Development and implementation of a fast implicit multigrid method for large eddy simulations.

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

Abstract

The present work presents a fast numerical procedure for large eddy simulations. This is an implicit fractional step method with a geometric multigrid pressure Poisson solver. It was found to be 20 times faster on a one million-node mesh than an explicit fractional step method with an incomplete Cholesky preconditioned conjugate gradient pressure Poisson solver. The flow in a ventilated enclosure is studied using the dynamic subgrid model.

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

Contents

1 Nomenclature

A	area of a control volume face
d	a non-diagonal nonzero coefficient in the discretized equation
k	turbulent kinetic energy
p	pressure
Re	Reynolds number
S_{ij}	strain rate tensor
Ar	cell aspect ratio
u_i	velocity vector
N	Number of nodes
C	Subgrid diffusion coefficient

Greek letters

μ	diffusion coefficient
ϵ	residual/initial residual
ν	kinematic viscosity
ρ	$\epsilon_{i+1}/\epsilon_i$ as $i \to \infty$
ρ_d	density
τ_{ij}	stress tensor
Δ	mesh size

2 Introduction

The Navier-Stokes equations are traditionally studied in their time-averaged form, often referred to as the Reynolds averaged Navier-Stokes equations (RNS). The resulting unknown stresses, $\overline{u_i u_j}$, stemming from the time averaging must be modeled with a RNS turbulence model. The most common turbulence model is the two-equation $k-\varepsilon$ model [1]. In industry today, the $k-\varepsilon$ model is routinely used and the model is included in most commercial CFD packages. The advantage of the $k-\varepsilon$ model from a numerical point of view is that it is robust and reliable. From a physical point of view, the physics is treated in a simplistic manner, but the model nevertheless works surprisingly well in many types of flows.

However, there are physical phenomena that eddy viscosity models such as the $k-\varepsilon$ model cannot capture, such as streamline curvature and the effect of irrotational strains, although these phenomena can be accounted for in full Reynolds Stress Models. However, they probably all fail for flow cases with a few distinct low frequencies. It is likely that these types of flows can be successfully predicted by large eddy simulations (LES), even on quite coarse meshes ($\sim 10^5$ nodes).

Since LES is consistent with the Naver-Stokes equations in the limit of zero mesh size, it is always possible to find a critical mesh size at which the LES methods performs better than all RNS methods. However, it seems as though this mesh size

is often quite small, resulting in meshes with 10^6 nodes or more to obtain better results with LES than with RNS turbulence models [2].

The small three-dimensional scales are convected far from their origin, and thus we need a sufficiently fine mesh in the whole domain in contrast to RNS models. In boundary layers, there exist scales that seem more important to resolve than others, however, and thus it is preferable to compress the mesh normal to walls, resulting in cell aspect ratios of 10 or higher [2].

Having concluded that a fine 3D mesh is required, it is found that we also need a fine time advancement. Even with a highly efficient numerical algorithm, LES is considerably more expensive than a steady-state turbulence models such as $k-\epsilon$, which forces us to use the most accurate and fastest algorithm possible. We therefore present a fully implicit second order discretization with a multigrid PPE solver that is 10-100 times faster than the explicit fractional step method in [2, 3].

3 Basic Equations

With a spatial, inhomogeneous filter (denoted by a bar) applied to the incompressible Navier-Stokes equations, we obtain the momentum and continuity equations for the large-scale motion. If the filter size is larger than the Kolmogorov scales, we need to model the non-resolved scales. We here use the dynamic sub-grid model proposed by Germano *et al.* [4, 5], where the constant relating the second moments to the strain tensor is not arbitrarily chosen but computed. It reads:

$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\bar{u}_i \bar{u}_j\right) = -\frac{1}{\rho_d}\frac{\partial \bar{p}}{\partial x_i} + \nu \frac{\partial^2 \bar{u}_i}{\partial x_j \partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j} \tag{1}$$

$$\frac{\partial \bar{u}_i}{\partial x_i} = 0 \tag{2}$$

where the sub grid stresses are given by

$$\tau_{ij} - \frac{1}{3}\delta_{ij}\tau_{kk} = -2C\Delta^2 |\bar{S}| \bar{S}_{ij}$$

where

$$\bar{S}_{ij} = \frac{1}{2}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right), \quad |\bar{S}| = \sqrt{2\bar{S}_{ij}\bar{S}_{ij}}, \quad C = \frac{\mathcal{L}_{ij} M_{ij}}{2 M_{ij} M_{ij}}$$

and

$$M_{ij} = -\left(\widehat{\Delta}^2 |\widehat{\bar{S}}| \widehat{\bar{S}}_{ij} - \Delta^2 \widehat{|\bar{S}| \bar{S}_{ij}}\right)$$

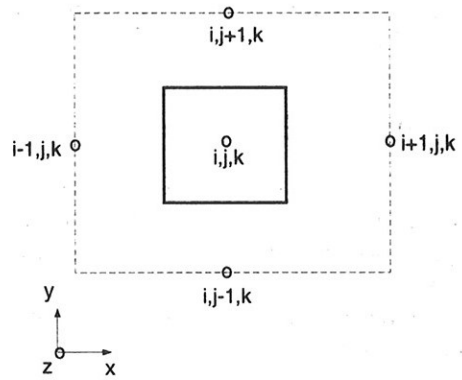

Figure 1: Grid cell (solid lines) and test cell (dashed lines).

$$\mathcal{L}_{ij} - \frac{1}{3}\delta_{ij}\mathcal{L}_{kk} = -2C\left(\widehat{\Delta}^2 |\widehat{\bar{S}}|\widehat{\bar{S}}_{ij} - \Delta^2 \widehat{|\bar{S}|\bar{S}_{ij}}\right)$$

$$\widehat{\bar{S}}_{ij} = \frac{1}{2}\left(\frac{\partial \widehat{\bar{u}}_i}{\partial x_j} + \frac{\partial \widehat{\bar{u}}_j}{\partial x_i}\right), \quad |\widehat{\bar{S}}| = \sqrt{2\widehat{\bar{S}}_{ij}\widehat{\bar{S}}_{ij}}$$

Following Germano *et al.* the ratio between the size of the test filter and the grid filter is set to two, i.e. $\widehat{\Delta}/\Delta = 2$ (see Fig. 1). The filtering procedure at the test level is carried out by integrating over the test cell assuming linear variation of the variables [6]. This is in fact the same stencil as the full weighting stencil used in the projection of a solution on a doubly as coarse grid in finite difference multigrid. The estimate of C is then averaged locally and span-wise (for more details see [2]).

We used a consistent pressure Poisson equation (PPE) rather than the continuity equation. The PPE is achieved by taking the divergence of the Navier-Stokes equations. This is referred to as a consistent PPE [7] and has the nice property that imposed divergence on the velocity field should decay in time. This may be important since we do not want to converge the PPE more than to the level of the truncation error. That would impose a divergence error on each time step, and thus it is important that no accumulation of divergence is possible. It was confirmed in our simulations that the divergence errors do not increase with time.

The consistent pressure Poisson equation reads:

$$\frac{\partial^2 \bar{p}}{\partial x_j \partial x_j} = -\frac{\partial^2 \bar{u}_i}{\partial x_i \partial_t} - \frac{\partial^2 (\bar{u}_i \bar{u}_j)}{\partial x_i \partial x_j} + \nu \frac{\partial^3 \bar{u}_i}{\partial x_i \partial x_j \partial x_j} - \frac{\partial^2 \tau_{ij}}{\partial x_i \partial x_j} \tag{3}$$

We have used the boundary condition $\partial\bar{p}/\partial n = 0$, approximating the physical correct one sufficiently well in most applications [7].

4 Discretization

The space discretization of Eq. 1 and Eq. 3 is a central difference finite volume method on a collocated grid, and the time discretization is the Crank-Nicholson scheme [8, 9] . In our previous explicit fractional step method, we used the Adam-Bashford time discretization [2].

In both the old formulation and the present one, the PPE is discretized via a fractional step method, reading:

$$\bar{u}_i^* = \bar{u}_i + \frac{\partial \bar{p}}{\partial x_i} \tag{4}$$

$$\frac{\partial^2 \bar{p}}{\partial x_j \partial x_j} = \frac{\partial \bar{u^*}_i}{\partial x_i} \tag{5}$$

As we use a zero Neumann boundary condition, we must satisfy the compatibility condition of the PPE. In the fractional step decomposition, it is satisfied if $\bar{u}_i^*$ (interpolated to the cell faces) satisfies global continuity, which we strongly enforce.

This method will be able to take significantly longer time-steps ($\sim$ 100 times) as compared with a fully explicit method (fully compressible formulation) where the time-step must be based on the speed of the acoustics. However, the success of a fractional step method will greatly depend on the effectiveness of the PPE matrix solver. This will be investigated in the next section.

When using an explicit velocity formulation, however, the CFL number based on convection and diffusion should be below 1. It turned out that it was necessary in practice to have CFL less then 0.3 everywhere to obtain numerical stability with the dynamic subgrid model [2, 3].

This may be a severe restriction if there are different scales in different parts of the domain or if the mesh is stretched. In terms of accuracy, however, it is possible to take larger time steps than $CFL = 1$, and we have experienced that the use of an implicit formulation allows us to use $CFL = 2$ without any degeneration of accuracy, which is the same as the experience reported in [10]. They claim that it is possible to use $CFL = 2 - 3$ without losing the accuracy for a second order time and space discretization.

The properties of the discretized problem should not be affected significantly if we use $CFL = 2$ instead of $CFL = 0.3$. Therefore the matrices resulting from discretizing the velocity-equations should converge within a few sweeps with a relaxation method, and the feed-back from the PPE to the velocity equations should be small.

These assumptions were verified on a 500 000 node mesh where, on each time step with ($CFL = 2$) the velocity equations converged to double machine precision within seven symmetric point Gauss-Seidel (spGS) sweeps. When an efficient matrix solver was used for the PPE, solving the PPE exactly in each iteration, it turned out that the whole system converged a factor of five for each global iteration. That confirms our assumption of weak feedback between the pressure and the velocities. Thus there is no use in reducing the error by more than a factor of five to ten in each sweep for each equation within each global iteration.

That is easily achieved by one to two spGS sweeps for the velocities but it is much more complicated to reduce the pressure equation by a factor of five. It is sufficient to converge the solution to the level of truncation errors, so converging the solution two decades will probably be sufficient in our cases. This means that 3-4 global iterations would be enough.

It was found that performing 3×4 (three velocity components) pGS sweeps takes approximately the same time as calculating the sub-grid viscosity, so it is of paramount interest to have as fast a pressure solver as possible.

5 Solution procedure of the pressure Poisson equation

LES calculations are very time consuming and it is therefore crucial to use the most efficient PPE solver possible. We thus compared, a geometrical multigrid method (GMG) with a modified incomplete Cholesky preconditioned conjugate gradient method (MIC-CG) and with an algebraic multigrid method (AMG) for some typical LES applications, i.e. for structured grids with varying cell aspect ratios.

The MIC-CG solver is a part of the `SLAP` package (Sparse Linear Algebra Package) available on `netlib` and was implemented by Renard and Gresser [11]. They found that the choice of preconditioner was important and that the incomplete Cholesky factorization was most efficient. The incomplete Cholesky factorization is fairly expensive in terms of CPU but, fortunately, the coefficients in the matrix stemming from the discretized Poisson equation (Eq. 7) are constants, which means that the preconditioner need to be applied only at the first time step.

The AMG solver is available from the database `mgnet.casper.yale.edu`. This is a black-box routine to solve a sparse system of algebraic equations and was applied directly to the PPE.

The GMG method for the Poisson equation is well developed and documented [12] [13]. The multigrid method is based on the fact that only N scales are representable on a N-node grid and that the most simple iterative methods (smoothers) are locally averaging, which quickly smooth an initial approximation. With the use of a sequence of grids with different grid sizes, all error components will be viewed as oscillatory on one of the grids and be reduced effectively by the smoother.

We have constructed two model cases in order to compare these solvers. Case

1 is a uniform 3D mesh of a unit cube with a randomly generated source term. Case 2 is a $10 \times 4 \times 2.4m$ domain with a $(4 * ii) \times (2 * ii) \times (ii)$ stretched mesh with aspect ratios of over 10 at all walls. This is a case in which where the GMG shows its worst behaviour, and it is also a typical LES application in ventilation [2, 3]. We used zero Neumann boundary conditions in both cases so the randomly generated source term is modified to satisfy the compatibility condition of the Neumann problem.

5.1 Construction of the 3D multigrid method

In LES we have a pressure field from the previous time step serving as an initial guess and thus we use only the MG and not the FMG method. Our coarse grid is obtained by merging eight fine grid cells into one coarse grid cell. Note that the nodes of the coarse grid are not a subset of the fine grid's nodes since we place the nodes in the geometric centre of the cells. We choose linear projections for both restriction and prolongation. In LES we can expect smooth meshes, and thus we use fixed weights in the projections as non-fixed weights will simply imply extra work.

The smoother must be designed carefully, since we have Neumann boundary conditions and we may have quite high cell aspect ratios, $Ar \sim 10 - 100$. The proper way to treat the Neumann boundary condition for finite differences is to reduce the stencil at the boundary using the boundary condition [12]. In a finite volume context, this corresponds to specifying the diffusive flux at the boundary. When doing so, we achieved for Case 2, with $ii = 16, log(\epsilon) = -10$, the asymptotic convergence factor $\rho = 0.15$. For the same case with homogeneous Dirichlet conditions instead of the Neumann conditions, we achieved an only slightly higher convergence rate ($\rho = 0.06$). When the Neumann boundary condition was imposed explicitly after each smoothing sweep, we noticed a significant slowdown ($\rho = 0.7$), probably due to bad smoothing near the boundary.

The second problem addressed earlier was how to treat aspect ratios. The aspect ratio enters the Poisson equation as anisotropic diffusion with $\mu_{11}/\mu_{22} = (Ar)^2$ and thus an aspect ratio of five will act as an anisotropy of 25. This results in bad smoothing in the direction of weak couplings [13]. A remedy for that is to coarsen the mesh only in the geometrically smooth directions of the residual. These directions vary in the domain, however, since there are high aspect ratios at each wall, making this concept difficult to realize within a geometrical multigrid. Instead, strongly connected points can simultaneously be relaxed, which provides good smoothing. This is made by using alternating plane relaxation, where each (y,z)-plane is first solved exactly, followed by each (x,z)-plane and each (x,y)-plane.

To solve such a 2D plane efficiently, we constructed a 2D multigrid, see Fig. 2. To be able to construct such a 2D multigrid, we must first note that what we actually wish to solve is a 3D problem, where the extent of the domain in the normal direction is one cell. In the normal direction, we have non-homogeneous Neumann boundary conditions. Note that the physical thickness of this 3D slice

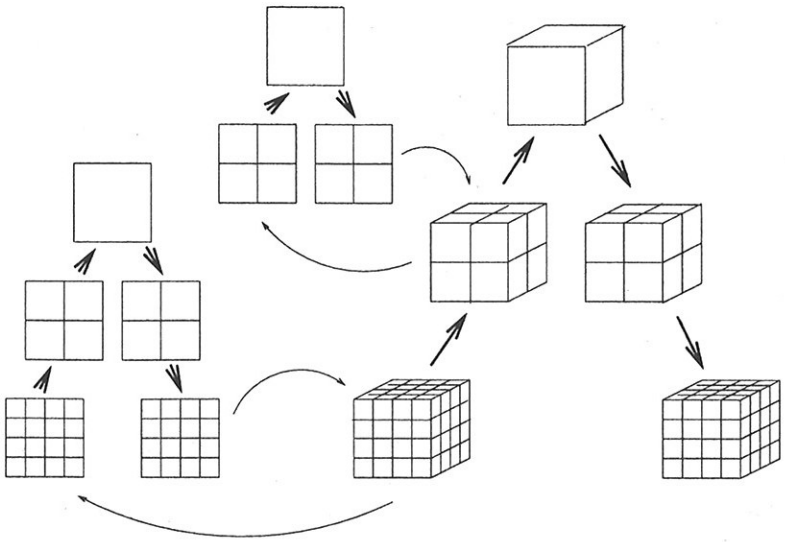

Figure 2: 3D-alternating plane GS multigrid

is constant throughout the 2D cycle.

A coefficient C in a finite volume discretization of the diffusion equation reads $C \sim \mu A/d$, where A is the face area, μ the diffusivity and d the distance between the two adjacent cell centres. We have no coarse grid operators (coefficients) for this quasi 2D-problem so they are constructed algebraically. Define first a coarse cell as the union of four fine cells. Second, note that the coefficients tangential to the plane are invariant with respect to coarsening on a non-stretched mesh, as both d and A increase by a factor of two when coarsening the mesh. To account for the stretching we average the two corresponding fine grid coefficients when defining the coarse grid coefficient. However, the coefficients normal to the plane correspond to a flux boundary condition whereby d remains the same on the fine and the coarse grid while A increases by a factor of four. Thus the coarse grid coefficient is achieved by summing the four corresponding fine grid coefficients together. Note that this corresponds to a pressure correction diffusion flux conservation in the normal direction.

Note that the 2D multigrid is performed for each plane on each level in the 3D multigrid but the number of levels in the 2D multigrid varies depending on the level at which we are in the 3D multigrid, see Fig. 2.

The smoother in the 2D multigrid is alternating line Gauss-Seidel, which has good smoothing properties for a stretched 2D grid and is performed by first a x-line Gauss-Seidel sweep followed by a y-line Gauss-Seidel sweep.

In this 2D multigrid we use linear restriction and prolongation analogously as in the 3D multigrid. We attempted to use injection for the prolongation operator (in both the 2D and the 3D multigrids) which in our case should correspond to the

pre,post	V-cycle		F-cycle		W-cycle	
relaxation	CPU-time	ρ	CPU-time	ρ	CPU-time	ρ
$(1,1)^*$	4.1	0.22	4.6	0.22	5.2	0.22
$(1,0)$	4.1	0.25	4.3	0.23	4.8	0.23
$(1,1)$	5.4	0.12	5.1	0.08	5.3	0.08
$(2,1)$	6.3	0.07	6.1	0.04	6.3	0.04

Table 1: Comparison of different cycles for Case 1 $log(\epsilon) = -10$ on the 32^3 mesh.

transpose of the restriction operator. However, for Case 2, the convergence rate was reduced from $\rho = 0.15$ to $\rho = 0.51$. Similar behaviour was observed when the same projection operators were applied to the Navier-Stokes equations [14]. Another choice would be to select the restriction operator as the transpose of the linear prolongation operator, but this choice also demonstrated a poorer performance ($\rho = 0.35$). We are unable to offer an explanation for this, but instead accept the fact and use the linear transfer operators.

We found that the number of cycles required in the 3D multigrid was almost independent of the type and the number of 2D cycles used. We therefore chose the cheapest alternative, which is $V(1,0)$, i.e. a V-cycle with one pre-smoothing sweep and zero post-smoothing sweeps.

The cycles tested (in the 3D GMG) were W-cycle, F-cycle and V-cycle. The most efficient cycle we found was $V(1,1)^*$, which is $V(1,1)$ without post-smoothing at the finest level, see Tables 1-2. The $V(1,1)^*$ and $V(1,0)$ have almost the same asymptotic behaviour but the first has a superior initial behaviour the first cycles. This is very important because, in practical calculations, we reduce the residual by only about two magnitudes.

It is possible on an isotropic mesh to use almost any type of smoother. In Case 1, with a $V(1,1)^*$ cycle, a symmetric point GS smoother had $\rho = 0.22$, an alternating line GS smoother had $\rho = 0.10$ and an alternating plane GS smoother $\rho = 0.04$. To do one smoothing sweep with alternating line GS is roughly three times as expensive as one symmetric GS sweep, however, and alternating plane GS is almost seven times more expensive than a symmetric pointGS, so it is obviously most efficient to use the point GS smoother for this case.

In practice, we usually have quite high aspect ratios in all directions in different parts of the domain which force us to use the more expensive alternating plane GS.

pre,post	V-cycle		F-cycle		W-cycle	
relaxation	CPU-time	ρ	CPU-time	ρ	CPU-time	ρ
$(1,1)^*$	20.2	0.15	23.5	0.16	24.5	0.15
$(1,0)$	20.2	0.17	22.2	0.18	23.3	0.17
$(1,1)$	29.8	0.10	25.1	0.05	28.0	0.05
$(2,1)$	40.1	0.08	33.1	0.035	35	0.035

Table 2: Comparison of different cycles for Case 2 $log(\epsilon) = -10$ on the $ii = 16$ mesh.

5.2 Comparison of the different solution methods.

In general, full-multigrid FMG (multigrid + nested iteration) proves to be asymptotic optimal($work = O(N)$), while the multigrid method is slightly non-optimal ($O(NlogN)$). The algebraic multigrid should have the same properties as the geometric multigrid, while the MIC-CG should have a $O(N^{1.17})$ dependency for the Dirichlet problem.

For the Neumann problem, however, it was found that the MIC-CG had $O(N^{1.33})$ dependency. The AMG solver also proved to have a slight N behaviour, while the geometric multigrid had grid independent convergence. Table 4 shows that the GMG is four times faster than the MIC-CG on the 128 x 64 x 32 mesh of Case 2. If the asymptotic behaviour is taken into consideration, the GMG is eight times faster than the MIC-CG method on a 256 x 128 x 64 mesh. This is significant, as the last mesh is relevant for LES.

This comparison was based on the asymptotic behaviour, although in practice we are interested only in the convergence rate the first two decades. Table 5 shows that the MIC-CG has a poorer initial behaviour than asymptotic behaviour, while the GMG has the same initial convergence rate as the asymptotic convergence rate. Therefore, in Case 2 on the 128 x 64 x 32 mesh, GMG is eight times faster than MIC-CG and, when extrapolating Table 5 on a 256 x 128 x 64 mesh, the GMG is 16 times faster than the MIC-CG, reducing the first two decades of error.

The GMG is even more superior on the unit cube with a uniform mesh size as compared with the MIC-CG because it is possible here to use a cheaper smoother, see Table 3. It should also be mentioned that the GMG needs only ten words per node work space while the MIC-CG uses 20. The AMG method, at the other hand, uses approximately 100 words per node, which is twice as much as the whole LES code uses, thus making AMG of limited interest. Furthermore, the AMG was two times slower than the GMG, see Tables 4-5.

	MG		AMG		MIC-CG	
	CPU-time	cycles	CPU-time	cycles	CPU-time	iterations
8^3	0.088	14	0.070	10	0.045	22
16^3	0.522	14	1.12	11	0.979	43
32^3	4.06	14	15.9	14	20.6	83
64^3	32.0	14	185	20	353	162

Table 3: Comparison of work for Case 1 $log(\epsilon) = -10$ using geometric multigrid, algebraic multigrid and the incomplete Cholesky preconditioned conjugate gradient method.

ii	MG		AMG		MIC-CG	
	CPU-time	cycles	CPU-time	cycles	CPU-time	iterations
4	0.44	12	0.89	15	0.070	36
8	2.11	13	3.78	16	1.60	78
16	22.2	14	29.0	22	38.6	164
32	180	14	360	26	733	334

Table 4: Comparison of work for Case 2 $log(\epsilon) = -10$ using geometric multigrid, algebraic multigrid and the incomplete Cholesky preconditioned conjugate gradient method.

ii	MG		AMG		MIC-CG	
	CPU-time	cycles	CPU-time	cycles	CPU-time	iterations
4	0.07	3	0.18	3	0.028	15
8	0.55	3	0.71	3	0.759	37
16	4.35	3	6.3	4	17.4	74
32	38.8	3	77	5	307	140

Table 5: Comparison of work for Case 2 $log(\epsilon) = -2$ using geometric multigrid, algebraic multigrid and the incomplete Cholesky preconditioned conjugate gradient method.

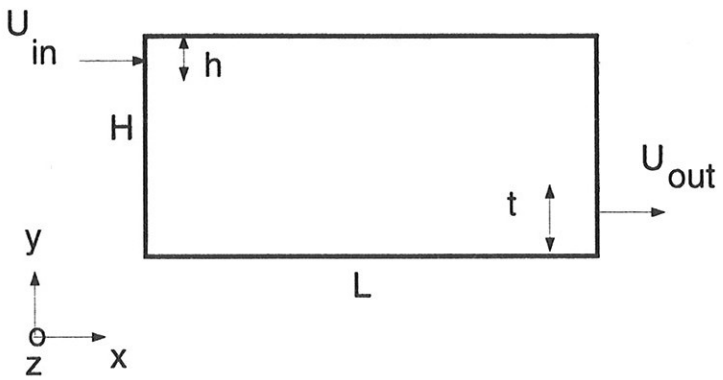

Figure 3: Ventilated room.

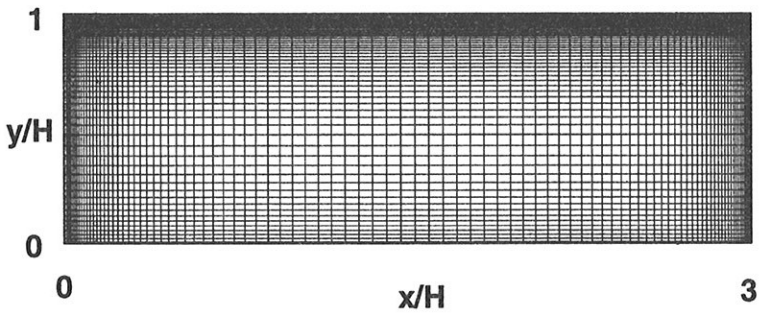

Figure 4: Grid in the $x - y$ plane.

6 Results: ventilated room

This section presents LES predictions of the flow in a three-dimensional room, Fig. 3. Unless otherwise stated, the results are obtained with a CFL number of approximately two, and we use values at all time steps when averaging. The predictions are compared with Laser-Doppler measurements of Restivo [15] (also available in [16]). Inlet boundary conditions and the geometry are given by:

$$L/H = 3,\ W/H = 1,\ h/H = 0.056,\ t/H = 0.16,\ Re = \frac{U_{in}h}{\nu}$$

where $H = 3$ m, $U_{in} = 0.455$ m/s, and air of 20°C.

Since we resolve the large-scale turbulence, we should impose turbulent flow conditions at the inlet. As an approximation, random fluctuations are superimposed on the time-averaged experimental flow conditions at the inlet according

$\Delta x_{min}/H$	$\Delta y_{min,f}/H$	$\Delta y_{min,c}/H$	$\Delta z_{min}/H$	$\Delta x_{max}/H$	$\Delta y_{max}/H$	$\Delta z_{max}/H$
0.0042	0.0076	0.00067	0.0034	0.061	0.042	0.025

Table 6: Geometrical details of the mesh. The *min* distances are from the near-wall node to the boundary. Indices c and f denote ceiling and floor, respectively.

to:

$$\bar{u}_{in} = U_{in} + random \cdot u'_{exp},\ \bar{v}_{in} = random \cdot u'_{exp},\ \bar{w}_{in} = random \cdot u'_{exp} \qquad (6)$$

Note that the random function is called at different times $\bar{u}_{in}$, $\bar{v}_{in}$ and $\bar{w}_{in}$, which means that the resolved velocities, $\bar{u}_{in}$, $\bar{v}_{in}$ and $\bar{w}_{in}$, at the inlet fluctuations are not correlated. Thus the shear stresses $\overline{\bar{u}\bar{v}}$, $\overline{\bar{u}\bar{w}}$ and $\overline{\bar{v}\bar{w}}$, are approximately zero at the inlet. The exit velocity at the outlet is computed from global continuity, and it is taken as constant over the outlet. Zero gradient is set for the remaining variables.

A steady computation is first carried out using the $k-\varepsilon$ model, and the results are used as initial start fields in the LES calculations. The number of time steps used in the calculations was 25000 when using $CFL = 2$, which corresponds to approximately 1300 seconds in real time. The stream-wise average of the peak velocity in the wall jet along the ceiling is close to $U_{av} = 0.5U_{in}$ ($= 0.228$ m/s). Thus the time it takes for a fluid particle to move from the inlet to the opposite wall can be estimated as $L/U_{av} \simeq 40$ seconds, which means that $1300/40 = 33$ characteristic time units (L/U_{av}) are covered in our simulations.

We used a $98 \times 66 \times 66$ mesh ($96 \times 64 \times 64$ interior cells), see Fig. 4. This means that the multigrid solver used for the pressure employs six multigrid levels. A geometrical stretching is used in the y-direction, and a hyperbolic tangent function is used in the x and the z-directions. These are summarized in Table 6.

In [2, 3], we found it necessary to place limits on C in Eq. 3 and to perform local and span-wise averaging in order to achieve numerical stability. Furthermore, the total viscosity (laminar plus turbulent) is not allowed to become negative, but we do not use a maximum limit on C. The same procedure was employed in this study and is similar to that chosen in [17]. The cross-diffusion terms are neglected in these simulations because they were causing problems with the dynamic subgrid model. These problems were present for all solution algorithms. However, they were not present when using a Smagorinsky model or a dynamic one-equation model [18].

6.1 Comparisons of the numerical methods for the ventilated enclosure

This section presents the speedup of our implicit fractional step method with the GMG PPE solver, compared to our explicit fractional step method with the MIC-

		CPU/time-step			
	CFL	U+V+W	μ_t	P	total
implicit-GMG	2	44	14	36	94
explicit-MICCG	0.3	6	14	180	200
explicit-GMG	0.3	6	14	25	45

Table 7: Time measuring of the parts in the implicit-GMG and in the explicit-MICCG algorithm. (For the implicit method we used three global iterations at each time step.)

CG PPE solver for the ventilated enclosure previously described. In the previous sections, we have seen the potential of this new method and we expect the same performance for this LES application. It is seen in Table 7 that the GMG PPE solver is about eight times faster than the MIC-CG PPE solver when $CFL = 0.3$. There is also also shown that the implicit GMG formulation requires only twice the CPU time per time step as does the explicit, even though the time step is 6.7 times longer in the implicit case.

The implicit method is also significantly more robust and converges nicely for $CFL > 5$, while the explicit method works only for $CFL \leq 0.3$.

Fig. 6- 7 shows that the difference is much smaller than the errors in the solution when using $CFL = 2$ instead of $CFL = 1$, and thus we expect that even longer time-steps could be used, without sacrifice accuracy. We should point out that here CFL is the highest value in the domain. Note that in Fig. 6- 7 have we used a somewhat modified subgrid model (SWIDSM). That because, recently it was found that the dynamic subgrid model were somewhat wrongly implemented in the code where we applied this implicit multigrid solver. However, as we only had time to recalculate the $CFL = 2$ case and as the results not were dramatically changed (see Fig. 5-6) we here show the influence of the CFL-number for this modified problem instead. We thus expect these results to hold also for the correctly implemented dynamic subgrid model. All other result shown are made with the correctly implemented dynamic subgrid model.

The speedup factor between GMG and MIC-CG is likely to increase even more when the number of nodes is increased, because the CPU time for the multigrid method increases linearly with the number of nodes, while the MICCG was found to use $CPU \sim N^{1.33}$.

However, with this comparison on the 400 000 node mesh, we find that the implicit GMG PPE method is 14.4 times faster than the explicit MIC-CG PPE method. However, on a one million-node mesh, we would achieve a speedup of 20 times. This is **VERY** important, as we then avoid using CPU months on such calculations, and use only CPU days on a work station.

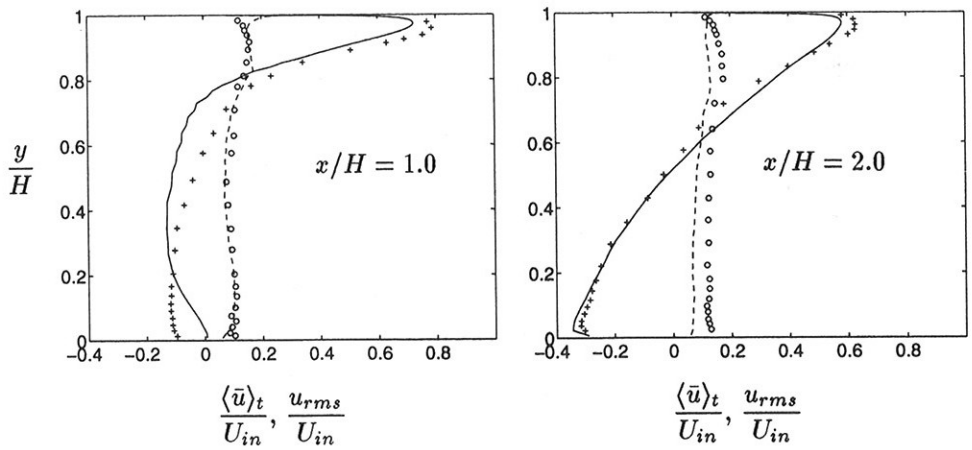

Figure 5: Time-averaged velocity and resolved rms velocity profiles. Symmetry plane $z/H = 0.5$. Solid lines: $\langle\bar{u}\rangle_t/U_{in}$; dashed lines: u_{rms}/U_{in}; +: experimental mean velocity; *o*: experimental fluctuations.

6.2 Results of the calculations

In Fig. 5, the time-averaged stream-wise velocities and resolved fluctuations are compared with experiments, showing fairly good agreement. The width of the wall jet seems to be under-predicted, and the magnitude of the reverse flow is too small. In general, the resolved rms fluctuations

$$u_{rms} = \sqrt{\langle (u''(t))^2 \rangle_t}, \quad u''(t) = \bar{u}(t) - \langle \bar{u}(t) \rangle_t$$

are under-predicted.

When making comparisons with experiments, we must time-average over a certain time, T, in the same way experimentalists measure over a certain time when they record a turbulent signal. The question is the length of time (T) required. In Fig. 8, we investigate the influence of T on the time-averaged $\bar{u}$ profile. In general, the $\langle\bar{u}\rangle_t$ does not seem to be very sensitive to the choice of T.

The resolved $\bar{u}$ velocities versus time at four chosen points are presented in Fig. 9. It can be seen that the fluctuations in $\bar{u}$ are strong. It can also be seen that the frequency of $\bar{u}$ is much higher in the wall jet near the ceiling (Figs. 9a,c) than in the middle of the room and in the back-flow region close to the floor (Fig. 9b,d).

Fig. 10 presents the C coefficient in the dynamic model (see Eq. 3). Fig. 10a shows the time history of C at one chosen point, in the wall jet close to the ceiling. The average of C in Fig. 10a is 0.048, which corresponds in the Smagorinsky model to

$$C_S = \sqrt{C} = 0.21$$

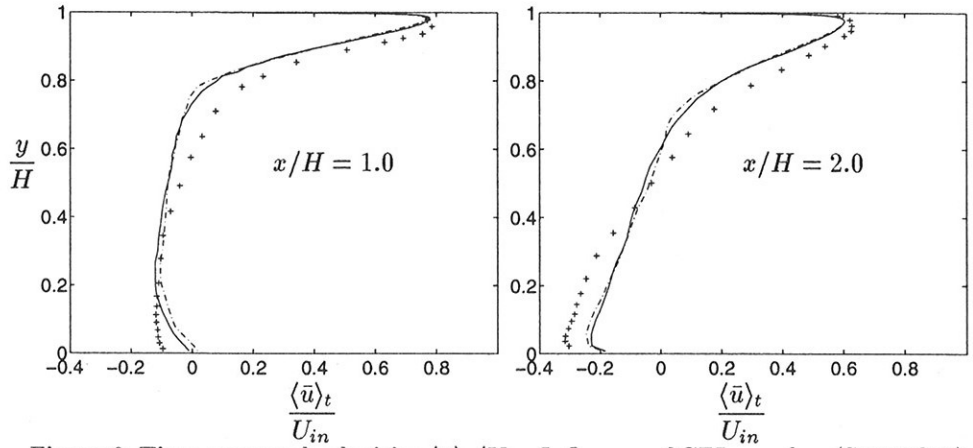

Figure 6: Time-averaged velocities $\langle \bar{u} \rangle_t / U_{in}$. Influence of CFL number (SWIDSM). Symmetry plane $z/H = 0.5$. Solid lines: $CFL = 2$; dashed lines: $CFL = 1$.

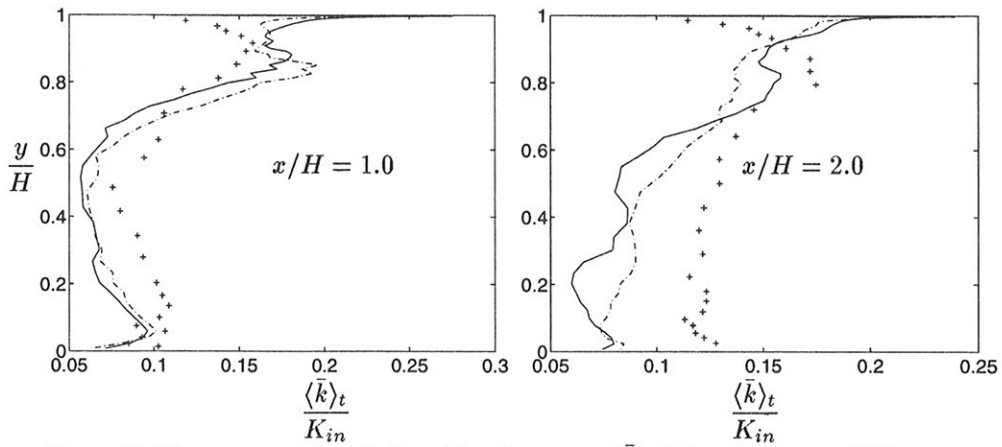

Figure 7: Time-averaged turbulent kinetic energy $\langle \bar{k} \rangle_t / K_{in}$. Influence of CFL number (SWIDSM). Symmetry plane $z/H = 0.5$. Solid lines: $CFL = 2$; dashed lines: $CFL = 1$.

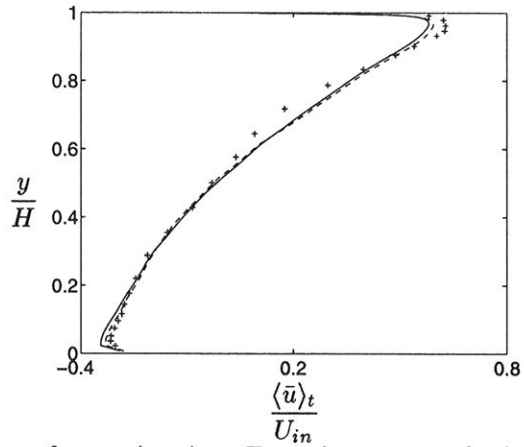

Figure 8: Influence of averaging time, T, on time-averaged velocity profiles. $CFL = 1$. Symmetry plane $z/H = 0.5$. +: experimental U/U_{in} velocity. Solid lines: $T = 1300$ seconds (25000 time steps); dashed lines: $T = 800$ seconds (the first 15000 time steps).

Fig. 10b shows the instantaneous C versus y. Fig. 11 shows the power density spectrum for the resolved stream-wise fluctuation $(u'')^2$. In fully turbulent flow, it should behave as $\Phi \propto n^{(-5/3)}$ (inertial region), which is included as a dashed line.

7 Conclusions

* The geometrical multigrid pressure Poisson solver is 10 times faster on a 10^6-node mesh than an incomplete Cholesky preconditioned conjugate gradient pressure Poisson solver.
* The implicit treatment of the Navier-Stokes equations allows CFL numbers 10 times larger than does the explicit without losing accuracy
* The overall speedup of this implicit multigrid solver is about 20 on a 10^6-node mesh.
* The flow in a ventilated room is predicted well with LES.

Acknowledgements

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

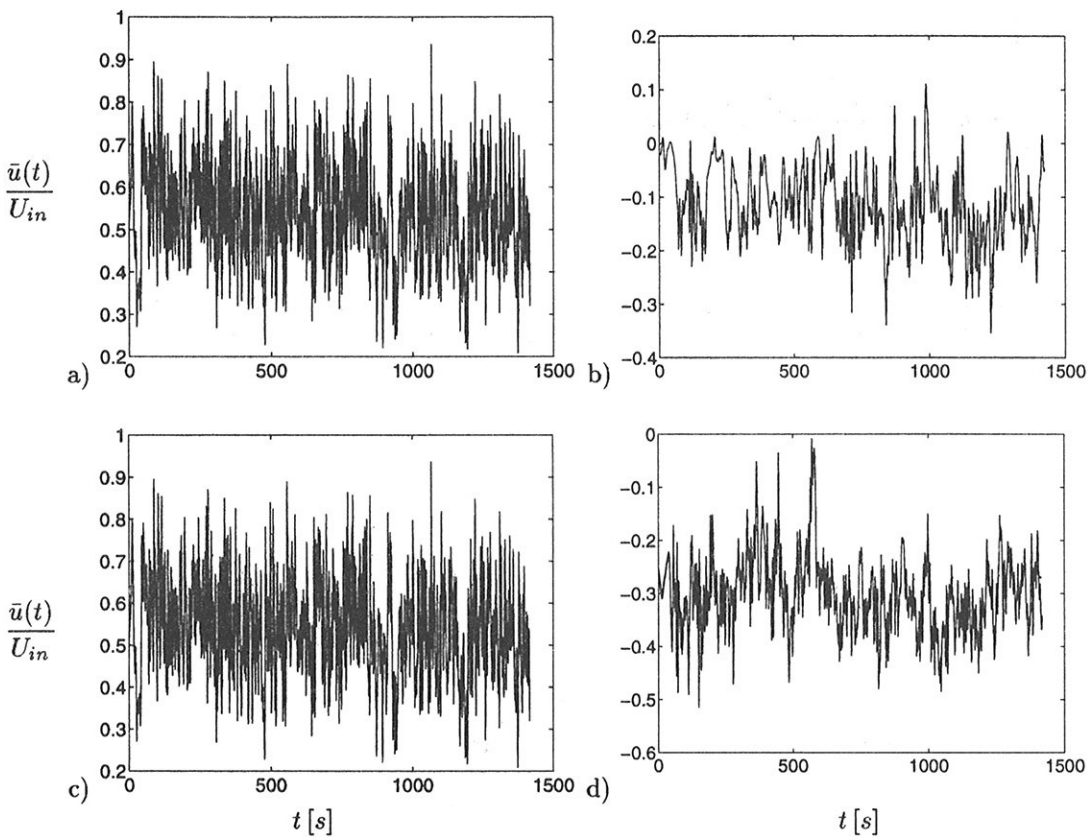

Figure 9: Time history of the $\bar{u}$ at four chosen cells. a) $x/H = 1.0, y/H = 0.92$, b) $x/H = 1.0, y/H = 0.5$, c) $x/H = 2.0, y/H = 0.92$, d) $x/H = 2.0, y/H = 0.14$.

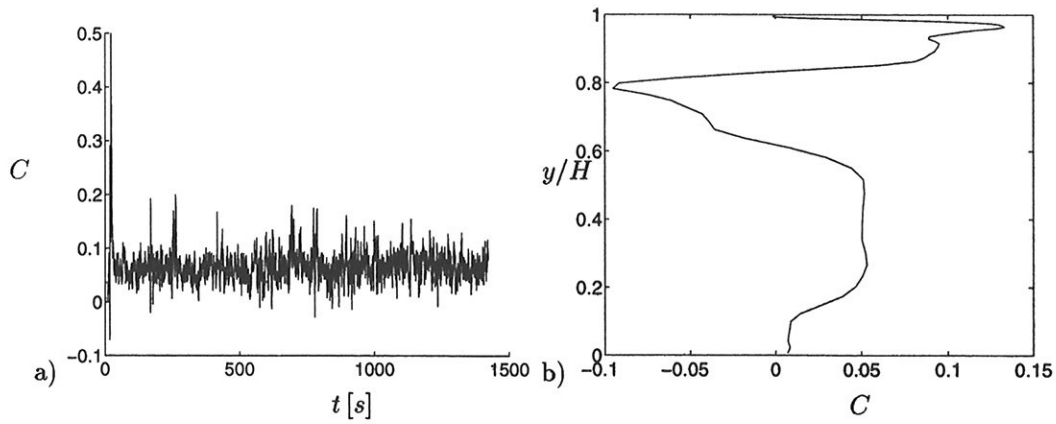

Figure 10: The C coefficient in the dynamic model. $CFL = 2$, $x/H = 1.0, z/H = 0.5$. a) Time history of C at one point. $y/H = 0.92$. b) Instantaneous vertical profile of C.

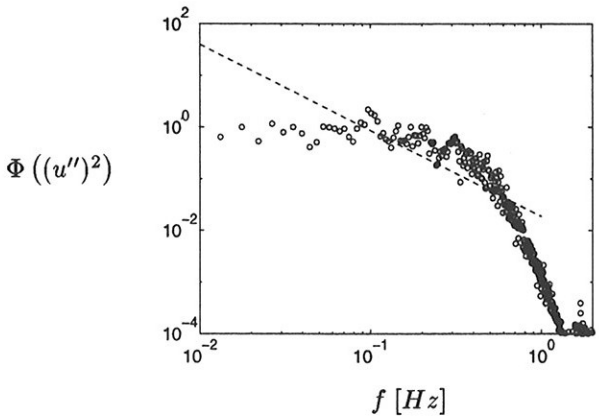

Figure 11: Power density spectrum $\Phi\left(\overline{(u'')^2}\right)$ $(z/H = 0.5.\ x/H = 1.0, y/H = 0.92)$.

References

[1] B.E. Launder and D.B. Spalding. The numerical computation of turbulent flows. *Computer Methods in Applied Mech. and Eng.*, 3:269–289, 1974.

[2] L. Davidson. Implementation of a large eddy simulation method applied to recirculating flow in a ventilated room. Report, ISSN 1395-7953 R9611, Dep. of Building Technology and Structural Engineering, Aalborg University, 1996.

[3] L. Davidson and P. Nielsen. Large eddy simulations of the flow in a three-dimensional ventilated room. In S. Murakami, editor, *5th Int. Conf. on Air Distributions in Rooms, ROOMVENT'96*, volume 2, pages 161–168, Yokohama, Japan, 1996.

[4] M. Germano, U. Piomelli, P. Moin, and W.H. Cabot. A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids A*, 3:1760–1765, 1991.

[5] M. Germano, U. Piomelli, P. Moin, and W.H. Cabot. Erratum. *Phys. Fluids A*, 3:3128, 1991.

[6] Y. Zang, R.L. Street, and J.R. Koseff. A dynamic mixed subgrid-scale model and its application to turbulent recirculating flows. *Phys. Fluids A*, 5:3186–3196, 1993.

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

[8] L. Davidson and B. Farhanieh. CALC-BFC: A finite-volume code employing collocated variable arrangement and cartesian velocity components for computation of fluid flow and heat transfer in complex three-dimensional geometries. Rept. 92/4, Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1992.

[9] S.V. Patankar. *Numerical Heat Transfer and Fluid Flow.* McGraw-Hill, New York, 1980.

[10] L. Choi and P. Moin. Effects of the computational time-step on numerical solutions of turbulent flow. *J. of Comp. Phys.*, 113:1–4, 1994.

[11] J. Renard and D. Gresser. Computational modelling of 2D hill flows. Diploma thesis. Rept. 95/6, Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1995.

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

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

[14] 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.

[15] A. Restivo. *Turbulent Flow in Ventilated Rooms.* PhD thesis, University of London, Imperial College of Science and Technology, Mechanical Engineering Department, 1979.

[16] P.V. Nielsen. Specification of a two-dimensional test case. Report, ISSN 0902-7513 R9040, Dept. of Building Technology and Structural Engineering,, Aalborg Universitetscenter, Aalborg, 1990.

[17] K.-S. Yang and J.H. Ferziger. Large-eddy simulation of turbulent obstacle flow using a dynamic subgrid-scale model. *AIAA J.*, 31:1406–1413, 1993.

[18] L. Davidson. Large Eddy Simulation: A dynamic one-equation model for three-dimensional recirculating flow. Report, Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1996.