INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 24, 1037–1047 (1997)

A NON-LINEAR ADAPTIVE FULL TRI-TREE MULTIGRID METHOD FOR THE MIXED FINITE ELEMENT FORMULATION OF THE NAVIER–STOKES EQUATIONS

SVEN ØIVIND WILLE[1]

[1]*Faculty of Engineering, Oslo College, Cort Adelersgate 30, N-0254 Oslo, Norway*

SUMMARY

The full adaptive multigrid method is based on the tri-tree grid generator. The solution of the Navier–Stokes equations is first found for a low Reynolds number. The velocity boundary conditions are then increased and the grid is adapted to the scaled solution. The scaled solution is then used as a start vector for the multigrid iterations. During the multigrid iterations the grid is first recoarsed a specified number of grid levels. The solution of the Navier–Stokes equations with the multigrid residual as right-hand side is smoothed in a fixed number of Newton iterations. The linear equation system in the Newton algorithm is solved iteratively by CGSTAB preconditioned by ILU factorization with coupled node fill-in. The full adaptive multigrid algorithm is demonstrated for cavity flow. © 1997 by John Wiley & Sons, Ltd. Int. j. numer. methods fluids 24: 1037–1047, 1997.

(No. of Figures: 5. No. of Tables: 1. No. of Refs: 11.)

KEY WORDS: tri-tree full multigrid; ILU coupled node fill-in preconditioner; element Reynolds number; adaptive refinements; adaptive recoarsement

1. INTRODUCTION

Kallinderis[1] has developed a method for generating adaptive hybrid prismatic/tetradhedral grids where the values of velocity differences and gradients across cell edges are monitored in order to divide/delete grid cells for inviscid transonic flow. The grid generation and method uses an oct-tree as superior tree structure and the grid is constructed from triangles in 2D and tetrahedra and prisms in 3D. The rules for triangle and tetrahedron division are similar to those presented in Wille[2].

Mavriplis and Martinelli[3] have investigated compressible turbulent flow by a multigrid method. They use non-structured, non-nested coarse and fine meshes in obtaining their solutions. The transfer of variables, residuals and corrections back and forth between various meshes is carried out using linear interpolation. The patterns for interpolating between non-nested unstructured meshes are determined in a preprocessing stage using an efficient search algorithm.

Bai *et al.*[4] have applied a multigrid method for predicting periodically fully developed flow. In their multigrid method they use four levels of regular conforming grids. The full multigrid method is implemented in finite volumes and uses a pressure correction scheme as smoother.

In a previous work[2] a triangular tree structure based on the tri-tree has been presented. In the

Contract grant sponsor: The Norwegian Research Council; Contract grant number: NN2461K

CCC 0271–2091/97/101037–11$17.50 *Received December 1995*

author's opinion the triangular tri-tree structure has advantages compared with the oct-tree structure. One advantage is that it is simpler to construct a triangular finite element grid from a triangular tri-tree structure.

In the full adaptive multigrid algorithm the tri-tree elements are refined or recoarsed by inspection of the Reynolds number of each tri-tree element.[5–7] If the Reynolds number of a tri-tree element exceeds a limit, the tri-tree element is refined. If the Reynolds number of tri-tree element is below the same limit and if no tri-tree elements with Reynolds number higher than the limit occur, a recoarsement will take place.

The tri-tree grid generation algorithm is able to generate both grids adapted to the solution and a hierarchy of grids. In the transition between the grids in the hierarchy the restriction of node variables is performed by simply using the function values for common nodes. In the projection of node variables the function values for common nodes are transferred directly and the function values for new nodes are found by linear interpolation.

The smoothing algorithm of the solution between grids consists of a fixed number of Newton iterations, where the update of the non-linear solution is found by solving the linearized equation system by preconditioned CGSTAB.[8]

2. EQUATIONS

The non-linear Navier–Stokes equations are given by

$$-\mu\nabla^2\mathbf{v} + \rho\mathbf{v}\cdot\nabla\mathbf{v} + \nabla p = 0 \quad \text{in } \Omega, \tag{1}$$

$$-\nabla\cdot\mathbf{v} = 0 \quad \text{in } \Omega, \tag{2}$$

where $\mathbf{v}$ is the velocity vector, p is the pressure and μ is the viscosity coefficient. The first equation is the equation of motion which contains a diffusion and a pressure gradient term. The second equation is the equation of continuity. A minus sign is introduced in the continuity equation in order to obtain the same sign for the pressure gradient as for the continuity equation in the finite element formulation. In the finite element formulation the velocities are approximated by quadratic basis functions and the pressure is approximated by linear basis functions on each element.[9] Denote quadratic polynomials N_i and linear polynomials L_i. Then by the Galerkin residual method and integration by parts the second-order finite element formulation of the Navier–Stokes equation system becomes

$$\begin{aligned} &\mathbf{F}_{\mathbf{v}} = \int_{\Omega} \mu\nabla N_i\cdot\nabla\mathbf{v}\mathrm{d}\Omega + \int_{\Omega} \rho N_i\mathbf{v}\cdot\nabla\mathbf{v}\mathrm{d}\Omega - \int_{\Omega} \nabla N_i\, p\mathrm{d}\Omega - \int_{\delta\Omega} \mu N_i\frac{\partial\mathbf{v}}{\partial n}\mathrm{d}\delta\Omega + \int_{\delta\Omega} N_i\, p\mathrm{d}\Omega = 0,\\ 6\\ &\mathbf{F}_p = -\int_{\Omega} L_i\nabla\cdot\mathbf{v}\mathrm{d}\Omega = 0. \end{aligned} \tag{3}$$

There are several methods to linearize this equation system. Usual linearization techniques involve the computation of gradients or approximate gradients as in the Newton method or steepest decent methods. The Newton linearization method is a global method of linearization.

3. NEWTON LINEARIZATION

The Navier–Stokes equations have one non-linear term, the convective acceleration, which requires a non-linear iterative solution procedure. The non-linear algorithm chosen as the Newton method, which is known to have a second-order convergence rate. The Navier–Stokes equations (3) will then have to be differentiated with respect to the unknowns and the linear equation system which has to be solved at each Newton step is

$$\begin{bmatrix} \partial \mathbf{F}_{\mathbf{v}}^n/\partial \mathbf{v} & \partial \mathbf{F}_{\mathbf{v}}^n/\partial p \\ \partial \mathbf{F}^n/\partial \mathbf{v} & 0 \end{bmatrix} \begin{bmatrix} \Delta \mathbf{v} \\ \Delta \mathbf{p} \end{bmatrix} = -\begin{bmatrix} \mathbf{F}_{\mathbf{v}}^n \\ \mathbf{F}_{\mathbf{p}}^n \end{bmatrix}, \tag{4}$$

where the matrix and the right-hand side are given by

$$\begin{bmatrix} \int_\Omega [\mu \nabla N_i \nabla N_j + \rho N_i (\nabla \mathbf{v} N_j + \mathbf{v} \nabla N_j)]\,\mathrm{d}\Omega & -\int_\Omega \nabla N_i L_j \mathrm{d}\Omega \\ -\int_\Omega L_i \nabla N_j \mathrm{d}\Omega & 0 \end{bmatrix} \begin{bmatrix} \Delta \mathbf{v} \\ \Delta \mathbf{p} \end{bmatrix}$$

$$= -\begin{bmatrix} \int_\Omega (\mu \nabla N_i \cdot \nabla \mathbf{v} + \rho N_i \mathbf{v} \cdot \nabla \mathbf{v} - \nabla N_i p)\,\mathrm{d}\Omega + \int_{\delta\Omega} (-\mu N_i \partial \mathbf{v}/\partial n + N_i p)\,\mathrm{d}\delta\Omega \\ -\int_\Omega L_i \nabla \cdot \mathbf{v} \mathrm{d}\Omega \end{bmatrix}, \tag{5}$$

$$\mathbf{v}^{n+1} = \mathbf{v}^n + \Delta \mathbf{v}, \tag{6}$$

$$\mathbf{p}^{n+1} = \mathbf{p}^n + \Delta \mathbf{p}. \tag{7}$$

If the initial solution $\mathbf{v}^0$ and p^0 is chosen close enough to the final solution, convergence of the non-linear equation system is guaranteed. The solution is then updated at each Newton step by the correction found by solving (5).

4. ADAPTIVE LINEARIZATION

An alternative or supplement to Newton linearization of the equation system is local grid adaptation to the solution, which will also contribute to the linearization of the equation system. From analytic integrations[5] the following formula is obtained:

$$\frac{\int_\Omega \rho N_i \mathbf{v} \cdot \nabla \mathbf{v} \mathrm{d}\Omega}{\int_\Omega \mu \nabla N_i \cdot \nabla \mathbf{v} \mathrm{d}\Omega} = \frac{a(\mathbf{v}) l}{b(\mathbf{v})}. \tag{8}$$

In the above formula, $a(\mathbf{v})$ and $b(\mathbf{v})$ are functions of the velocities inside the element only and are independent of the element size. The length l is some characteristic length of the element. The formulae show that the magnitude of the matrix coefficient of the convection can be reduced arbitrarily by local refinements compared with the diffusion coefficient in the implicit equation system. The above relation is valid in both two and three dimensions and for first- and second-order polynomial approximations of the Navier–Stokes equations. By reducing the element size where the convection is large, the equation system becomes more and more linear and symmetric. Provided that the local element size is reduced sufficiently, this implicit adaptive linearization will for many Navier–Stokes applications appear to be sufficient and satisfactory.

5. MULTIGRID ALGORITHM

The Navier–Stokes equations (5) can be expressed as an equation system $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, where

$$\mathbf{f} = \begin{bmatrix} \mathbf{f}_\mathbf{v} \\ \mathbf{f}_\mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathscr{A}(\mathbf{F}_\mathbf{v}) \\ \mathscr{A}(\mathbf{F}_\mathbf{p}) \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} \mathbf{x}_\mathbf{v} \\ \mathbf{x}_\mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathscr{A}(\mathbf{v}) \\ \mathscr{A}(\mathbf{p}) \end{bmatrix}. \tag{9}$$

The function $\mathscr{A}$ denotes the assembly of element vectors into a global vector. $\mathbf{f}_\mathbf{v}$ and $\mathbf{f}_\mathbf{p}$ are velocity and pressure equations assembled from the element velocity equation vector $\mathbf{F}_\mathbf{v}$ and pressure equation vector $\mathbf{F}_\mathbf{p}$. The global velocity, $\mathbf{x}_\mathbf{v}$, and pressure, $\mathbf{x}_\mathbf{p}$, vectors are assembled from $\mathbf{v}$ and $\mathbf{p}$ respectively.

With the notation defined above, the non-linear Navier–Stokes multigrid algorithm[10] is defined as follows:

```
void Multigrid()
{
  $\mathbf{b}_N = \mathbf{0}$, choose $\bar{\mathbf{x}}_N$
  $Smooth(\mathbf{x}_N, \bar{\mathbf{x}}_N, \mathbf{f}_N, \mathbf{b}_N)$
  for $(k = N; k \geqslant 1; k--)$
  {
    $\mathbf{x}_{k-1} = \bar{\mathbf{x}}_{k-1} = \mathbf{R}_{k-1}\mathbf{x}_k$
    $\mathbf{b}_{k-1} = \mathbf{f}_{k-1}(\bar{\mathbf{x}}_{k-1}) + \mathbf{R}_{k-1}(\mathbf{b}_k - \mathbf{f}_k(\mathbf{x}_k))$
    $Smooth(\mathbf{x}_{k-1}, \bar{\mathbf{x}}_{k-1}, \mathbf{f}_{k-1}, \mathbf{b}_{k-1})$
  }
  for $(k = 1; k \leqslant N; k++)$
  {
    $\mathbf{x}_k = \mathbf{x}_k + \mathbf{P}_k(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1})$
    $Smooth(\mathbf{x}_k, \bar{\mathbf{x}}_k, \mathbf{f}_k, \mathbf{b}_k)$
  }
}

void Smooth($x, \bar{x}, \mathbf{f}, \mathbf{b}$)
{
  $\mathbf{x}_1 = \bar{\mathbf{x}}$
  for $(n = 1; n \leqslant P; n++)$
  {
    $\nabla\mathbf{f}(\mathbf{x}_n)\boldsymbol{\Delta}\mathbf{x}_n = -(\mathbf{f}(\mathbf{x}_n) - \mathbf{b})$
    $\mathbf{x}_{n+1} = \mathbf{x}_n + \boldsymbol{\Delta}\mathbf{x}_n$
  }
  $\mathbf{x} = \mathbf{x}_{P+1}$
}
```

The function $\mathbf{R}_k$ denotes the restriction of grid variables from a finer to a coarser grid. The restriction function $\mathbf{R}_k$ is a filter which uses the function values in each node in the fine grid for the function values of the nodes present in the coarse grid. The function $\mathbf{P}_k$ is the projection of function values from the coarse grid to the fine grid. The function values for nodes in the fine grid are the same as in the coarse grid for common nodes. For nodes present in the fine grid but not in the coarse grid, linear interpolation is applied to obtain function values for these nodes. The new nodes in the fine grid are always located at the midpoint of a side in a tri-tree element. The function value at the midpoint is therefore simply the average value for the endpoints of the side.

The smoothing algorithm *Smooth*$(\mathbf{x}, \bar{\mathbf{x}}, \mathbf{f}, \mathbf{b})$ returns a solution vector $\mathbf{x}$ which is closer to the solution of the equation system $\mathbf{f}(\mathbf{x}) = \mathbf{b}$ than is the start vector $\bar{\mathbf{x}}$. The smoothing algorithm consists of a fixed number of Newton iterations by solving the linear equation system $\nabla\mathbf{f}(\mathbf{x}_n)\Delta\mathbf{x}_n = -(\mathbf{f}(\mathbf{x}_n) - \mathbf{b})$. This equation system is solved iteratively by the iterative equation solver CGSTAB preconditioned by coupled node fill-in ILU factorization.[8]

The most time-consuming parts of the multigrid algorithm are the computations of the vector $\mathbf{f}(\mathbf{x})$ and the matrix $\nabla\mathbf{f}(\mathbf{x}_n)$.

6. GRID ADAPTION

The Reynolds number for fluid flow is usually defined as

$$Re = \rho U_{\mathrm{b}} d/\mu, \tag{10}$$

where ρ is the density and μ the viscosity of the fluid. The velocity U_{b} and the length d are some characteristic velocity and diameter in the flow geometry. For flow in a straight tube, U_{b} is the mean inlet velocity and d is the diameter. For more complex geometries it is not possible to use a single number to characterize the flow conditions. The element Reynolds number is defined as

$$Re_{\mathrm{e}} - \frac{\sum_i N_i^{\mathrm{c}} \left\| \rho \int_\Omega (N_i \mathbf{v} \cdot \nabla\mathbf{v})\,\mathrm{d}\Omega \right\|}{\sum_i N_i^{\mathrm{c}} \left\| \mu \int_\Omega (\nabla N_i \cdot \nabla\mathbf{v})\,\mathrm{d}\Omega \right\|} < \epsilon_{\mathrm{A}}, \tag{11}$$

where N_i^{c} is the basis function evaluated at the geometrical centre. The refinements and recoarsements of elements are decided on the basis of the element Reynolds number, which is calculated from the nodal values and weighted with the basis function evaluated at the geometrical centre. The size of the element Reynolds number Re_{e} indicates the degree of non-linearity in the equation system. By reducing the element size by refinements, the magnitude of the non-linear coefficients in the equation matrix will also decrease.

$$Re_{\mathrm{e}} > \epsilon_{\mathrm{A}}. \tag{12}$$

Before the finite element grid is adapted to the previous solution of lower Reynolds number, the solution at all nodes is scaled by $U_{\mathrm{b}}^{\mathrm{n}}/U_{\mathrm{b}}^{\mathrm{o}}$, the ratio between the new and the old velocity boundary condition. The scaled solution is then projected from the finite element grid to the tri-tree grid. The element Reynolds number is computed for the tri-tree elements. First the tri-tree elements are recoarsed. During recoarsing, the element Reynolds number of the tri-tree element above the terminal element in the tri-tree is computed. If the element Reynolds number Re_{e} of this tri-tree element is below the adaption limited ϵ_{A}, this tri-tree element is made terminal and the four leaf elements at the finer adaption levels are discarded. The recoarsement algorithm starts from the termina leaves of the tri-tree and transverses towards the root of the tri-tree. Owing to the tri-tree hierarchic tree structure, the recoarsement algorithm becomes recursive. At the end of the recoarsement procedure of the tri-tree contains elements with element Reynolds numbers just below the adaption limit ϵ_{A} and elements with element Reynolds numbers above the adaption limit which have not been recoarsed. At this stage no elements in the tri-tree can be recoarsed without introducing a new tri-tree element with element Reynolds number above the adaption limit. When the recoarsement algorithm is finished, the refinements are performed. The tri-tree is then traversed from the root towards the leaves. The refinements will then be recursive. When the element Reynolds number is above the refinement limit, the tri-tree element is refined into four new tri-tree elements. The element Reynolds number of these four elements will be computed later in the refinement algorithm and they will be refined later if their element Reynolds number is above the refinement limit ϵ_A.

7. MULTIGRID GENERATION

The generation of grids at different multigrid levels begin with the grid which is adapted to the scaled solution. The recoarsements of the grid are based on the tri-tree grid. The tri-tree elements are traversed from the root towards the terminal elements of the tri-tree. For each terminal tri-tree element the parent element is investigated. When the parent tri-tree element has children which are terminals and at the same level of refinement, the child tri-tree elements are discarded and the parent tri-tree element is made a terminal tri-tree element. During the recoarsement procedure, only those tri-tree elements which have the same level of refinement and belong to the same parent tri-tree element are allowed to be recoarsed. There is only one recoarsement per grid level so that a parent tri-tree element will be the terminal element for the multigrid level.

Figure 1 shows three multigrid levels. The tri-tree elements are shown to the left and the corresponding finite elements are shown to the right. The recoarsements are first applied to the tri-tree grid which forms the basis for the finite element grid. The initial tri-tree and finite element grids are shown at the top of Figure 1. The tri-tree elements are then recoarsed once and the resulting tri-tree and finite element grids are shown in the middle of Figure 1. The tri-tree and finite element grids after another recoarsement are shown at the bottom of the figure. The finite element grids generated in this way are used for both the fine-to-coarse and the coarse-to-fine transition. The grids involved in the computations are generated when they are needed in the solution algorithm. The CPU time for generating the grids is approximately 10% of the CPU time for solving the equations.[11] The most time-consuming part of the grid adaption algorithm is the computation of the element Reynolds number.

The multigrid solution algorithm consists of several iterative operations before a final solution is obtained. At present the most efficient way seems to be as follows:

```
Solve equation system for low velocities on a coarse grid
Repeat
  Increase the boundary velocity and scale the solution correspondingly
  Compute the $Re_e$ for each element using the scaled solution
  Recoarse the grid while $Re_e < \epsilon_A$ for all elements
  Refine the grid until $Re_e < \epsilon_A$ for all elements
  Project the coarse solution to the fine grid by linear interpolation
  Repeat
    Smooth the solution using the interpolated solution as start vector
    Recoarse the grid
  Until Coarse grid level
  Repeat
    Refine the grid
    Smooth the solution using the interpolated solution as start vector
  Until Fine grid level
Until convergence at selected Reynolds number
```

8. CONVERGENCE CRITERIA

The tri-tree multigrid solver consists of three iterative algorithms inside each other. The inner iterative algorithm is CGSTAB, the linear equation solver. For each Newton iteration a set of linear

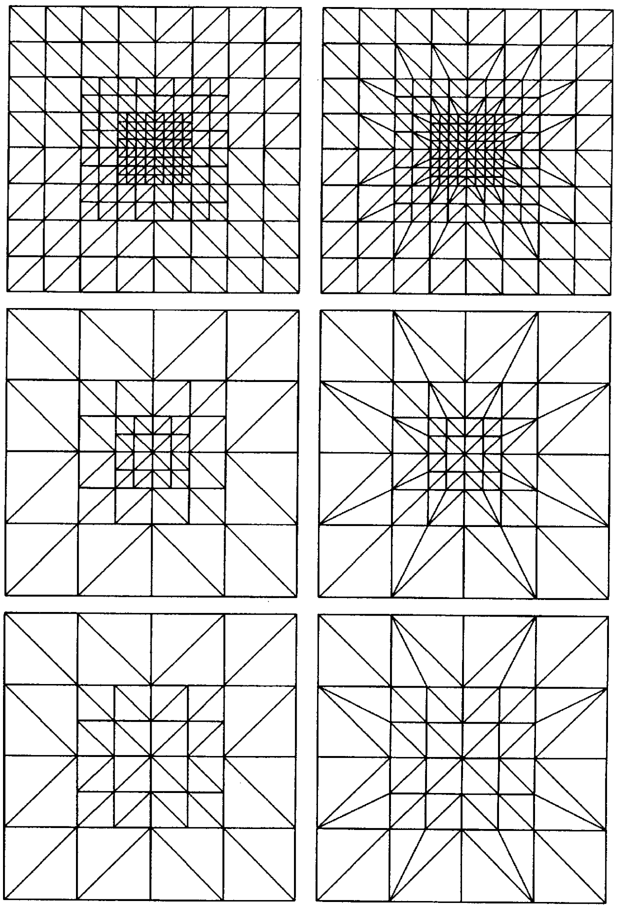

Figure 1. Tri-tree elements (left) and finite elements (right) for arbitrary multigrid cycle. The initial tri-tree and finite elements are shown at the top. The first level of recoarsements is shown in the middle and the second level of recoarsements is shown at the bottom

equations is solved. The iterative Newton algorithm is performed for each grid level, which is the outer level of iterations. The three convergence criteria used are

$$\frac{\|\delta^{\mathrm{L}}\mathbf{x}_{\mathbf{v}}^{k}\|}{U_{\mathrm{b}}} < \epsilon_{\mathrm{L}} \qquad \frac{\|\delta^{\mathrm{N}}\mathbf{x}_{\mathbf{v}}^{k}\|}{U_{\mathrm{b}}} < \epsilon_{\mathrm{N}}, \qquad \frac{\|\delta^{\mathrm{G}}\mathbf{x}_{\mathbf{v}}^{k}\|}{U_{\mathrm{b}}} < \epsilon_{\mathrm{G}}, \tag{13}$$

where U_{b} is the velocity boundary condition, $\delta^{\mathrm{L}}\mathbf{x}_{\mathbf{v}}$ is the update of the velocity solution in the linear equation solver, $\delta^{\mathrm{N}}\mathbf{x}_{\mathbf{v}}$ is the velocity update of the solution in the Newton iterations and $\delta^{\mathrm{G}}\mathbf{x}_{\mathbf{v}}^{\mathrm{k}}$ is the velocity difference between the projected start vector and the velocity solution at grid level k. ϵ_{L} is the linear, ϵ_{N} is the non-linear and ϵ_{G} is the grid convergence criterion.

9. NUMERICAL EXPERIMENT

The adaptive full multigrid method is tested for driven cavity flow.[7] The solution is found for the Reynolds numbers 200, 400, 600, 800, 1000 and 1200. The multigrid path and the adaptive Newton path are shown in Figure 2. In the multigrid cycles, three grids are applied. The Reynolds number is

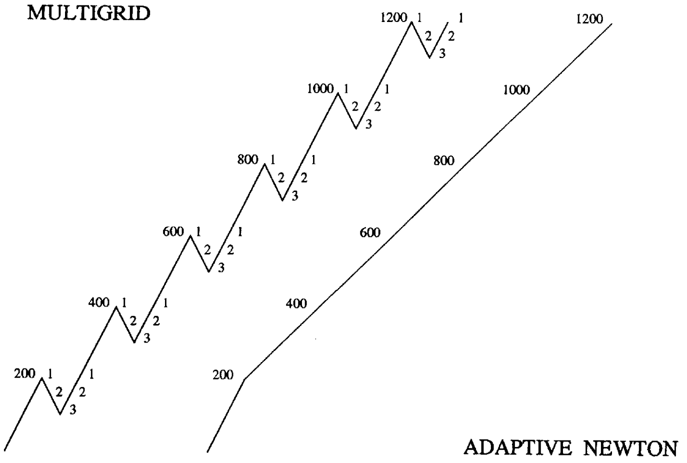

Figure 2. Multigrid path (left) and adaptive Newton path (right). Each multigrid cycle consists of three grids

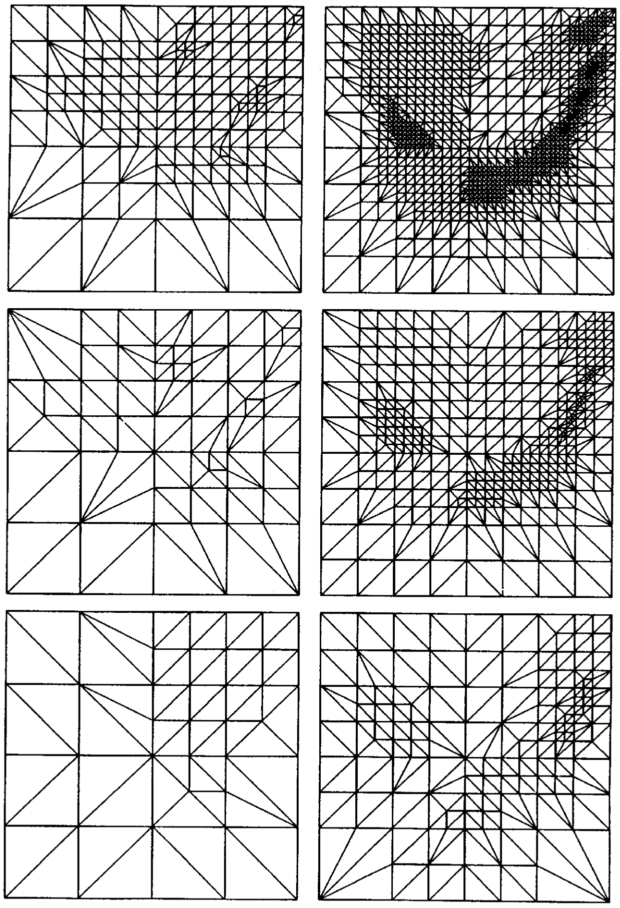

Figure 3. Three multigrid levels for Reynolds numbers 200 (left) and 400 (right). The initial level in each multigrid cycle is obtained by adaption of the grid to the solution

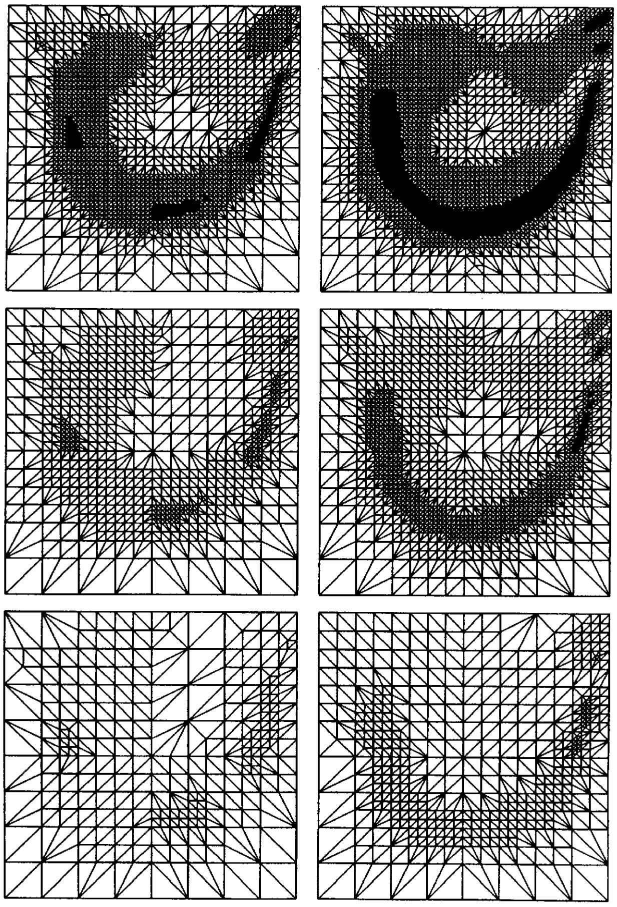

Figure 4. Three multigrid levels for Reynolds numbers 600 (left) and 800 (right). The initial level in each multigrid cycle is obtained by adaption of the grid to the solution

increased by scaling the solution with the relative increase in boundary velocity condition. The grid is then adapted to the scaled solution. The adaption limit ϵ_A is 1·0. The adaption limit ensures that all element Reynolds numbers initially are less than 1·0. The scaled solution is used as start vector in the multigrid algorithm. The linear convergence criterion is $\epsilon_L = 0{\cdot}001$ and the number of Newton iterations is fixed at two in the smoothing algorithm at each grid level. The multigrid cycles consists of three multigrid levels at each Reynolds number as shown in Figures 3–5. In the adaptive Newton algorithm the grids at the top of these figures are applied for the corresponding Reynolds number.

The multigrid algorithm is compared with the Newton method with five non-linear iterations. The Newton method is applied to solve the equation system for each Reynolds number using the scaled solution from the lower Reynolds number as start vector. The results of these comparisons are shown in Table I. The table shows that for all the Reynolds numbers investigated, the adaptive Newton method with five iterations is both faster and achieves greater accuracy than the multigrid method.

10. DISCUSSION

In the present work a new adaptive full multigrid finite element method has been developed. For a three-level multigrid cycle the multigrid method has been tested and compared with solving the

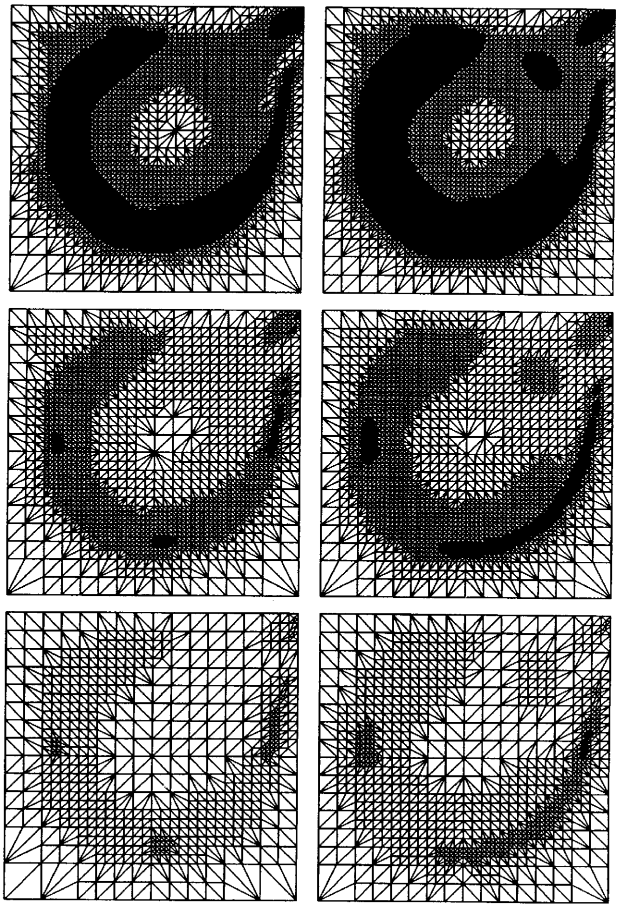

Figure 5. Three multigrid levels for Reynolds numbers 1000 (left) and 1200 (right). The initial level in each multigrid cycle is obtained by adaption of the grid to the solution

Table I. Reynolds numbers, CPU times and relative errors for adaptive Newton method and multigrid method. The times are given in seconds and include both grid generation and computations of the solution of the equation system

	Adaptive Newton		Multigrid	
Re	Time	Error ϵ_M	Time	Error ϵ_N
200	33.72	0.0024	48.46	0.0506
400	223.14	0.0028	307.81	0.0574
600	516.27	0.0124	753.83	0.0710
800	878.62	0.0083	1312.89	0.0597
1000	1494.82	0.0072	2148.64	0.0517
1200	2139.57	0.0476	3034.14	0.0289

Navier–Stokes equations with the adaptive Newton method. The results of these tests are that with the present smoothing and multigrid algorithms the adaptive Newton method is faster than the multigrid method. The multigrid algorithm is applied to the non-linear equation system. There are several reasons for the relatively slow convergence rate of the multigrid algorithm. One is that the start vector of the multigrid cycle is not close enough to the final solution. Another reason might be the high efficiency of the ILU preconditioner with coupled node fill-in.

By inspection of the numerical part of the adaptive multigrid algorithm, the function $\mathbf{f}(\mathbf{x})$ is evaluated twice in going from fine to coarse grid and once in going from coarse to fine grid. Inside the smoothing algorithm the vector $\mathbf{f}(\mathbf{x})$ and the matrix $\nabla\mathbf{f}(\mathbf{x})$ are evaluated once for each adaptive Newton iteration. The computationals of these non-linear vectors and matrices are expensive owing to the large computational time. However, the tri-tree grid generation algorithm reveals advantageous properties both concerning the CPU time and in providing adequate finite element grids.[11] Thus less expensive smoothing and more efficient numerical multigrid transition algorithms will be sought.

ACKNOWLEDGEMENTS

The author is grateful to Demosthenes Skipitaris for skilled computer advice to Trygve Svoldal for valuable discussions of the manuscript. The project has been supported by The Norwegian Research Council, grant NN2461K, which has financed part of the computer run time expenses.

REFERENCES

1. Y. Kallinderis, ‘Adaptive hybrid prismatic/tetrahedral grids’, *Int. j. numer. methods fluids*, **20**, 1023–1037 (1995).
2. S. Ø. Wille, ‘A structured tri-tree search method for generation of optimal unstructured finite element grids in two and three dimensions’, *Int. j. numer. methods fluids*, **14**, 861–881 (1992).
3. D. J. Mavriplis and L. Martinelli, ‘A Multigrid solution of compressible turbulent flow on unstructured meshes using a two-equation model’, *Int. j. numer. methods fluids*, **18**, 887–914 (1994).
4. L. Bai, N. K. Mitra, M. Fiebig and A. Kost, ‘A multigrid method for predicing periodically fully developed flow’, *Int. j. numer. methods fluids*, **18**, 843–852 (1994).
5. S. Ø. Wille, ‘A local solution adapted tri-tree multigrid generator and equation solver for mixed fintie element formulation of the Navier–Stokes equations’, *Comput. Methods Appl. Mech. Eng.*, **131**, 109–139, (1996).
6. S. Ø. Wille, ‘A local predictive convenction–diffusion refinement indicator for the tri-tree adapted finite element multigrid algorithm of the Navier–Stokes equations’, *Comput. Methods Appl. Mech. Eng.*, **134**, 181–196, (1996).
7. S. Ø. Wille, ‘A tri-tree recoarsement algorithm for the finite element formulation of the Navier–Stokes equations’, *Comput. Methods Appl. Mech. Eng.*, **135**, 129–142 (1996).
8. O. Dahl and S.Ø. Wille, ‘An ILU preconditioner with coupled node fill-in for iterative solution of the mixed finite element formulation of the 2-D and 3-D Navier–Stokes equations’, *Int. j. numer. methods fluids*, **15**, 525–544, (1992).
9. C. Taylor and P. Hood, ‘A numerical solution of the Navier–Stokes using the finite element technique’, *Comut. Fluids*, **1**, 73–100 (1973).
10. P. Wesseling, *An Introduction to Multigrid Methods*, Wiley, New York, 1992.
11. S. Ø. Wille, ‘Adaptive linearization and grid iterations with the tri–tree multigrid refinement–Recoarsement algorithm for the Navier–Stokes equations’, *Int. j. numer. methods fluids*, **24**, 155–168 (1997).