4	NI5	NI6	For CF 2		For CF 1
								Outer-loop iterative number	CPU time(s)	Convergence behavior
0.23	1.	30	10	8	4	4	2	904	106.8	F
	2.	30	10	8	8	8	2	F	—	F
	3.	30	0	15	0	9	2	849	96.1	F
	4.	30	0	10	0	8	2	782	87.9	F
	5.	20	0	10	0	8	2	887	69.9	F
	6.	20	10	8	5	4	2	835	71.4	F
	7.	15	4	3	2	2	2	1033	88.8	F
	8.	15	10	10	6	5	2	809	56.3	F
	9.	10	5	4	2	2	2	1081	84.7	F
	10.	10	10	10	6	5	2	872	46.9	F
	11.	10	8	4	2	2	2	1081	84.7	F
	12.	10	5	4	2	2	2	1081	84.7	F
	13.	10	5	5	2	2	2	1067	78.4	F
	14.	8	5	4	2	2	2	F	—	F
	15.	10	5	4	4	3	2	897	69.8	F
	16.	10	5	4	4	4	2	875	43.6	F
	17.	10	5	5	6	5	2	888	43.5	F
	18.	10	5	5	4	4	2	874	42.5	F
	19.	10	5	4	5	4	2	887	76.8	F
	20.	10	5	4	4	4	4	894	75.9	F

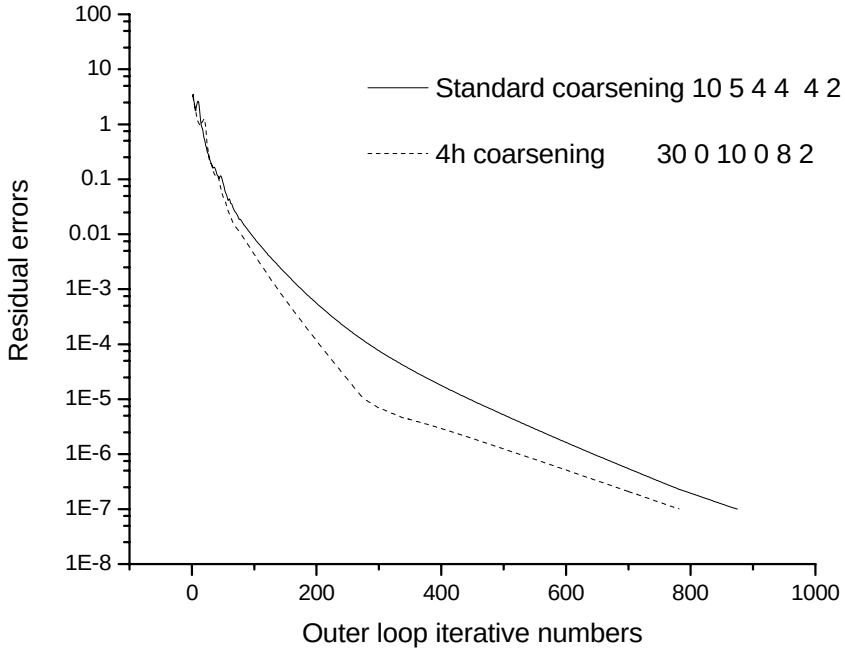


Fig. 3.11 Comparison of convergence histories between standard and 4h coarsening (finest mesh size of 129×129, W=0.23, CUI=MCUI=0.12, Re=5000, CF 2)

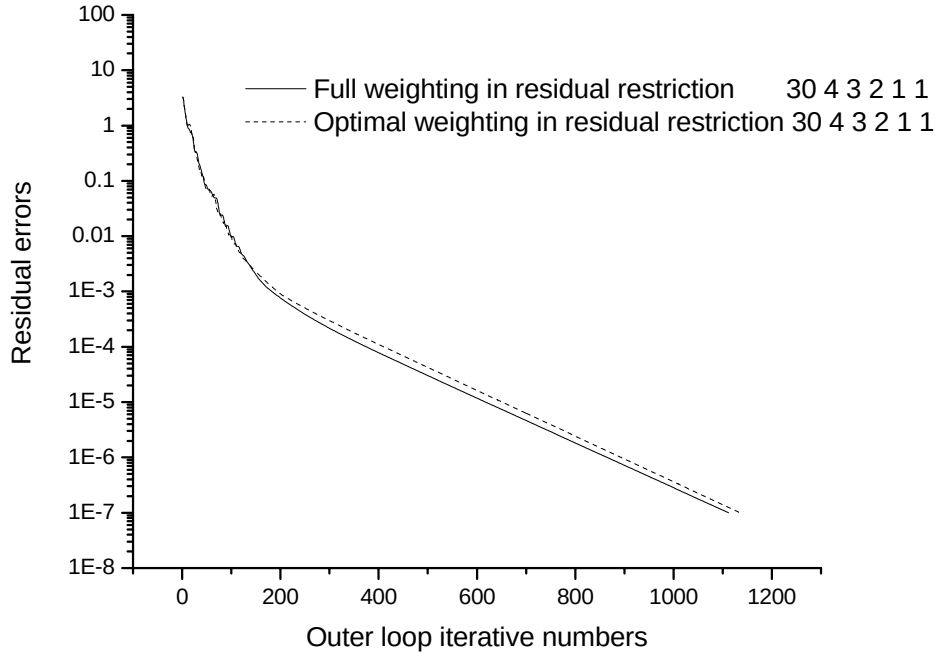


Fig. 3.12 Comparison of convergence histories between full and optimal weighting in residual restriction (finest mesh size of 129×129, W=0.23, CUI=MCUI=0.12, Re=5000, CF 2)

Table 3.6 Second-order Comparison for different residual error restriction operator (CF 2, finest mesh size of 257×257, Re=10000, MCUI=CUI=0.18)

Different smoothing times in each level	10,10,8,5,4,3	10,10,5,5,4,2	15,10,8,5,4,3
Full weighting	3628	3791	3576
Optimal weighting	F	F	3578

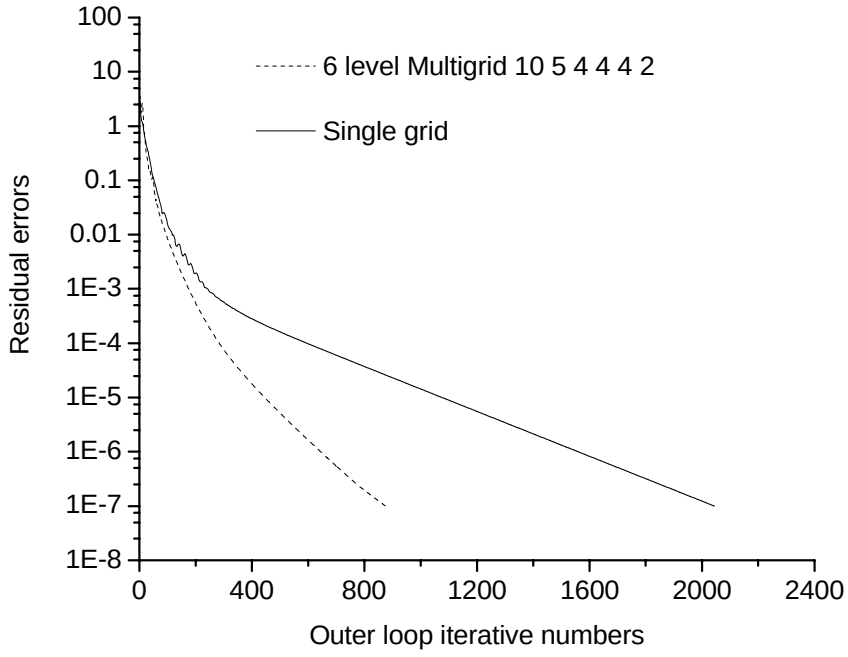


Fig. 3.13 Comparison of convergence histories between multigrid and single-grid computations. (CF 2, W=0.23, CUI=MCUI=0.12, Re=5000, multigrid: finest mesh point of 129×129; single-grid: mesh size of 129×129)

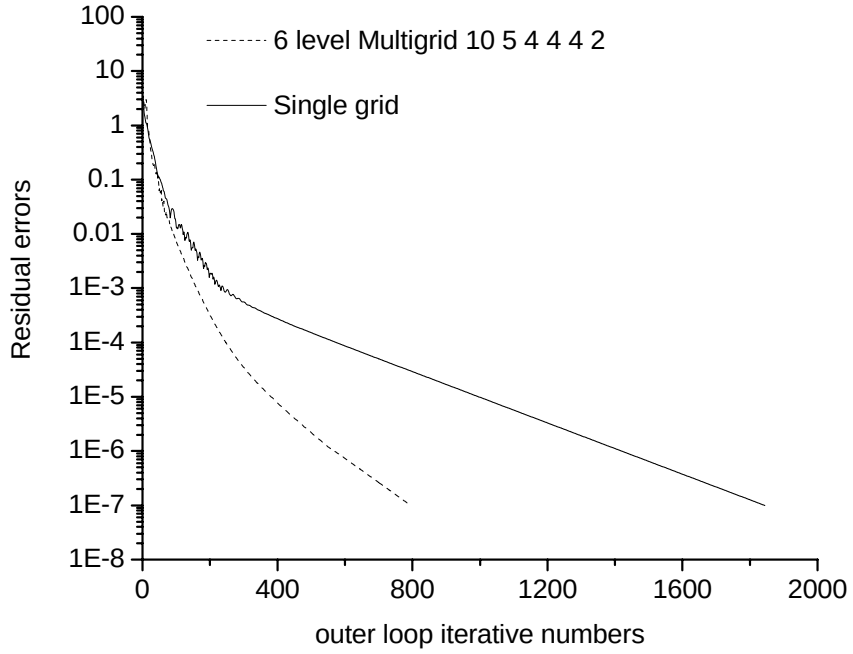


Fig. 3.14 Comparison of convergence histories between multigrid and single-grid computations. (CF 3, α=0.5, β=0.9, W=0.23, CUI=MCUI=0.12, Re=5000, multigrid: finest mesh point of 129×129; single grid: mesh size of 129×129)

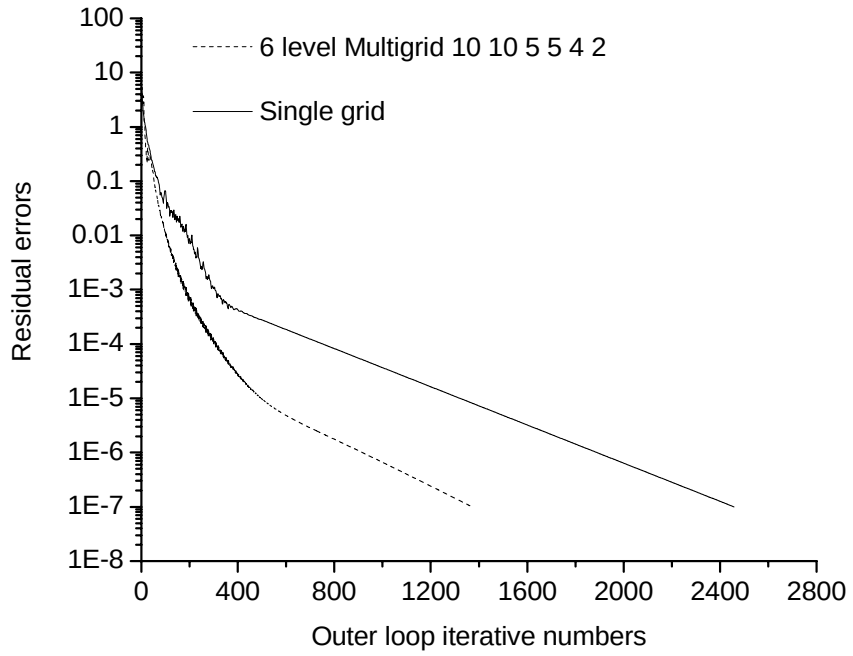


Fig. 3.15 Comparison of convergence histories between multigrid and single-grid computations. (CF 3, α=0.8, β=0.6, W=0.23, CUI=MCUI=0.12, Re=5000, multigrid: finest mesh point of 161×161; single grid: mesh size of 161×16)

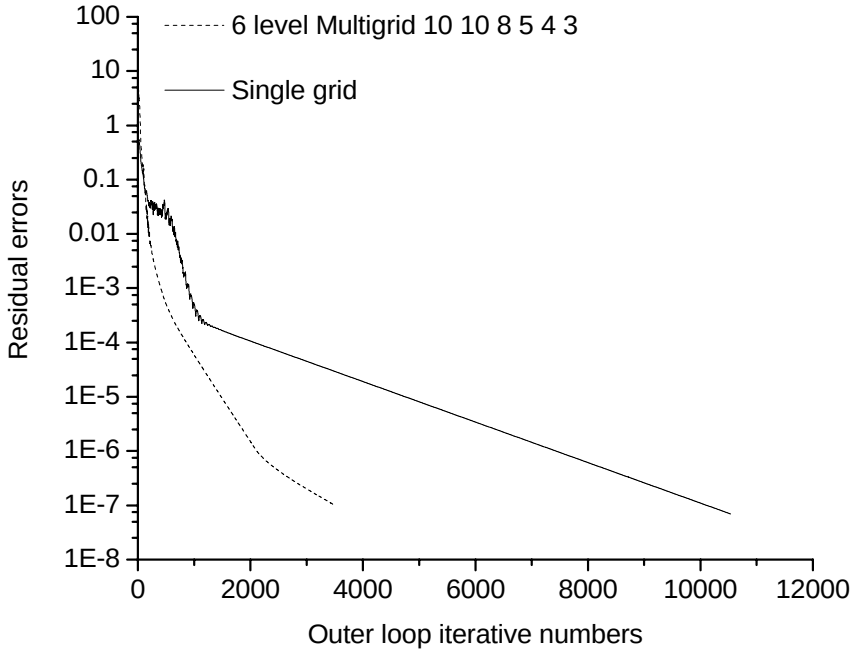


Fig. 3.16 Comparison of convergence histories between multigrid and single-grid computations. (CF 2, W=0.23, CUI=MCUI=0.18, Re=10000, multigrid: finest mesh point of 257×257; single-grid: mesh size of 257×257)

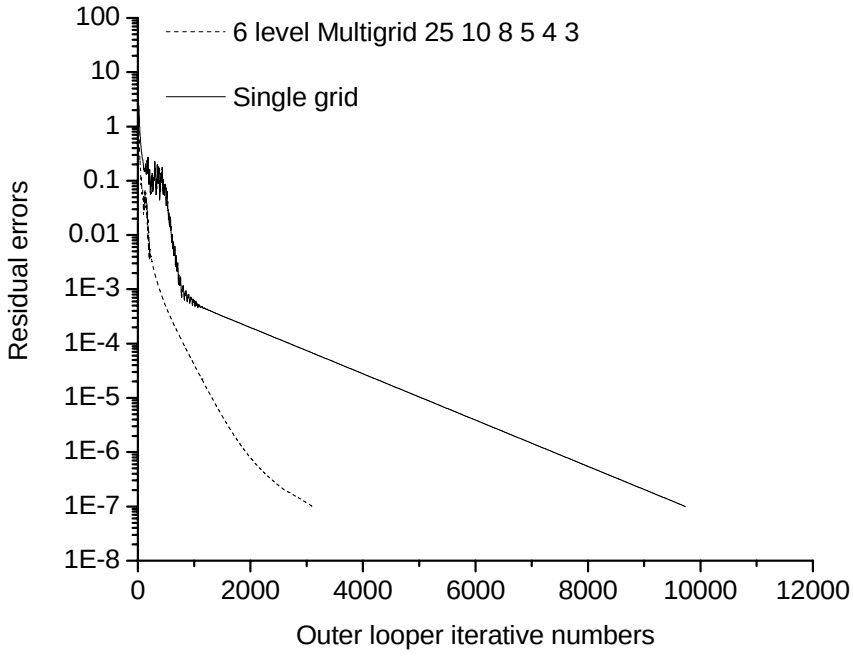


Fig. 3.17 Comparison of convergence histories between multigrid and single-grid computations. (CF 3, α=0.8, β=0.6, W=0.23, CUI=MCUI=0.18, Re=10000, multigrid: finest mesh point of 257×257; single grid: mesh size of 257×257)

Table 3.7 Comparison of iteration number and CPU running time for single and multigrid computations with CF 2 and same parameters.

Mesh size	Reynolds number	Single-grid method		6 level Multigrid method	
		Outer-loop iterative numbers	CPU time (s)	Outer-loop iterative numbers	CPU time (s)
65×65	1000	331	11.9	515	5.25
129×129	1000	1295	225.8	1268	77.6
129×129	5000	2044	391.8	874	42.5
161×161	5000	2696	1090.0	1502	136.5
257×257	10000	10533	15607	3628	920

Table 3.8 Comparison of iteration number and CPU running time for single and multigrid computations with CF 3 and same parameters.

Mesh size	Reynolds number	Single-grid method		6 level Multigrid method	
		Outer-loop iterative numbers	CPU time (s)	Outer-loop iterative numbers	CPU time (s)
65×65	1000	284	10	338	5.6
129×129	1000	1114	192	1140	59
129×129	5000	1844	314.4	795	39.5
161×161	5000	2458	657	1372	119
257×257	10000	8868	12055	3198	710

3.4 Fourth-order refinement

Up to now, all the results are obtained by the second-order algorithms and second-order boundary conditions. In this section, the fourth-order algorithms with fourth-order boundary conditions are examined to understand the advantages of the fourth-order scheme. Firstly, the accuracy of fourth-order scheme must be confirmed. Table 3.9 lists the maximum errors for five different meshes ranging from 61×61 to 161×161. The maximum error is defined with reference to the solution obtained with a grid of 481×481. Fig. 3.18 shows the relationship between the maximum errors and the mesh number. The least square fitted line has a slope of -3.41, somewhat less than the -4.00 that we had expected. We believe this is caused by the fact that the reference solution is itself an approximate solution. Secondly, the numerical convergence may not be rigorous enough given that we are now dealing with fourth-order scheme.

Fig. 3.19 compares the V-velocity profile of the solution obtained on a mesh of 61×61 with the fourth-order scheme with corresponding profile of the solution obtained on a finer mesh of 129×129 using the original second-order scheme. The Reynolds number is at 1000. Similarly, the U velocity profiles along the vertical centre line through the cavity are illustrated in Figure 3.20. It can be seen that the U and V profiles obtained from fourth-order accuracy computation with mesh size of 65×65 almost overlaps with that obtained from second-order accuracy with mesh size of 129×129. It means that with the same tolerance value the fourth-order algorithms can achieve the same or more accurate solution with half of the mesh size of the second-order algorithm. Comparing the error for fourth-order scheme with mesh 81×81 (Table 3.9) with the corresponding error for the second-order scheme with mesh 161×161 (Table 3.4), it

would appear that the fourth-order scheme is more accurate with only half the mesh resolution. Consequently, the fourth-order difference scheme can reduce the size of equation systems by possibly more than three quarters whilst preserving or even enhancing the accuracy of the solutions.

Table 3.9 Maximum U velocities differences for various mesh sizes with fourth-order scheme. (Re=1000, CUI=MCUI=0.10, W=0.23, CF 2)

	Mesh sizes	$\left\|U-U^{ref}\right\|_{\Omega}$
Reference	418	—
1	161	4.59E-04
2	121	8.46E-04
3	97	0.00168
4	81	0.00372
5	61	0.01216

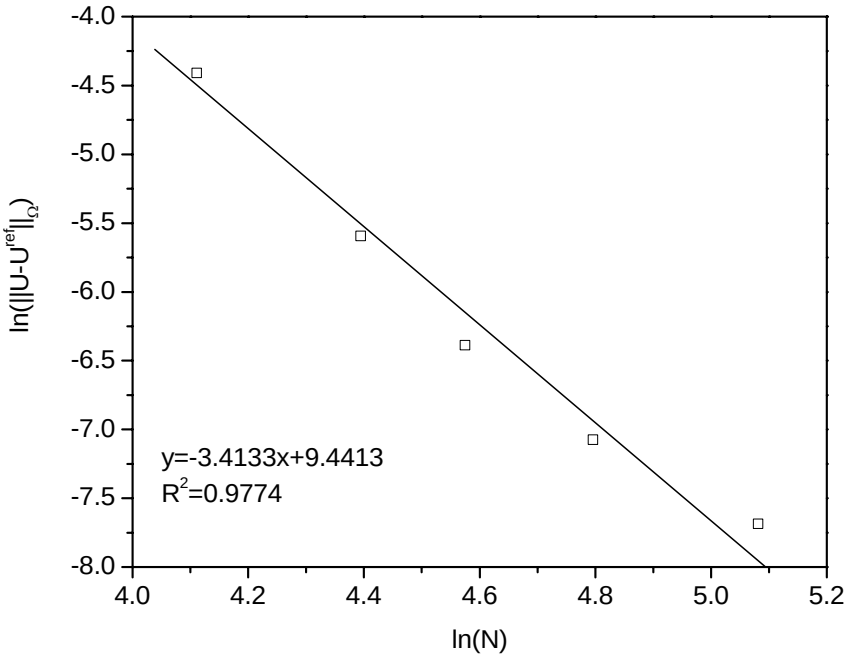


Fig. 3.18 Maximum error as a function of mesh sizes

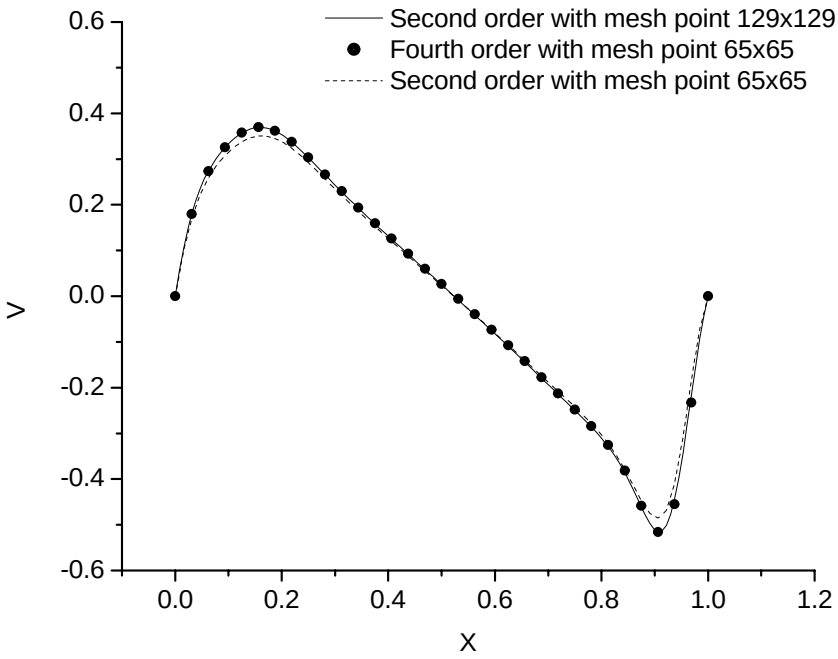


Fig. 3.19 The comparison of V velocity profiles along the horizontal line passing through the geometric centre of the cavity for different meshes and schemes. (Re=1000)

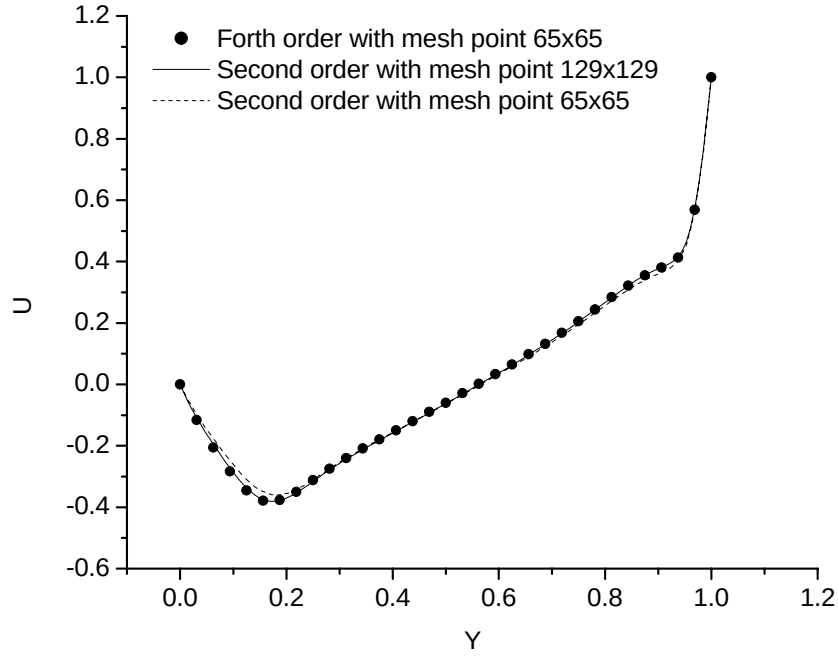


Fig. 3.20 The comparison of U velocity profiles along the horizontal line passing through the geometric centre of the cavity for different meshes and schemes. (Re=1000)

Besides gain in accuracy, the fourth-order scheme also appears to have better convergence performance if the mesh is fine enough. As shown in Table 3.10, where Re=1000 and correction function 2 are employed, the fourth-order scheme converges faster than the second-order scheme at both mesh sizes of 65×65 and 129×129. The convergence histories for the latter cases are illustrated in Figure 3.21. It is interesting that for the first 300 iterations, there almost is no difference in the convergence histories of the two schemes. However, the fourth-order scheme shows the advantage over the second-order scheme after 300 iterations, requiring only 933 iterations instead of 1295 iterations to reach the residual error below 10^{-7}. According to the computational time, Table 3.10 also contains the information that the fourth-order scheme has the desirable performance at both mesh sizes.

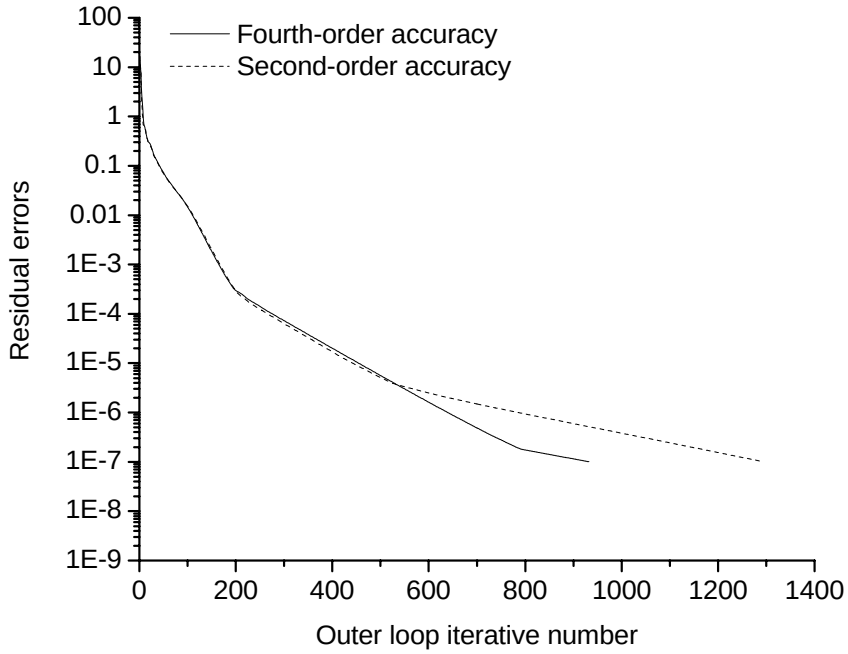


Fig. 3.21 Comparison of convergence histories between second- and fourth- order schemes. (finest mesh size of 129×129, W=0.23, CUI=MCUI=0.1, Re=1000, CF 2)

Table 3.10 The convergence comparison for second- and fourth-order schemes. (Re=1000)

Mesh size	Second-order		Fourth-order	
	Outer-loop iteration numbers	CPU time (s)	Outer-loop iteration numbers	CPU time (s)
65×65	331	12	285	11
129×129	1295	348	933	229

There is seemingly one important limitation to the free application of the fourth-order scheme. The mesh must be adequately fine for the fourth-order scheme to converge. This is especially so as one goes to higher Reynolds number. For instance, with Re=5000, the grid size of 129×129 is large enough for second-order scheme to solve the equations. However, for the fourth-order scheme, a mesh finer than 181×181 is needed. For convenience, in the following multigrid computations, the mesh 193×193 is applied to compute the flow with Re=5000.

The multigrid method is quite efficient for the second-order difference scheme. It also works well with the fourth-order scheme if suitable parameters are used. As shown in Figure 3.22, for Re=5000 with 4h coarsening, not only is the convergence behaviour for multigrid scheme monotonic but it is also desirably much faster than that of the single-grid scheme. As a result, the CPU running time needed for multigrid scheme is less than one third that of the single-grid scheme, 497 seconds versus 1553.6 seconds. The comparison of the convergence histories for three correction functions is given in figure 3.23. The correction function 3 again shows slightly better performance than the correction function 2. Due to the convergence failure of the Newton's method (correction function 1 and $s \equiv 1$), only two history curves are given in the figure.

Figures 3.24 and 3.25 show the pressure contours for the second-order and fourth-order discretization schemes at Re=1000 respectively. A magnified view of the geometric center in the dashed square is also included. The fourth-order scheme has smooth pressure contours whereas the second-order scheme shows some checkerboard pressure oscillation that is not uncommon with second-order scheme on collocated primitive-variables grid. This is because the fourth-order discretization scheme employs a 9-point template.

The Figure 3.26 shows the streamline for the cavity flow at the Re=10000 obtained on a uniform grid 385×385. The relative convergent criterion is set to be $\|h\|/\|h^1\| < 10^{-12}$ combined with fourth-order discretization. A magnified view of the various secondary vortices is also included. In terms of the notation shown in this figure, the letters T, B, L, and R denote top, bottom, left, and right, respectively. The subscript numeral denotes the hierarchy of these secondary vortices. For example, BR_2 refers to the second vortex in the sequence of secondary vortices that occur in the bottom right corner of the cavity. Five vortices are shown clearly in the streamline plot. In the magnified left bottom corner view, the second vortex just begins to appear. The location of the primary vortices, TL, BL_1, BL_2 BR_1, and BR2 BR_3 are listed in table 3.11. The physical extensions of the various secondary vortices and the inception of the third vortices at the right bottom are in excellent agreement with that reported by Ghia *et al*. (1982); Schreiber and Keller (1983) and Hwang and Cai (2003). Table 3.12 and table 3.13 tabulate the numerical values corresponding to the velocity profiles shown in figures 3.27 and 3.28 for lines passing through the geometric centre of the cavity. Only typical points including the maxima and minima, rather than the entire set of computational points, along these profiles are listed.

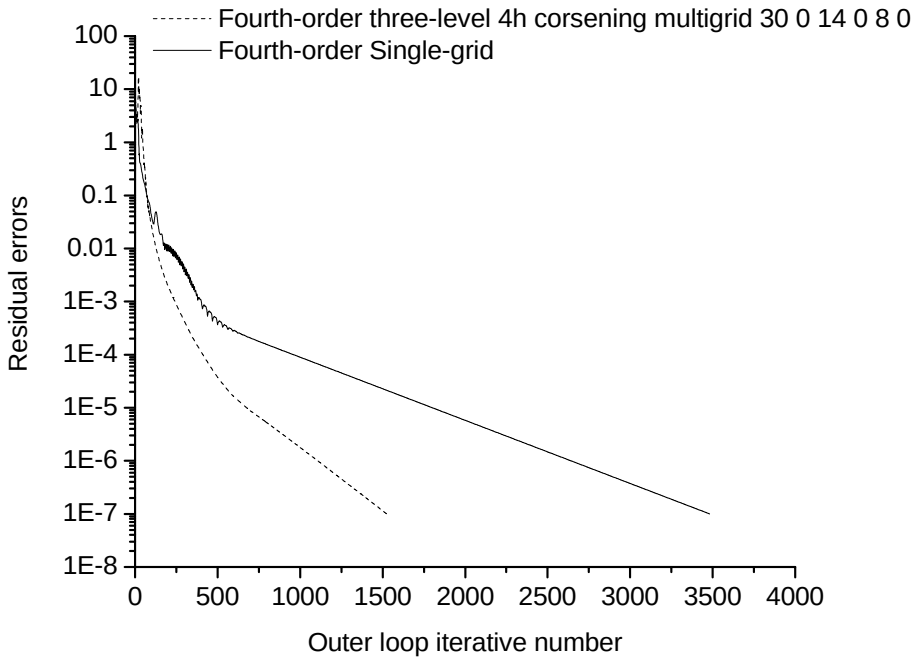


Fig. 3.22 Comparison of fourth order convergence histories between multigrid and single-grid methods. (multigrid: finest mesh point of 193×193; single grid: mesh size of 193×193, CF 2, W=0.23, CUI=MCUI=0.12, Re=5000)

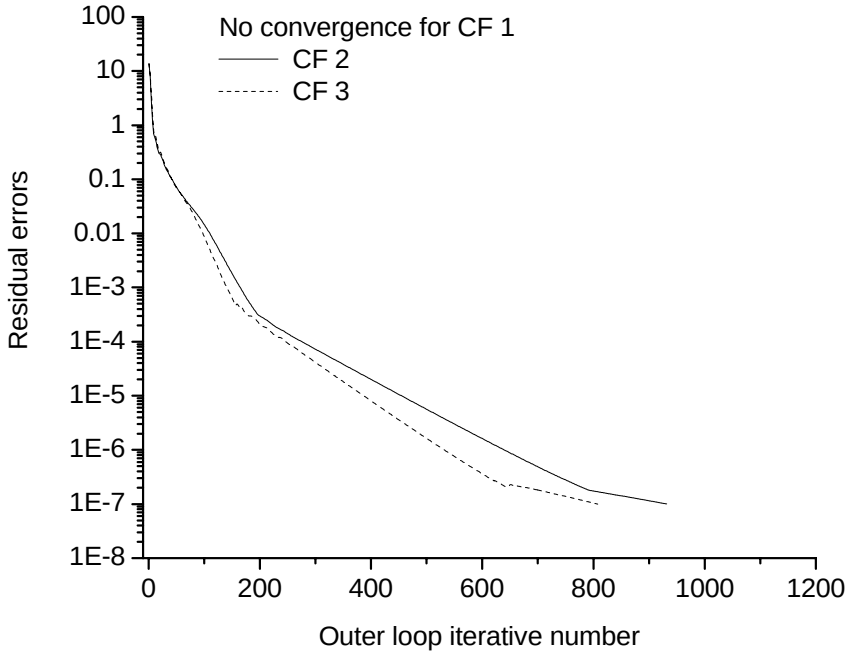
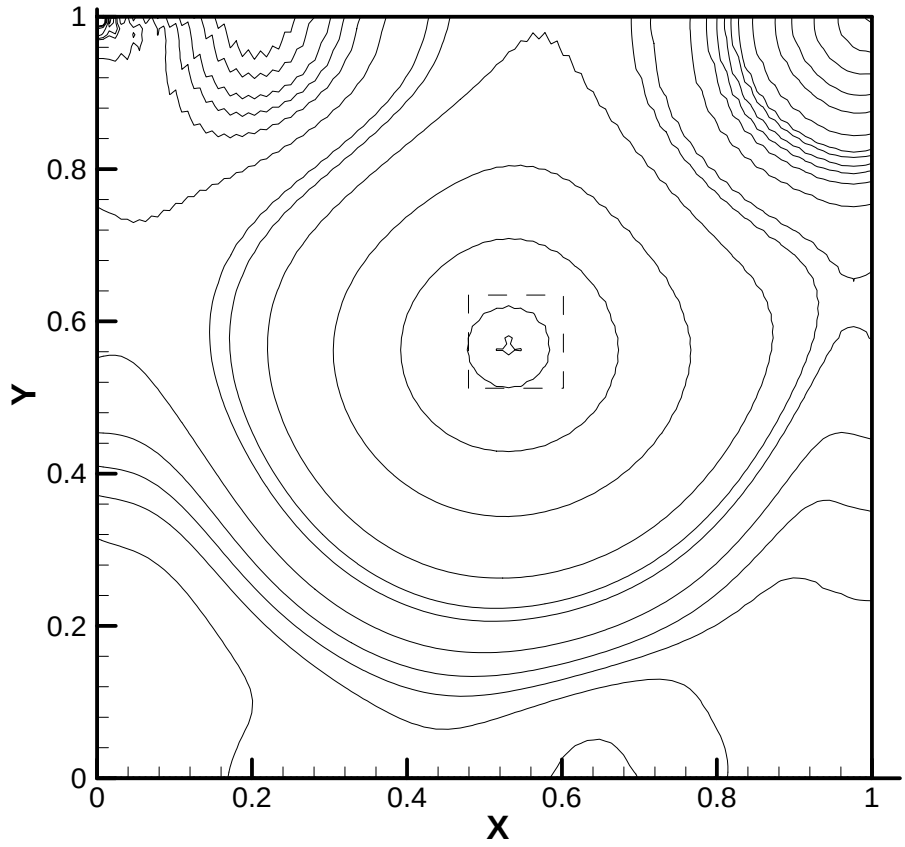
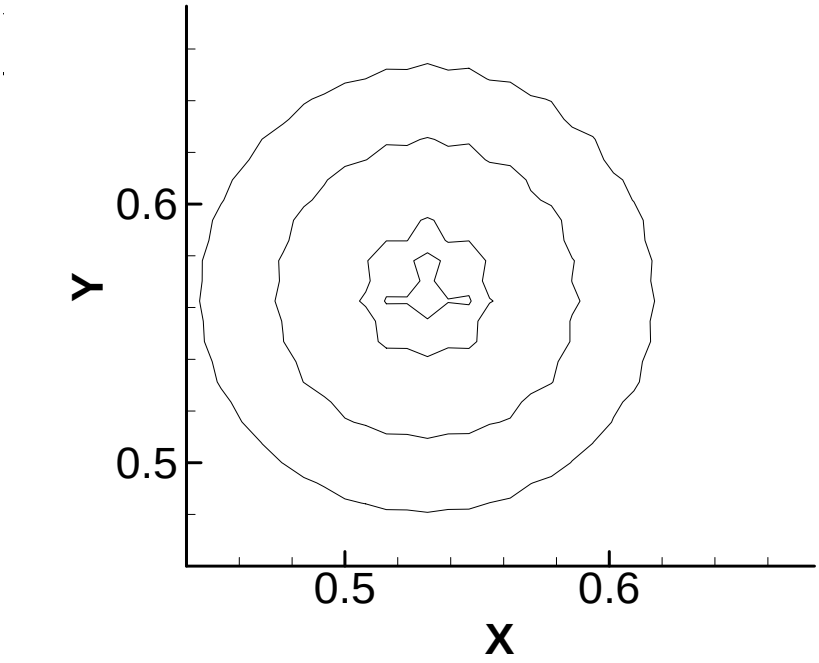


Fig. 3.23 Comparison of fourth-order convergence histories between CF 2 and CF 3 using single-grid methods. (mesh size of 129×129, α=0.9, β=0.6, W=0.23 CUI=MCUI=0.10, Re=1000)

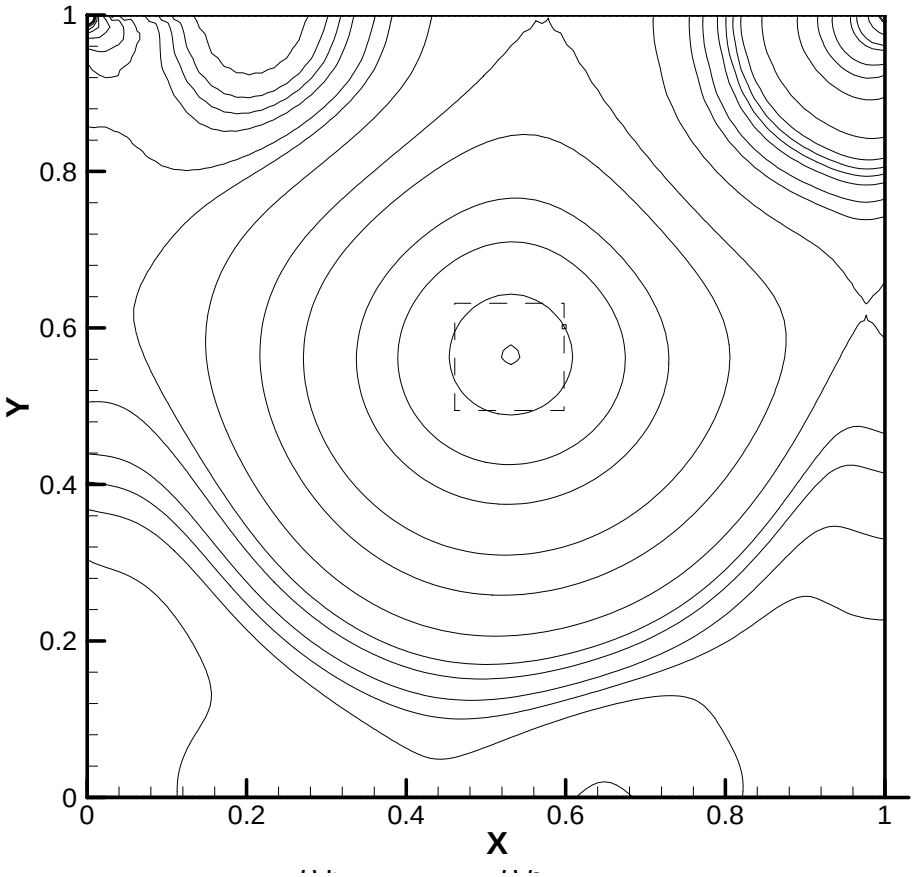


(a)

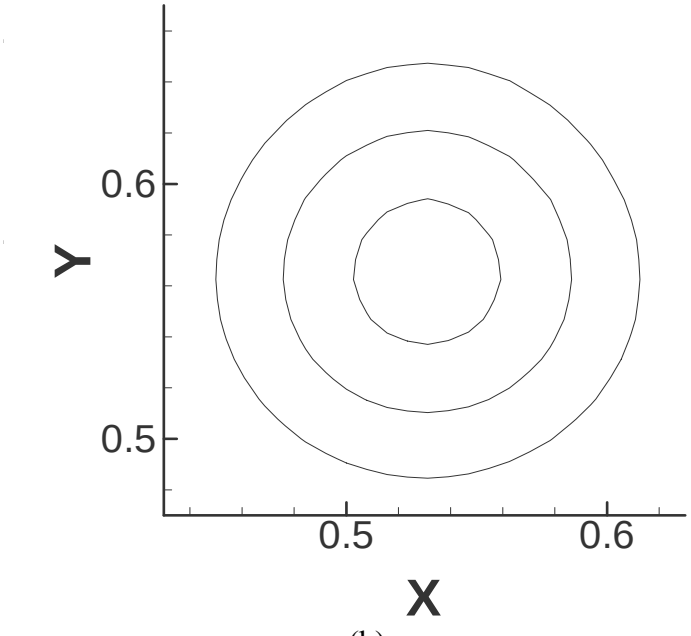


(b)

Fig. 3.24 (a) Pressure contour using second-order scheme, (b) magnified view of the center (Re=1000, mesh size of 129×129)



(a)



(b)

Fig. 3.25 (a) Pressure contour using fourth-order scheme, (b) magnified view of the center (Re=1000, mesh size of 129×129)

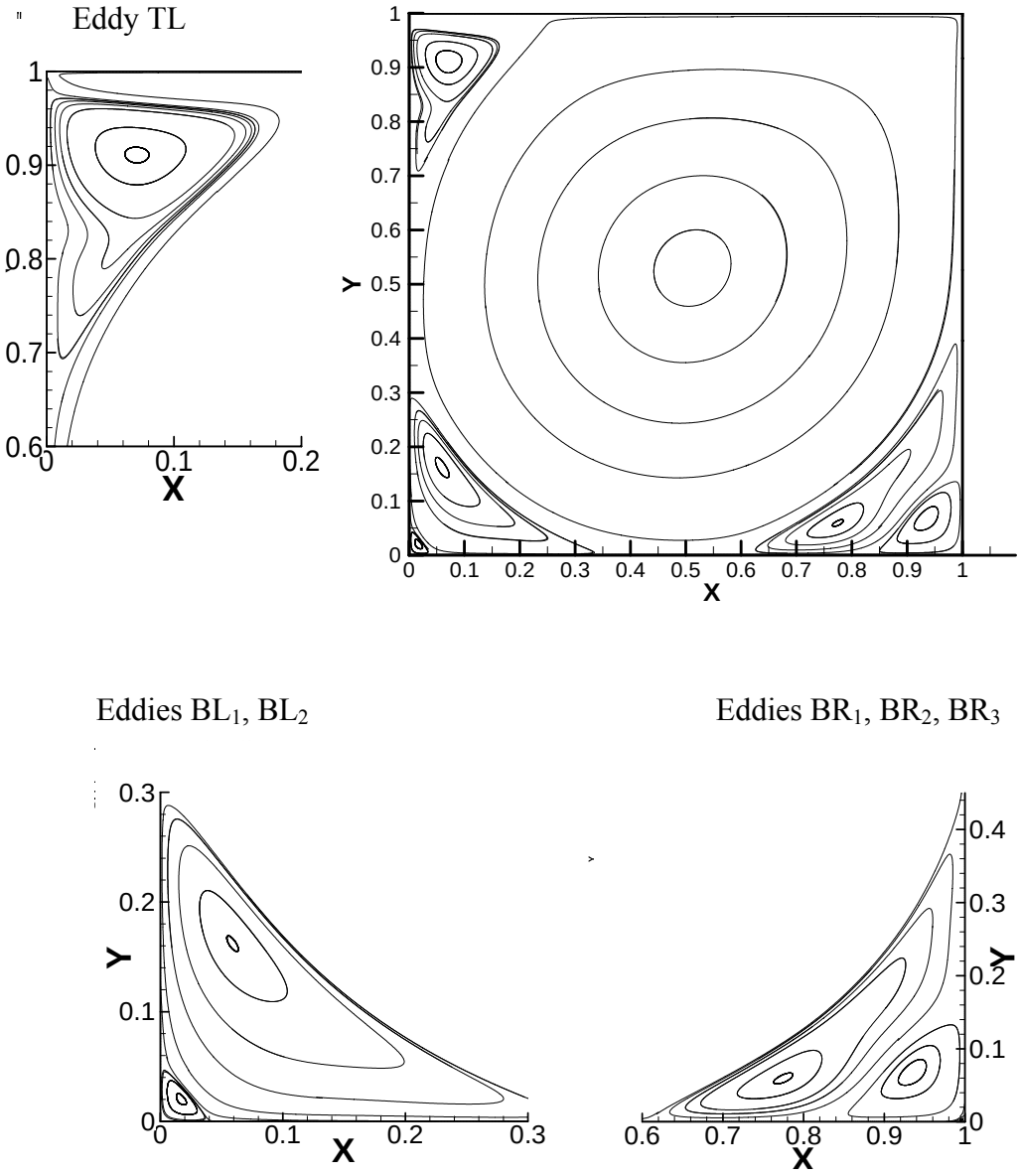


Fig. 3.26 Streamline pattern for primary, secondary, and additional corner vortices (Re=10000, Finest grid 385×385, correction function, CF 2, CUI=MCUI=0.18)

Table 3.11 Position of the vortex centres

Number	Primary	TL	BL_1	BL_2	BR_1	BR_2	BR_3
Location *x*, *y*	0.51194, 0.53001	0.0707, 0.9108	0.0589, 0.1621	0.0171, 0.0200	0.7750, 0.0593	0.9351, 0.0677	0.9951, 0.0031

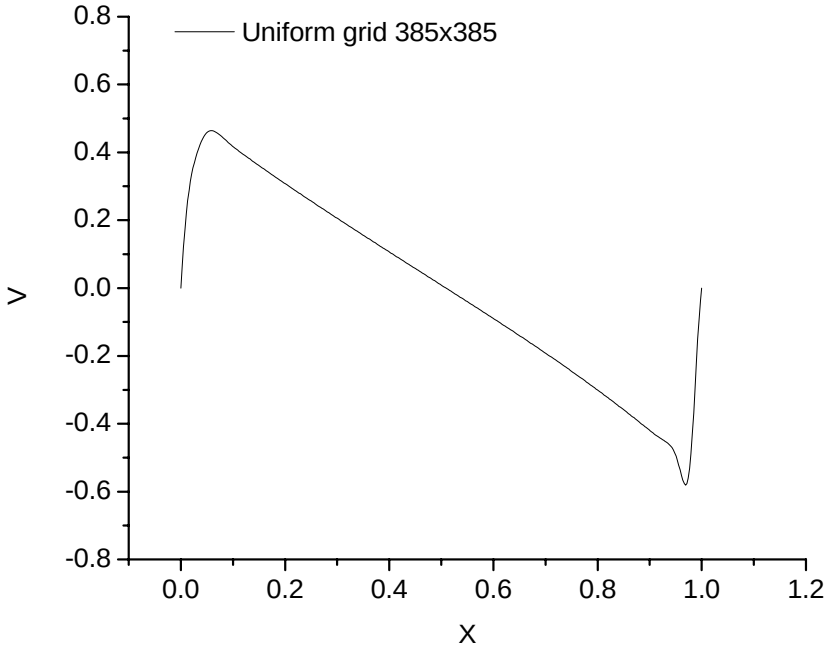


Fig 3.27 V velocity profile along the horizontal line passing through the geometric centre of the cavity for Re=10000, correction function CF 2.

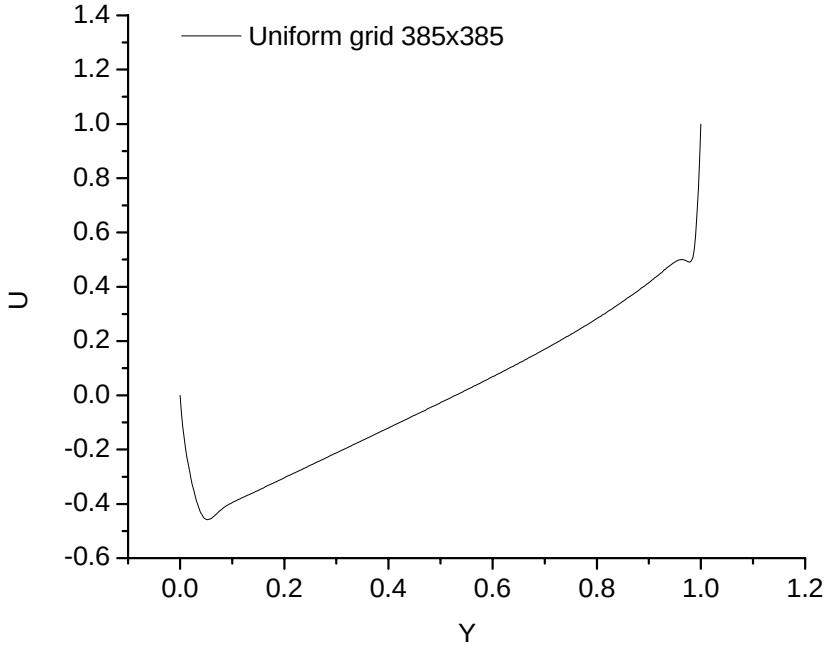


Fig 3.28 U velocity profile along the vertical line passing through the geometric centre of the cavity for Re=10000, correction function CF 2.

Table 3.12 Results for U velocity along the vertical line through geometric centre of the cavity for Re=10000, finest mesh size 385×385.

385-grid pt. No.	y	U
385	1.000000	1.000000
377	0.979167	0.490892
376	0.9765625	0.491147
375	0.973958	0.493350
370	0.960938	0.499678
369	0.958333	0.498693
368	0.955729	0.497063
338	0.877604	0.383288
308	0.783854	0.263313
278	0.705729	0.175934
248	0.643229	0.111262
218	0.549479	0.019638
193	0.500000	-0.027131
166	0.429688	-0.092599
139	0.361979	-0.155090
112	0.289062	-0.222167
86	0.213542	-0.291523
70	0.179688	-0.322617
54	0.138021	-0.361060
34	0.085938	-0.410948
23	0.057291	-0.456007
22	0.054688	-0.457699
21	0.052083	-0.457993
20	0.049479	-0.456616
1	0.000000	0.000000

Table 3.13 Results for V velocity along the horizontal line through geometric centre of the cavity for Re=10000, finest mesh size 385×385.

385-grid pt. No.	x	V
385	1.000000	0.000000
373	0.968750	-0.580678
372	0.966146	-0.576393
371	0.963542	-0.566011
370	0.960938	-0.552066
365	0.947917	-0.485293
340	0.882812	-0.398381
320	0.830729	-0.336054
295	0.765625	-0.262326
255	0.661458	-0.151877
225	0.583333	-0.073174
205	0.531250	-0.021824
193	0.500000	0.008768
175	0.453125	0.054556
155	0.401042	0.105544
130	0.335938	0.169828
105	0.270833	0.235125
80	0.205729	0.301949
60	0.153646	0.357171
40	0.101562	0.415398
28	0.070313	0.456680
26	0.065104	0.461417
25	0.062500	0.462945
24	0.059896	0.463754
1	0.000000	0.000000

3.5 Application to other problems

The error reduction methodology is also applied to flows in two rectangular driven cavities. The driving surface remains the top boundary, moving from left to right. The shorter walls in those two cases are used as the reference length to define the Reynolds number. The flows at Re=5000 are solved in a uniform single grid. Figure 3.29 and figure 3.31 show the streamline plot for these two cases. The convergence histories are given in figure 3.30 and figure 3.32, respectively. Correction functions 2 and 3 are tested on these two types of flow. It is obviously in figure 3.30 that correction function 3 is slightly fewer in term of the total of iteration number. Correction function 3 also displays higher rate of convergence, which is critically important if one is interested in a highly converged solution. However, correction function 2 has less oscillation at the beginning of the convergence history. Nevertheless, correction function 3 remains highly stable (robust) in terms of success in achieving convergence. It would appear that the correction function 3 is able to achieve successful convergence even in the presence of more oscillations.

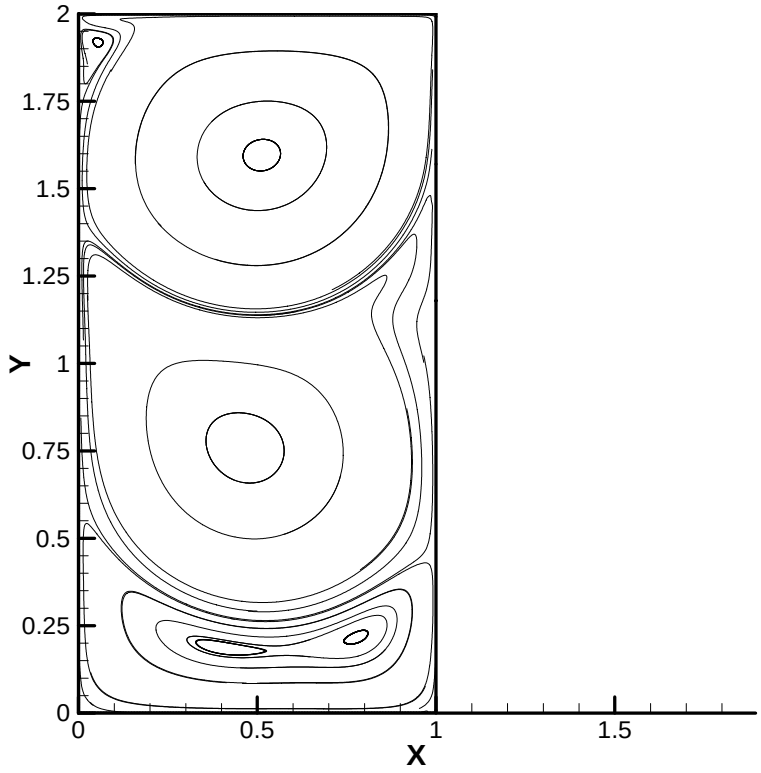


Fig. 3.29 The streamline pattern of flows in a rectangular driven cavity at Re=5000 with uniform grid 129×257.

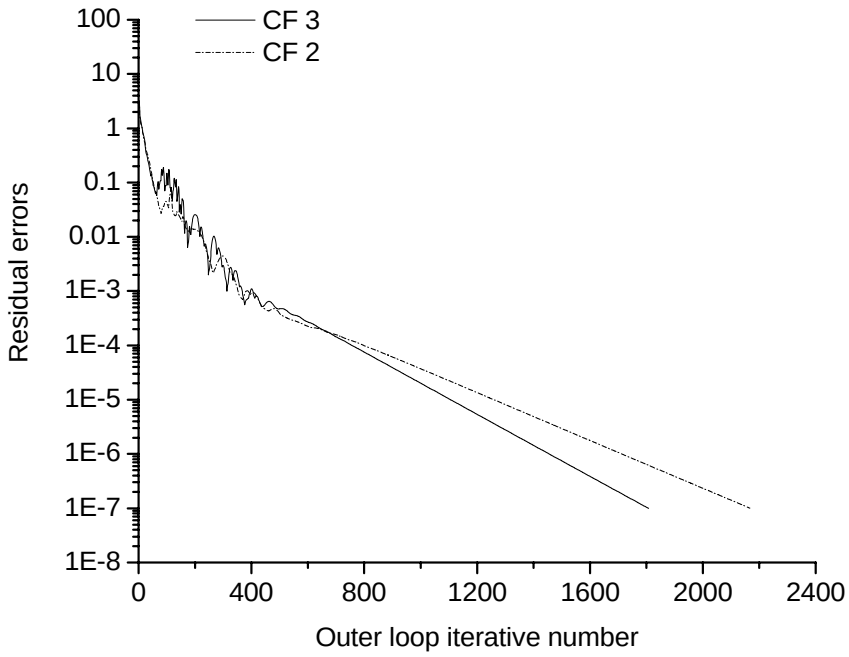


Fig. 3.30 The convergence histories for flow in a rectangular driven cavity at Re=5000, uniform grid 129×257, using two different correction functions.

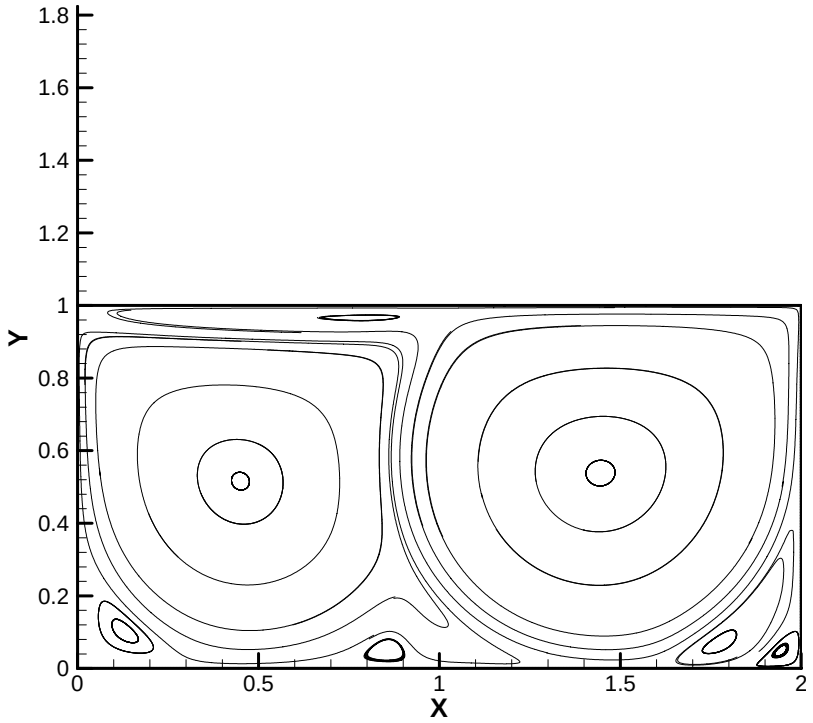


Fig. 3.31 The streamline pattern of flow on a rectangular driven cavity at Re=5000 with uniform grid 257×129.

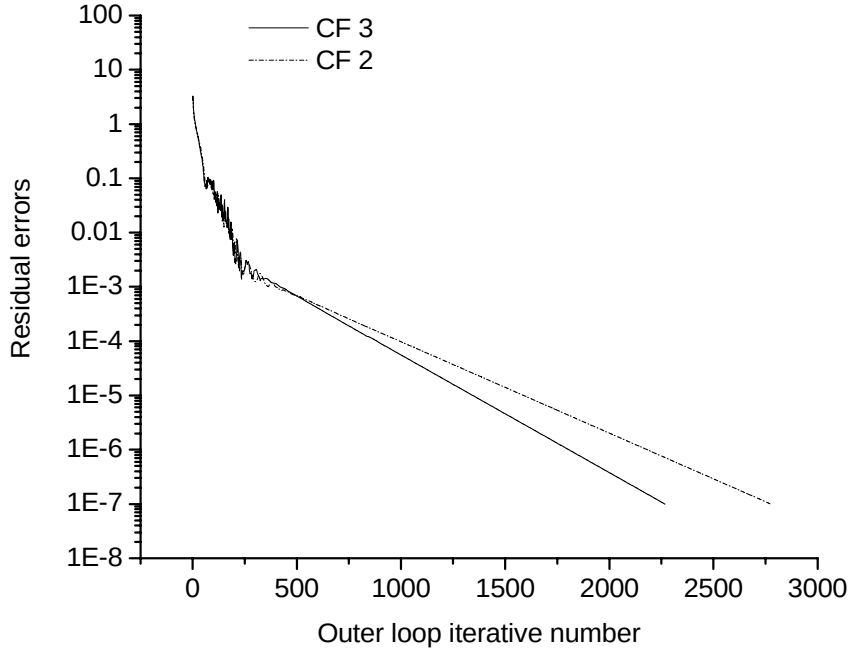


Fig. 3.32 The convergence histories for flow in a rectangular driven cavity at Re=5000, uniform grid 129×257, using two different correction functions.

Chapter 4 Conclusions and recommendations

4.1 Conclusion

Today, numerical simulation is an essential and indispensable tool for fluid dynamics research. The effort to improve the efficiency and accuracy of numerical computation has been under way since the beginning of CFD. In the present project, a numerical method based on the principle of monotonic residual error reduction developed by Liu (2002) has been improved incorporate to multigrid iteration. The second-order difference scheme has also been refined to fourth-order scheme. Comparisons have been made between this new method and the customary Newton's method.

The prototypical two-dimensional driven cavity flow problem is set as the basic test problem. Two kinds of correction functions have been designed to compare with the performance of the Newton's method. To analyze the convergent performance and the monotonic property of the numerical method, the residual errors are plotted against the number of outer-loop step iterations. It is concluded that if Newton's method converges, it is slightly better in terms of numerical efficiency. However, Newton's method has been known and has also been proved here to be more sensitive to the parameters and the initial conditions than current residual reduction scheme and correction functions. The rate of successful convergence of the Newton's method is much lower compared to the present scheme. This is especially for multigrid implementation. We also can conclude that the correction function 3 shows slightly better performance than correction function 2.

This work shows that the proposed method can lead to nearly monotonic decrease in the residual errors no matter whether single-grid or multigrid method has been used in small Reynolds number problems. As Re No. increases, the residual behaviour for single grids becomes more and more oscillatory. However, for the multigrid case, this residual reduction quality is kept to a larger extent with increase in Re. In respect of the efficiency, employing the multigrid process tends to retain the initial decay rate almost during the whole computation, while the single-grid calculations exhibit rapid decay of the residuals only during the first few tens of iterations. As a result, the multigrid method can save a lot of computing cost compared with single-grid scheme. The various components and parameters in the multigrid procedure are examined. The use of full weighting is found to be slight superior to optimal weighting. The finest mesh size employed in the grid sequence continues to be a very significant parameter. The smoothing factor of the iteration scheme is seen to be influenced by the physical problem parameters, namely, Re.

The fourth-order refinement scheme offers important gains over the standard second-order scheme. Examples show that the fourth-order scheme preserves or even enhances the accuracy of the solutions computed using far fewer mesh points – typically one quarter or lesser of the number required of corresponding second-order scheme. For a given mesh, the fourth-order scheme also appears to have better convergence performance and to require less total CPU running time to achieve the same level of residual error reduction. The pressure solution is also free of checkerboard fluctuations that may degrade the accuracy of the second-order primitive-variables schemes on collocated grid. However, for high Reynolds number flows, the fourth-order scheme has a limitation to its free application in that it may require the mesh size to be

sufficiently fine to achieve convergence. Since the fourth-order refinement scheme does not incur a large CPU-time penalty for the accuracy gain, it is a useful variation of the present method for problems that require high accuracy solutions.

4.2 Recommendations

Owing to time limitation, the smoother used in the multigrid procedure is the SOR method. A more efficient smoothing method such as incomplete LU matrix decomposition is worth trying. Meanwhile, more sophisticated multigrid strategies also can be areas for future research.

Up to now, this proposed method has been applied to two-dimensional incompressible steady flow. The extension to three-dimensional incompressible steady flow could be further investigated. And this method could also be modified to solve the unsteady flow problems.

Other forms of correction functions could also be designed to further improve rate of convergence to solution as well as its success rate in acquiring solution, which is already much higher than the traditional Newton's method.

References

1. Anderson, J.D., Degrez, G., Dick, E. and Grundmann, R. (1996), Computational Fluid Dynamics, edited by Wendt, J.F., Pringer, New York.
2. Chacón, L., Barmes, D.C., Knoll, D.A. and Miley, G.H. (2000), An implicit energy-conservative 2D Fokker-Planck algorithm, J. Comput. Phys., 157, pp. 654-682.
3. Chang, J.L.C., and Kwak, D., (1984), On the method of pseudo compressibility for numerically solving incompressible flows, AIAA, Paper 84-0252.
4. Chang, J.L.C., Kwak, D., Rogers, S.E. and Yang, R.J. (1988), Numerical simulation methods of incompressible flows and an application to the space shuttle maine engine, Int. J. Numer. Method Fluids, 8, pp. 1241-1268.
5. Chorin, A.J. (1967) A numerical method for solving incompressible viscous flow problems, J. Comput. Phys., 2, pp.12-26.
6. Coelho, P.J. and Pereira, J.C.F. (1992), Finite volume computation of the turbulent flow over a hill employing 2D or 3D non-orthogonal collocated grid systems, Int. J. Numer. Method Fluids, 14, pp. 423-441.
7. Dembo, R.S., Eisenstat, S.C. and Steihaug T. (1982), Inexact Newton methods, SIAM J. Numer. Anal., 19, pp. 400-408.
8. Doormaal, V.J.P. and Raithby, G.D. (1984), Enhancements of the SIMPLE method for predicting incompressible fluid flows, Numerical Heat Transfer, 7, pp. 147-163.
9. Edwards, W.S., Tuckerman, L.S., Friesner, R.A., and Sorensen, D.C. (1994), Krylov methods for the incompressible Navier-Stokes equations, J.Comput. Phys., 110, pp. 82-102.
10. Eisenstat, S.C. and Walker, H.F. (1994), Globally convergent inexact Newton Methods, SIAM J. Optimization, 4(2), pp. 393-422.
11. Eisenstat, S.C. and Walker, H.F. (1996), Choosing the forcing terms in an inexact Newton method, SIAM J. Sci., Comput., 17(1), pp. 16-32.
12. Freund, R.W., Folub, G.H., and Nachtigal, N.M. (1992), Iterative solution of linear systems, Acta Numerica, 1, pp. 87-100.

13. Ghia, U., Ghia, K.N., and Shin, C.T. (1982), High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method, J. Comput. Phys., 48, pp. 387-411.
14. Gresho, P.M. and Sani, R.L. (1987), On pressure boundary conditions for the incompressible Navier-Stokes equations, Int. J. Numer. Methods Fluids, 7, pp. 1111-1145.
15. Hadjidimos, A. (2000), Successive over-relaxation (SOR) and related methods, Journal of Computational and Applied Mathematics, 123, pp. 177-199.
16. Hageman, L.A. and Young, D.M. (1981), Applied Iterative Method, Academic Press, New York.
17. Harlow, F.H. and Welch, J.E. (1965), Numerical calculation of time-dependent viscous incompressible flow with free surface, Phy. Fluids, 8(12), pp. 2182-2189.
18. Hwang, F.N. and Cai, X.C., (2003), A parallel nonlinear additive schwart precondioned inexact Newton algorithm for incompressible Navier-Stokes equations, 2003 Copper Mountain Conference on Multigrid Methods, Copper Mountain, Colorado (USA).
19. Knoll, D.A. and Mousseau, V.A. (2000), On Newton-Krylov-multigrid methods for the incompressible Navier-Stokes equations, J. Comput. Phys., 163, pp. 262-267.
20. Koh, Y.M. (2000), On boundary conditions and numerical methods for the unsteady incompressible Navier-Stokes equations, Computers and Fluids, 29, pp. 791-812.
21. Kopteva, N., (2003), Error expansion for an upwind scheme applied to a two-dimensional convection-diffusion problem, SIAM, J. Numer. Ana., 41(5), pp. 1851-1869.
22. Kupferman, R. and Tadmor, E. (1997), A fast, high resolution, second-order central scheme for incompressible flows, Proc. Natl. Acad. Sci. USA, 94, pp. 4848–4852.
23. Kwak, D., Chang, J.L.C., Shanks, S.P., and Chakravarthy, S. (1986), A three-dimensional incompressible Navier-Strokes flow solver using primitive variables, AIAA J., 24(3), pp. 390-396.
24. Kwak, D., Kiris, C., Dacles-Mariani, J., Rogers, S., Yoon, S. (1998), Incompressible Navier-Stokes solvers using primitive variables, Computational Fluid Dynamics Review, edited by Hafez, M. and Oshima, K., World Scientific.

25. Li, Y. (2000), Comparison of implicit multigrid schemes for three-dimensional incompressible flows, J. Comput. Phys., 177, pp. 134–155.
26. Liu, C., Zheng, X., Sung, C.H. (1998), Preconditioned multigrid methods for unsteady incompressible flows, J. Comput. Phys., 139, pp. 35-57.
27. Liu, K.L. (2002), A Monotonical Approximate Method for Steady Incompressible Navier-Stokes Equation, Master thesis, National University of Singapore.
28. Mallinson, G.D., Davis, V.G. (1977), Three-dimensional natural convection in a box - a numerical study, J. Fluid Mech., 83(1), pp. 1-31.
29. McHugh, P.R. and Knoll, D.A. (1994), Inexact Newton's method solution to the incompressible Navier-Stokes and energy equations using standard and matrix-free implementations, AIAA J., 32 (12), pp. 2394-2400.
30. Melaaen, M.C. (1991), Analysis of fluid flow in constricted tubes and ducts using bodyfitted non-staggered grids, Int. J. Numer. Methods Fluids, 15, pp. 895-923.
31. Mousseau, V.A., Knoll, D.A., and Rider, W.J. (2000), Physics-based preconditioning and Newton-Krylov method for non-equilibrium radiation diffusion, J. Comput. Phys., 160, pp. 743-765.
32. Oosterlee, C.W. and Washio, T. (1998), An evaluation of parallel multigrid as a solver and a preconditioner for singularly perturbed problems, SIAM J. Sci. Comput., 19(1), pp. 87-110.
33. Ortega, J.M. and Rheinboldt, W.C. (1970), Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York.
34. Patankar, S.V. and Spalding, D.B. (1972), A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, Int J. Heat Mass Transfer, 15, pp. 1787-1806.
35. Patankar, S.V. (1980), Numerical Heat Transfer and Fluid Flow, Hemisphere Publishing Co., New York.
36. Pernice, M. and Tocci, M.D. (2000), A multigrid-preconditioned Newton-Krylov method for the incompressible Navier-Stokes equations, SIAM J. Sci. Comput. 23(2), pp. 398-418.
37. Rider W.J., Knoll D.A., and Olson G.L. (1999), A multigrid Newton-Krylov method for multimaterial equilibrium radiation diffusion, J. Comput. Phys., 152, pp. 164-191.
38. Rizzi, A. and Eriksson, L.E. (1985), Computation of inviscid incompressible flow with rotation, J. Comp. Phys., 153, pp. 275-312.

39. Rogers, S.E., Kwak, D. and Kiris C. (1991), Steady and unsteady solutions of the incompressible Navier-Strokes equations, AIAA J., 29(4), pp. 603-610.
40. Saad, Y. and Schultz, M. (1986), GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Statist. Comput., **7**, pp. 856-869.
41. Schreiber, R. and Keller, H.B. (1983), Driven cavity flows by efficient numerical techniques, J. Comput. Phys., 49, pp. 310-333.
42. Shadid J.N., Tuminaro R.S., Walker H.F. (1997), An inexact Newton method for fully coupled solution of the Navier–Stokes equations with heat and mass transport, J. Comput. Phys., 137, pp. 155-185.
43. Tamamids, P., Zhang, G.Q., Assanis, D.N. (1996), Comparison of pressure-based and artificial compressibility method for solving 3D steady incompressible viscous flows, J. Comput. Phys., 124, pp. 1-13.
44. Tidriri, M.D. (1997), Preconditioning techniques for the Newton-Krylov solution of compressible flows, J. Comput. Phys., 132, pp. 51-61.
45. Trottenberg, U., Oosterlee, C.W., Schüller, A. (2001), Multigrid, Academic, London.
46. Turkel, E. (1987), Preconditioned methods for solving the incompressible and low speed compressible equations, J. Comput. Phys., 72, pp. 277-298.
47. Vorst, V.D.H.A. (1990), BI-CGSTAB: A fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems, SIAM J. Sci. Statist. Comput., 13 , pp. 631-644.