An unstructured Finite Element Method with a Full MultiGrid SIMPLE solver for turbulent flow

Peter Emvin[1] and Lars Davidson
Thermo and Fluid Dynamics
Chalmers University of Technology
S-412 96 Gothenburg, Sweden

ABSTRACT

An implementation of a SUPG (Streamline Upwind Petrov Galerkin) finite element method for unstructured triangular meshes is presented. The solution process is the Full Multigrid algorithm with a SIMPLE smoother. The SIMPLE smoother is specially designed to handle multiple free outlets with maintained performance. The code automatically refines the mesh on the basis of on the finite element residual, and from this process we extract a mesh hierarchy which is used in the multigrid procedure. We investigated the behaviour of this SIMPLE solver with and without multigrid for the $P1/P1$ and $P2/P2$ Lagrangian triangular elements.

We also compare the performance of this method to our previous work based on finite volumes on structured grids. The third order unstructured FEM (Finite Element Method), was found to be superior to the second order FEM which showed to be as accurate as a structured second order FVM (Finite Volume Method). Using the third order FEM we present benchmark solutions for the lid-driven cavity for Re=100, 400, 1000, 3200 and 5000. We also implemented a $k-\omega$ turbulence model to simulate high Reynolds number flows and applied this to backwards-facing step flows.

For all applications presented, the FMG-SIMPLE method showed grid independent convergence within CPU time comparable with $100-200$ iterations on the finest grid (i.e. 100-200 WU).

[1] Peter Emvin changed his name from Peter Johansson in September 1995

Contents

1 Introduction

For an engineer there exists a large number of applications based on fluid motion. The basic equations describing fluid flow are complex but form a closed set. The complexity requires numerical methods to simulate the flow but, the computer capacity of today usually forces us to model the turbulence and hence solve a stationary problem instead.

When constructing numerical methods for fluid flow with or without turbulence models, there are great differences in scales in the domain. This requires stabilised methods, as resolving all scales is prohibitively expensive. There has been a vast amount of research into stabilised methods based on several different techniques.

In finite volumes, the stability is often introduced into a central difference discretization via the addition of second or fourth derivatives discretized by undivided central differences or special reconstructions. For applications on structured grids see [1, 2, 3, 4] and for unstructured grids see, [5, 6, 7, 8, 9, 10].

The leading error is third order for undivided fourth derivatives and thus is the second order accuracy usually preserved when adding such dissipation. However, adding undivided second derivatives means that the method degenerates to first order, and this artificial term normally contaminates the whole solution. Such a method is the Upwind or the HYBRID/Upwind method and is not often used today when simulating the Navier-Stokes equations although a special case is compressible flow, where a first order viscosity term often is added in the vicinity of shocks. However, since the spatial extent of a shock is small compared to the whole domain, it does not contaminate the solution significantly.

Another problem when using stencils with a large support is that special treatment of the boundary is required. While the stability problem of the dissipation stencil in unstructured finite volumes methods is partially solved, the implementational problems near the boundaries still remain [6, 11]. When using higher order methods, these problem increase for the finite volume technique [7, 8, 9].

In finite elements, a proof of a stabilised finite element method with equal order linear elements, was presented in [12], and it has been generalised to higher order elements in [13]. These proofs show the slightly non-optimal convergence $O(h^{p+0.5})$, where p denotes the polynomial order even though all of our calculations indicate the optimal $O(h^{p+1.0})$ behaviour. Since this slight non-optimality most often not is visible [14] we will in the rest of the paper consider these methods to be optimal.

To raise the order of the method in FEM, additional nodes are simply added to each element. Because, it is common when implementing finite elements to evaluate the integrals in each element and distribute the contributions to the nodes, the presence of boundaries does not add any extra complexity when raising the order of the method. FEM also manages curved boundaries if the geometry is mapped with a polynomial of an order of two or more.

Having discretized the PDE, we need a solution algorithm. The full multigrid

(FMG) has been shown to be very efficient [15]. We earlier used the multigrid method for a structured finite volume method (FVM) [3, 16, 17, 18]. In that multigrid algorithm, we used the SIMPLE smoother, which turned out to have nice smoothing properties and was also found to be robust. We thus applied that FMG-SIMPLE concept to our unstructured FEM.

Simulating turbulent flows we require a turbulence model and today are eddy viscosity models (especially $k-\epsilon$) routinely used in industry. They are often used in combination with wall functions, where the boundary layer is replaced by a log-law approximation. Such a model is quite robust, reliable and rather cheap to use. One way to improve the accuracy of such an eddy viscosity model is to resolve the boundary layers. The $k-\epsilon$ model, unlike the $k-\omega$ model is not consistent in the boundary layer and thus must be modified with a low-Reynolds method. When resolving the boundary layer, the size of the problem is increased by a factor of two or more and, usually, the code also becomes more unstable as a result of the model and/or the required stretching of the grid.

We have in previous work with structured finite volumes used the $k-\epsilon$ model with a two-layer wall model [19]. However, this requires the normal distance from the wall, as most low-Reynolds models do. These wall distances are easily evaluated for structured grids while they can be very expensive to evaluate for an unstructured grid. We have therefore avoided methods relying on wall distances.

We decided to have the higher accuracy of resolved boundary layers and, encouraged by the work of [20, 21, 22], we chose the $k-\omega$ model (without wall functions).

2 Basic Equations

The equations of motion with the $k-\omega$ model read:

$$R_d(U_i) := \rho \partial_i U_i = 0 \tag{1}$$

$$R_u(U_i, P) := Q(U_i, \mu + \mu_t) + \partial_i P = 0 \tag{2}$$

$$R_k(k, \mu + \sigma_k \mu_t) := Q(k, \mu + \sigma_k \mu_t) - PD(1, \beta^* \omega k) = 0 \tag{3}$$

$$R_\omega(\omega, \mu + \sigma_\omega \mu_t) := Q(\omega, \mu + \sigma_\omega \mu_t) - PD(\gamma \omega / k, \beta \omega^2) = 0 \tag{4}$$

$$\mu_t := \rho k / \omega \tag{5}$$

where

$$Q(u, \mu) = \rho U_j \partial_j u - \partial_j (\mu \partial_j u)$$

$$PD(\gamma, \beta\omega^2) = \gamma \mu_t \partial_i U_j (\partial_i U_j + \partial_j U_i) - \rho \beta \omega^2$$

and

$$\beta = 3/40, \quad \beta^* = 0.09, \quad \gamma = 5/9, \quad \sigma_\omega = \sigma_k = 0.5$$

3 Variational form

A variational form, in the sense of SUPG, of the presented set of differential equations reads: Find $[U, V, P, k, \omega] = [U_i, P, k, \omega] \in V_h$ such that

$$\underbrace{\{R_d(U_i), q\}}_{1a} + \underbrace{QV(U_i, \mu + \mu_t, w_i) - \{P\delta_{ij}, \partial_j w_i\}}_{2a} +$$

$$\underbrace{QV(k, \mu + \sigma_k \mu_t + \mu^*(k), t) - \{PD(1, \beta^* \omega k), t\}}_{3a} +$$

$$\underbrace{QV(\omega, \mu + \sigma_\omega \mu_t + \mu^*(\omega), d) - \{PD(\omega\gamma\ /k, \beta\omega^2), d\}}_{4a} +$$

$$\underbrace{\sum_{NEL} [\delta_2 R_d(U_i), R_d(w_i)]}_{1b} + \underbrace{\sum_{NEL} [\delta_1 R_u(U_i, P), R_u(w_i, q)]}_{2b} +$$

$$\underbrace{\sum_{NEL} [\delta_1 R_k(k, \mu + \sigma_k \mu_t + \mu^*(k)), Q(k, \mu + \sigma_k \mu_t + \mu^*(k))]}_{3b} +$$

$$\underbrace{\sum_{NEL} [\delta_1 R_\omega(\omega, \mu + \sigma_\omega \mu_t + \mu^*(\omega)),\ Q(\omega, \mu + \sigma_\omega \mu_t + \mu^*(\omega))]}_{4b} = 0$$

$$\forall [w_i, q, t, d] \in V_h^5 \tag{6}$$

where

$$QV(u, \mu, w) = \{\rho U_j \partial_j u, w\} + \{\mu \partial_j u, \partial_j w\}$$

$$\{a, b\} = \int_\Omega ab d\Omega \quad [a, b] = \int_K ab dK$$

$$\delta_1 = (4\rho^2 (U^2 + V^2) h^{-2} + 16(\mu + \mu_t\ + \mu^*)^2 h^{-4})^{-0.5}$$

$$\delta_2 = 0.5\rho h (U^2 + V^2)^{0.5} \min(3^{-1} \rho h (U^2 + V^2)^{0.5}\ (\mu + \mu_t + \mu^*)^{-1}, 1)$$

$$\mu^*(a) = 0.5h |R_a(a, \mu + \sigma_a \mu_t)| ((\partial_i a)^2 + (\max_\Omega\ a)^2 h^{-2})^{-0.5}$$

Ω is a bounded domain in 2D with the Dirichlet boundary Γ_D and the Neumann boundary Γ_N. Ω is split into isotropic elements element K of the size h. On Ω, the test functions $[w_i, q, t_i]$ span the Sobolev space V_h and are continuous and piecewise linear or piecewise quadratic within each K.

Eq. 6 is the SUPG formulation of Eq. 1-5 and in Eq. 6 the term $(1a)$ correspond to the standard Galerkin method of Eq. 1 and the term $(1b)$ is the least squares of Eq. 1 scaled by δ_2. Analogously the terms $(2a)$ and $(2b)$ correspond to the standard Galerkin and least squares of Eq. 2, etc. The scaling δ is chosen to

give sufficient stability and still retaining high accuracy [23]. However, it turns out that an extra diffusion μ^* is required to remove "undershoots" and it was found necessary to add such artificial viscosity for the $k-\omega$ equations. Note that, for the turbulent equations, we have not used the full residual on the test function but neglected the source term. The production and the dissipation terms can be magnitudes larger than the convection diffusion terms and are highly nonlinear. If they are introduced in the test function we must use another scaling δ in the turbulent equations. We have made calculations were we have accounted for the source terms in δ and used the full residual. This method worked but the most robust choice was to exclude the source terms in the test function. Note that it is a consistent method in both cases, and thus should we retain the high accuracy in both cases, whereby we chose the most stable method.

For clarity we might point out that if we had put more effort into tuning δ it may have been found that the full residual in the test function is still the best choice.

4 Discretization

This section describes the details in the discrete approximation of Eq. 6. This discrete approximation consists of a set of nonlinear algebraic equations represented by a matrix equation:

$$\left(\begin{bmatrix} Q & G & 0 \\ D & 0 & 0 \\ 0 & 0 & Q+S \end{bmatrix} + \begin{bmatrix} SUPG \end{bmatrix} \right) \begin{bmatrix} U_i \\ P \\ T \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ P_k \end{bmatrix} \tag{7}$$

$$\Leftrightarrow$$

$$A\Phi = F \tag{8}$$

where Q is the convection-diffusion operator, G the gradient operator, D the divergence operator, P_k the turbulent production, S the dissipation operator and $T = [k, \omega]$.

4.1 The discrete approximation

The domain is divided into triangles and nodes are introduced into each triangle in the sense of the Lagrangian element type. In each triangle, the continuous field is approximated by a polynomial of degree p based on the nodal degrees of freedom. The approximation U_i, P, k, ω will thus be continuous, while the derivatives will be continuous only at the interior of the elements.

We chose the SUPG formulation [24] which has also proven stable when using equal order elements for both pressure and velocities. In SUPG, the standard

Galerkin method is modified by perturbing the test-function. This perturbation term then controls the residual and acts in a similar way to that of the dissipation term used in FVMs. Note that it is a consistent method whereby the order of the method is preserved and corresponds to higher order dissipations in FVM.

Comparing SUPG with FVM, we can loosely say that we add a first order skew upwind scheme but compensate in the discretization of the pressure gradient such that we retain a consistent method with preserved accuracy. That means second order accuracy in most cases for $P1/P1$ ($p = 1$) elements and third order for $P2/P2$ ($p = 2$) elements. Note that SUPG also contain a term similar as the Rhie and Chow interpolation used in FVM. The integrals in (Eq. 6) are approximated by a symmetric Gauss quadrature [25] where the integrals are replaced by a sum of products of weights and integrand values in specific points. To calculate the integrand, we need derivatives of the field variables at the Gauss points and therefore map the element from the physical domain (x, y) onto a unit triangle in the computational domain (ξ, η). We used an isoparametric mapping, i.e. the geometry (x, y) is mapped using the same basis functions as the field variables. Using the chain rule, we can express the physical first and second derivatives of the basis functions (in an element) in terms of the mapping and the derivatives in the computational domain. The derivatives of the basis functions φ with respect to (x, y) can then be expressed as:

$$\begin{bmatrix} \varphi_x \\ \varphi_y \end{bmatrix} = \begin{bmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{bmatrix}^{-1} \begin{bmatrix} \varphi_\xi \\ \varphi_\eta \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} \varphi_{xx} \\ \varphi_{xy} \\ \varphi_{yy} \end{bmatrix} = \begin{bmatrix} x_\xi^2 & 2y_\xi x_\xi & y_\xi^2 \\ x_\xi x_\eta & y_\eta x_\xi + y_\xi x_\eta & y_\eta y_\xi \\ x_\eta^2 & 2y_\eta x_\eta & y_\eta^2 \end{bmatrix}^{-1} \begin{bmatrix} \varphi_{\xi\xi} - (\varphi_x x_{\xi\xi} + \varphi_y y_{\xi\xi}) \\ \varphi_{\xi\eta} - (\varphi_x x_{\xi\eta} + \varphi_y y_{\xi\eta}) \\ \varphi_{\eta\eta} - (\varphi_x x_{\eta\eta} + \varphi_y y_{\eta\eta}) \end{bmatrix} \tag{10}$$

4.2 Notes on implementing boundary conditions in FEM

Dirichlet conditions are usually specified in the FEM in the same fashion as in the FVM, while Neumann conditions are usually implemented weakly in the FEM via boundary integrals. Such integrals are much easier to implement with preserved accuracy as compared with evaluating the normal derivatives in the FVM. As a by-product of the Greens theorem, we also have boundary integrals that appear in the variational form and, if they are neglected, they are weakly set to zero at free boundary nodes.

That means that for an outlet, the zero traction condition :

$$(P\delta_{ij} - (\mu + \mu_t)\partial_j U_i)n_j = 0 \tag{11}$$

is weakly satisfied automatically (n_j the outward normal). Also, as we are solving for zero divergence on the boundary, we weakly satisfy the correct pressure boundary condition:

$$(\rho U_j \partial_j U_i + \partial_i P - (\mu + \mu_t)\partial_j \partial_j U_i)n_i = 0 \tag{12}$$

Thus, by assembling the coefficient matrix, specifying the Dirichlet nodes, and solving the equations in the other nodes, we satisfy the zero traction outlet condition for the velocities and the appropriate pressure boundary condition [26]. These boundary conditions are complex to implement in a structured second order FVM method and are usually simplified. In a higher order FVM and/or in an unstructured FVM it will be very hard to implement these boundary conditions correctly.

4.3 Adaptivity

For an efficient mesh distribution, we must adopt the mesh such that the discretization error is minimised. The first naive approach is to make two calculations on two meshes with different mesh sizes and then refine regions in which the solution differs a great deal. This will give a bad grid distribution since errors are convected from their origin, i.e. regions of high truncation error τ (τ = discrete operator on continuous solution). One way to estimate τ is to apply the coarse grid operator on the fine grid solution, a procedure which is easily implemented in an FMG process [15, 17, 27]. The counterpart to τ in FEM is the residual R, defined as the continuous operator on the discrete solution.

The residual R has the advantage over τ that it is uniquely determined on the same grid on which the approximation is made. A FEM posteriori error bound based on the residual is given in [24]. We have simplified that somewhat and it reads:

$$E := \| u - U \| \leq RN \tag{13}$$

with

$$RN = C(u,U)(\| h^2 R_1(U)/\mu \| + \| hR_2(U) \|) \tag{14}$$

where in each triangle

$$R_1(U) = (|\rho U_i \partial_i U_j - \partial_i(\mu + \mu_t)\partial_i U_j + \partial_j P|) \tag{15}$$

$$R_2(U) = (|\rho \partial_i U_i|) \tag{16}$$

and where $|\times|$ is the Euclidean norm, $\| \times \| := \{\times, \times\}^{1/2}$ and U the approximation of the exact solution u. We wish to minimise E with respect to a given set of nodes and to insert a sufficient amount of nodes to obtain $E < TOL$. First we assume a value for the stability constant C as it is complex to evaluate. We approximate the minimum of RN with respect to the grid distribution by an equidistribution, i.e all elements should give the same contribution RN_K to RN. This gives a condition to select elements to be refined. The selected elements are refined by inserting four triangles within the original coarse element, see Fig. 1. If a non-refined triangle is next to two refined triangles, it will automatically be refined.

This is iteratively performed until no non-refined triangles are next to two refined ones. A triangle next to only one refined triangle will be split into two new triangles (so called green triangles) in order to avoid hanging nodes. These green

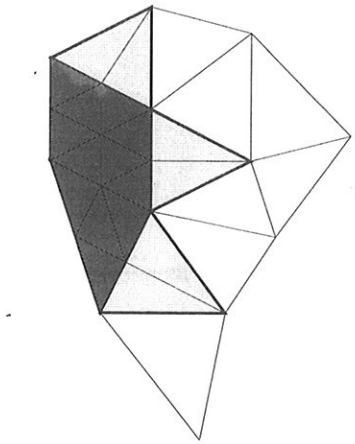

Figure 1: Adaptivity and location of nodes. Dark gray triangles are refined once, light gray triangles are green elements and white triangles are non-refined ones.

elements are removed prior to the next level of refinement to avoid degeneration of the mesh quality.

Fig. 2 shows the node distribution for a triangle that has been refined once. Using $P1/P1$ elements, nodes 1-3 exist on both the coarse and the refined level, while nodes 4-6 exist only on the fine level. Nodes 7-15 do not exist on either the coarse or the fine level, using $P1/P1$ elements. Using $P2/P2$ elements, nodes 1-6 exist on both the coarse and refined levels, while nodes 7-15 exist only on the fine level. When a coarse triangle is refined into two green triangles, one extra node is inserted for $P1/P1$ elements and three nodes for $P2/P2$ elements.

5 Iterative method

This section presents the iterative method. We first describe the SIMPLE method. It is usually used as a solver but, as it has a non optimal performance we use it only as a smoother in the full multigrid method (FMG), which is an optimal solver.

In the first section we present the SIMPLE method and how we have applied it to this FEM. We then present an extension of the SIMPLE method, handling zero traction boundaries. Finally we present the implementation of this full multigrid method with the SIMPLE smoother in conjunction with the FEM.

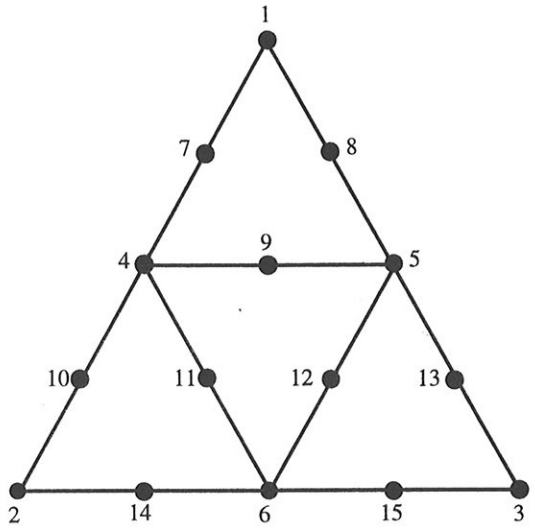

Figure 2: Node distribution on a refined element.

5.1 The SIMPLE smoother

In solving Eq. 7 we first notice that we do not have the pressure in the equation for the pressure except in the SUPG term. We must therefore post condition A with B in order to obtain a matrix $M = AB$ which allows a regular splitting (to use a simple local averaging smoother such as Jacobi, SSOR etc.). Choose B as:

$$B = \begin{bmatrix} I & -Q^{-1}G & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \tag{17}$$

with I equal to the identity operator, then

$$M = \begin{bmatrix} Q & 0 & 0 \\ D & -DQ^{-1}G & 0 \\ 0 & 0 & Q+S \end{bmatrix} \tag{18}$$

With $A^{-1} = (MB^{-1})^{-1} = BM^{-1}$ and $r = F - A\Phi$, and if $\overline{M}$ is an easy invertible approximation of M, we obtain a cheap and robust iterative method reading:

$$\Phi^{i+1} = B\overline{M}^{-1}r^i + \Phi^i \tag{19}$$

where the superscript i denotes the iteration number. We chose to run symmetric Gauss-Seidel (SGS) on each block of $\overline{M}$ and therefore had to put some restrictions on $\overline{M}$. First we needed diagonal dominance of $\overline{M}$ so we replace Q by $\overline{Q}$, which

have enough artificial viscosity to be an M-matrix (positive diagonal dominant). After that, $\overline{Q}$ is under relaxed by adding $(1-\alpha_u)\text{diag}(\overline{Q})/\alpha_u$, where $\alpha_u = 0.7$ in all calculations . Second we need Q^{-1} in $-DQ^{-1}G$ and thus here approximate Q by $\text{diag}(Q) \sim |U|/h \sim \delta_1^{-1}$ where δ_1 is the streamline diffusion parameter.

We also need a boundary condition for the pressure correction matrix:

$$M_p = -\delta_1 DGP_c \tag{20}$$

If we consider BM^{-1} we see that the velocities are corrected by the pressure corrections in order to satisfy zero divergence. We should not have any divergence corrections on a Dirichlet boundary. By identifying the velocity divergence corrections on the boundary, we find by applying Green's theorem on Eq. 21, that it corresponds to a zero Neumann pressure correction condition. If we reformulate the pressure correction equation weakly and use Green's theorem, we obtain the following variational form.

Find $P_c \in V_h$ such that

$$\{\delta_1 \nabla P_c, \nabla w_c\} = r_p \quad \forall w_c \in V_h \tag{21}$$

As we have dropped the surface integral arising from Green's theorem, we weakly satisfy $\partial_n P_c = \partial_i P_c n_i = 0$ simply by also solving Eq. 21 on the boundary.

To sum up we present how the method is implemented in practice:

1. Evaluate the residual r and the M-matrix for the U-equation. Smooth by a few SGS sweeps and update U.
2. Evaluate the residual r and the M-matrix for the V-equation. Smooth by a few SGS sweeps and update V.
3. Evaluate the residual r and the matrix for the continuity equation. Smooth by a few SGS sweeps and update P, U and V.
4. Evaluate the residual r and the M-matrix for the k-equation. Smooth by a few SGS sweeps and update k.
5. Evaluate the residual r and the M-matrix for the ω-equation. Smooth by a few SGS sweeps and update ω.
6. Update the turbulent viscosity.
7. Repeat 1. to 7. until convergence.

5.2 Extension of the SIMPLE method to multiple free outlets

Assume that, at the outlet we have a zero traction boundary condition instead of a Dirichlet velocity condition . The pressure correction outlet condition should then read $P_c = \mu \partial_n U_{cn} = -\mu \partial_n^2 P_c$, a boundary condition that is difficult to implement in FEM. If that boundary condition is reduced to $P_c = 0$ it strongly implies $P = 0$, which obviously is wrong as we should satisfy the zero traction boundary condition weakly.

However, if we use $\partial_n P_c = 0$, we obtain a singular pressure correction problem and, because the velocities are free at the outlet, we will not satisfy the compatibility condition of the pressure correction equation (global mass conservation). That is shown as very poor convergence of the SIMPLE algorithm. A remedy to this problem often used in the FVM's is to change the normal velocity at the outlets in order to satisfy the compatibility condition.

A dual point of view of that remedy, is that weakly enforced on the outlet is a pressure correction gradient, corresponding to an added constant velocity such that the compatibility condition is satisfied. However, together with a zero traction outlet, this condition gives a conflict situation, because the velocity modifications result in a pressure correction drop near an outlet. In particular, it will drop differently at different outlets, and our experience is that these conflicts results in very poor convergence (often a slowdown of a factor of 10 or more compared to Dirichlet outlets). If we simplify the zero traction outlet boundary condition by imposing zero velocity gradients, we avoid the conflict situation above. However, we then only obtain accurate results for developed channel flows, while for more general flow situations such outlet condition will detoriate the solution if it converges at all.

If we consider the zero traction outlet boundary condition, we see that it implies a pressure level in the domain such that it forces the velocities at the outlets to satisfy global mass conservation. Thus during the course of iterations, it is desirable to raise the pressure correction in the inner domain such that it gives a velocity correction at the outlets enforcing the global mass conservation condition. At the same time, we want to correct the pressure by means of the local divergence errors.

We therefore decoupled these two processes during each iteration. A constant mass source $\dot{m}_{add}$ is first added so that the compatibility condition is satisfied. Second, the pressure correction equation with zero Neumann boundary conditions is smoothed. Finally, we add a constant pressure correction α, corresponding to the global mass imbalance, to all nodes except the zero traction nodes. This is realised by solving a one degree of freedom Galerkin FEM problem based on Eq. 21. It has a test function and a base function equal to unity for all inner nodes and Dirichlet boundary nodes, while at zero traction boundary nodes are they set to zero. This one-d.o.f problem has a source term equal to $-\dot{m}_{add}$ and it reads:

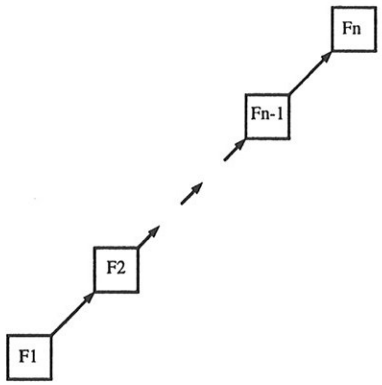

Figure 3: Full multigrid description

Find α such that:

$$\{\delta_1 \nabla \phi_{gc}, \nabla \phi_{gc}\}\alpha = -\dot{m}_{add} \tag{22}$$

where

$$\dot{m}_{add} = -\sum_{i=1,N} r_p \quad \text{and} \quad \phi_{gc} = \sum_{i \in \Upsilon} \phi_i \tag{23}$$

and $\phi_i = 1$ in node i and zero for all other nodes and Υ contain all N nodes except free boundary nodes. This will give a raised pressure correction level in the inner domain and, by applying B in Eq. 17 on this pressure correction, the velocities are forced to match global mass conservation without changing the pressure correction on the zero traction boundary.

The algorithm described above proved to be equally fast and robust for flow situations with multiple zero traction outlets as for flow situations with only one Dirichlet outlet. It should also be mentioned that this decoupled SIMPLE method, using the pressure correction stiffness matrix, is implemented for arbitrary elements, arbitrary numbers of outlets, in arbitrary directions, 2D or 3D, within 20 extra lines of Fortran 77, i.e. is very easy.

5.3 The full multigrid method

In the full multigrid process, the algebraic set of equations is solved to the same accuracy as the discretization error on a coarse mesh. The approximation is then projected onto a finer mesh and the algebraic error is also reduced on this mesh to the level of the discretization error. This approximation is projected on an even finer mesh and so on, see Fig. 3. The solver on each mesh is the multigrid method with a F-cycle, see Fig. 4.

The multigrid method uses a simple iterative method to smooth the residual and a sequence of coarser grids to evaluate cheap corrections to the smooth part

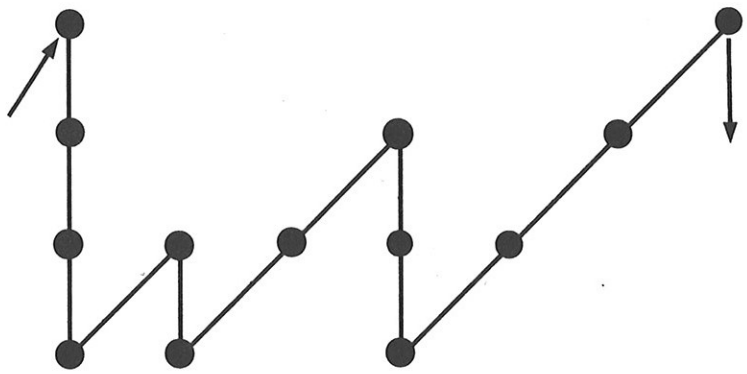

Figure 4: F-cycle description

of the residual. Fortunately, the SIMPLE method, in combination with a local averaging method, has good smoothing properties (at least for $Pe = Uh/\nu \sim 25$ or smaller).

A sequence of coarser grids is defined via the adaptivity process whereby we already have all coarse meshes and their operators defined, see Fig. 5. To use this sequence of grids in the multigrid process, we must define the transfer operators [28]. The prolongation operator, projecting the coarse grid corrections Δ_{2h} on the fine grid, is defined by:

Find $\Delta_h \in V_h$ such that

$$\{\Delta_h - \Delta_{2h}, v_h\} = 0, \quad \forall v_h \in V_h \tag{24}$$

or, in matrix notation,

$$D^h(\Delta^h - P\Delta^{2h}) = D^h\Delta^h - C\Delta^{2h} = 0. \tag{25}$$

Since the meshes are nested, i.e. $V_{2h} \subset V_h$, the L_2 projection above simply reduces to the nodal interpolant. For the restriction, we first note that the nodal residual, r^h, can be viewed as an L_2 projected residual field b^h, as:

$$D^h b^h = r^h \tag{26}$$

This residual field is then projected onto Ω_{2h}. That means:

$$D^{2h}b^{2h} - C^T b^h = 0 \Rightarrow r^{2h} = C^T D^{-h} r^h = P^T r^h \tag{27}$$

Thus the restriction operator is the transpose of the prolongation operator, which in the case of conformal cells is the transpose of the nodal interpolant.

In the multigrid method, the nonlinearity can be handled either by a global linearization or by a nonlinear iteration via the full approximation scheme (FAS).

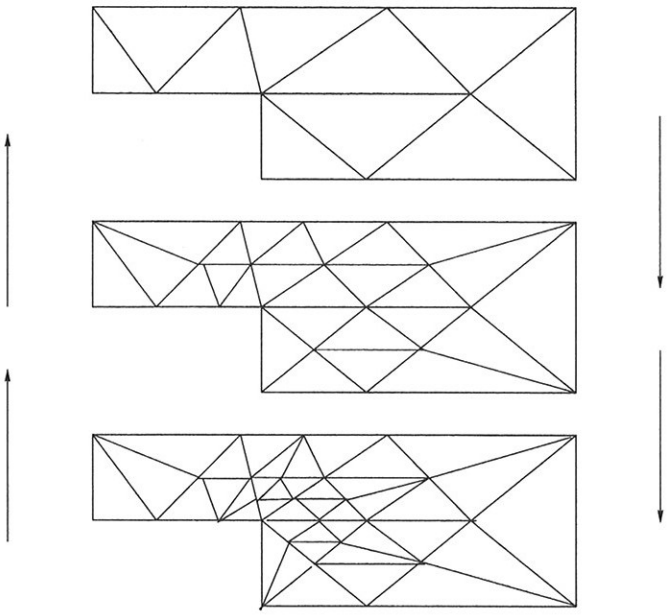

Figure 5: Multigrid grid hierarchy

The corrections on the coarse mesh in FAS are then the changes from the approximation that was projected from the fine mesh and used in the nonlinear iteration procedure. It has been stated that the Newton linearization does not give any acceleration owing to the frequency with which the residual in the FAS algorithm is evaluated and the negligible extra cost of performing a FAS cycle compared to a CS (correction scheme [15]) cycle. This has also been shown in explicit tests [29] and, thus have we used the FAS. We used the F-cycle as it has been shown to be more robust and somewhat faster than the V-cycle and the W-cycle [17].

With the grid hierarchy shown in Fig. 5, the method does not follow the asymptotic optimal behaviour. However, as long as the number of levels with a small amount of refined cells is not too many (2-4), we have seen a slowdown of only a factor of 2-3 at most. The advantage is that we smooth the whole domain on each level, and also smooth over the interior "interfaces" created within the adaptivity process. This gives us a more robust solver, which is important when solving complex turbulent flows.

6 Data structures

We store the geometry and the solution in all NNO nodes. The coefficient matrix is implemented in a compressed fashion where only the nonzero quantities are stored using a pointer for all coefficients. The size of the array for the coefficients

and the pointer is approximately $10 \times NNO \times 8$ bytes (double precision) for $P1/P1$ elements and $35 \times NNO \times 8$ bytes for the $P2/P2$ elements.

We also store pointers for all NEL cells such as the node connectivity $3p \times NEL \times 4$ bytes, parent-child connectivity $5 \times NEL \times 4$ bytes and neighbour connectivity of the size $3 \times NEL \times 4$ bytes. In addition, about $5 \times NNO \times 4$ bytes pointers are stored for the renumbering in the FMG process, boundary conditions etc. In the FMG process, we must also store a coarse grid solution and a residual field, which correspond to $2 \times 7 \times NNO \times 8$ bytes. Altogether including the solution $(7 \times NNO \times 8bytes)$, we store approximately $60 \times NNO \times 8$ bytes in a LIFO-stack for both the $P1/P1$ and the $P2/P2$ methods.

Note that this is a small memory usage compared with most commercial packages, which usually use 10-20 times more memory than does this FEM code. When programming this code we had a great help of the data-structure from the free-ware Femlab [30]. It is an adaptive 2D code for the scalar Poisson equation, with a direct solver and with $P1/P1$ elements. We greatly appreciated the use of Femlab even though we only could use about 10 % of the source code.

7 Some issues on the implementation of the $k-\omega$ turbulence model

We will present here some details about our implementation of the $k-\omega$ turbulence model [21]. We first need a wall boundary condition ω_{wall} for ω. Asymptotically, as $y \to 0$, $\omega = 6\nu/\beta^* y^2$, where y is the normal distance from the wall. If we just try to set a very large value of ω_{wall} we get a very large numerical diffusion of the source term in the k equation resulting in non-positive k values. This can easily be seen when a one-dimensional, two-element discretization of a fully developed channel flow is considered.

We therefore approximated $y \to 0$ by the diameter d_{ew} of the elements close to the wall. However on very coarse meshes, ω becomes too small and, consequently, the length scale becomes too large. To obtain realistic results on very coarse meshes in the FMG method, we limit μ_t and set $\omega_{wall} = \max((2IU/l), (6\nu/\beta^* d_{ew}^2))$, where l is the integral length scale and I the turbulence intensity.

Note that in the cycles in the multigrid we only produce corrections around an approximation from the current level in the FMG method and thus we can use the same wall value of ω_{wall} on all meshes in the cycles. The discretization of the $k-\omega$ model often needs to satisfy a discrete maximum principle to avoid numerical troubles. That is realized via the artificial viscosity μ^*, reducing the accuracy locally to first order, but the overall convergence should be close to optimal, similar to the TVD schemes in the FVM. The nonlinearity is handled by Picard iteration every SIMPLE sweep (using the values from the previous iteration).

The $k-\omega$ equations are strongly coupled via the nonlinear source terms and thus we must damp the iterative method for $k-\omega$. We therefore include the strictly negative dissipation term in the stiffness matrix. In a similar fashion, is the source

term appearing in the multigrid used to damp the $k-\omega$ equations on a coarse grid. If the multigrid source term is negative, we use the realisability of $k-\omega$ and lump it into the diagonal of the stiffness matrix (divide the multigrid source term by the nodal turbulent variable) [16]. This does not affect the consistency of the multigrid but gives an initial (the first iterations) behaviour similar to what would be found if the production term were increased.

The corrections from the smoother and the multigrid are filtered for the turbulent quantities such that temporarily non-positive $k-\omega$ values are avoided. The filter is nonlinear and reads: $k \leftarrow k*(k+\max(\Delta k,0))/(k-\min(\Delta k,0))$, where Δk is the correction before the filter [16, 31].

8 Results

We chose the lid-driven cavity as a test case ($Re = 100$ and $Re = 3200$), to evaluate our discretization and solver. We first compared our calculations with the benchmark solutions of Ghia *et. al.* [32]. We achieved grid independent solutions deviating five percent from those of Ghia *et. al.* In particular, we obtained the best agreement with Ghia *et. al.* when we used $P1/P1$ elements and the same number of nodes as in [32]. If the mesh is further refined or $P2/P2$ elements were used, the results converged to the solution differing five percent from those in [32].

We also calculated the same two flow cases using our finite volume code [18]. This second order FVM code converged to the same solutions as did our $P1/P1$ and $P2/P2$ FEM code. We therefore evaluated the accuracy and efficiency of our FEM formulations towards our FVM formulation instead of Ghia's results. We also compare the performance of our second order $P1/P1$ method with the sharpest scheme available in FLUENT/UNS [33]. Following this, we present a section in which we calculate benchmark solutions for $Re = 100, 400, 1000, 3200$ and $Re = 5000$ using the $P2/P2$ FEM.

The fourth subsection presents the performance of a calculation with two inlets and two outlets using our decoupled SIMPLE smoother. In the final subsection, we show the performance of the $P1/P1$ method in turbulent calculations of three backwards-facing step flows.

8.1 Evaluation of the $P1/P1$ and $P2/P2$ FEM methods

The $P1/P1$ and $P2/P2$ finite element methods with the full multigrid SIMPLE solver are here compared with the collocated structured finite volume code in [18]. This FVM code uses the second order QUICK scheme and has a FMG-SIMPLE line SSOR solver. We use a structured uniform Cartesian grid for the FVM and, for the two FEM methods ($P1/P1$ and $P2/P2$) we have for all calculations used a non-uniform triangular base mesh, see Fig. 6. This base mesh has 100 non-stretched elements of similar size. It is important to use an unstructured triangulation, and not triangulation based on a structured mesh with split quadrilaterals, because

Method	Nodes	cycles	CPU-time	Adaptivity
FVM	10000	3	60	No
P1/P1	8000	4	46	No
P2/P2	3000	4	98	No
P1/P1	3600	4	50	Yes
P2/P2	2300	3	86	Yes

Table 1: The number of nodes required for different methods to reduce E_{xu} below 0.5×10^{-4} for $Re = 100$.

Method	Nodes	cycles	CPU	Adaptivity
FVM	700000	21	18000	No
P1/P1	600000	34	21000	No
P2/P2	70000	15	8300	No
P1/P1	220000	20	16000	Yes
P2/P2	14000	9	2600	Yes

Table 2: The numbers of nodes required for different methods to reduce E_{xu} below 0.5×10^{-4} for $Re = 3200$.

cancellation can occur on the latter grid, giving results that are too good and not showing the real behaviour of the unstructured method. As a triangle has higher approximation error than a quadrilateral, we expect the structured FVM to be superior to the unstructured $P1/P1$ FEM.

However, we find the opposite in Tables 1-2, which indicates the power of the SUPG concept. Tables 1-2 include, for $Re = 100$ and $Re = 3200$, respectively, the number of nodes required to achieve a solution in which $E_{xu} = \max(u - U)/\max(u) < 5 \times 10^{-4}$ for $x = 0.5$, where u is the exact solution and U is the approximation of u. The results in Tables 1-2 are based on an estimated exact solution in the sense of Rickardsson extrapolation based on calculations of a sequence of grids up to 200 000 nodes for all three methods.

The required CPU time is measured on a DEC 3000/700. We can see that, for $Re = 100$, the $P2/P2$ method is about twice as expensive as the other two. The number of nodes for the $P2/P2$ only is 30% of the other methods when not using adaptivity. Using adaptivity, the number of nodes can be reduced by 50% for $P1/P1$ and by 25% for $P2/P2$.

We see for $Re = 3200$ that the $P2/P2$ method is three times as fast as the $P1/P1$ method and twice as fast as the FVM. However, the $P2/P2$ method uses only 15% of the memory of the $P1/P1$ method. Using adaptivity, the $P2/P2$ method is even more efficient than the $P1/P1$ because of the third order convergence of the $P2/P2$ method compared with the second order convergence for the $P1/P1$ method.

The overhead in CPU time for the $P2/P2$ method as compared with the $P1/P1$ method is per iteration approximately a factor of two with an equal number of nodes in the $P1/P1$ and the $P2/P2$ methods. That is because we use seven-point quadrature for $P2/P2$ elements while only one-point for $P1/P1$ elements. Using the $P2/P2$ method, we must perform three pre- and post-smoothing sweeps instead of one for the $P1/P1$ method. We also needed to increase the number of SSOR sweeps from one to four, but then the number of cycles was reduced by a factor of two or three. We tested four-point quadrature for $P2/P2$ elements and no degeneration of the results. This means that the $P2/P2$ CPU times could have been reduced by 25%.

The cost of the $P1/P1$ is approximately the same per iteration as of the structured code, but the SIMPLE smoother in the structured code uses an alternating symmetric line Gauss-Seidel smoother, which is twice as expensive as the symmetric point Gauss-Seidel relaxation method used in the FEM code. Thus we find the overhead using the unstructured data structure to be about 2-3 in 2D.

It should be mentioned that without multigrid the $P2/P2$ solver converges approximately six times more slowly than does the $P1/P1$. In the present study, the multigrid method is used for all calculations, since a single grid calculation would probably be approximately 1000 times slower than a corresponding multigrid calculation would on a 200 000 node mesh.

It is important to have a method which is robust with respect to mesh quality, preventing spurious oscillations. We have found the SUPG to have good stability and an example with $Re = 3200$ on a coarse grid is shown in Fig.s 7-9. Fig. 7 shows the adaptively refined mesh where the mesh size is changed abruptly in some regions by a factor of two. Fig.s 8-9 show a velocity vector plot and a pressure plot. We can clearly conclude that no oscillations occur, even though we have a coarse mesh with 294 elements (173 nodes) and the mesh size varies a great deal locally, see Fig. 7.

The effects of a piecewise linear contour plot technique (clearly visible in Fig. 9) is not to be confused with spurious oscillations. We can also see that the major flow structures are resolved on this coarse mesh. The good stability of the $P1/P1$ method is shown here, but even better regularity is obtained with the $P2/P2$ method, maybe because the viscous perturbation of the test-function is present there.

It should be mentioned that the use of adaptivity does not increase the number of cycles in the multigrid, but rather a decreased number of cycles were used if the refined region not is too large, see Tables 1-2.

8.2 Comparison of the $P1/P1$ FEM and FLUENT/UNS

We have included the part of the investigation [34] concerning the lid-driven cavity $Re = 3200$. We have compared both the $P1/P1$ FEM and the most accurate scheme in FLUENT/UNS on a serie of pairwise equivalent grids (generated by two different mesh generators). We have in Table 3 monitored the convergence of

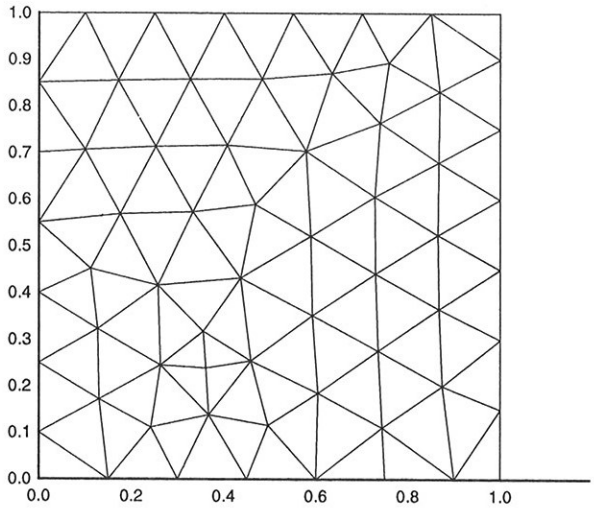

Figure 6: Base mesh for lid-driven cavity

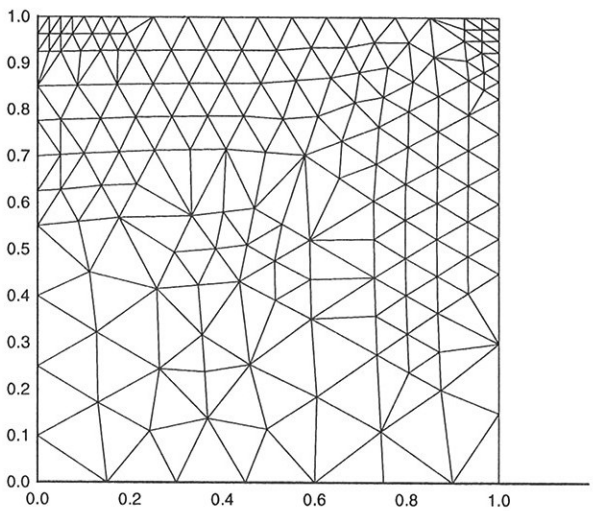

Figure 7: An adaptively refined grid with 173 nodes for $Re = 3200$.

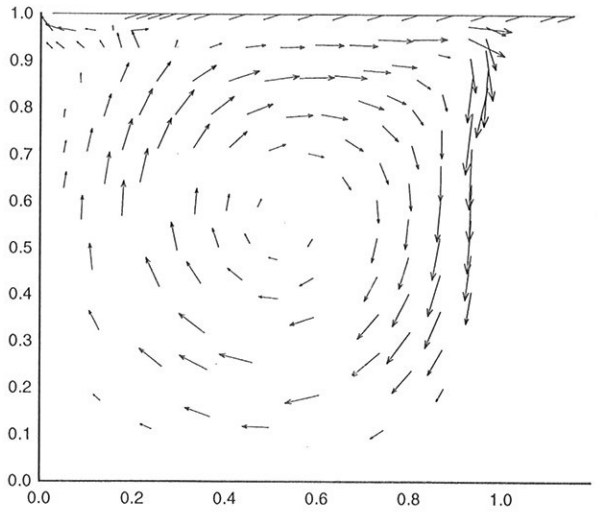

Figure 8: Velocity vectors for $Re = 3200$.

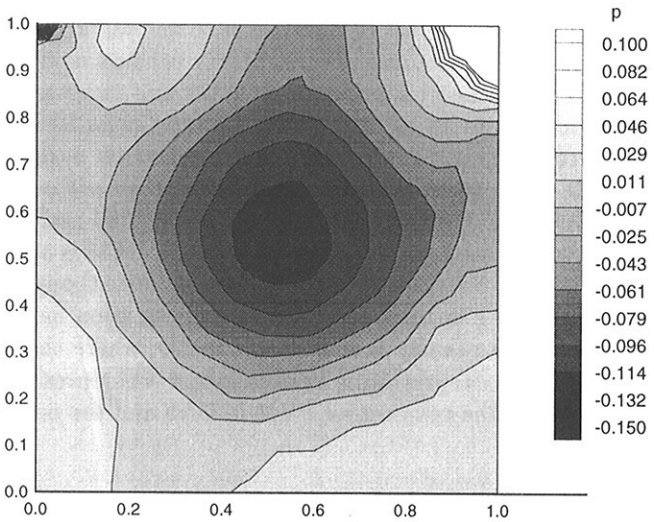

Figure 9: Pressure contours for $Re = 3200$. Note that the piecewise linear approximation is visible in the contours but that no spurious oscillation occurs.

Method	Nodes	$U(0.5, 0.1)$	Iterations	CPU (min)
UNS/AMG	68	-0.112	230	
UNS/AMG	241	-0.211	830	
UNS/AMG	905	-0.313	1500	6.5
UNS/AMG	3505	-0.381	4100	70
FEM/GS	65	0.016	320	
FEM/GS	229	-0.301	950	
FEM/GS	857	-0.391	3500	2.5
FEM/GS	3313	-0.422	12000	35
FEM/FMG	3313	-0.422	407	2.3
Benchmark		-0.4338		

Table 3: Comparison of the performance of the $P1/P1$ FEM compared with FLUENT/UNS for the lid-driven cavity $Re = 3200$. These calculations were performed on an Indigo2/R10000 SGI machine.

$U(x = 0.5, y = 0.1)$ as it has shown to well represent the errors in the whole cavity. It is shown in Table 3 that the $P1/P1$ SUPG method presented here is much more accurate than the cell-centred finite volume method used in FLUENT/UNS . In fact, the $O(h^2)$ scheme in FLUENT/UNS seems to be only about $O(h^1)$ accurate, while the formally $O(h^{1.5})$ accurate $P1/P1$ SUPG behaves as $O(h^2)$. We can also see that FLUENT/UNS uses ten times more RAM and twice as much CPU time on the same amount of nodes as our SIMPLE point Gauss-Seidel solver without multigrid. Note that FLUENT uses an AMG method on each linearised scalar equation and still needs twice as much CPU time. If we use our FMG solver, it converges 35 times faster than FLUENT/UNS on the 3500 node case.

On the 3500 node mesh, we see that the error for FEM is only 20% of that of the FVM in FLUENT/UNS. This difference is present even though FLUENT/UNS use twice as many d.o.f's in their cell-centred discretization as the node centred $P1/P1$ FEM does. This penalty is even larger in 3D, where there are about five times more cells than nodes, resulting in even larger differences in CPU time and memory usage between the-cell centred FVM in UNS and the node centred SUPG FEM.

8.3 Benchmark solutions for the lid-driven cavity

Tables 4-9 present our benchmark solutions for the lid-driven cavity and give an error bound of the data given. This error bound is based on a comparison in all points given here of calculations made on three different meshes. These three

Y	$Re = 100$	$Re = 400$	$Re = 1000$	$Re = 3200$	$Re = 5000$
0.05000	-0.0343	-0.0753	-0.1682	-0.3367	-0.3995
0.10000	-0.0635	-0.1440	-0.2967	-0.4338	-0.4195
0.15000	-0.0903	-0.2138	-0.3797	-0.3702	-0.3583
0.20000	-0.1163	-0.2788	-0.3759	-0.3187	-0.3122
0.25000	-0.1419	-0.3212	-0.3189	-0.2721	-0.2653
0.30000	-0.1668	-0.3254	-0.2591	-0.2249	-0.2187
0.35000	-0.1891	-0.2923	-0.2079	-0.1780	-0.1722
0.40000	-0.2060	-0.2370	-0.1597	-0.1311	-0.1257
0.45000	-0.2139	-0.1753	-0.1112	-0.0842	-0.0791
0.50000	-0.2091	-0.1150	-0.0620	-0.0369	-0.0321
0.55000	-0.1894	-0.0570	-0.0121	0.0110	0.0155
0.60000	-0.1543	0.0009	0.0390	0.0600	0.0642
0.65000	-0.1050	0.0600	0.0920	0.1106	0.1144
0.70000	-0.0440	0.1207	0.1479	0.1636	0.1668
0.75000	0.0279	0.1813	0.2079	0.2197	0.2223
0.80000	0.1149	0.2387	0.2721	0.2799	0.2817
0.85000	0.2321	0.2904	0.3353	0.3460	0.3463
0.90000	0.4083	0.3528	0.3842	0.4182	0.4187
0.95000	0.6719	0.5410	0.4592	0.4612	0.4778

Table 4: U-velocity for $X = 0.5$ with $E_{xu} < 5 \times 10^{-4}$.

meshes are a conformal sequence of grids refined over the whole domain having $NNO, 4NNO$ and $16NNO$ number of nodes, where $NNO = 12500$. We used the $P2/P2$ discretization and compared the solutions on these three meshes. We clearly saw the third order convergence, and this comparison gives an estimate of the error on the finest mesh in the sense of Richardsson extrapolation.

8.4 An example of multiple outlet ventilated enclosure.

Below we present a laminar model problem of a ventilated room with two inlets with $Re = 120$ and $Re = 40$, respectively, and two free outlets. It gives a complex flow pattern within the room and different mass fluxes at the two outlets. A wall jet becomes turbulent when $Re_{inlet} = 100 - 150$, so it is indeed a convection-dominated flow case.

We used the adaptive process and stopped at 30 000 nodes. The mesh is shown in Fig. 10. The $P2/P2$ FEM discretization with the FMG decoupled SIMPLE solver was used. It converged (grid independently) within 12 $F(3,3)$ cycles which, is approximately 125 WU. We can clearly see that the convergence is as fast as for the lid-driven cavity, and the CPU time on a DEC 3000/700 workstation was 1.1 hours. Fig.s 11-12 shows the major features of the flow in the streamline plot and the pressure distribution plot.

Y	$Re = 100$	$Re = 400$	$Re = 1000$	$Re = 3200$	$Re = 5000$
0.05000	-0.00003	-0.00039	0.00020	-0.00166	-0.00380
0.10000	-0.00019	-0.00229	-0.00294	-0.00408	-0.00699
0.15000	-0.00030	-0.00323	-0.00072	-0.00501	-0.00857
0.20000	0.00014	0.00044	0.00430	-0.00490	-0.00801
0.25000	0.00189	0.00979	0.00732	-0.00325	-0.00618
0.30000	0.00594	0.02229	0.00927	-0.00075	-0.00352
0.35000	0.01315	0.03387	0.01219	0.00238	-0.00028
0.40000	0.02426	0.04215	0.01624	0.00599	0.00339
0.45000	0.03932	0.04763	0.02089	0.00998	0.00742
0.50000	0.05753	0.05206	0.02581	0.01426	0.01173
0.55000	0.07709	0.05674	0.03083	0.01871	0.01620
0.60000	0.09546	0.06195	0.03577	0.02318	0.02069
0.65000	0.10973	0.06723	0.04038	0.02742	0.02498
0.70000	0.11717	0.07163	0.04431	0.03110	0.02872
0.75000	0.11542	0.07392	0.04715	0.03374	0.03145
0.80000	0.10270	0.07266	0.04833	0.03477	0.03255
0.85000	0.07835	0.06609	0.04668	0.03358	0.03136
0.90000	0.04513	0.05033	0.04031	0.02969	0.02735
0.95000	0.01336	0.02051	0.02323	0.02127	0.01970

Table 5: V -velocity for $X = 0.5$ with $E_{xv} < 5 \times 10^{-5}$.

Y	$Re = 100$	$Re = 400$	$Re = 1000$	$Re = 3200$	$Re = 5000$
0.05000	0.01492	0.01781	0.01423	0.00456	0.00910
0.10000	0.01475	0.01677	0.00880	-0.01288	-0.01097
0.15000	0.01429	0.01345	-0.00522	-0.03316	-0.02984
0.20000	0.01329	0.00596	-0.02547	-0.05026	-0.04650
0.25000	0.01142	-0.00707	-0.04497	-0.06518	-0.06098
0.30000	0.00826	-0.02480	-0.06077	-0.07782	-0.07317
0.35000	0.00332	-0.04404	-0.07339	-0.08812	-0.08303
0.40000	-0.00380	-0.06128	-0.08340	-0.09604	-0.09056
0.45000	-0.01331	-0.07469	-0.09084	-0.10158	-0.09575
0.50000	-0.02492	-0.08398	-0.09565	-0.10471	-0.09858
0.55000	-0.03778	-0.08942	-0.09778	-0.10544	-0.09906
0.60000	-0.05050	-0.09131	-0.09726	-0.10380	-0.09722
0.65000	-0.06148	-0.08983	-0.09416	-0.09988	-0.09316
0.70000	-0.06933	-0.08523	-0.08863	-0.09382	-0.08701
0.75000	-0.07323	-0.07804	-0.08091	-0.08590	-0.07903
0.80000	-0.07305	-0.06925	-0.07141	-0.07653	-0.06962
0.85000	-0.06926	-0.06035	-0.06108	-0.06639	-0.05946
0.90000	-0.06279	-0.05281	-0.05174	-0.05645	-0.04956
0.95000	-0.05515	-0.04709	-0.04560	-0.04880	-0.04162

Table 6: Pressure for $X = 0.5$ with $E_{xp} < 5 \times 10^{-5}$.

X	$Re = 100$	$Re = 400$	$Re = 1000$	$Re = 3200$	$Re = 5000$
0.05000	-0.00634	-0.00507	-0.00066	0.00510	0.00760
0.10000	-0.02222	-0.02064	-0.01376	-0.00148	0.00255
0.15000	-0.04362	-0.04149	-0.02661	-0.00781	-0.00388
0.20000	-0.06775	-0.06211	-0.03516	-0.01358	-0.00926
0.25000	-0.09285	-0.07960	-0.04059	-0.01830	-0.01369
0.30000	-0.11792	-0.09270	-0.04485	-0.02223	-0.01748
0.35000	-0.14244	-0.10143	-0.04894	-0.02575	-0.02094
0.40000	-0.16609	-0.10697	-0.05307	-0.02918	-0.02435
0.45000	-0.18858	-0.11101	-0.05735	-0.03281	-0.02799
0.50000	-0.20915	-0.11505	-0.06205	-0.03690	-0.03211
0.55000	-0.22659	-0.12013	-0.06740	-0.04163	-0.03685
0.60000	-0.23892	-0.12702	-0.07354	-0.04706	-0.04227
0.65000	-0.24317	-0.13682	-0.08035	-0.05313	-0.04830
0.70000	-0.23552	-0.15134	-0.08733	-0.05960	-0.05471
0.75000	-0.21217	-0.17161	-0.09373	-0.06605	-0.06105
0.80000	-0.17152	-0.19107	-0.10077	-0.07185	-0.06665
0.85000	-0.11735	-0.18501	-0.11622	-0.07557	-0.07056
0.90000	-0.06057	-0.12535	-0.12827	-0.07676	-0.07034
0.95000	-0.01676	-0.03930	-0.06330	-0.08294	-0.07753

Table 7: U-velocity for $Y = 0.5$ with $E_{yu} < 5 \times 10^{-5}$.

X	$Re = 100$	$Re = 400$	$Re = 1000$	$Re = 3200$	$Re = 5000$
0.05000	0.0794	0.1596	0.2500	0.3642	0.4062
0.10000	0.1315	0.2387	0.3407	0.4326	0.4347
0.15000	0.1622	0.2799	0.3762	0.3852	0.3725
0.20000	0.1767	0.3009	0.3591	0.3254	0.3177
0.25000	0.1792	0.3009	0.3071	0.2714	0.2647
0.30000	0.1722	0.2775	0.2461	0.2186	0.2128
0.35000	0.1567	0.2338	0.1882	0.1666	0.1618
0.40000	0.1326	0.1771	0.1334	0.1155	0.1114
0.45000	0.0997	0.1148	0.0795	0.0648	0.0615
0.50000	0.0575	0.0521	0.0258	0.0143	0.0117
0.55000	0.0063	-0.0095	-0.0282	-0.0364	-0.0382
0.60000	-0.0526	-0.0703	-0.0830	-0.0877	-0.0887
0.65000	-0.1161	-0.1322	-0.1389	-0.1400	-0.1402
0.70000	-0.1778	-0.2000	-0.1960	-0.1938	-0.1931
0.75000	-0.2278	-0.2810	-0.2533	-0.2496	-0.2478
0.80000	-0.2529	-0.3767	-0.3137	-0.3080	-0.3049
0.85000	-0.2407	-0.4499	-0.4032	-0.3680	-0.3650
0.90000	-0.1866	-0.4070	-0.5209	-0.4305	-0.4231
0.95000	-0.0994	-0.2128	-0.3780	-0.5669	-0.5604

Table 8: V-velocity for $Y = 0.5$ with $E_{yv} < 5 \times 10^{-4}$.

X	$Re = 100$	$Re = 400$	$Re = 1000$	$Re = 3200$	$Re = 5000$
0.05000	-0.00382	-0.00068	-0.00625	-0.01416	-0.00844
0.10000	-0.00553	-0.00322	-0.01238	-0.02689	-0.02312
0.15000	-0.00709	-0.00870	-0.02432	-0.04337	-0.03919
0.20000	-0.00890	-0.01758	-0.03992	-0.05838	-0.05377
0.25000	-0.01119	-0.02958	-0.05548	-0.07152	-0.06661
0.30000	-0.01398	-0.04351	-0.06875	-0.08264	-0.07743
0.35000	-0.01712	-0.05742	-0.07931	-0.09161	-0.08611
0.40000	-0.02033	-0.06947	-0.08731	-0.09831	-0.09256
0.45000	-0.02313	-0.07847	-0.09277	-0.10269	-0.09672
0.50000	-0.02492	-0.08398	-0.09565	-0.10471	-0.09858
0.55000	-0.02506	-0.08602	-0.09594	-0.10437	-0.09812
0.60000	-0.02292	-0.08484	-0.09367	-0.10171	-0.09539
0.65000	-0.01820	-0.08066	-0.08893	-0.09682	-0.09045
0.70000	-0.01114	-0.07351	-0.08188	-0.08981	-0.08344
0.75000	-0.00289	-0.06300	-0.07284	-0.08086	-0.07451
0.80000	0.00457	-0.04829	-0.06219	-0.07018	-0.06384
0.85000	0.00897	-0.03013	-0.04947	-0.05802	-0.05167
0.90000	0.00894	-0.01506	-0.03226	-0.04493	-0.03833
0.95000	0.00484	-0.01099	-0.01787	-0.02830	-0.02306

Table 9: Pressure for $Y = 0.5$ with $E_{yp} < 5 \times 10^{-5}$.

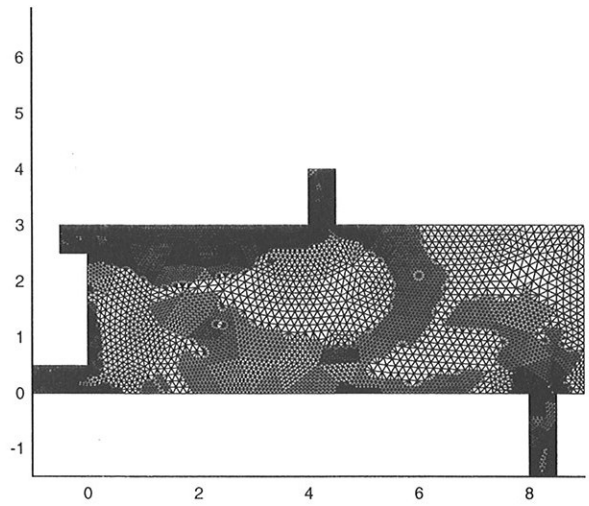

Figure 10: Grid distribution for the multiple outlet ventilated enclosure.

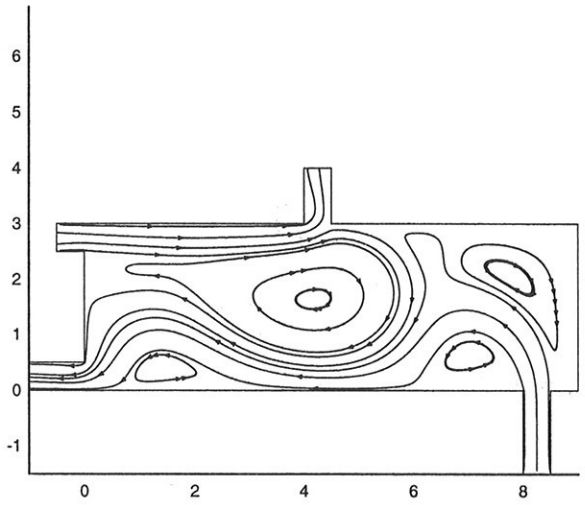

Figure 11: Velocity stream line contours for the multiple outlet ventilated enclosure.

8.5 The backwards-facing step

The final application is the backwards-facing step. We evaluated our implementation of the $k-\omega$ model by comparing our predictions with three different measurements. The geometry is described in Fig. 13.

The inlet is located $10(h+H)$ upstream the step, and the outlet is located $10(h+H)$ downstream the step . As inlet boundary condition we use a fully developed channel flow profile for U, k, ω and we have a free outlet for the outlet, i.e the zero traction condition and symmetry for $k-\omega$. We simulated three different cases (on a DEC 2100 computer) and compared with measurements in Tables 10-11. The attachment points X_R in our predictions are compared with the measured X_R and are in close agreement. We chose to use the $P1/P1$ method since we expect to need almost the same amount of nodes for the $P2/P2$ method owing to the sharpness of the boundary layers.

We needed to use up to 11 levels of refinement near the walls in order to resolve the boundary layers and, since we use non-stretched elements, we were forced to use up to 650 000 nodes. The solver converged within four to six cycles, similar to the laminar cases presented earlier in the previous sections. As we use the F(2,1)-cycle with the slightly non-optimal grid hierarchy shown in Fig. 5, these four to six cycles correspond to work equivalent to about 80-120 iterations on the finest grid. That is probably 100-1000 times faster than the single grid solver. Such

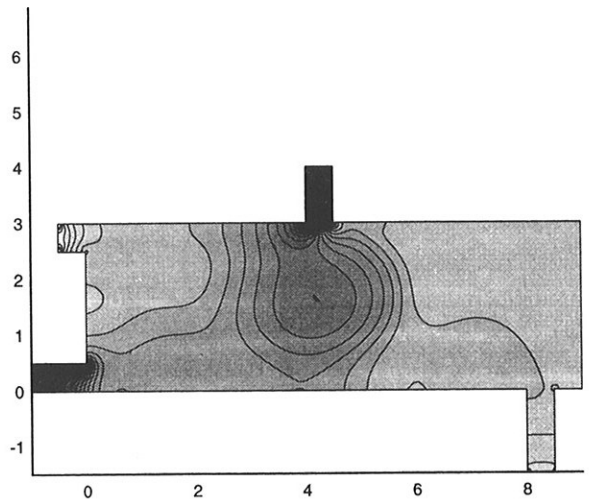

Figure 12: Contours of constant pressure for the multiple outlet ventilated enclosure.

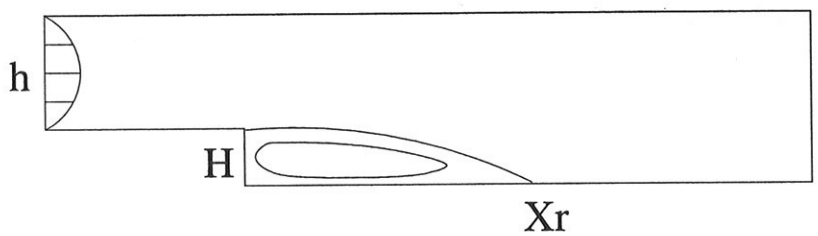

Figure 13: The geometry of the backwards-facing step.

Source	Re_H	h/H	X_R
Kim[35]	46000	2.0	6-8
Eaton[36]	38000	1.5	7.95
Kasagi[37]	5500	2.0	6.51

Table 10: Definition of the three experiments and their measured reattachment lengths.

Re_H	h/H	X_R	Nodes	Cycles	MG-levels	CPU
46000	2.0	7.2	645000	6	11	15H
38000	1.5	8.0	644000	5	11	12H
5500	2.0	6.9	166000	4	9	2.5H

Table 11: The predicted reattachment lengths in our simulations and the performance of the FMG solver using the $k-\omega$ model.

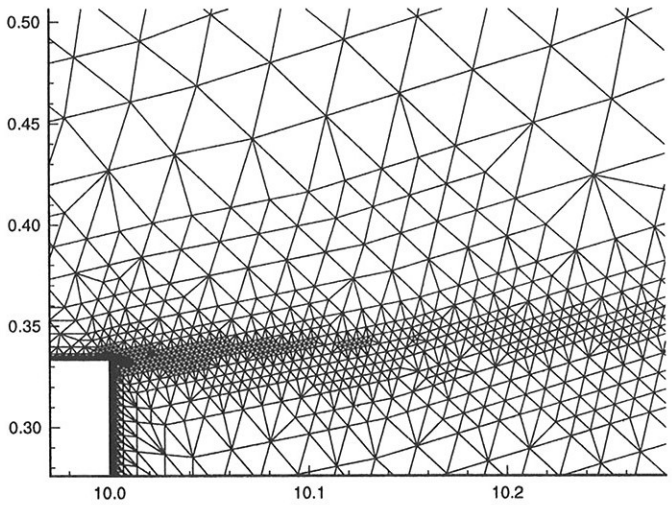

Figure 14: The grid of the shear layer after the step.

speedup is analogous to the results we obtained with our structured FMG-FVM with the $k-\epsilon$ model [18].

Fig.s 14-15 show the grid distribution around the step and close to a wall. We can see in Fig. 14 see the refinement of the shear layer from the corner of the step,as well as the many levels of refinements near the wall in Fig. 15. Profiles of U and k from two of the flow cases are compared with the measured profiles in Fig.s 16-17, and the agreement is good.

We also applied the solver to other geometries and the performance was found to be similar to that for the backwards-facing step presented here. To resolve the boundary layers in 3D with the same brute force method used here is of course impossible. However, that is purely a mesh generation problem and can be solved using using stretched tetrahedrons, prisms or hexagons.

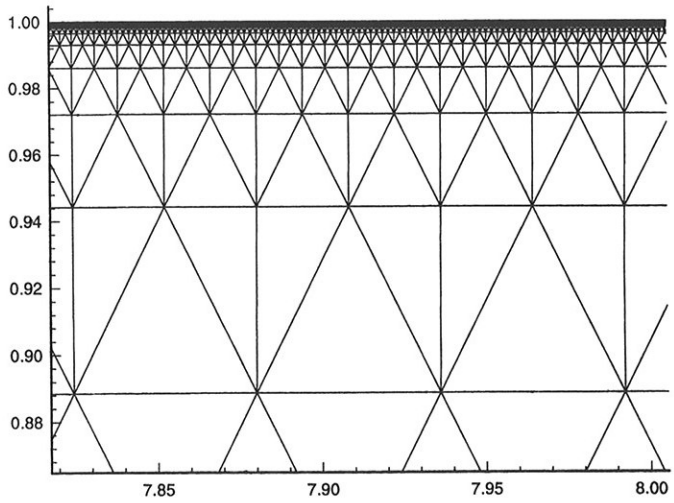

Figure 15: The grid near the wall

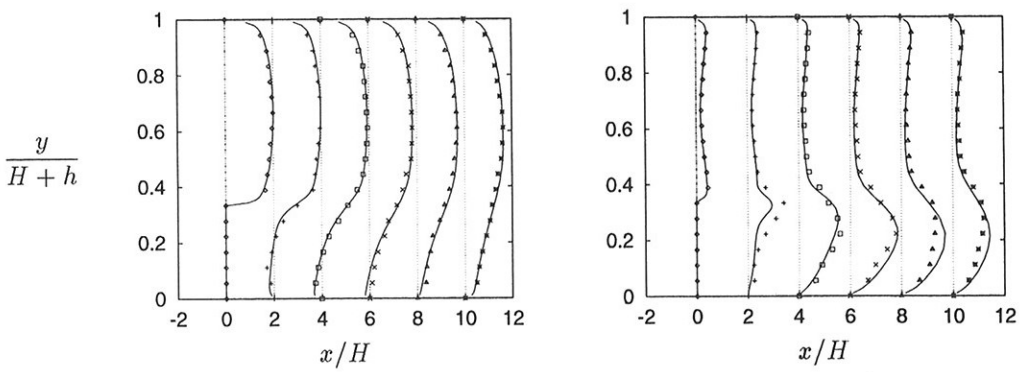

Figure 16: Profiles of U (to the left) and k (to the right) for $Re_H = 5500$. Lines represent calculations and dots measurements [37].The width of a field ($x/L = 2$) corresponds to U/U_0 and $k/(0.04U_0^2)$.

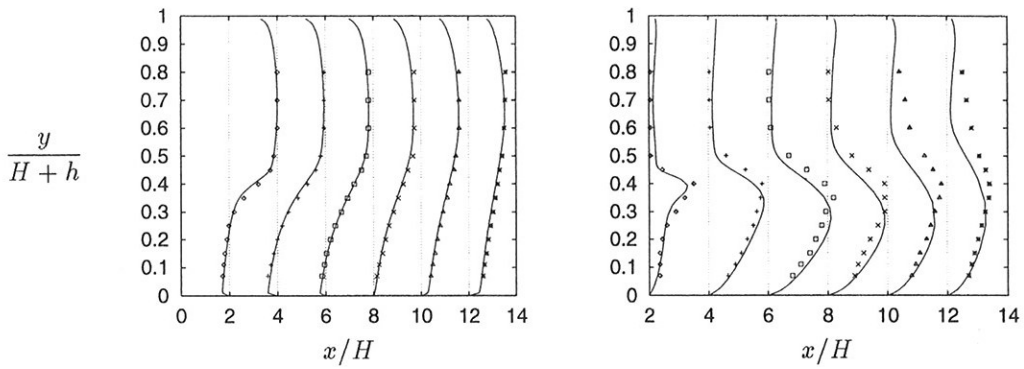

Figure 17: Profiles of U (to the left) and k (to the right) for $Re_H = 38000$. Lines represent calculations and dots measurements [36].The width of a field ($x/L = 2$) corresponds to U/U_0 and $k/(0.04U_0^2)$.

We would also like to mention that we have performed calculations, using wall functions approximating the boundary layers, resulting in problems with approximately 10 000 nodes instead. With wall functions the FMG solver showed the same behaviour as in the resolved case presented in this section but the predictions is somewhat worse.

9 Conclusions

* The $P1/P1$ and $P2/P2$ FEM methods are stable and do not give any spurious oscillations, even though the mesh quality may be poor as a result of adaptivity.
* The second order unstructured $P1/P1$-FEM is on an unstructured triangulation as accurate as a second order structured QUICK-FVM for a given set of nodes.
* The third order unstructured $P2/P2$-FEM is more accurate than the $P1/P1$ FEM.
* The FMG-SIMPLE method on a unstructured mesh is as fast with $P1/P1$ FEM as a structured SIMPLE-FMG FVM. Using $P2/P2$ FEM, the work is two times greater on a given set of nodes.
* The $P1/P1$ FMG-FEM is approximately 100 times faster, five times more accurate and uses ten times less RAM than FLUENT/UNS. The $P2/P2$ FMG-FEM method is even more superior.
* The modified (decoupled) SIMPLE method gives the same performance for multiple free outlets as for a single Dirichlet outlet.

* The FMG converges with grid-independent speed (100-200 WU) for both laminar and turbulent flows.

Acknowledgements

The support of the Swedish Research Council for Engineering Sciences (TFR) is gratefully acknowledged.

References

[1] C.M. Rhie and W.L. Chow. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA J.*, 21(11):1525–1532, 1983.

[2] B.P. Leonard. A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Comp. Meth. in Appl. Mech. Eng.*, 19:59–98, 1979.

[3] P. Johansson and L. Davidson. Modified collocated SIMPLEC algorithm applied to buoyancy-affected turbulent flow using a multigrid solution procedure. *Num. Heat Transfer, Part B*, 28:39–57, 1995.

[4] A. Jameson, W. Schmidt, and E. Turkel. Numerical solutions of the Euler equations by finite volume methods with runge-kutta time-stepping schemes. AIAA Paper 81-1259, USA, 1981.

[5] H. Guillard. Mixed element volume methods in computational fluid dynamics. Computational fluid dynamics lecture notes at von-Kármán Institute, 1995.

[6] P.R.M. Lyra, K. Morgan, J. Peraire, and J. Peiro. TVD algorithms for the solution of the compressible Euler equations on unstructured meshes. *Int. J. for Num. Meth. in Fluids*, 19:827–847, 1994.

[7] J. Casper. Finite-volume implementation of high-order essentially nonoscillatory schemes in two dimensions. *AIAA J.*, 30(12):2829–2835, 1992.

[8] J. Casper and H.L. Atkins. A finite-volume high-order ENO scheme for two-dimensional hyperbolic systems. *J. of Comp. Phys.*, 106:62–76, 1993.

[9] R. Abgrall. On essentially non-oscillatory schemes on unstructured meshes: Analysis and implementation. *J. of Comp. Phys.*, 114:45–58, 1994.

[10] T.J. Barth. Recent developments in high order k-exact reconstruction on unstructured meshes. AIAA Paper 93-0668, Reno, USA, 1993.

[11] H. Lou, J.D. Baum, and R. Löhner. An improved finite volume scheme for compressible flows on unstructured grids. AIAA Paper 95-0348, Reno, USA, 1995.

[12] P. Hansbo and A. Szepessy. A velocity-pressure streamline diffusion finite element method for the incompressible Navier-Stokes equations. *Comp. Meth. Appl. Mech Eng.*, 84:175–192, 1990.

[13] L.P. Franca and S.L. Frey. Stabilized finite element methods:II. The incompressible Navier-Stokes equations. *Comp. Meth. Appl. Mech. Eng.*, 99:209–233, 1992.

[14] T. Peterson. A note on the convergence of the discontinuous Galerkin method for a scalar hyperbolic equation. *SIAM J. Num. Anal.*, 28(1):133–140, 1991.

[15] A. Brandt. Multigrid techniques: 1984 guide with applications to fluid dynamics. Technical report, Von-Kármán Institute, 1984.

[16] P. Johansson and L. Davidson. A full multigrid method applied to turbulent flow using the SIMPLEC algorithm together with a collocated arrangement. In *Multigrid Methods IV*, pages 245–256. Birkhäuser Verlag, 1994.

[17] P. Emvin and L. Davidson. A local mesh refinement algorithm applied to turbulent flow, to appear. *Int. J. for Num. Meth. in Fluids*, 24(5):519–530, 1997.

[18] P. Emvin and L. Davidson. PEC-FLTBB, a CFD code for turbulent buoyant flow based on a local mesh refinement full multigrid strategy using boundary-fitted coordinates. Report 96/7, Dept. of Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1996.

[19] H.C. Chen and V.C. Patel. Practical near-wall turbulence models for complex flows including separation. AIAA Paper 87-1300, Reno, USA, 1987.

[20] D.C. Wilcox. *Turbulence modeling for CFD.* DCW Industries Inc., La Canada , California, 1993.

[21] D.C. Wilcox. Reassessment of the scale-determining equation for advanced turbulence models. *AIAA J.*, 26(11):1299–1310, 1988.

[22] J. Larsson. Numerical simulation of turbine blade heat transfer. Thesis for the degree of Licentiate of Engineering, report 96/4, Dept. of Thermo and Fluid Dynamics, Chalmers Univ. Techn., Göteborg, 1996.

[23] M. Behr and T.E. Tezduyar. Finite element solution strategies for large-scale flow simulations. *Comp. Meth. Appl. Mech. Eng.*, 112:3–24, 1994.

[24] P. Hansbo and C. Johnson. Streamline diffusion finite element methods for fluid flow. Computational fluid dynamics lecture notes at von-Kármán Institute, 1995.

[25] D. Dunavant. High degree efficient symmetrical Gaussian quadrature rules for the triangle. *Int. J. Numer. Meth. Eng*, 21:1129–1148, 1985.

[26] P.M. Gresho and R. Sani. On pressure boundary conditions for the incompressible Navier-Stokes equations. *Int. J. Num. Meth in Fluids*, 7:1111–1145, 1987.

[27] S.C. Caruso, J.H. Ferziger, and J. Oliger. Adaptive grid techniques for elliptic fluid-flow problems. Report no. TF-23, Thermosciences Division Department of Mechanical Engineering, Stanford University, 1985.

[28] P. Hansbo. Finite elements in practice. Lecture notes, Dept. of Math., Chalmers Univ. Techn., Göteborg, 1994.

[29] A. Brandt, W. Joppich, J. Linden, G. Lonsdale, A. Schüller, B. Steckel, and K. Stüben. Arbeitspapiere der GMD 690. Technical report, Gesellschaft für mathematik und datenverarbeitung, 1992.

[30] P. Hansbo and M. Levenstam. FEMLAB. Public domain software available on internet , (http.www.math.chalmers.se/Research/Femlab), Dept. of Mathematics, Chalmers University of Technology, Göteborg, 1995.

[31] F.S Lien and M.A. Leschziner. Multigrid convergence accelleration for complex flow including turbulence. In *Multigrid Methods III*, pages 277–288. Birkhäuser Verlag, 1991.

[32] U. Ghia, K.N. Ghia, and T. Shin. High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *Comp. Phys.*, 48:387–411, 1982.

[33] User's guide for FLUENT/UNS & RAMPANT Release 4.0. Centerra Resource Park , 10 Cavendish Court lebanon, NH 0376, Fluent Inc., 1996.

[34] P. Emvin. Installation and validation of the unstructured flow simulation package FLUENT together with the parametric mesh generator AML. Report 9970-1907, Volvo Aero Corporation, 1996.

[35] J. Kim, S.J. Kline, and P. Johnston. Investigation of separation and reattatchment of a turbulent shear layer: flow and the structure behind a backwards-facing step. Md 37, Thermosciences Division, Dept. of Mech. Eng., Stanford University, 1978.

[36] J. K. Eaton and J. P. Johnston. Turbulent flow reattatchment: an experimental study of the flow and the structure behind a backwards-facing step. Md 39, Thermosciences Division, Dept. of Mech. Eng., Stanford University, 1980.

[37] N. Kasagi, S. Kawara, and A. Matsunaga. Turbulence measurement in a separated and attatching flow over a backwards-facing step with the aid of a three-dimensional partikle tracking velocimetry,. In *The symposium of the society of instrument and control engineers*, 1991.