INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 12, 535–557 (1991)

CALCULATION OF LAMINAR RECIRCULATING FLOWS USING A LOCAL NON-STAGGERED GRID REFINEMENT SYSTEM

P. COELHO, J. C. F. PEREIRA AND M. G. CARVALHO

Mechanical Engineering Department, Instituto Superior Técnico, Av. Rovisco Pais, 1096 Lisboa Codex, Portugal

SUMMARY

A grid-embedding technique for the solution of two-dimensional incompressible flows governed by the Navier–Stokes equations is presented. A finite volume method with collocated primitive variables is employed to ensure conservation at the interfaces of embedding grids as well as global conservation. The discretized equations are solved simultaneously for the whole domain, providing a strong coupling between regions of different refinement. The formulation presented herein is applicable to uniform or non-uniform Cartesian meshes. The method was applied to the solution of two scalar transport equations, to cavity flows driven by body and shear forces and to a sudden plane contraction flow. The numerical predictions are compared with the exact solutions when available and with experimental data. The results show that neither the convergence rate nor the stability of the method is affected by the presence of embedded grids. Embedded grids provide a better distribution of grid nodes over the computational domain and consequently the solution accuracy was improved. The grid-embedding technique proved also that significant savings in computing time could be achieved.

KEY WORDS Embedding non-staggered grids Incompressible recirculating flows Finite volume method

1. INTRODUCTION

The numerical solution of fundamental or practical engineering flow problems requires in most cases the solution of the Navier–Stokes equations. Numerical solutions based on finite differences/finite volume methods require the overlapping of a discrete computational domain over the physical domain. To decrease local truncation errors in regions of steep gradients or to resolve different flow length scales, it is necessary to use a very large number of mesh points. However, the use of a refined mesh very often brings high densities of mesh points in regions where they are required and also where they are not required. This leads to the use over the whole computational domain of too many mesh points and to very long computing times to achieve convergence. Several different alternatives have been devised to overcome this problem, namely overlapping grids, zonal methods and grid embedding. Other techniques such as the multigrid method and adaptive schemes can be used and combined with these methods.

A technique often employed for treating complicated flow geometries is to use several grids, each one optimized for an individual zone of the flow field. The problem of grid generation is greatly simplified when grids are allowed to overlap,[1] but the transfer of information between the grids becomes complicated and it is difficult to ensure global conservation. Thus this method has been mainly used for the prediction of transonic flows together with a full potential formulation[2]

0271–2091/91/060535–23$11.50

Received November 1989
Revised March 1990

or solving the Euler equations.[3] A more popular approach is the zonal method where the individual grids are patched together rather than overlapped. It is also possible to solve different equation sets in the different zones. This method has been used to solve the full potential equations,[4] the Euler equations,[5,6] the Euler/Navier–Stokes equations[7–9] and the incompressible Navier–Stokes equations.[10]

Grid embedding is a technique where a single coarse grid covers the whole domain and local refinement is carried out in the regions of high gradients without changing the basic grid structure. This method has been applied to the prediction of transonic flows using the full potential equations,[11] the small-disturbance potential equation[12] or the Euler equations[13] and laminar incompressible flows.[14] Adaptive schemes used in conjunction with grid embedding have been applied to compressible flows using either the Euler equations[15–17] or the Navier–Stokes equations.[18,19] The multigrid method has also been coupled with grid embedding in the prediction of transonic flow fields using a potential flow analysis.[20] Adaptive techniques and the multigrid method have both been used together with grid embedding to predict incompressible flows.[21–23]

Each zone in zonal methods or each embedding mesh in grid-embedding techniques is generally considered independent. As recognized by Fuchs,[21] the only interaction among the different subdomains is done by transferring boundary data. This degrades the efficiency and perhaps the stability of the numerical method.

In this paper a new grid-embedding method is presented where the whole domain is treated simultaneously regardless of the level of grid refinement. This enhances the coupling between regions of different level of refinement and does not reduce either the efficiency or the stability of the method. Conservation at grid interfaces is ensured as well as full conservation. The Navier–Stokes equations are solved in non-staggered grids using a primitive variable, finite volume method. The method has been used to predict two-dimensional, steady, laminar, incompressible flows.

In the next section the numerical method developed is described for a general scalar transport equation and for the flow equations in particular. The method has been applied to the solution of two test cases described by only one scalar transport equation as well as to the flow equations for three test cases: a cavity flow driven by combined shear and body forces, the classical lid-driven cavity flow and a sudden plane contraction. Results and discussion are presented in Section 3. In the last section the main conclusions of the present study are drawn.

2. DISCRETIZATION PROCEDURE AND SOLUTION ALGORITHM

2.1. Basis of local grid refinement

The physical domain is discretized using a coarse rectangular mesh, either uniform or non-uniform. In regions of high gradients, finer mesh spacing is required and embedded cells are used. These embedded cells are generated by halving on a cell-by-cell basis the mesh spacing in both the x- and y-direction. This process can be repeated, leading to as many levels of grid embedding as desired.

Each cell is numbered sequentially and its neighbours are stored in unidimensional arrays. Four arrays are necessary to store the north, south, east and west neighbours of each grid node. The dimension of these arrays is slightly higher than the total number of grid nodes. This small overhead in the dimension of the arrays is required to keep track of interfaces between regions of different grid refinement. Three additional points for each cell interface between regions of different level of refinement are stored: two on the coarser side of the interface and one on the

other side. These auxiliary points are also stored sequentially. Thus each array contains sequentially the information concerning the neighbours of the grid nodes, followed by the neighbours of auxiliary points on the finer side of the grid interface and finishing with the neighbours of auxiliary points on the coarser side of the grid interface.

This data structure can be exemplified by referring to Figures 1 and 2. Grid node N1 (see Figure 1) on the finer side of the interface has the south neighbour P1. This is an auxiliary point whose west and east neighbours are grid nodes W and P respectively. Grid node S (see Figure 2)

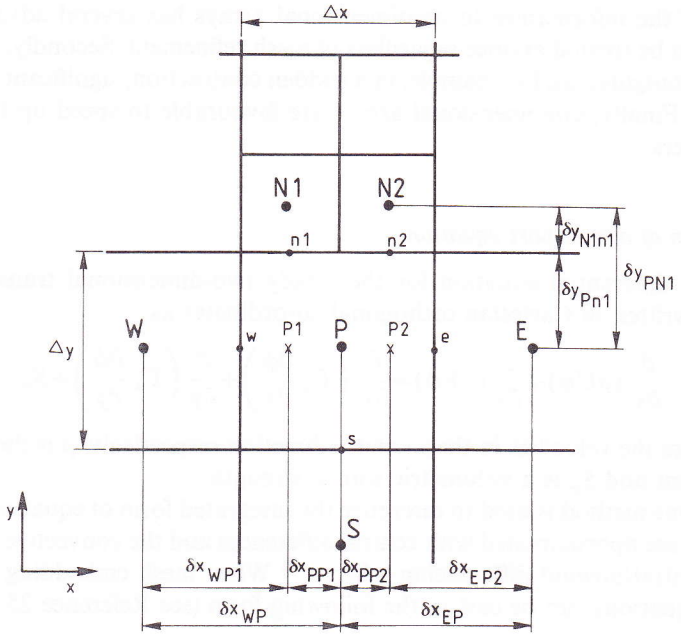


Figure 1. Coarse-grid control volume

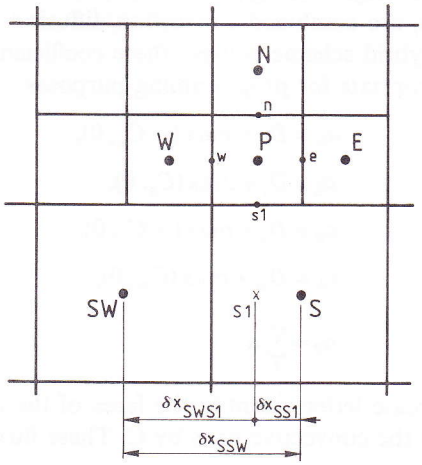


Figure 2. Fine-grid control volume

on the coarser side of the interface has the north neighbour e, an auxiliary point whose west and east neighbours are grid nodes P and E respectively.

Notice that the dimension of the arrays where the dependent variables are stored is equal to the number of grid nodes. Only the dimension of the four arrays containing the neighbours of each cell is slightly higher, as explained above.

With this data structure and provided the refinement ratio is kept equal to two, all possible interfaces are of the kind depicted in Figures 1 and 2 irrespective of the refinement level. No additional complications arise from the use of several levels of grid refinement. It is only necessary to give as input the refinement level desired for each cell of the coarsest grid.

The storage of the information in unidimensional arrays has several advantages. First, the whole domain can be treated at once regardless of mesh refinement. Secondly, when the physical domain is not rectangular, as, for example, in a sudden contraction, significant savings in storage can be achieved. Finally, unidimensional arrays are favourable to speed up the calculations in vectorial computers.

2.2. *Discretization of a transport equation*

The governing differential equation for the steady two-dimensional transport of a general scalar ϕ can be written in Cartesian orthogonal co-ordinates as

$$\frac{\partial}{\partial x}(\rho U\phi)+\frac{\partial}{\partial y}(\rho V\phi)=\frac{\partial}{\partial x}\left(\Gamma_\phi \frac{\partial\phi}{\partial x}\right)+\frac{\partial}{\partial y}\left(\Gamma_\phi \frac{\partial\phi}{\partial y}\right)+S_\phi, \tag{1}$$

where U and V are the velocities in the x- and y-direction respectively, ρ is the density, Γ_ϕ is the diffusion coefficient and S_ϕ is a volumetric source strength.

The finite volume method is used to discretize the integrated form of equation (1). The diffusive and source terms are approximated with central differences and the convective term is discretized by the hybrid central/upwind differencing scheme.[24] When mesh embedding is not considered, the discretized equations can be cast in the following form (see Reference 25 for details):

$$a_P\phi_P=\sum_i a_i\phi_i+S_u, \tag{2}$$

where the index i runs over all neighbouring points (N, S, E, and W), S_u denotes the source term and the coefficients a_P and a_i are combined convection/diffusion fluxes across the faces of the control volume. When the hybrid scheme is used, these coefficients are given by the following well-known expressions appropriate for programming purposes:

$$a_N=D_n+\max(-C_n, 0), \tag{3a}$$

$$a_S=D_s+\max(C_s, 0), \tag{3b}$$

$$a_E=D_e+\max(-C_e, 0), \tag{3c}$$

$$a_W=D_w+\max(C_w, 0), \tag{3d}$$

$$a_P=\sum_i a_i. \tag{3e}$$

In these equations the lowercase letters identify the faces of the control volume. The diffusive fluxes are denoted by D and the convective ones by C. These fluxes are evaluated as

$$C_n=\rho_n V_n\Delta x, \tag{4a}$$

$$C_e = \rho_e U_e \Delta y \tag{4b}$$

and

$$D_n = \Gamma_n \Delta x / \delta y_{NP}, \tag{5a}$$

$$D_e = \Gamma_e \Delta y / \delta x_{EP}, \tag{5b}$$

and similarly for the south and west faces. Distances between nodes N and P and nodes E and P are denoted by δy_{NP} and δx_{EP} respectively.

When grid embedding is used, interfaces between regions of different refinement need particular attention. In all such interfaces there is one grid node on one side of the interface and two nodes on the other side, as sketched in Figure 1. To exemplify the method developed here, only one interface between regions of different refinement is considered, at the north face of the control volume, but other interfaces can be handled as well.

Perhaps the most attractive feature of the control volume formulation is that the resulting solution implies integral conservation of quantities such as mass, momentum and energy over any group of control volumes, including the whole domain. Thus special care was taken in order to ensure conservation across the interfaces. This can be automatically ensured if fluxes are calculated in the same way for grid nodes on both sides of the interface. Hence the fluxes are calculated using the dependent variable values at grid nodes N1 and N2 and the auxiliary points P1 and P2 represented in Figure 1. Points P1 and P2 have the same y-co-ordinate of the grid node P and the x-co-ordinates are equal to the x-co-ordinates of points N1 and N2 respectively. When the control volume associated with node N1 is considered, the south flux is calculated using node N1 and point P1. Similarly, node N2 and point P2 are used to calculate the south flux for the control volume centred in node N2. Finally, the flux across the north interface for the node P control volume is the sum of two terms, one calculated using node N1 and point P1 and the other calculated using node N2 and point P2. In this way the discretized equation for the control volume centred in node P can be written as

$$a_P \phi_P = \sum_i a_i \phi_i + a_{N1}(\phi_{N1} - \phi_{P1}) + a_{N2}(\phi_{N2} - \phi_{P2}) + S_u, \tag{6}$$

where the summation extends over points S, E and W only, as well as the summation for evaluating a_P.

The problem of dealing with an interface between regions with different refinement has been replaced by the problem of handling the second and third terms of the right-hand side of the previous equation. If the two terms were included in the source term S_u, this would strongly weaken the link between node P and nodes N1 and N2. There would be an explicit link through the source term S_u rather than an implicit link through the coefficients a_i and so the interface would behave as a boundary. Consequently the rate of convergence would decrease significantly. On the other hand, if the terms $a_{N1}\phi_{N1}$ and $a_{N2}\phi_{N2}$ were included in the summation and the terms $a_{N1}\phi_{P1}$ and $a_{N2}\phi_{P2}$ were included in the source term, the aforementioned problem would be solved but in this case we would end up with a coefficient matrix which is not diagonally dominant. This would be only a restriction to the choice of the solver but we prefer to avoid this additional complication. The approach followed here begins with the evaluation of ϕ_{P1} and ϕ_{P2} by means of linear interpolation from neighbouring grid nodes:

$$\phi_{P1} = (\delta x_{WP1}\phi_P + \delta x_{PP1}\phi_W)/\delta x_{WP}, \tag{7a}$$

$$\phi_{P2} = (\delta x_{EP2}\phi_P + \delta x_{PP2}\phi_E)/\delta x_{EP}. \tag{7b}$$

Introducing these expressions in equation (6), one obtains, after simple algebraic manipulations,

$$a_P\phi_P=\sum_i a_i\phi_i+\frac{\delta x_{WP1}}{\delta x_{WP}}a_{N1}(\phi_{N1}-\phi_P)+\frac{\delta x_{EP2}}{\delta x_{EP}}a_{N2}(\phi_{N2}-\phi_P)$$

$$+\frac{\delta x_{PP1}}{\delta x_{WP}}a_{N1}(\phi_{N1}-\phi_W)+\frac{\delta x_{PP2}}{\delta x_{EP}}a_{N2}(\phi_{N2}-\phi_E)+S_u. \tag{8}$$

The second and third terms on the right-hand side of this equation can be included in the summation, ensuring a strong coupling between grid nodes on opposite sides of the north interface, whereas the fourth and fifth are included in the source term. Thus the final discretized equation can be cast in a form similar to equation (2):

$$a'_P\phi_P=\sum_i a'_i\phi_i+S'_u, \tag{9}$$

where the summation includes the five neighbours (S, E, W, N1 and N2). Combined convection/diffusion coefficients are given by

$$a'_S=a_S, \tag{10a}$$

$$a'_E=a_E, \tag{10b}$$

$$a'_W=a_W, \tag{10c}$$

$$a'_{N1}=\frac{\delta x_{WP1}}{\delta x_{WP}}a_{N1}=\frac{\delta x_{WP1}}{\delta x_{WP}}[D_{n1}+\max(-C_{n1},0)], \tag{10d}$$

$$a'_{N2}=\frac{\delta x_{EP2}}{\delta x_{EP}}a_{N2}=\frac{\delta x_{EP2}}{\delta x_{EP}}[D_{n2}+\max(-C_{n2},0)], \tag{10e}$$

$$a'_P=\sum_i a'_i, \tag{10f}$$

where

$$C_{n1}=\rho_{n1}V_{n1}(\Delta x/2), \tag{11a}$$

$$D_{n1}=\Gamma_{n1}(\Delta x/2)/\delta y_{PN1}, \tag{11b}$$

and similarly for C_{n2} and D_{n2}. The source term is written as

$$S'_u=S_u+\frac{\delta x_{PP1}}{\delta x_{WP}}a_{N1}(\phi_{N1}-\phi_W)+\frac{\delta x_{PP2}}{\delta x_{EP}}a_{N2}(\phi_{N2}-\phi_E). \tag{12}$$

The same ideas are used to write a discretized equation for a control volume on the opposite side of the interface as represented in Figure 2, where point P is the correspondent to point N1 in Figure 1. For the sake of simplicity, only one interface between regions of different refinement is considered again. The flux across the south boundary is calculated with the help of point S1, where the dependent variable is calculated from

$$\phi_{S1}=(\delta x_{SWS1}\phi_S+\delta x_{SS1}\phi_{SW})/\delta x_{SSW}. \tag{13}$$

The introduction of this expression in the discretized equation yields

$$a_P\phi_P=\sum_i a_i\phi_i+\frac{\delta x_{SWS1}}{\delta x_{SSW}}a_S(\phi_S-\phi_P)+\frac{\delta x_{SS1}}{\delta x_{SSW}}a_S(\phi_{SW}-\phi_P)+S_u. \tag{14}$$

This is the equation corresponding to equation (8) and the summation now includes neighbours E, W and N. Equation (9) still holds with the summation extending over neighbours N, S, E and W and with coefficients given by

$$a'_{\mathrm{N}} = a_{\mathrm{N}}, \tag{15a}$$

$$a'_{\mathrm{E}} = a_{\mathrm{E}}, \tag{15b}$$

$$a'_{\mathrm{W}} = a_{\mathrm{W}}, \tag{15c}$$

$$a'_{\mathrm{S}} = \frac{\delta x_{\mathrm{sws1}}}{\delta x_{\mathrm{ssw}}} a_{\mathrm{S}} = \frac{\delta x_{\mathrm{sws1}}}{\delta x_{\mathrm{ssw}}} [D_{\mathrm{s}} + \max(C_{\mathrm{s}}, 0)], \tag{15d}$$

$$a'_{\mathrm{P}} = \sum_i a'_i + \frac{\delta x_{\mathrm{ss1}}}{\delta x_{\mathrm{ssw}}} a_{\mathrm{S}} \tag{15e}$$

and

$$S'_u = S_u + \frac{\delta x_{\mathrm{ss1}}}{\delta x_{\mathrm{ssw}}} a_{\mathrm{S}} \phi_{\mathrm{sw}}. \tag{16}$$

Thus the discretized equations for any control volume can be written as equation (9). Whenever common interfaces are presented, the coefficients of this equation are calculated in the conventional way using equations (3)–(5). In the boundaries of embedding meshes the coefficients are modified as described above.

The method used to obtain values required for the flux balance at interfaces ensures conservation and is based on linear interpolations. Although more complex interpolation schemes could have been used, the present one was chosen because despite its simplicity it has yielded accurate predictions for the test cases studied.

2.3. Discretization of the flow equations on a non-staggered grid

The governing differential equations for mass and momentum conservation in a 2D, steady, laminar, incompressible flow can be written as

$$\frac{\partial}{\partial x}(\rho U) + \frac{\partial}{\partial y}(\rho V) = 0, \tag{17}$$

$$\frac{\partial}{\partial x}(\rho UU) + \frac{\partial}{\partial y}(\rho VU) = \frac{\partial}{\partial x}\left(\mu \frac{\partial U}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu \frac{\partial U}{\partial y}\right) - \frac{\partial p}{\partial x}, \tag{18}$$

$$\frac{\partial}{\partial x}(\rho UV) + \frac{\partial}{\partial y}(\rho VV) = \frac{\partial}{\partial x}\left(\mu \frac{\partial V}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu \frac{\partial V}{\partial y}\right) - \frac{\partial p}{\partial y}, \tag{19}$$

where p is the static pressure and μ is the dynamic viscosity. The terms of these equations are identical to those of equation (1). Indeed, setting $\phi = 1$ in equation (1), the continuity equation (17) is recovered, whereas setting $\phi = U$ or $\phi = V$ leads to the momentum equations. Thus the discretization of these equations follows the steps previously described. However, the coupling between pressure and velocity poses new problems which traditionally have been solved by means of a staggered grid. In such a grid the location of the control volumes for the velocity components is shifted in such a way that the velocities at the faces of the scalar control volumes become directly available. Since there are three sets of control volumes in the two-dimensional case, the complexity and the storage requirements are much greater than for a collocated grid. In

the last few years several authors[26,27] have shown that non-staggered grids can produce as accurate solutions as staggered ones with no increase in computational effort provided that a special interpolation is used to calculate the velocities at the faces of the control volumes. The grid-embedding structure would become much more complicated if a staggered grid were used. Hence a non-staggered grid was used in this work.

2.4. *Pressure–velocity coupling*

The method described by Peric *et al.*,[27] which is a modified version of the SIMPLE algorithm applied to non-staggered grids, is used here, although underrelaxation is introduced in a different way as explained below.

The U- and V-momentum equations are solved first using guessed values for the pressure field and for the mass fluxes. Pressures at the control volume faces are calculated by linear interpolation. In general, the solution of these equations, denoted by U^* and V^*, does not satisfy continuity and thus a source term appears in the continuity equation given by

$$S_u = C_n^* - C_s^* + C_e^* - C_w^*, \tag{20}$$

where the asterisk identifies a guessed value. The modification of this expression when an interface between grids of different refinement is present is straightforward. Evaluation of the convective fluxes requires the calculation of the velocity components at the faces of the control volumes. Peric *et al.*[27] have shown that if linear interpolation were used to calculate the velocity components at the interfaces, decoupling between pressure and velocities would occur. To see how this can be avoided, we write the discretized x-momentum equations for grid nodes P and E, in Figure 1 or 2, as

$$U_P^* = \left(\frac{\sum_i a_i U_i^* - \Delta y (p_e^* - p_w^*)}{a_P} \right)_P, \tag{21a}$$

$$U_E^* = \left(\frac{\sum_i a_i U_i^* - \Delta y (p_e^* - p_w^*)}{a_P} \right)_E. \tag{21b}$$

These equations come directly from equation (9) when the primes are dropped for simplicity. The asterisks are introduced to identify guessed values and the source term is explicitly written as the pressure gradient. The U^*-velocity at the cell interface is obtained from equations (21) using linear interpolation for the summations (denoted by the overbar in the equation terms) and the pressure difference between grid nodes on each side of the interface. This yields

$$U_e^* = \left(\frac{\overline{\sum_i a_i U_i^*}}{a_P} \right)_e - \left(\frac{\overline{1}}{a_P} \right)_e \Delta y (p_E^* - p_P^*). \tag{22}$$

Hence the cell face velocity depends directly on the pressures at the two neighbour nodes, which is the basis of the staggering practice.

For the interfaces between regions of different refinement the same ideas are used. Thus, referring to Figure 1, the north cell face velocity is given by

$$V_{n1}^* = \left(\frac{\overline{\sum_i a_i V_i^*}}{a_P} \right)_{n1} - \left(\frac{\overline{1}}{a_P} \right)_{n1} \frac{\Delta x}{2} (p_{N1}^* - p_{P1}^*), \tag{23}$$

where the overbar denotes the linearly interpolated value from grid node N1 and point P1. Values at point P1 are linearly interpolated between grid nodes W and P. A similar equation is used to calculate V^*_{n2}. On the opposite side of the interface (see Figure 2) we have

$$V^*_{s1}=\left(\frac{\overline{\sum_i a_i V^*_i}}{a_P}\right)_{s1}-\left(\frac{\overline{1}}{a_P}\right)_{s1}\frac{\Delta x}{2}(p^*_P-p^*_{S1}), \tag{24}$$

where the overbar denotes the linearly interpolated value from grid node P and point S1. Values at point S1 are linearly interpolated between grid nodes S and SW.

To enforce mass conservation, velocity and pressure corrections are introduced as in the SIMPLE algorithm:

$$U'_e=-\left(\frac{\overline{1}}{a_P}\right)_e\Delta y(p'_E-p'_P). \tag{25}$$

Identical expressions can be written at the boundaries of embedding grids. When the velocity components are expressed as a sum of estimated plus corrected (identified by the prime) components and introduced in the continuity equation, a pressure correction equation results. The solution of this equation gives the corrections used to update velocities and pressure as well as mass fluxes at the control volume faces. These mass fluxes are used to calculate the convective terms in the momentum equations.

To attain convergence, underrelaxation factors for velocities and pressure are used. The general discretized equation (9) is changed according to

$$\phi_P=\alpha_\phi\frac{\sum_i a_i\phi_i+S_u}{a_P}+(1-\alpha_\phi)\phi_P^{old}, \tag{26}$$

where the primes have been dropped for simplicity, α_ϕ is the underrelaxation factor and ϕ_P^{old} is the ϕ-value at node P in the previous iteration. The velocity at a cell face is calculated according to the pressure-weighted interpolation method suggested by Majumdar[28] and Miller and Schmidt[29] and examined by Kobayashi and Pereira[30] for flows with rapidly varying pressure gradient regions yielding solutions independent of relaxation factors. Using this method, equations (21) and (22) read

$$U^*_P=\alpha_U\left(\frac{\sum_i a_i U^*_i-\Delta y(p^*_e-p^*_w)}{a_P}\right)_P+(1-\alpha_U)U_P^{old}, \tag{27a}$$

$$U^*_E=\alpha_U\left(\frac{\sum_i a_i U^*_i-\Delta y(p^*_e-p^*_w)}{a_P}\right)_E+(1-\alpha_U)U_E^{old}, \tag{27b}$$

$$U^*_e=\alpha_U\left(\frac{\overline{\sum_i a_i U^*_i}}{a_P}\right)_e-\alpha_U\left(\frac{\overline{1}}{a_P}\right)_e\Delta y(p^*_E-p^*_P)+(1-\alpha_U)U_e^{*old}. \tag{28}$$

Grid embedding does not present further complexities.

For the solution of the systems of linear equations, two different methods have been used aiming to identify which one would be more suitable to employ with the grid-embedding

technique. One is the biconjugate gradient method[31] using an incomplete block-LU decomposition[32] as preconditioning. The other is a modified version of the Gauss–Seidel line-by-line iteration procedure. With grid embedding a grid node can have more than four neighbours and this requires a modification of the usual version of the method. The new feature is the increase of the number of sweeps whenever an interface between regions with different grid refinement is found in order to make it possible to apply the Thomas algorithm in each sweep. This can be better explained with the help of Figure 1. A sweep in the x-direction presents no problems since the values of the variable at the south and north interfaces are temporarily assumed as known. But in the y-direction there is a problem since grid node P has two north neighbours. The idea is to perform two sweeps. In the first one the values of the variable at grid nodes W, E and N2 are assumed as known and in the second sweep the values at grid nodes W, E and N1 are assumed as known. Thus the number of sweeps in a given direction is dictated by the maximum level of grid refinement in that direction.

The convergence criterion is to demand the normalized sum of the absolute residuals for each variable over all the control volumes to be less than or equal to a prespecified value. The normalization value is the inlet mass flow rate for the pressure correction equation and the inlet momentum for the momentum equations.

3. RESULTS AND DISCUSSION

3.1. Case 1: scalar transport equation

The general transport equation (1) has been solved for a case where an analytic solution exists[33] given by

$$\phi = \sin(\pi x)\cos(\pi y), \tag{29a}$$

$$U = -\lambda\pi \sin(\pi x)\sin(\pi y), \tag{29b}$$

$$V = -\lambda\pi \cos(\pi x)\cos(\pi y), \tag{29c}$$

$$S_\phi = 2\pi^2\phi. \tag{29d}$$

The density and the diffusion coefficient were set equal to unity and λ is an independent solution parameter which may be used to change the Peclet number. The calculated analytic solution (equation (29a)) is shown in Figure 3 for the domain $x \in [0, 1]$, $y \in [-0{\cdot}5, 0{\cdot}5]$. Owing to the symmetry of the problem, the calculation domain was restricted to $0{\cdot}5 \leqslant x \leqslant 1$, $0 \leqslant y \leqslant 0{\cdot}5$ with symmetry boundary conditions imposed on $x = 0{\cdot}5$ and $y = 0$.

The analytical and numerical solutions were compared by means of the average absolute errors calculated over the computational domain:

$$\bar{E}_\phi = \frac{\sum_{i=1}^{N} [\phi_i - \sin(\pi x)\cos(\pi y)]}{N}, \tag{30}$$

where N is the number of grid nodes and ϕ_i denotes the numerical solution of each grid node.

Figure 4(a) shows a 40×40 uniform grid and the corresponding contour levels of the local absolute errors obtained with this grid. It can be seen that maximum errors occur close to the left bottom corner of the calculation domain and so this region was covered with an embedding grid as shown in Figure 4(b). When a conventional code is used, this refinement must be extended up to the opposite boundaries and such a grid is represented in Figure 4(c). The corresponding

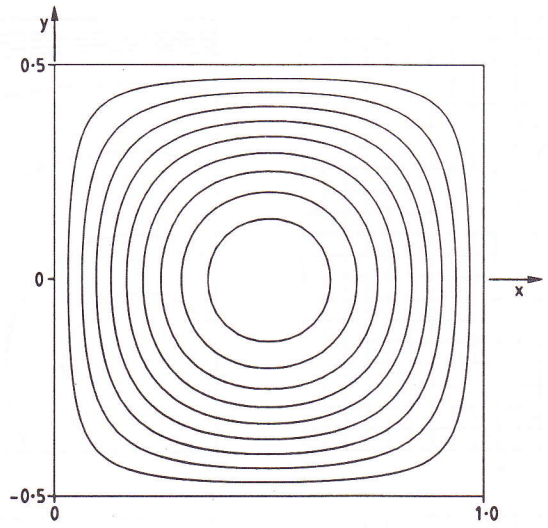


Figure 3. Streamlines of the flow

Table I. CPU time and mean value of absolute errors for the grids used in test case 1

Base grid	Type of grid (see Figure 4)	Embedding grids	Number of grid nodes	CPU time (s)	$\bar{E}_\phi$
20 × 20	4(a)	No	400	15	$3{\cdot}6 \times 10^{-4}$
20 × 20	4(b)	Yes	448	18	$3{\cdot}2 \times 10^{-4}$
24 × 24	4(c)	No	576	25	$2{\cdot}1 \times 10^{-4}$
20 × 20	4(b)	Yes	592	26	$2{\cdot}1 \times 10^{-4}$
28 × 28	4(c)	No	784	39	$1{\cdot}4 \times 10^{-4}$
40 × 40	4(a)	No	1600	98	$0{\cdot}8 \times 10^{-4}$
40 × 40	4(b)	Yes	1792	107	$0{\cdot}7 \times 10^{-4}$
48 × 48	4(c)	No	2304	150	$0{\cdot}4 \times 10^{-4}$
40 × 40	4(b)	Yes	2368	179	$0{\cdot}5 \times 10^{-4}$
56 × 56	4(c)	No	3136	268	$0{\cdot}3 \times 10^{-4}$

contours of absolute errors show that the local errors decrease in the regions where grid refinement was chosen.

All the meshes used in the solution of this problem are described in Table I. Only one level of grid refinement was used for embedding grids. Results presented in Table I were obtained using the preconditioned biconjugate gradient method. This method yielded for this linear problem a faster convergence than the line Gauss–Seidel method. The λ-parameter (see equations (29)) was set equal to unity, yielding a maximum local Peclet number of less than two. Therefore central differences are used for convection discretization and very small solution errors were found (see Table I). Thus the idea of using finer grids in this case is to investigate the effect of the use of embedding grids on convergence and computing time. Convergence was achieved when the sum of the residuals was less than 10^{-3}.

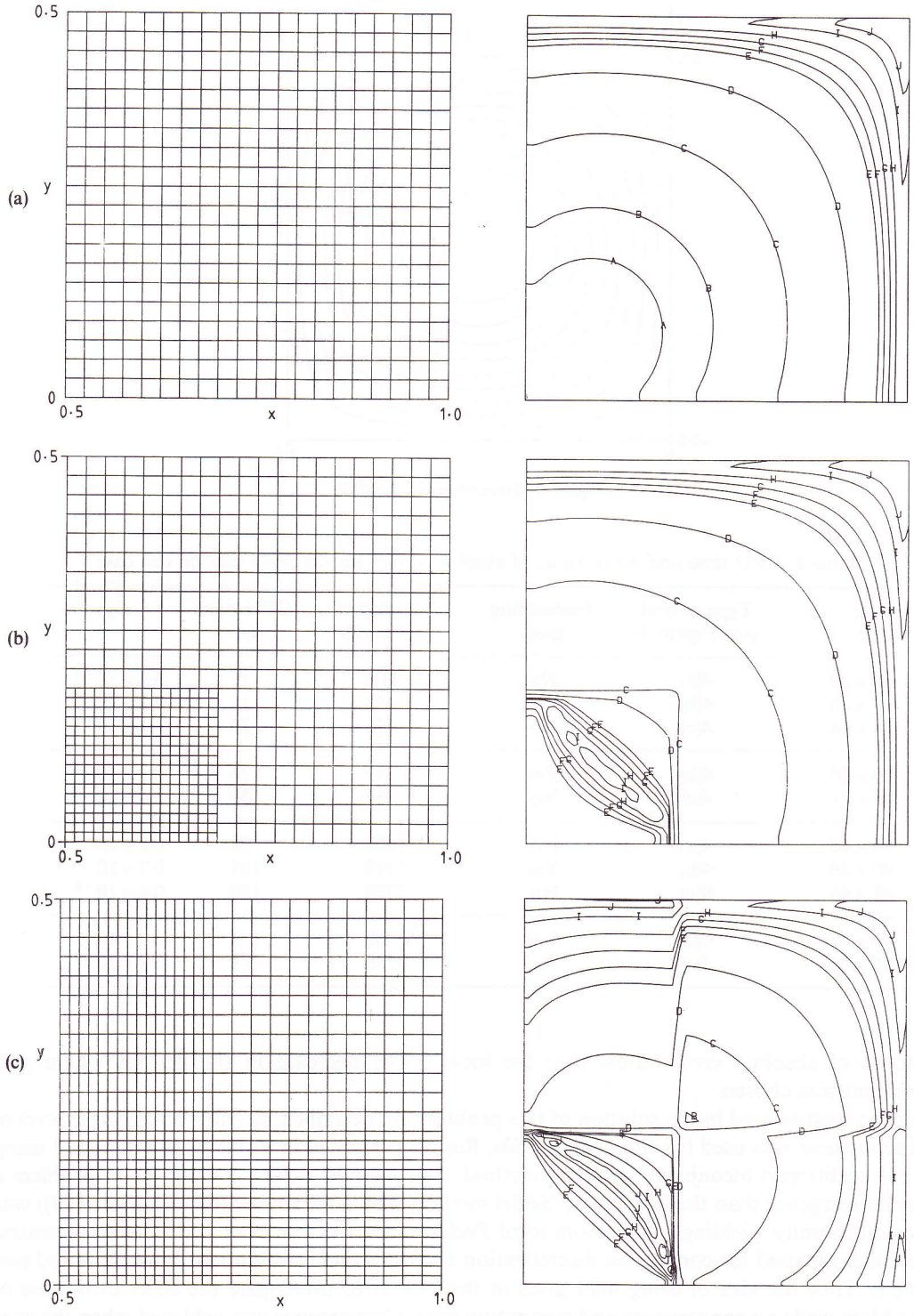


Figure 4. Typical meshes and local absolute error contours for test case 1: (a) uniform 20×20 mesh; (b) 20×20 coarse grid with one level of refinement; (c) non-uniform 28×28 mesh. Contour values: A, 7×10^{-4}; B, 6×10^{-4}; C, 4×10^{-4}; D, 2×10^{-4}; E, 1×10^{-4}; F, $0{\cdot}8 \times 10^{-4}$; G, $0{\cdot}6 \times 10^{-4}$; H, $0{\cdot}4 \times 10^{-4}$; I, $0{\cdot}2 \times 10^{-4}$; J, $0{\cdot}1 \times 10^{-4}$

The results listed in Table I show, for example, that the solutions obtained with the grid represented in Figure 4(c) with 576 grid nodes and a grid similar to the one shown in Figure 4(b) but with a larger embedding grid (592 grid nodes) required about the same CPU time. However, a reduction of 30% in CPU time can be obtained with an embedding grid when compared with a conventional grid with the same refinement extended up to the boundaries. The CPU time can be markedly reduced if more than one level of grid refinement is used. Calculations were also performed by changing the parameter λ between 0·1 and 100, but they are not presented here since the results are qualitatively similar.

3.2. Case 2: transport of a step change

The transport of a step change in a uniform velocity field is a classical test case for assessing convection discretization schemes. Here the idea is not to assess the discretization scheme but to show that the accuracy can be improved when a given number of grid nodes are distributed according to the features of the solution by means of embedding grids.

The scalar transport equation (1) with no source term is solved in a square domain. The velocity field is uniform and its direction defines the boundary conditions. A line with the direction of the velocity passing through the centre of the domain originates two regions. The boundary ϕ-value is equal to unity on the upper region and zero elsewhere. Thus the solution presents a steep gradient close to that line. When the velocity is aligned with the x- or y-direction, the solution is easily predicted, but when the velocity vector is skewed with grid lines and local Peclet numbers are high, false diffusion yields poor results for most convection discretization schemes.

Solutions are presented for a Reynolds number equal to 500, taking the side of the square domain as the characteristic dimension. The angle θ between the velocity vector and the x-direction was considered equal to 45° and 22° for the two cases studied. Figure 5 shows some of the meshes used. Several levels of embedded grids were used, with a maximum level of refinement along the direction of the velocity and passing through the centre of the domain where a steep gradient occurs.

Figure 6(a) shows the profiles corresponding to the vertical centreline and $\theta = 45°$ for several meshes with a similar number of grid nodes. Each profile presents the results corresponding to a uniform grid and an embedding one (obtained from a coarser grid refined along the direction of the velocity vector). As expected, the numerical solution only becomes close to the exact one when the dimension of the cells leads to a local absolute value of the Peclet number of less than two. In fact, this is the limit value above which the first-order upwind differencing scheme suffering from false diffusion is used. However, the point we want to stress here is that for two grids with about the same number of grid nodes, the one which has been locally refined yields significantly better results. To attain the same level of accuracy with a conventional code, the refinement would have to be extended from one boundary to the opposite one. This would require a much larger number of grid nodes, with a consequent increase in CPU time.

When the angle θ is reduced to 22°, the behaviour of the solutions shown in Figure 6(b) is identical to that for $\theta = 45°$ but the problem of false diffusion is not as severe as previously.

In both cases, $\theta = 22°$ and 45°, Figure 6 shows that the numerical solution through the interfaces of the embedding grid is smooth.

3.3. Case 3: cavity flow driven by combined shear and body forces

The 2D steady, laminar, incompressible flow equations were solved for a cavity where the flow is driven by combined shear and body forces. This test case was studied because the exact solution is known and given by Shih *et al.*[34] The boundary U- and V-velocities are zero everywhere except

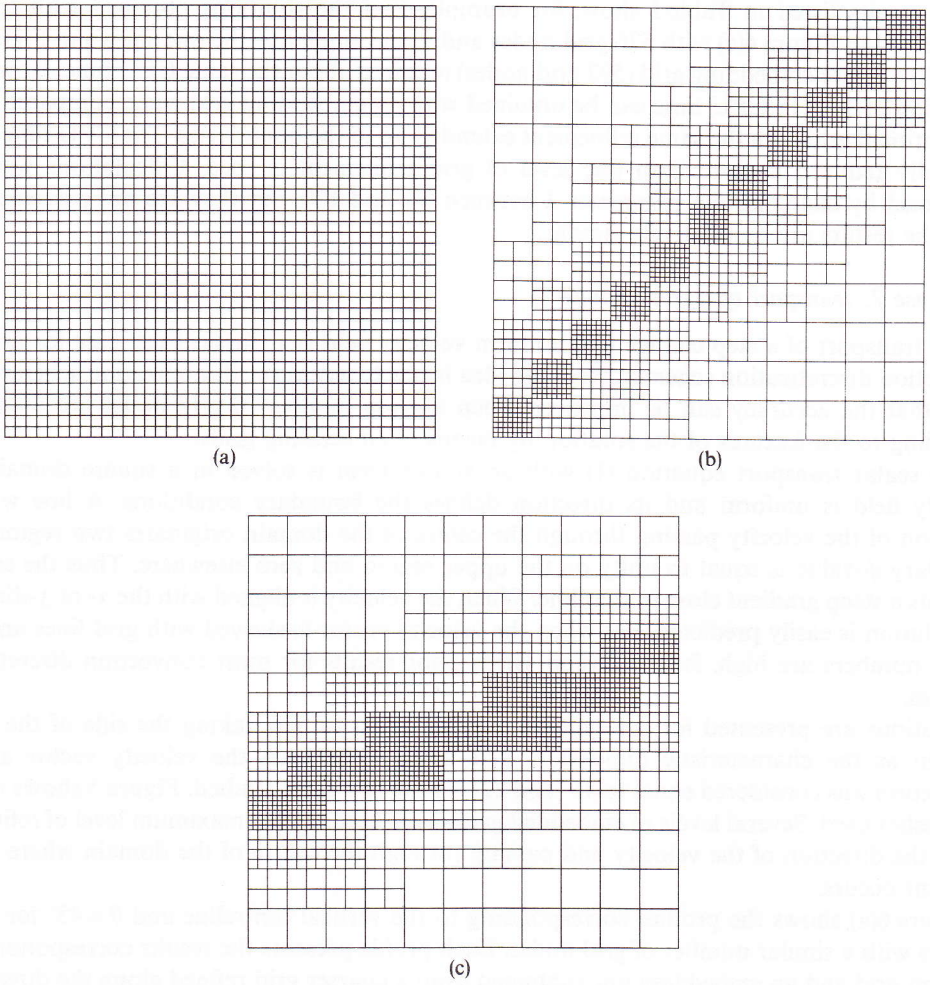


Figure 5. Typical meshes for test case 2: (a) uniform 40×40 mesh (1600 grid nodes); (b) embedding grid for $\theta = 45°$ (1474 grid nodes); (c) embedding grid for $\theta = 22°$ (1726 grid nodes)

along the top surface, where V is zero but there is a positive U-velocity driving the flow.

Calculations were performed for $Re = 1$ and 10, but since no significant differences were found, only the results for $Re = 1$ will be presented here. Both of the aforementioned methods for the solution of the systems of linear equations were tried. It was found that when the underrelaxation parameters are optimized as well as the number of solver iterations, the preconditioned biconjugate gradient method converges faster than the Gauss–Seidel line-by-line iteration procedure. However, the convergence rate is much more dependent on the underrelaxation parameters when the biconjugate gradient method is employed. Thus all the results presented from now on were obtained using the line-by-line iteration procedure. The underrelaxation parameters used were 0·9 and 0·1 for the velocities and for the pressure respectively.

The solution of the problem using a uniform 20×20 mesh has shown that the results were close to the exact solution except near the bottom and side boundaries. Thus a coarser base mesh with embedded grids close to those boundaries was chosen. Both grids are shown in Figure 7. Figure 8

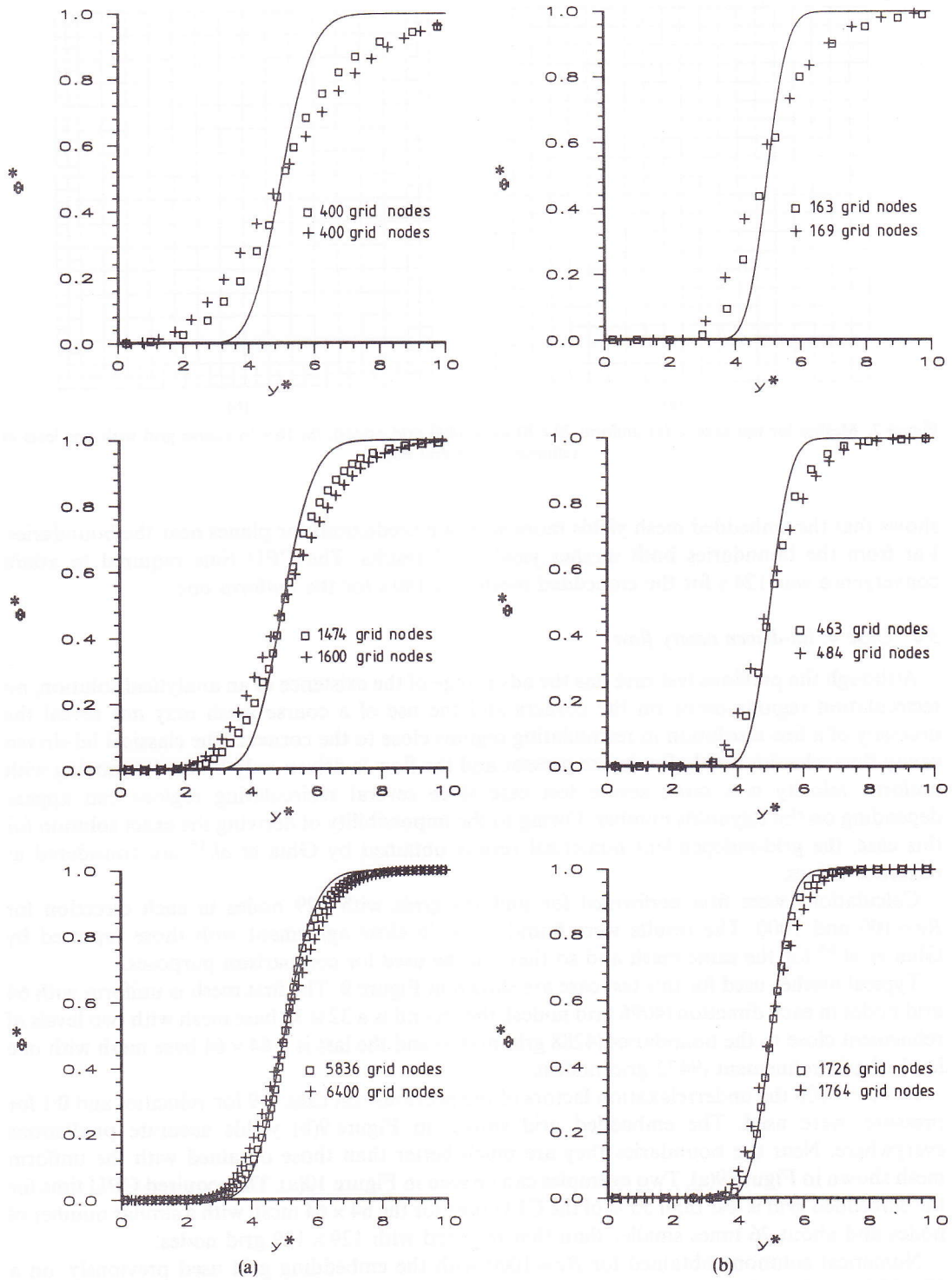
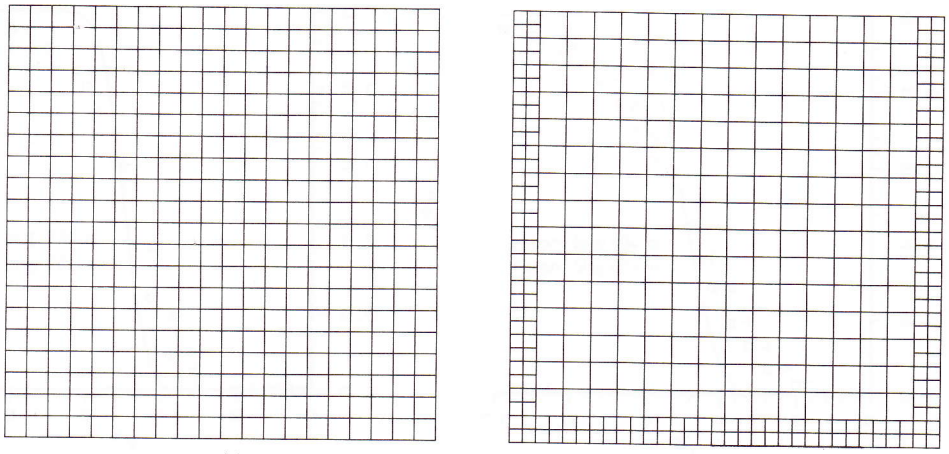


Figure 6. Step transport solutions on vertical centreline (——, exact solution; □, embedding grid; +, uniform grid) (a) $\theta = 45°$; (b) $\theta = 22°$



(a) (b)

Figure 7. Meshes for test case 3: (a) uniform 20×20 mesh (400 grid nodes); (b) 16×16 coarse grid with one level of refinement (394 grid nodes)

shows that the embedded mesh yields more accurate predictions for planes near the boundaries. Far from the boundaries both meshes yield good results. The CPU time required to attain convergence was 124 s for the embedded mesh and 140 s for the uniform one.

3.4. Case 4: lid-driven cavity flow

Although the previous test case has the advantage of the existence of an analytical solution, no recirculation regions occur on the corners and the use of a coarse mesh may not reveal the necessity of a fine resolution in recirculating regions close to the corners. The classical lid-driven cavity flow where no body forces are present and the flow is driven only by the lid moving with uniform velocity is a more severe test case since several recirculating regions can appear depending on the Reynolds number. Owing to the impossibility of deriving the exact solution for this case, the grid-independent numerical results obtained by Ghia *et al.*[35] are considered as reference values.

Calculations were first performed for uniform grids with 129 nodes in each direction for $Re = 100$ and 1000. The results were found to be in close agreement with those reported by Ghia *et al.*[35] for the same mesh and so they will be used for comparison purposes.

Typical meshes used for this test case are shown in Figure 9. The first mesh is uniform with 64 grid nodes in each direction (4096 grid nodes), the second is a 32×32 base mesh with two levels of refinement close to the boundaries (4288 grid nodes) and the last is a 64×64 base mesh with one level of grid refinement (9472 grid nodes).

For $Re = 100$ the underrelaxation factors of the previous test case, 0·9 for velocities and 0·1 for pressure, were used. The embedded grid shown in Figure 9(b) yields accurate predictions everywhere. Near the boundaries they are much better than those obtained with the uniform mesh shown in Figure 9(a). Two examples can be seen in Figure 10(a). The required CPU time for the embedded grid is less than 50% of the CPU time for the 64×64 mesh with a similar number of nodes and about 26 times smaller than that required with 129×129 grid nodes.

Numerical solutions obtained for $Re = 1000$ with the embedding grid used previously, on a 32×32 base, show that the grid is too coarse since the predicted results (not shown here) were far from those predicted using a very refined mesh. On the other hand, the 64×64 uniform grid leads

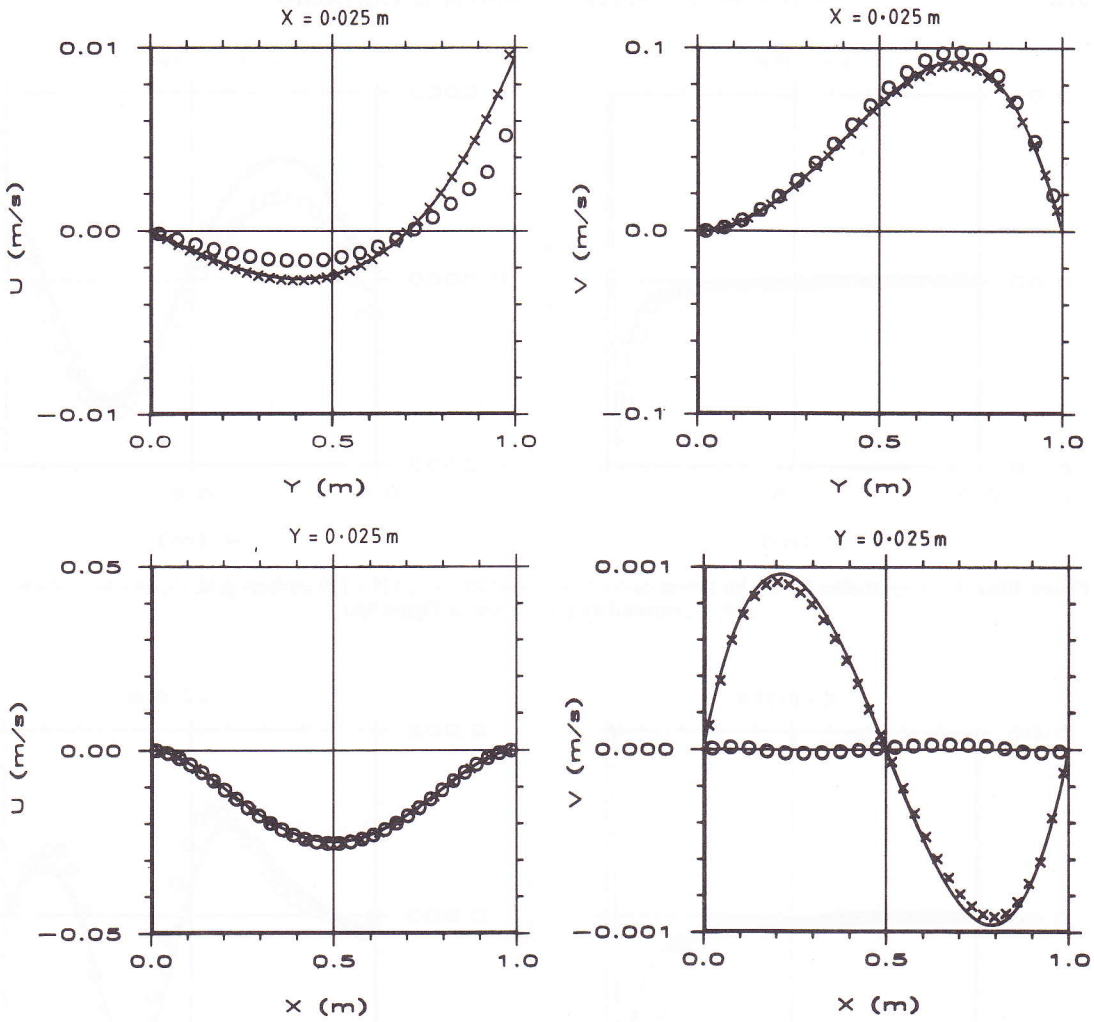
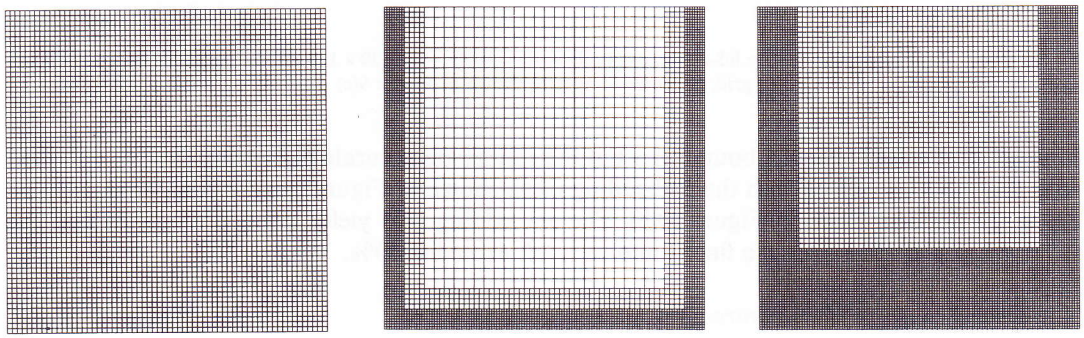


Figure 8. Velocity profiles for the cavity flow driven by combined shear and body forces: ——, exact solution; ○, uniform grid shown in Figure 7(a); ×, embedding grid shown in Figure 7(b)



(a) (b) (c)

Figure 9. Meshes for test case 4: (a) uniform 64×64 mesh (4096 grid nodes); (b) 32×32 coarse grid with two levels of refinement (4288 grid nodes); (c) 64×64 coarse grid with one level of refinement (9472 grid nodes)

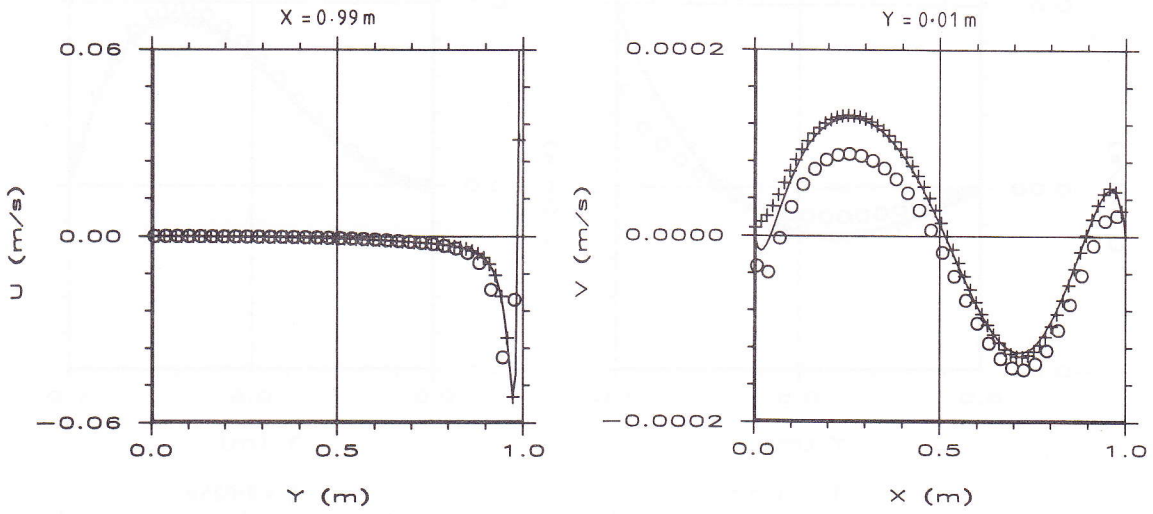


Figure 10(a). Velocity profiles for the lid-driven cavity flow, $Re = 100$: ——, 129 × 129 uniform grid; ○, 64 × 64 uniform grid; +, embedding grid shown in Figure 9(b)

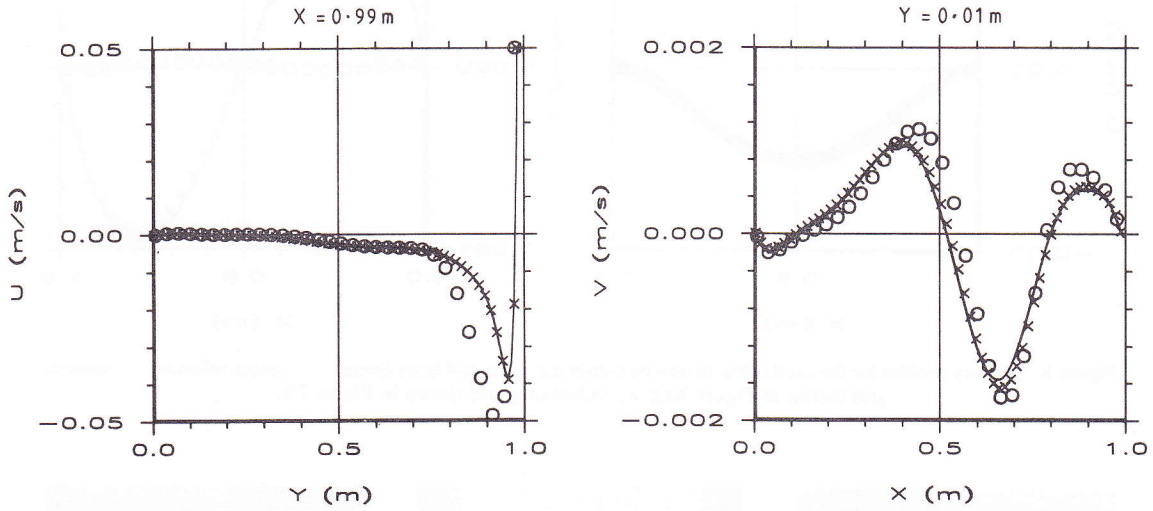


Figure 10(b). Velocity profiles for the lid-driven cavity flow, $Re = 1000$: ——, 129 × 129 uniform grid; ○, 64 × 64 uniform grid; ×, embedding grid shown in Figure 9(c)

to significant errors near the boundaries but the results are accurate elsewhere. Results obtained with this mesh as well as with the embedding mesh shown in Figure 9(c) are compared with the 129 × 129 uniform mesh in Figure 10(b). The embedding grid yields good predictions and the savings in CPU time over the finer uniform mesh exceeded 60%.

3.5. Case 5: sudden plane contraction

The experimental data obtained by Durst *et al.*[36] for laminar two-dimensional flow through a plane duct with a sudden contraction constitute the last test case. The channel flow consists of a

duct 10 mm in height and a contraction ratio of 4:1, yielding 2·5 mm for the duct height downstream of the contraction. Owing to the symmetry of this geometry, computations were only carried out for one half of the duct. Inlet conditions were taken from the experimental data. The Reynolds number based on the inlet mean velocity and on the duct height was equal to 95.

Computations were carried out on uniform grids with 40×24, 80×48, 160×96 and 320×192 grid nodes in the longitudinal and transverse co-ordinate directions respectively. Error estimation based on Richardson extrapolation[22] was carried out and the results are given in Figure 11. They

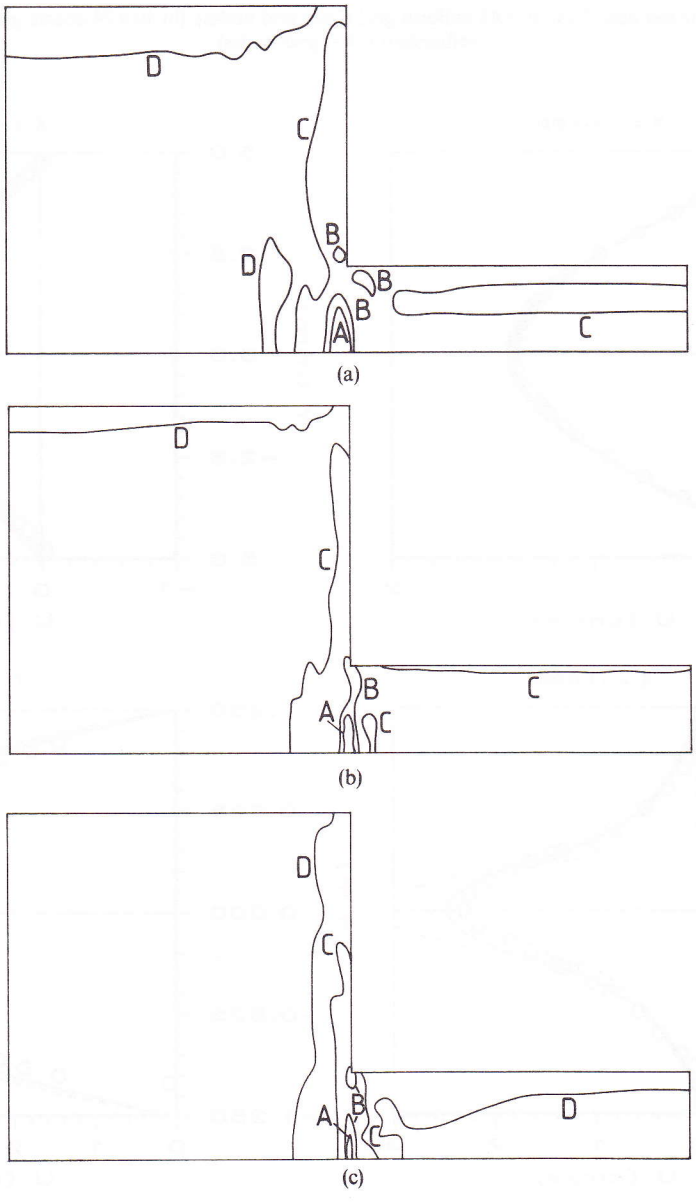


Figure 11. Error estimation for the sudden plane contraction, $|U_h - U_{2h}|/U_{max,inlet}$ (A, 0·2; B, 0·1; C, 0·01; D, 0·001): (a) $2h$, 40×24 uniform grid; h, 80×48 uniform grid; (b) $2h$, 80×48; h, 160×96; (c) $2h$, 160×96; h, 320×192

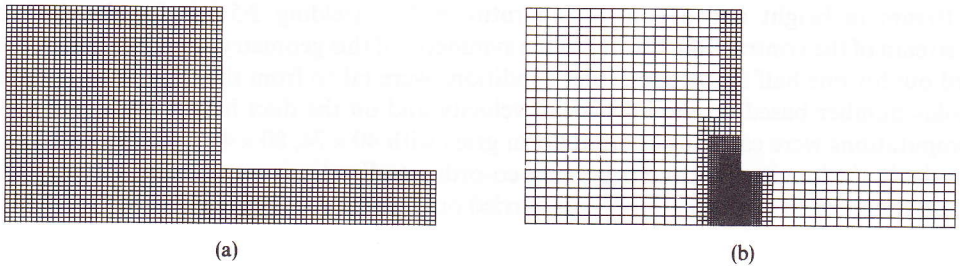


Figure 12. Meshes for test case 5: (a) 80×48 uniform grid (2400 grid nodes); (b) 40×24 coarse grid with three levels of refinement (2430 grid nodes)

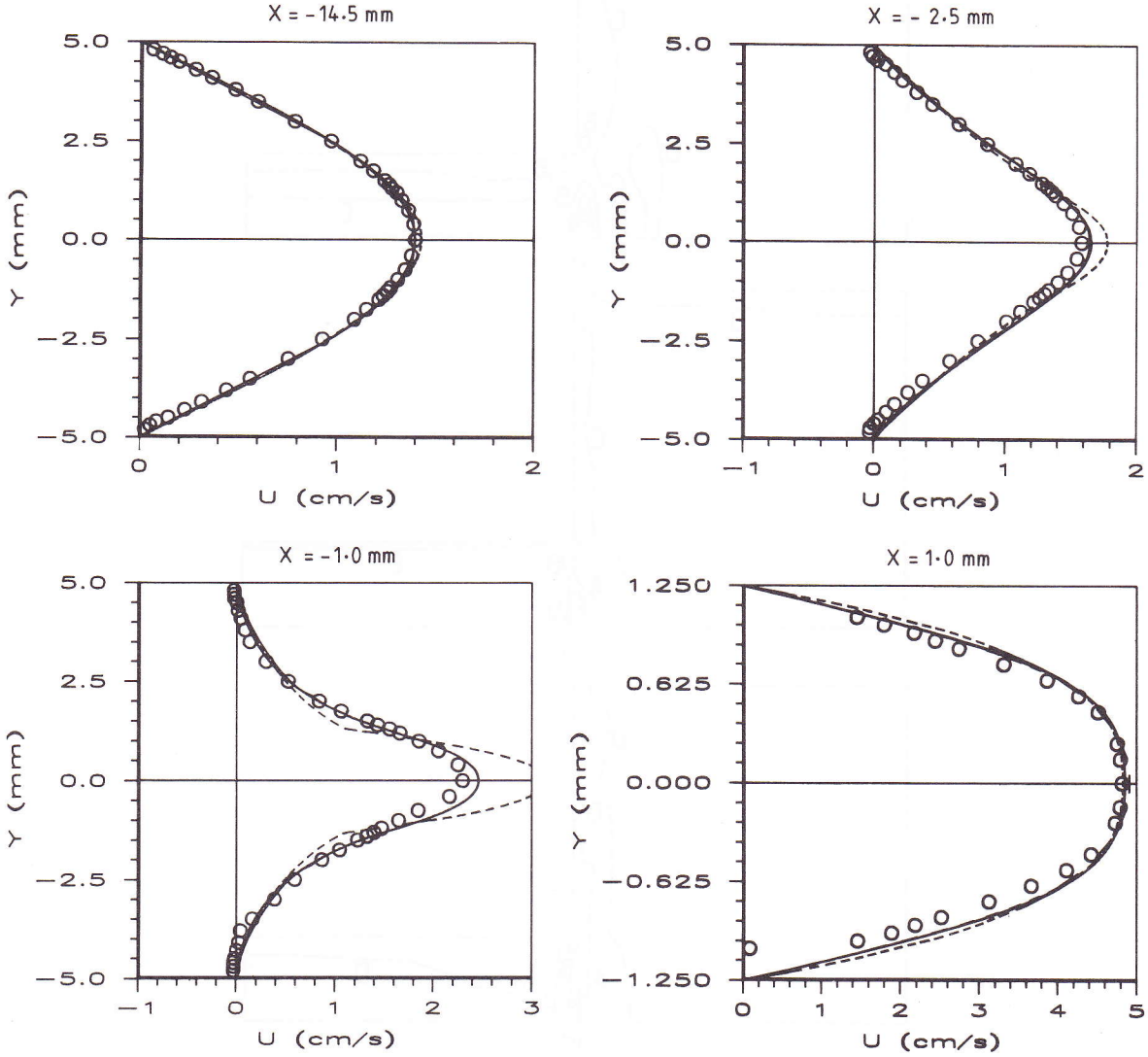


Figure 13. Velocity profiles for a sudden plane contraction: ○, experimental data; ——, embedding grid shown in Figure 12(b); - - - -, 80×48 uniform grid; ...., 320×192 uniform grid

show that larger errors occur near the contraction and so this is the region where embedding grids should be used.

Figure 12 represents the embedding grid used (2430 grid nodes) as well as the 80×48 uniform mesh with a comparable number of grid nodes (2400). Several predicted velocity profiles are shown in Figure 13 along with the available data. The co-ordinates are taken from a frame located at the contraction plane. Solutions obtained for the meshes shown in Figure 12 and for the finer uniform mesh used (320×192) were plotted. However, it was found that the embedding grid yields everywhere results virtually equal to those obtained with the finer uniform mesh and the two solutions are hardly distinguishable. They both compare favourably with the data and the differences are in the range of experimental uncertainty. The velocity profiles at $x = -2{\cdot}5$ and $-1{\cdot}0$ mm confirm that the solution computed with an embedding grid is smooth through interfaces between regions of different levels of refinement. The 80×48 uniform mesh leads to good results except near the contraction, where it departs markedly from the other solutions. The CPU time required to obtain convergence was similar for the meshes with about the same number of grid nodes. However, if the accuracy attained with the embedding grid is to be obtained with a conventional mesh, then savings in CPU time become evident. For example, the solution for the 160×96 uniform grid requires about 14 times more CPU time than the solution obtained using the embedding grid.

4. CONCLUSIONS

A grid-embedding technique for the solution of two-dimensional incompressible flows governed by the Navier–Stokes equations and continuity equation was presented using the finite volume method and a non-staggered grid system. The main features of the method consist of the simultaneous solution of the finite difference equations for the whole computational domain for any degree of local grid refinement. Five different test flow cases were solved and the main conclusions may be summarized as follows.

(i) The grid-embedding technique proves to be adequate to improve the accuracy of the solution of the elliptic form of the flow equations when compared with standard Cartesian meshes using the same number of control volumes.

(ii) The interpolation practices described in the paper for the calculation of control volume fluxes across the interfaces have proved to yield stabilizing effects in the iterative procedure.

(iii) The use of the SIMPLE solution algorithm for pressure–velocity coupling was successfully implemented with the grid-embedding technique. Good agreement was found between the predictions and the analytical solutions or experimental results for the five test flow cases analysed.

(iv) The solutions obtained with the grid-embedding technique yielded a large reduction in computing time compared with standard grids to achieve the same accuracy.

(v) For complex flow geometries where rectangular solid boundaries are immersed in the computational domain, e.g. flow in a sudden plane contraction, the present technique can yield large memory savings.

(vi) The present technique can be improved by the incorporation of an intelligent adaptive embedding grid based on any error estimation analysis.

REFERENCES

1. J. L. Steger, F. C. Dougherty and J. A. Benek, 'A chimera grid scheme', in K. Ghia and U. Ghia (eds), *Advances in Grid Generation, Vol. 5*, FED, ASME, New York, 1983, pp. 59–69.

2. E. A. Atta and J. Vadyak, 'A grid overlapping scheme for flowfield computations about multicomponent configurations', *AIAA J.*, **21**, 1271–1277 (1983).
3. J. A. Benek, J. L. Steger and F. C. Dougherty, 'A flexible grid embedding technique with application to the Euler equations', *AIAA Paper 83-1944*, 1983.
4. E. A. Atta and J. Vadyak, 'A grid interfacing zonal algorithm for three-dimensional transonic flows about aircraft configurations', *AIAA Paper 82-1017*, 1982.
5. M. M. Rai, 'A conservative treatment of zonal boundaries for Euler equation calculations', *J. Comput. Phys.*, **62**, 472–503 (1986).
6. K. A. Hessenius and M. M. Rai, 'Applications of a conservative zonal scheme to transient and geometrically complex problems', *Comput. Fluids*, **14**, 43–58 (1986).
7. J. Flores, 'Convergence acceleration for a three-dimensional Euler/Navier–Stokes zonal approach', *AIAA J.*, **24**, 1441–1442 (1986).
8. Ü. Kaynak and J. Flores, 'Advances in the computation of transonic separated flows over finite wings', *Comput. Fluids*, **17**, 313–332 (1989).
9. J. Flores, N. M. Chaderjian and R. L. Sorenson, 'Simulation of transonic viscous flow over a fighter-like configuration including inlet', *J. Aircraft*, **26**, 295–301 (1989).
10. C. Yung, T. Keith and K. Witt, 'Numerical simulation of axisymmetric turbulent flow in combustors and diffusers', *Int. j. numer. methods fluids*, **9**, 167–183 (1989).
11. F. T. Johnson, S. S. Samant, M. B. Bieterman, R. G. Melvil, D. P. Young, J. E. Bussoletti and M. D. Madson, 'Application of TRANAIR rectangular grid approach to the aerodynamic analysis of complex configurations', *AGARD Specialists' Meeting on Application of Mesh Generation to Complex 3D Configurations*', Loen, Norway, 1989, Paper 21.
12. S.-M. Liang and K.-Y. Fung, 'Refined numerical solution of the transonic flow past a wedge', *AIAA J.*, **25**, 1171–1175 (1987).
13. R. A. Mitcheltree, M. D. Salas and H. A. Hassan, 'Grid embedding technique using cartesian grids for Euler equations', *AIAA J.*, **26**, 754–756 (1988).
14. S. Kotake and T. Kodama, 'Coarse–fine mesh method for heat and mass transfer in locally complex flows', *Int. j. numer. methods eng.*, **25**, 347–355 (1988).
15. M. J. Berger and J. Oliger, 'Adaptive mesh refinement for hyperbolic partial differential equations', *J. Comput. Phys.*, **53**, 484–512 (1984).
16. M. J. Berger and A. Jameson, 'Automatic adaptive grid refinement for the Euler equations', *AIAA J.*, **23**, 561–568 (1985).
17. M. J. Berger and P. Colella, 'Local adaptive mesh refinement for shock hydrodynamics', *J. Comput. Phys.*, **82**, 64–84 (1989).
18. F. Bassi, F. Grasso and M. Savini, 'Numerical solution of compressible Navier–Stokes flows', *AGARD CP-437*, 1988, Paper 8.
19. Y. G. Kallinderis and J. R. Baron, 'Adaptation methods for a new Navier–Stokes algorithm', *AIAA J.*, **27**, 37–43 (1989).
20. J. J. Brown, 'An embedded mesh potential flow analysis', *AIAA J.*, **22**, 174–178 (1984).
21. L. Fuchs, 'A local mesh refinement technique for incompressible flows', *Comput. Fluids*, **14**, 69–81 (1986).
22. M. C. Thompson and J. H. Ferziger, 'An adaptive multigrid technique for the incompressible Navier–Stokes equations', *J. Comput. Phys.*, **82**, 94–121 (1989).
23. M. C. Thompson, 'An adaptive multigrid method for the steady three-dimensional Navier–Stokes equations', in N. W. M. Ko and S. C. Kot (eds.), *The Fourth Asian Congr. of Fluid Mechanics*, Hong Kong, 1989, F140–F143.
24. D. B. Spalding, 'A novel difference formulation for differential expressions involving both first and second derivatives', *Int. j. numer. methods in engineering*, **4**, 551–559 (1972).
25. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, Hemisphere, New York, 1980.
26. C. M. Rhie and W. L. Chow, 'Numerical study of the turbulent flow past an airfoil with trailing edge separation', *AIAA J.*, **21**, 1525–1532 (1983).
27. M. Peric, R. Kessler and G. Scheurer, 'Comparison of finite volume numerical methods with staggered and collocated grids', *Comput. Fluids*, **16**, 389–403 (1988).
28. S. Majumdar, 'Role of underrelaxation in momentum interpolation for calculation of flow with nonstaggered grids', *Numer. Heat Transfer*, **13**, 125–132 (1988).
29. T. F. Miller and F. W. Schmidt, 'Use of a pressure-weighted interpolation method for the solution of the incompressible Navier–Stokes equations on a nonstaggered grid system', *Numer. Heat Transfer*, **14**, 213–233 (1988).
30. M. H. Kobayashi and J. C. F. Pereira, 'Numerical comparison of momentum interpolation methods and pressure–velocity algorithms using non-staggered grids', to be published in *Commun. Appl. Numer. Methods* (1990).
31. R. Fletcher, 'Conjugate gradient methods for indefinite systems', in G. A. Watson (ed.), *Lecture Notes in Mathematics, Vol. 506*, pp. 73–89, Springer-Verlag, Berlin, 1976.
32. Z. Mikic and E. Morse, 'The use of a preconditioned bi-conjugate gradient method for hybrid plasma stability analysis', *J. Comput. Phys.*, **61**, 154–185 (1985).
33. J. J. D. Domingos and J. B. Lopes, 'Numerical stability and false diffusion in recirculating flows', Technical Institute of Lisbon, Internal Report, 16/81 (1981).

34. T. M. Shih, C. H. Han and B. C. Hwang, 'Effects of grid staggering on numerical schemes', *Int. j. numer. methods fluids*, **9,** 193–212 (1989).
35. U. Ghia, K. N. Ghia and C. T. Shin, 'High-*Re* solutions for incompressible flow using the Navier–Stokes equations and a multigrid method', *J. Comput. Phys.*, **48,** 387–411 (1982).
36. F. Durst, W. F. Schierholz and A. M. Wunderlich, 'Experimental and numerical investigations of plane duct flows with sudden contraction', *J. Fluids Eng.*, **109,** 376–383 (1987).