Publikation Nr 92/5

CHALMERS TEKNISKA HÖGSKOLA
Institutionen för Termo- och Fluiddynamik

CHALMERS UNIVERSITY OF TECHNOLOGY
Department of Thermo- and Fluid Dynamics

A THREE-DIMENSIONAL LAMINAR MULTIGRID METHOD APPLIED TO THE SIMPLEC ALGORITHM

by

Peter Johansson

Examensarbete i Termo- och Fluid Dynamik

Göteborg, Juni 1992

First I would like to thank my supervisor Lars Davidson for all the help and support wich have been very important and helped me very much in this work.

I would also like to thank all the personel at the Department of Thermo- and Fluid Dynamics for their help and support to this Diploma Thesis.

Summary

The subject of this Diploma Thesis is a three-dimensional laminar multigrid implementation. The multigrid method is here used as a convergence accelerator for the system of nonlinear algebraic equations obtained from the discretization of the Reynolds avaraged Navier Stoke's equations.

The solution process is the SIMPLEC algorithm. A line Gauss-Seidel (TDMA) solver is used as a smoother for all equations.

For fine grids significantly computational savings have been achieved. The savings is of the order that a three-dimensional grid with half a million cells could be calculated over a night on a work station. But since the computer memory on the workstation used in the present study was to small for such domains the largest three-dimensional grid used here is a 40x40x40 grid.

For two-dimensional calculations has a 160x160 grid been used and the CPU-time was reduced by a factor 16 using multigrid instead of singlegrid.

Implementation of the multigrid algorithm for turbulent calcultations with the $k - \epsilon$ turbulence model is also started. But due to the nature of the k and the ϵ equations special care be taken and further reserch must be done before any calculations can be done.

Contents

1 Introduction

1.1 Preface

Except for some simple cases, calculations of flow problems are always bounded to numerical methods. These numerical methods can be based on finite elements, boundary elements, finite volumes etc.

To solve the system of algebraic equations in the finite volume method, a smoother used, for example Gauss-Seidel solver or Stone's Strongly Implicit Procedure (SIP). Smoothers like these have a high convergence rate for the first iterations but after a couple of iterations it is slower. Since the truncation error from the discretization process becomes smaller as the grid gets finer, a solution of high accuracy demands a fine grid.

Most smoothers needs more relaxation sweeps to smooth out the algebraic errors, as the mesh gets finer. Each relaxation sweep also takes longer time at a fine grid. The computer time increases therefore rapidly with an increased number of nodes. To reach a certain accuracy can therefore be prohibitively expensive. If the relaxation process could be speeded up, could thereby a higher accuracy be afforded.

One way to increase the convergence rate is multigrid. Multigrid uses a serie of grids applied to the same physical problem. These grids have different mesh sizes and they are in a nested way used to correct each other, to obtain a solution at the finest grid.

If this is done properly can the computing time be reduced by an order of a magnitude or two. The multigrid principles from the concept presented 1977 by Brandt[2] has been further developed by, Peric[4, 5, 6, 7], Fuchs[8, 9, 10] , Briggs[11],Lien [12] among others.

This present work is done with a finite volume based code CALC-BFC, devoloped at Department of Thermo and Fluid Dynamics by Davidson and Farhanieh [1]. CALC-BFC uses the SIMPLEC algorithm and has a collocated arrangement for all field quantities. As a smoother a line-Gauss-Seidel is used and boundary-fitted coordinates can be handled.

In CALC-BFC a multigrid code for two dimensional laminar flow situations had been implemented [14]. It was found that this multigrid code contained so many bugs and conceptual errors that a new multigrid code had to be written. This new two-dimensional laminar multigrid code was then extended to three dimensions and an implementation of the $k-\epsilon$ turbulence is started.

It should not be forgotten that the old multigrid code [14] has been a good help to the structure of the implementation in CALC-BFC. The same pointer system has also been used.

2 Equations

The following differential equations are discretized and calculated in the code CALC-BFC. They are first lined up and later represented with the general transport equation.

2.1 Basic Equations

Continuity equation:

$$\frac{\partial \rho}{\partial \tau} + \frac{\partial}{\partial x_i}(\rho U_i) = 0 \tag{1}$$

Reynolds averaged Navier Stokes Equation:

$$\frac{\partial}{\partial \tau}(\rho U_i) + \frac{\partial}{\partial x_j}(\rho U_j U_i) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left\{\mu_{eff}\frac{\partial U_i}{\partial x_j}\right\} \tag{2}$$

The $k-\varepsilon$ turbulence model

$$\frac{\partial}{\partial \tau}(\rho k)+\frac{\partial}{\partial x_i}(\rho U_i k)=\frac{\partial}{\partial x_i}\left\{\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_i}\right\}+P_k-\rho\epsilon \tag{3}$$

$$\frac{\partial}{\partial \tau}(\rho \epsilon)+\frac{\partial}{\partial x_i}(\rho U_i \epsilon)=\frac{\partial}{\partial x_i}\left\{\left(\mu+\frac{\mu_t}{\sigma_\epsilon}\right)\frac{\partial \epsilon}{\partial x_i}\right\}+\frac{\epsilon}{k}(C_{\epsilon 1}P_k-C_{\epsilon 2}\rho\epsilon) \tag{4}$$

$$P_k=\mu_t\frac{\partial U_j}{\partial x_i}\left\{\frac{\partial U_j}{\partial x_i}+\frac{\partial U_i}{\partial x_j}\right\} \tag{5}$$

$$\mu_t=C_\mu \rho k^2/\epsilon \tag{6}$$

$$\mu_{eff}=\mu_t+\mu \tag{7}$$

$\sigma_k=1;\qquad \sigma_\epsilon=1.3;\qquad C_{\epsilon 1}=1.44;\qquad C_\mu=0.09;\qquad C_{\epsilon 2}=1.92$

2.2 General Transport Equation

To represent the equations above the general transport equation is defined as:

$$\frac{\partial}{\partial \tau}(\rho\Phi)+\frac{\partial}{\partial x_j}(\rho U_j\Phi)=\frac{\partial}{\partial x_j}\left(\Gamma_\Phi\frac{\partial \Phi}{\partial x_j}\right)+S_\Phi \tag{8}$$

Table.1 shows the parameters in the general transport equation.

Define a flux vector J_j containing both convection and diffusion:

$$J_j=\rho U_j\Phi-\Gamma_\Phi\frac{\partial\Phi}{\partial x_j} \tag{9}$$

Then the general transport equation can then be written as:

$$\frac{\partial}{\partial \tau}(\rho\Phi)+\frac{\partial J_j}{\partial x_j}=S_\Phi \tag{10}$$

EQUATION	Φ	Γ	S
Continuity	1	0	0
Navier Stokes	U_i	μ_{eff}	$-\frac{\partial p}{\partial x_i}$
Kinetic energy	k	$\mu + \frac{\mu_t}{\sigma_k}$	$P_k - \rho\epsilon$
Dissipation	ϵ	$\mu + \frac{\mu_t}{\sigma_\epsilon}$	$\frac{\epsilon}{k}(C_{\epsilon 1}P_k - C_{\epsilon 2}\rho\epsilon)$

Table 1: The parameters in the general transport equation

Integrating this equation over a control volume, with volume V and surface area A ,using Gauss' law yields:

$$\int_V \frac{\partial}{\partial \tau}(\rho\Phi)dV + \int_A n_j J_j dA = \int_V S_\Phi dV \tag{11}$$

This is the equation that is discretized and solved for the different $\Phi's$ in Table 1.

3 The CALC-BFC code

3.1 Basics

In this section is the finite volume program CALC-BFC described briefly. For more details see [1]. CALC-BFC uses three dimensional boundary fitted coordinates. A collocated arrangement is used for all field quantities, i.e. all information is stored at the cell center. To aviod non-physical oscillations, a fourth order artificial pressure diffusion is introduced by Rhie and Chow interpolation. Cartesian velocity components are also used. The collocated arrangement for a 2D cartesian grid is showed in fig. 2.

Since only steady calculations are made in this report, only the steady part of CALC-BFC is presented. If equation 11 is discretized using a finite volume formulation it yields:

$$(n_j J_j A)_e + (n_j J_j A)_w + (n_j J_j A)_n + (n_j J_j A)_s + (n_j J_j A)_l + (n_j J_j A)_h = S\delta V \tag{12}$$

which can be written on the form:

$$a_P \Phi_P = \sum_{nb} a_{nb}\Phi_{nb} + S_C \tag{13}$$

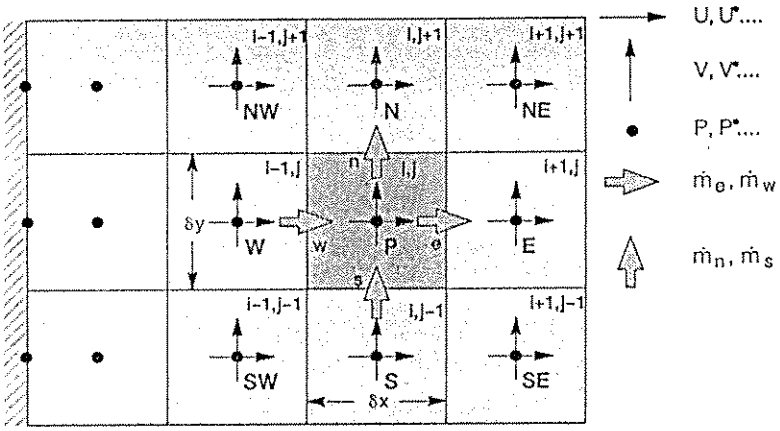

Figure 1: Grid nomenclature

where

$$a_P = \sum_{nb} a_{nb} - S_P \tag{14}$$

The coefficients a_{nb} contains contributions due to both convection and diffusion, and the source terms S_C and S_P contains the remaining terms.

3.2 Convection

The convective part of the flux vector multiplied with the facearea is defined as the massflux multiplied with the field variable.

$$(n_j J_j^{conv} A)_e = \dot{m}_e \Phi_e \tag{15}$$

The mass flux through an eastface is defined:

$$\dot{m}_e = \{\rho A(U n_x + V n_y + W n_z)\}_e \tag{16}$$

A is the area of the face and n_i is the east face normal vector. To avoid nonphysical oscillations Rhie and Chow interpolation is used for the face velocities. For an east face velocity U_e we obtain:

$$U_P^\circ = U_P - \frac{-(P_e - P_w)\delta V}{|we|(a_P)_P} \qquad and \qquad U_E^\circ = U_E - \frac{-(P_{ee} - P_e)\delta V}{|e(ee)|(a_P)_E} \tag{17}$$

$$U_e = f_x U_E^o + (1 - f_x) U_P^o - \frac{P_E - P_P}{|PE|(a_P)_e} \tag{18}$$

where

$$f_x = \frac{|Pe|}{|Pe| + |eE|} \tag{19}$$

P_e etc are calculated by linear interpolation. $|we|$ is the spatial distance between the west and the east face. CALC-BFC can calculate the field quantities used to calculate the convective fluxes with three different schemes. Only Hybrid Upwind/Central Differencing Scheme will be presented since it is the only scheme used in this report.

Hybrid Upwind/Central Differencing Scheme

$$\Phi_w = \begin{cases} \Phi_W & \text{if } |Re_{\delta x}| > 2 \quad \text{and } U_w > 0 \\ \Phi_P & \text{if } |Re_{\delta x}| > 2 \quad \text{and } U_w < 0 \\ f_x \Phi_P + (1 - f_x)\Phi_W & \text{if } |Re_{\delta x}| < 2 \end{cases} \tag{20}$$

3.3 Diffusion

The diffusiv part of the flux vector multiplied with a face area is defined:

$$(n_j J_j^{diff} A)_e = -\left\{ \Gamma_\Phi A n_j \frac{\partial \Phi}{\partial x_j} \right\}_e \tag{21}$$

If this equation is transformed into curvlinear coordinates with $(g_i)_m$ as the covariant base vectors and g^{ij} as the metric tensor it yields:

$$(n_j J_j^{diff} A)_e = -\left\{ \Gamma_\Phi A n_m (g_1)_m g^{1j} \frac{\partial \Phi}{\partial \xi_j} \right\}_e \tag{22}$$

since for an east face is $n_m(g_2)_m = n_m(g_3)_m = 0$. For more details, see [1].

3.4 Pressure correction equation

The pressure correction equation is obtained by applying the SIMPLEC algorithm, on the collocated grid. The mass flux is divided into one old value $\dot{m}^*$, and one correction value, $\dot{m}'$. The mass flux correction at the east face can be calculated by

$$\dot{m}'_e = (\rho A n_m (g^j)_m U'_j)_e \tag{23}$$

where $(g^j)_m$ are the contravariant basevectors. The covariant velocity correction components U'_j are related to the pressure gradient by:

$$U'_j = -\frac{\delta V}{a_P} \frac{\partial p'}{\partial x_j} \tag{24}$$

If these two equations are combined

$$\dot{m}'_e = \rho A_e (n_m \frac{-1}{a_P} \frac{\partial}{\partial x_j} (p')(g^j)_m) \tag{25}$$

The continuity equation reads:

$$\dot{m}_e - \dot{m}_w + \dot{m}_s - \dot{m}_n + \dot{m}_l - \dot{m}_h = 0 \tag{26}$$

If $\dot{m} = \dot{m}^* + \dot{m}'$ and eq.25 are substituted into eq.26 a diffusion equation is obtained. It can be solved in the same manner as the momentum equation if the convection is zero, $\Phi = p'$, and the source term is the mass flux imbalance for a control volume using $\dot{m}^*$.

3.5 The $k - \varepsilon$ turbulence model

The $k - \varepsilon$ turbulence model is discretized and calculated in the same way as the momentum equations. Only the coefficents and the source differ. Further information how this is implemented is described in [1].

4 Multigrid theory

4.1 Basics

The multigrid method was originally devoloped as an convergence accelerator for linear algebraic equations. These algebraic equations could arise from discretized linear differential equations, or discretized linearized nonlinear differential equations.

For linear problems, the first multigrid algorithm, the correction scheme (CS), was devoloped by Brandt. It was then extended to handle nonlinear problems, such as fluid dynamics, and presented as FAS (Fully Storage Approximation) 1977 by Brandt [2]. Before FAS and CS is described, is a short description of the basic iterative methods and their behaviour shown.

4.2 Definitions of errors

Define the discretized equation as:

$$Au = f \tag{27}$$

where A is a coefficient matrix, f is a sourceterm and u is the exact solution to the discrete problem. It can also be written as:

$$Av = f + r \tag{28}$$

where v is an approximation to u and r is the residual

If A depends on u then the discretized problem is nonlinear. The algebraic error e is defined as:

$$e = u - v \tag{29}$$

Let the exact solution to the differential equation be U. Then are the truncation error t defined by:

$$t = U - u \tag{30}$$

The total error E in any approximation v is therefore defined as:

$$E = t + e = U - v \tag{31}$$

It is important not to confuse these three errors. The truncation error depends on the gradients G, and the mesh size h like, $G^a h^b$. The only way to reduce the truncation error is therefore to reduce the mesh size (or use a higher order discretization scheme). The algebraic error on the other hand can be reduced with the number of relaxation sweeps. It is this rate of reducing the algebraic error wich can be speeded up with multigrid.

The relaxation process is strongly dependent of the number of nodes N in a characteristic direction. The number of iterations needed to reach a certain accuracy is $O(N^a)$ where a is around two for singlegrid calculations and close to zero for multigrid calculations.

Consider the computer time as limited when a solution as accurate as possible is needed. With multigrid the algebraic error is reduced faster and therefore a finer grid with lower truncation errors can be used. Therefore multigrid gives indirectly more accurate solutions.

For areas where the gradients are small, the reduction of truncation error using a finer grid is negligible, compared to the reduction in areas where the gradients are significant. Therefore is it a waste of computer time to refine domains with small gradients, and that is why local mesh refinement is so effective to get high accuracy for a low computing cost.

To measure the error matrix, a suitable norm on E or e can be used. But since neither u or U is known is it common to use the residual as a substitute for the algebraic error. The measure for the residual, is here choosen to be the sum of the absolute of the components in the residual matrix scaled with a flow characteristic.

The residual is defined as the imbalance of the discretized equation when the approximation v is used instead of the exact discrete solution u:

$$r = f - Av \tag{32}$$

It should be mentioned that if r is low e is normally also low. Fortunately if r is zero, e is always zero.

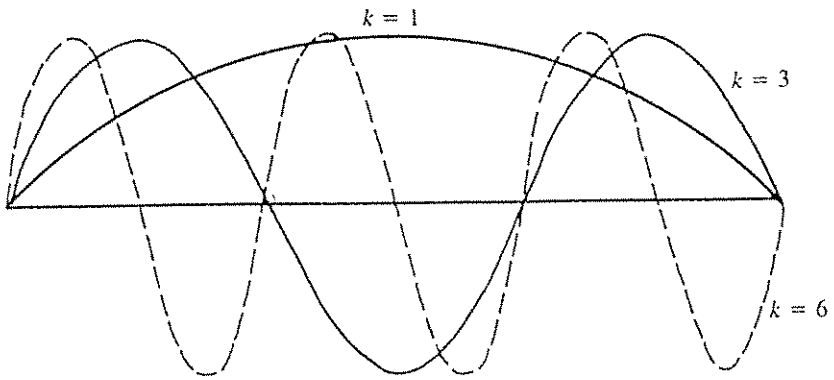

Figure 2: One-dimensional Fourier modes

4.3 Fourier expansion of the algebraic error

First consider an one-dimensional problem and expand the algebraic error e into Fourier series. For a one-dimensional grid with six nodes can the error e be expressed as $\sum_{k=1}^{6} A_k sin(\pi kj/6)$ where j is the node number and k is the wavenumber. Fig. 2 showes 3 of these error modes. Notice that it is the continous modes that are shown, i.e. $N \rightarrow \infty$..

The components with a wave length less then four times the mesh size will be called oscillatory modes, and the rest is called smooth modes. It can analougus be extended to higher dimensions. Most smoothers like Thomas, Gauss-Seidel or Jacobi have different smoothing properties for different error components. The oscillative modes are smoothed out rapidly by just a few iterations whereas the smoothest (largest wave length) components are almost not effected within 10-30 iterations.

Fig. 3 shows this typical behaviour for a one-dimensional model problem with a weighted-Jacobi smoother. Assume a discretization where $3a$ is concidered as a smooth mode and $3b$ as an oscillative error mode. The figures to the left show the initial errors and the figures to the right show the same error modes after ten relaxation sweeps. These two modes superponated are shown in $3c$.

Further information of the local mode Fourier analys se Brandt[2, 3] and Shaw[13]

4.4 The multigrid idea

First assume that a full spectra of error components are represented on the fine grid k. A full spectra of Fourier modes for a one-dimensional problem with N nodes are $\sum_k A_k sin(jk\pi/N)$ $1 \leq k \leq N$. After a couple of iterations at a fine gridlevel k are only smooth error components e_{sm}^k left. If the mesh size is halved then e_{sm}^k does not consist of only smooth modes from the coarse grid's point of view.

Let e_{sm}^k be represented on the coarse gridlevel $k-1$ by e^{k-1}. On grid level $k-1$ e_{k-1} is a full spectra of error modes that can be represented on this grid. After a couple of iterations on this grid level will there only be low frequency errors e_{sm}^{k-1} left. They can now be represented at a still coarser grid as a full spectra and so on.

Let the coarser grids be used to liberate the fine grid approximation from the error components that looks oscillative on respectively coarse grid. Use then a serie of grids, where the coarsest grid is coarse enough to correct the fine grid solution, from the error component with largest wave length. Then the computing time would theoretically depend on the smoothing rate of the oscillative components, and not on the smoothing rate of the smooth error components.

The basic idea of multigrid is to do these corrections in a consistent manner in order to

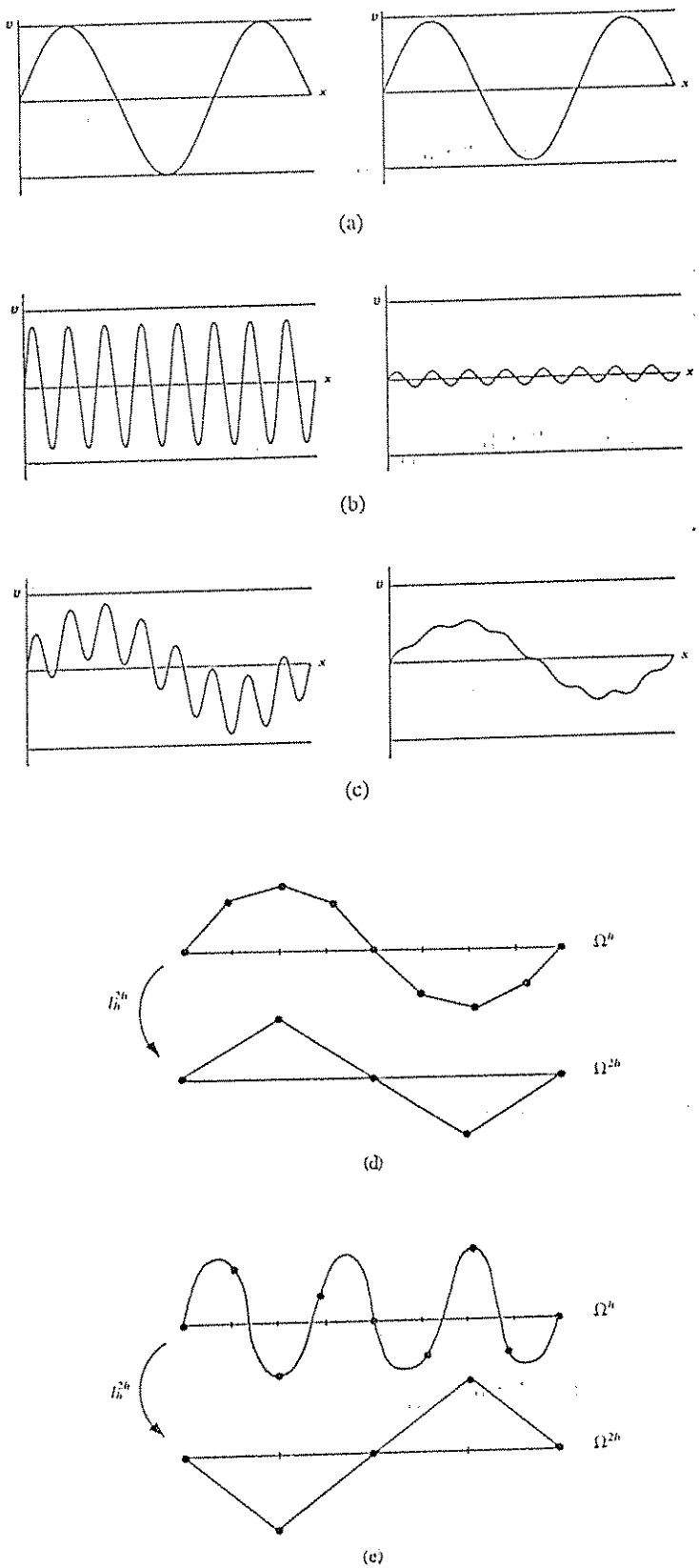

Figure 3: Typical smoothing effects. (a) one smooth, (b) one oscillative error mode and (c) both superponated. The left curves show the initial error and the right curves show what is left after 10 iterations. (d) shows how a smooth mode is restricted. (e) show how an oscillative error mode is aliased as a smooth mode after restriction

significantly reduce the computing time.

Note, if oscillative error components are left on a fine grid, they will mistakenly, be regarded as smooth components by the next coarser grid. This is called the alias fenomena and Fig. 3d shows how a smooth mode on a fine grid is represented on a coarse grid and Fig. 3e shows how an oscillative error mode at a fine grid is mistakenly regarded as a smooth mode on a coarse grid. Let N be the number of nodes in a one-dimensional problem, j the node number and k the mode number. The kth error mode lookes then like $sin(\pi jk/N), 1 \leq k \leq N$. Modes with $k > N/2$ are oscillative and cannot be represented on the coarse grid but they are aliased as the N-k mode on the coarse grid. That is shown in Fig. 3e.

To have oscillatory errors and exchange them against smooth errors has disasterous effects on the convergence rate if convergence is obtained at all. This is a phenomena that can arise from bad smoothers, bugs, inconsistency with boundary conditions or other conceptual mistakes.

4.5 The CS-concept

CS is the first multigrid concept presented by Brandt[2], and it was developed for linear problems. The discretized equation for the interior domain at grid level k yeilds:

$$L^k u^k = f^k \tag{33}$$

where f^k is the residual left by the approximation v^{k+1}, that is:

$$f^k = I_{k+1}^k (f^{k+1} - L^{k+1} v^{k+1}) \tag{34}$$

where $k+1$ is the next finer gridlevel and I_{k+1}^k is the interpolation operator that restrict the residuals from the fine grid to the coarse one.

However, to handle nonlinear problems, Brandt extended this concept and presented it as FAS.

4.6 The FAS-algorithm

The Fully Approximation Storage algorithm is the general multigrid algorithm that can handle linear as well as nonlinear problems. FAS was presented 1977 by Brandt[2] and a lot of work has been carried out to adopt this algorithm into different problems and different codes, see among others, Vanka[15], Fuchs[8, 9, 10], Peric[4, 5, 6, 7], Lien[12]

The FAS algorithm stores the full current approximation at each grid level, while CS stores only corrections due to the residuals from the finest grid. For the interior domain:

$$L^k u^k = F^k \tag{35}$$

where

$$F^k = L^k (I_{k+1}^k v^{k+1}) + I_{k+1}^k (F^{k+1} - L^{k+1} v^{k+1}) \tag{36}$$

and

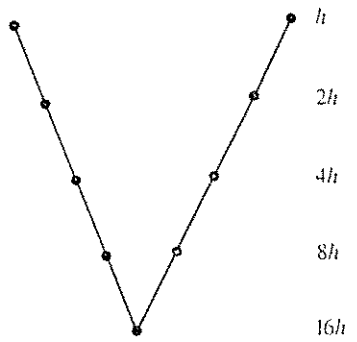

Figure 4: The V-cycle. h denotes mesh size.

$$v^{k+1} \leftarrow v^{k+1} + I_k^{k+1}(v^k - I_{k+1}^k v^{k+1}) \tag{37}$$

where $k+1$ is the next finer gridlevel, I_{k+1}^k is the interpolation operator that restrict the residuals from the fine grid to the coarse one and I_k^{k+1} is the interpolation operator that prolongates the corrections from the coarse grid to the fine one.

Brandt suggests that for fluid dynamic problems linear or bilinear interpolations are used for both restriction and prolongation operations.

4.7 The V-cycle and FMG

The V-cycle is a way to combine different grids. The V-cycle starts with a couple of iterations at the finest grid and then are the field variables stored, and restricted together with the residuals to the next coarser grid level. There are a couple of iterations made and then are the field variables stored, and restricted together with the residuals to the next coarser grid level and so on.

When the coarsest grid is reached the computed variables on this grid are compared to the restricted variables, and the differences are prolongated to correct the second grid level. A couple of iterations to smooth interpolation errors are made before prolongation to the third level and so on. The V-cycle is shown in Fig.4 where the mesh size is used to label the different grids.

The FMG-algorithm calculates an approximation at grid level 1 and then prolongates that to level 2 as an initial guess. The V-cycle is for these two grids used until a given accuracy at level 2 is obtained. The approximation at level 2 is then prolongated to the third grid level as an initial guess and the V-cycle is now used with three levels to obtain an approximation at grid level 3 to a given accuracy and so on.

For the prolongation of the initial guess Brandt suggested bicubic interpolation.

5 FAS applied to SIMPLEC

5.1 SIMPLEC algorithm

A SIMPLEC iteration is defined as a procedure of relaxation sweeps applied to the equations mentioned in ??. Here is how SIMPLEC is implemented:

1. Start with an initial guess for all quantities.

2. Update coefficients for U-momentum equation. Calculate corresponding residual.

3. Make a relaxation sweep with the smoother at U-momentum equation.

4. Update coefficients for V-momentum equation. Calculate corresponding residual.

5. Make a relaxation sweep with the smoother at V-momentum equation.

6. Update coefficients for W-momentum equation. Calculate corresponding residual.

7. Make a relaxation sweep with the smoother at W-momentum equation.

8. Calculate the massfluxes and the continuity error.

9. Do approx. 5 sweeps with the smoother at the pressure correction equation

10. Correct U, V, W and the massfluxes using the pressure corrections

11. Update coefficients for k-momentum equation. Calculate corresponding residual.

12. Make a relaxation sweep with the smoother at k-momentum equation.

13. Update coefficients for ϵ-momentum equation. Calculate corresponding residual.

14. Make a relaxation sweep with the smoother at ϵ-momentum equation.

15. return to 2. if not convergence is obtained.

5.2 FAS-algorithm applied to SIMPLEC

The description below is done with two grid levels, but is easily extended to more grid levels by the V-cycle. The FAS algorithm described earlier is here described in more detail and specially applied to SIMPLEC. This special multigrid method has been devoloped by Perić it et. al. [4, 5, 6, 7] For any variable on the fine grid level 2, the corresponding discretized equation can be written as:

$$A_P^2 \Phi_P^2 = \sum_{nb} A_{nb}^2 \Phi_{nb}^2 - S_\Phi^2 \tag{38}$$

where Φ^2(superscript 2 denotes grid level 2) is the exact discrete solution.

If that equation is relaxed with a smoother (i.e. a number of iterations are performed), an error in the field variable will always remain. Therefore is it always an imbalance in the discretized equation since the smoothed field variable is an approximation ϕ^2 to the discrete solution Φ^2. This imbalance is called the residual r_Φ^2. With an approximative solution, the discretized equation can be written as:

$$a_P^2 \phi_P^2 = \sum_{nb} a_{nb}^2 \phi_{nb}^2 - s_\Phi^2 + r_\Phi^2 \tag{39}$$

If the residuals $r_\Phi^2 = 0$ the corrections from the coarse grid to the fine grid should be zero. That is satisfied if the discretized equations, on the coarse grid level (grid 1) are satisfied with the restricted fields originating from the exact discrete solution on the finest grid level (grid 2).

This condition is satisfied by the FAS-algorithm:

$$A_P^1 \Phi_P^1 = \sum_{nb} A_{nb}^1 \Phi_{nb}^1 - S_\Phi^1 + [\overline{a}_P^1 \overline{\phi}_P^1 - \sum_{nb} \overline{a}_{nb}^1 \overline{\phi}_{nb}^1 + \overline{s}_\Phi^1 - \overline{r}_\Phi^1] \tag{40}$$

or with an approximation ϕ^1 to the exact discrete solution Φ^1:

$$a_P^1 \phi_P^1 = \sum_{nb} a_{nb}^1 \phi_{nb}^1 - s_\Phi^1 + [\overline{a}_P^1 \overline{\phi}_P^1 - \sum_{nb} \overline{a}_{nb}^1 \overline{\phi}_{nb}^1 + \overline{s}_\Phi^1 - \overline{r}_\Phi^1] + r_\Phi^1 \tag{41}$$

where $\overline{\phi}^1$ is the restricted field variable and $\overline{r}_\Phi^1$ is the restricted residual obtained from the fine grid residual r_Φ^2. The source term $\overline{s}_\Phi^1$ and the coefficients $\overline{a}^1$ are calculated at the coarse grid using the restricted field variable. All overlined terms are held constant under the course of coarse grid iterations.

As an initial guess for a^1, ϕ^1 and s^1 the overlined quantities are used. At the coarse grid a^1, ϕ^1 and s^1 are changed due to the restricted residual $\overline{r}_\Phi^1$ while iterating at the coarse grid. The changes $\phi^1 - \overline{\phi}^1$ are then prolongated to correct the approximation ϕ^2 obtained earlier at the fine grid.

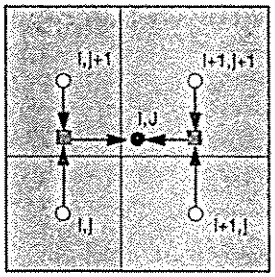

Figure 5: Restriction from fine to coarse grid

If $r_\Phi^2 = 0$ then are also $\overline{r}_\Phi^1 = 0$ and with an initial guess where a^1, ϕ^1 and s^1 the same as the resticted ones, eq.41 vanishes, and the changes $\phi^1 - \overline{\phi}^1 = 0$. The condition that a convergent solution should not be corrected by a coarse grid iteration is thereby satisfied.

Note that the exact discretized solution at the coarse grid in the multigrid method, is not the same as a singlegrid solution on the same coarse grid. This has to do with the extra source term in the coarse grid equation, when multigrid is used. This is not an important fact since the coarse grids are only used to push the finest grid to converge faster.

Of course is the finest grid solution with multigrid identical to corresponding single grid solution.

5.3 Restriction of the fine grid field variable ϕ^2 and the fine grid residual r_Φ^2

The two-dimensional coarse grid control volume is obtained by merging four fine grid cells together, and a three-dimensional coarse grid control volume by merging eight fine grid cells together.

With the definitions in Fig.5 the bilinear restriction of a uniform two-dimensional cartesian grid reads:

$$\phi_{I,J}^1 = \frac{1}{4}(\phi_{i,j+1}^2 + \phi_{i+1,j+1}^2 + \phi_{i+1,j}^2 + \phi_{i,j}^2) \tag{42}$$

Analougus is the trilinear restriction for a three-dimensional uniform grid:

$$\phi_{I,J,K}^1 = \frac{1}{8}(\phi_{i,j,k}^2 + \phi_{i+1,j,k}^2 + \phi_{i,j+1,k}^2 + \phi_{i+1,j+1,k}^2 + \phi_{i,j,k+1}^2 + \phi_{i+1,j,k+1}^2 + \phi_{i,j+1,k+1}^2 + \phi_{i+1,j+1,k+1}^2) \tag{43}$$

If a nonuniform distorted grid is used is the interplotator changed very little since the level of distorsion or expansion is seldom more than a percent or two for a fine grid. These small errors in the interpolator due to the nonuniformity should be compared to the pretty ruff approximation to the correction field that bilinear interpolation is in itself.

However the interpolations based on a uniform grid has the advantage that they are simple and requires little computer time.

It is very important that the execution is short since these interpolations are made after every third fine grid iteration. Therefore the interpolator based on a uniform grid is also used for a nonuniform grid.

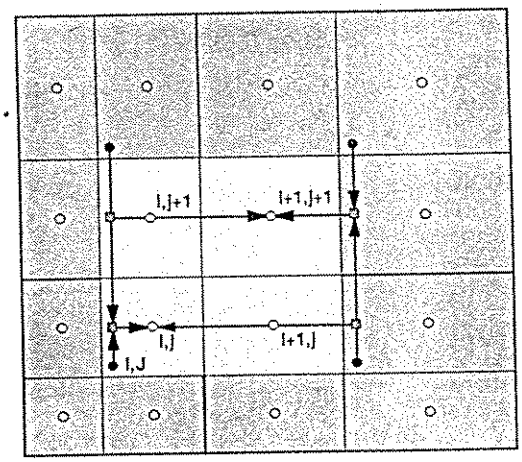

Figure 6: Prolongation from coarse to fine grid

Since the pressure operator is linear is it not necessary to restrict the pressure. At each coarse grid level the pressure is therefore initially set to zero. And instead of prolongating a pressure difference, the total pressure is used to correct the pressure at the finer grid.

The residuals are representing a flux imbalance according to eq.11 and are therefore restricted to the coarse grid by a summation of the fine grid residuals that correspond to the fine grid cells that define the coarse grid cell.

5.4 Prolongation of the coarse grid changes $\phi^1 - \overline{\phi^1}$ to the fine grid

First are the changes δ^1 between the current approximation ϕ^1 and the restricted approximation $\overline{\phi^1}$ at the coarse grid calculated. These changes are then prolongated to the fine grid to liberate the fine grid approximation ϕ^2 from low frequency errors.

The interpolation operator for prolongation was chosen to be bilinear for two-dimensional problems and trilinear for three dimensions. The reason of these choises was the same as for restriction. The prolongation interpolation operator assumes uniform grids but is also used on distorted nonuniform grids like the restiction interpolator.

With the definitions in Fig. 6 the bilinear prolongation of a uniform two-dimensional cartesian grid is:

$$\delta^2_{i,j} = \frac{1}{16}(9\delta^1_{I,J} + 3\delta^1_{I+1,J} + 3\delta^1_{I,J+1} + \delta^1_{I+1,J+J}) \tag{44}$$

Analougus the trilinear prolongation for a three-dimensional uniform grid reads:

$$\begin{aligned}\delta^2_{i,j,k} = \frac{1}{64}(&27\delta^1_{I,J,K} + 9\delta^1_{I+1,J,K} + 9\delta^1_{I,J+1,K} + 9\delta^1_{I,J,K+1} + \\ &3\delta^1_{I+1,J+1,K} + 3\delta^1_{I+1,J,K+1} + 3\delta^1_{I,J+1,K+1} + \delta^1_{I+1,J+1,K+1})\end{aligned} \tag{45}$$

The prolongation near the boundaries is not exactly the same as for the interior nodes. Since collapsed control volumes are used for the boundary velocities as Fig. 7 shows, the weights are not the same as for the interior domain.

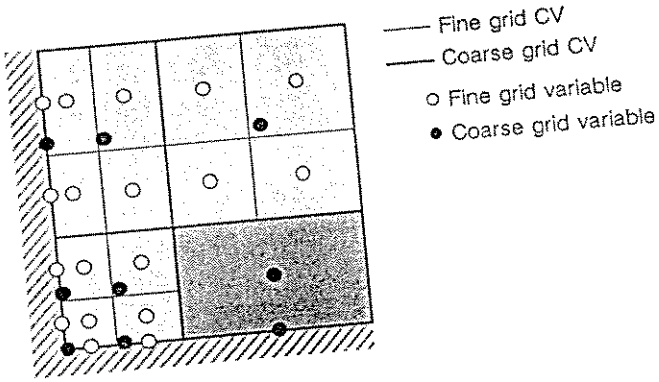

Figure 7: Prolongation at boundaries

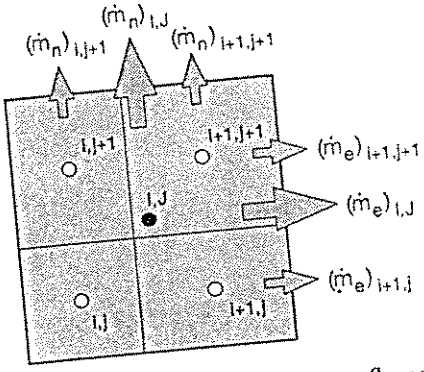

Figure 8: Restriction of mass fluxes

This fact makes the coding more tedious and specially in three dimensions. The boundary prolongation interpolator is not presented here but is easily obtained from eq.45 by substituting some 0.75 and 0.25 weights to 0.5 weights.

It has been found to be advantageble to put the pressure changes at the coarse grid boundary to zero if Dirchlet conditions is used for the velocities at the same boundary. That is done even though there has been changes of the pressure at the coarse grid boundary. This treatment is consistent with the treatment of the velocities, since their changes at the boundaries become zero for Dirchlet boundary conditions.

If the velocity boundary condition is of Neumann type it has been found to be more effective to use the coarse grid pressure changes at the boundary for the prolongation. That is also consistent with the velocities since their changes at the boundaries for Neumann conditions at the coarse grid differ from zero. The velocity changes at the boundary are also used for the prolongation.

5.5 Special treatment for the mass fluxes

The coarse grid control volumes are created by assembling together eight fine grid control volumes in 3D and four control volumes in 2D. To avoid that the solution at the coarser grid drift away too much from the fine grid solution, the mass fluxes at a coarse grid are restricted by a summation of the corresponding fine grid mass fluxes, see Fig.8. No prolongation of the massflux changes at the coarse grid is done.

If continuity is satisfied at the fine grid it is automatically ensured at the coarse grid too.

During the iterations at the coarse grid the mass fluxes are then corrected with a mass flux correction. The mass flux correction is calculated by using Rhie and Chow interpolation for the difference between the current approximation v^k and the restricted approximation $\overline{v}^k$.

5.6 Conservation of the global mass flux at the coarse grid when Dirchlet velocity boundary condition is used

When a velocity profile is used as a boundary condition special care must be taken so the total massflux is equal at al grid levels. This example will show why special care must be taken:

Assume a parabolic inlet velocity profile as for the backwards facing step. Set the density, the inlet area and the maximum velocity to 1. With two control volumes the velocities would be 0.75 and thereby the total mass flux 0.75. With four control volumes at the inlet boundary would two velocities be 0.9375 and the other two to 0.4375. The total inlet massflux is now 0.6875 and that shows why special care must be taken with the restriction of the Dirchlet velocity boundary condition.

To avoid the problem above the velocities at the fine grid are evaluated from the parabolic profile. For the coarse grid the mass fluxes are first restricted as described in the previous section. Then the coarse grid velocities are calculated from the massflux by $u^1_{boundary} = \left\{\frac{\dot{m}^1_e}{A^1_e \rho}\right\}_{boundary}$

5.7 Some notes on the $k-\epsilon$ turbulence model

It was found that the k and the ϵ transport equations could not be treated in the same manner as the velocity transport equations. All attemps to simply apply the laminar multigrid method to the $k-\epsilon$ equation without any justifications resulted in total divergence. One trial was made with a rearrangement of the sources so the equations was more implicit and the corresponding $k-\epsilon$ coefficient matrix more diagonal dominant. If in addition no prolongation of the turbulence quantities were performed a V-cycle with two grid levels could converge.

Another trial was made where the $k-\epsilon$ equations were solved only at the finest grid level and the turbulent viscocity was interpolated to the coarse grid and was there held constant. This was working fine for a V-cycle using two grids and converged three times faster than corresponding singlegrid calculation.

Both these trials were unstable when a three level V-cycle was used and on such a weak basis have I chosen not to present any results.

6 Results

In this section calculations of different cases are presented. These calculations have been made with the multigrid method as well as with the singlegrid method.

This work is not dedicated to present optimum convergence rates for each specific test case by testing different combinations of under-relaxation factors and number of iterations at each gridlevel. The goal has instead been to design a robust multigrid that can use the same under-relaxation factors and the same cycles for different flow situations.

All calculations have therefore been made with the under relaxation factors $\alpha_u = \alpha_v = \alpha_w = 0.7, \alpha_p = 0.3$ and with the V-cycle described in Fig.9. Three iterations at the finest grid and four iterations at the coarsest grid are performed and for the other grids only one iteration is done for both restriction and prolongation. Note that the work of the four iterations at the coarsest grid is negligible compared to one iteration at the finest grid. The reason of these choises is that Lien[12] and Perić [4, 5, 6, 7] used the same.

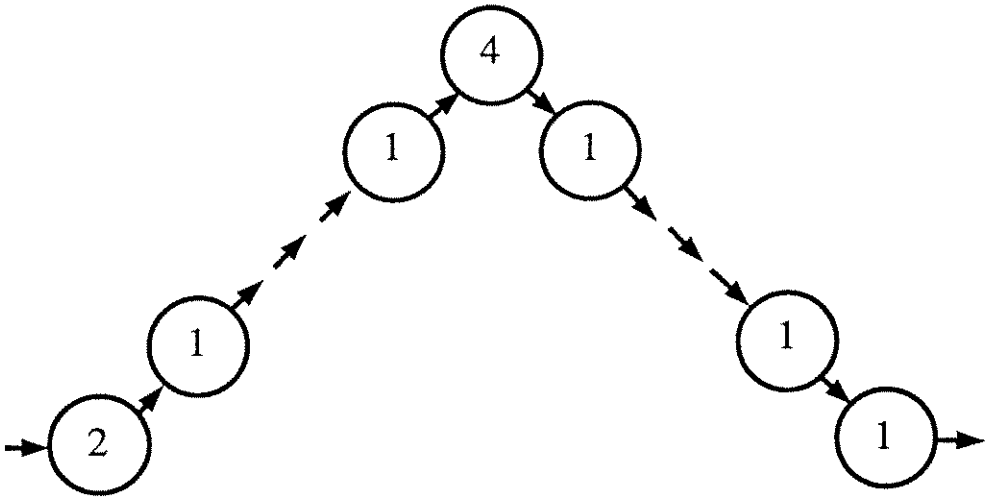

Figure 9: The V-cycle used for all cases

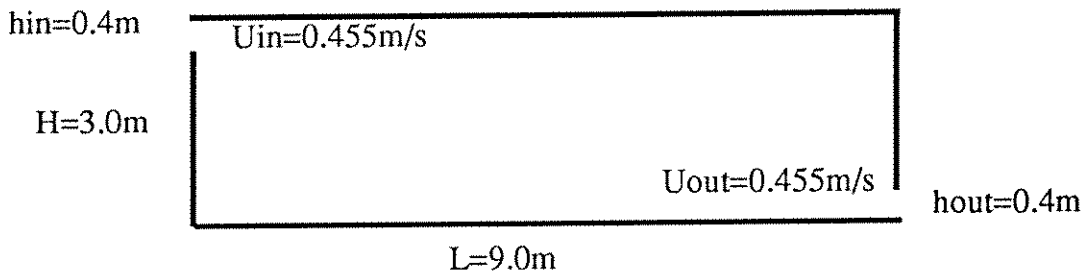

Figure 10: Two-dimensional laminar ventilated enclosure

If $\alpha_p = 1.3$ for the three dimensional laminar ventilated enclosure the computing time was reduced with a factor 2 which verifies that optimizations can be done. Grid plots and velocity plots are presented below for each case.

6.1 Two-dimensional laminar ventilated enclosure using non-uniform grids

The first test case is a two-dimensional laminar ventilated enclosure with Reynolds number 110 based on the inlet height and the inlet velocity, see Fig. 10. The tests has been made with either uniform or nonuniform grids.

Only the calculations with the nonuniform grid are presented. The convergence properties for the uniform grid is similar to the results at the non-uniform setup.

For the non-uniform grid with 160x160 nodes the expansion factor is 1.015. That gives a highest aspect ratio of 10. If the expansion factor was incrased to 1.025 convergence was obtained only with four multigridlevels which could indicate some sensitivity for the multigrid to high aspect ratios.

The convergence plot in Fig. 11 shows the maximum residual as a function of workunits(WU).

$$WU = \sum_{ng} it_{cg} 2^{d(cg-ng)} \tag{46}$$

where it_{cg} is the the number of iterations at the current grid level, ng is the number of grid levels, d is the spatial dimension and cg is the current grid level.

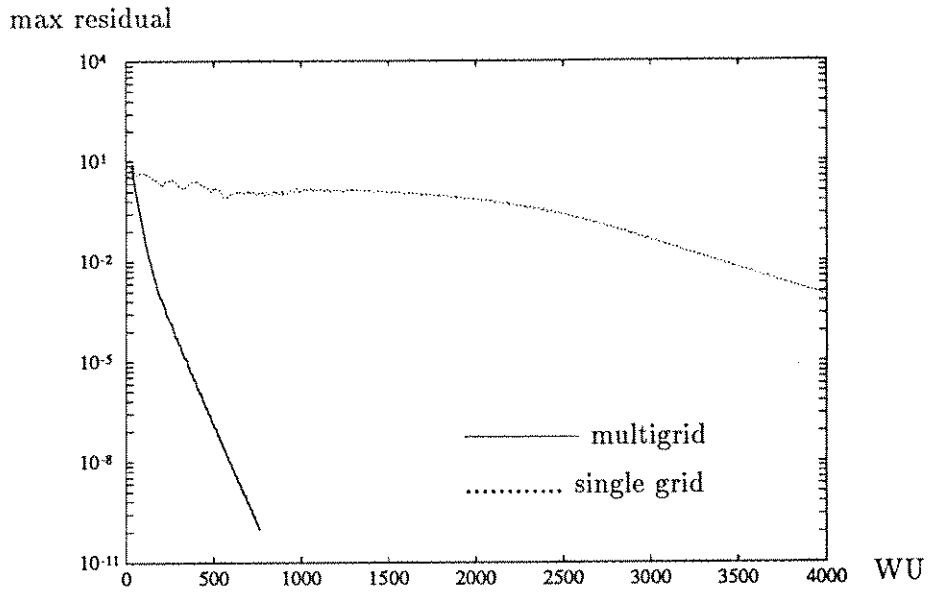

Figure 11: Convergence history plot for the two-dimensional laminar ventilated enclosure using a 160x160 non-uniform grid

The dashed line represent the singlegrid calculation at a 160x160 grid. The solid line represent calculations at the same case with the multigrid method. This multigrid calculation is made with five grid levels.

If a multigrid code contain conceptual errors or bugs the result is often convergence stagnation. The level of the stagnation depends of the influence of the bugs and is therefore case dependent. If no stagnation occurs it is not a guarrantee of a bug-free code but it shows at least that a presumptive bug has only little influence.

That is the reason why the convergence plots for multigrid calculations is done to a maximum residual $< 10^{-10}$ even though a common convergence criteria is 10^{-3} or 10^{-2}.

The WU is not an exact measure of the total amount of work made with the multigrid method since the interpolation work is neglected. The interpolation work is here only 25% of the total work since the bilinear and trilinear interpolations are very simple.

For consistency WU and CPU-times for single grid and multigrid calculations are showed in Table 2 and Table 3. Note that the demanded CPU-time inreases linearly by the number of nodes for the multigrid method, and quadratically for the singlegrid calculations as expected. The multigrid calculation at 160x160 grid converges 16.21 times faster than corresponding singlegrid calculation. If a 320x320 grid would be used should the same ratio be around 50.

The modest convergence accelerations for the 20x20 and 40x40 grids has probably to do with non-linear effects, big differences in truncation error between the multigrid meshes and that the asymptotic smoothing behaviour of the single grid calculation is not devoloped since these grids are too coarse. Note that the measure WU based on 80x80 grid is not equal to the WU based on a 160x160 grid

The convergence line for the multigrid calculation in Fig. 11 shows small hacks due to the corrections from the coarser grids. For linear problems this should not be possible for a correct implemented algorithm but for systems and especially nonlinear systems, small amplifications could sometimes be expected.

GRIDSIZE	CPU-TIME	WU
160x160	101534	3977
80x80	6570	1028
40x40	517	317
20x20	68.9	168
10x10	22.8	120

Table 2: CPU-times for the two-dimensional laminar ventilated enclosure using non-uniform single grid $\alpha_u = \alpha_v = 0.7, \alpha_p = 0.3$ and max residual< 0.001

GRIDSIZE	MULTIGRID LEVELS	CPU-TIME	WU	SG/FMG CPU-TIME
160x160	5	6261	189	16.21
80x80	4	1575	182	4.17
40x40	3	388.5	190	1.33
20x20	2	92.1	188	0.74
10x10	1	22.8	120	1.0

Table 3: CPU-times for the two-dimensional laminar ventilated enclosure using non-uniform grids in the multigrid method $\alpha_u = \alpha_v = 0.7, \alpha_p = 0.3$ and max residual< 0.001

6.2 Two-dimensional laminar backwards facing step

The next test case was the two-dimensional laminar backwards facing step and is shown in Fig.12. The big difference between this case and the two-dimensional laminar ventilated enclosure is the outlet boundary condition for the velocity. Backwards facing step uses Neumann condition ($\partial U/\partial x = 0$) instead of the Dirchlet condition ($U = U_{out}$) for the ventilated enclosure.

Fig.13 shows the convergence history for multigrid calculations at backwards facing step 160x80 nodes, 4 grid levels. The singlegrid calculation at the same case did not converge and is therefore not presented. But with a modified outlet condition and other relaxation parameters would convergence be accieved also for singlegrid. Table 4 shows the CPU-times and the numbers of work units used for multigrid and singlegrid calculations at backwards facing step.

GRIDSIZE	MULTIGRID LEVELS	MULTIGRID CPU-TIME	MULTIGRID WU	SINGLEGRID CPU-TIME	SINGLEGRID WU
160x80	4	2901	184	no conv.	no conv.
80x40	3	519	150	no conv.	no conv.
40x20	2	138	148	226	290
20x10	1	29.4	142	29.4	142

Table 4: CPU-times for single- and multigrid calculations of backwards facing step $\alpha_u = \alpha_v = 0.7, \alpha_p = 0.3$ and max residual< 0.001

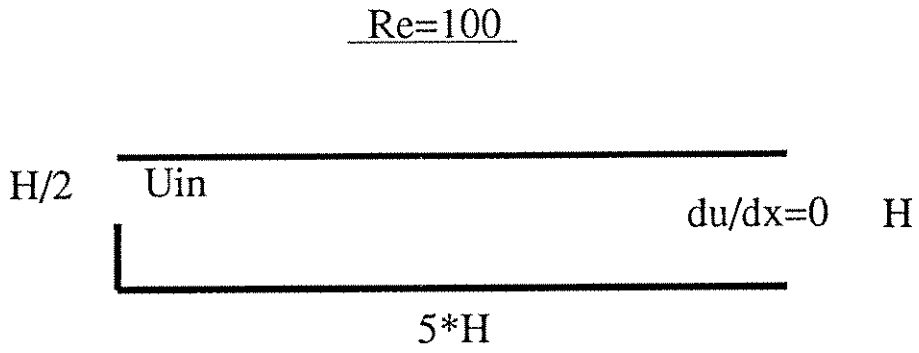

Figure 12: Backward facing step

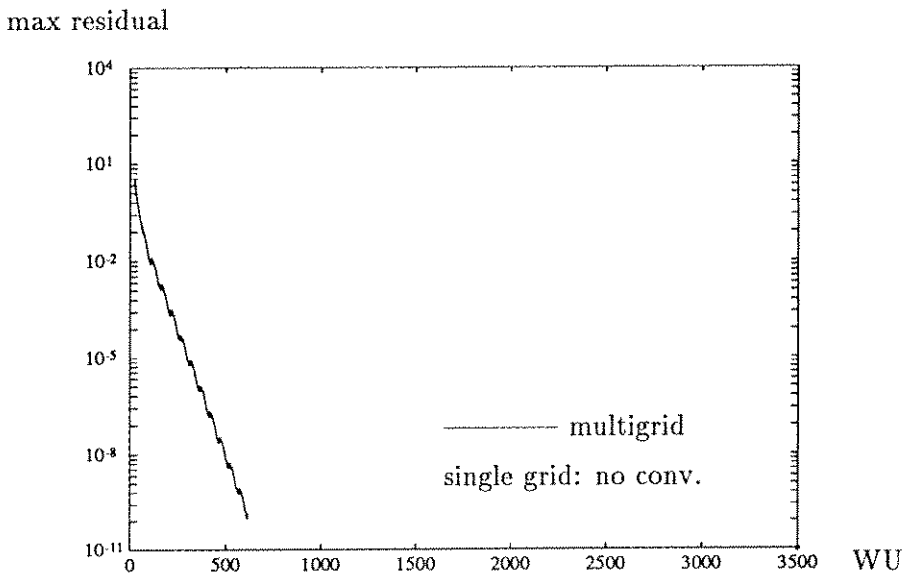

Figure 13: Convergence history plots for the two-dimensional laminar backwards facing step

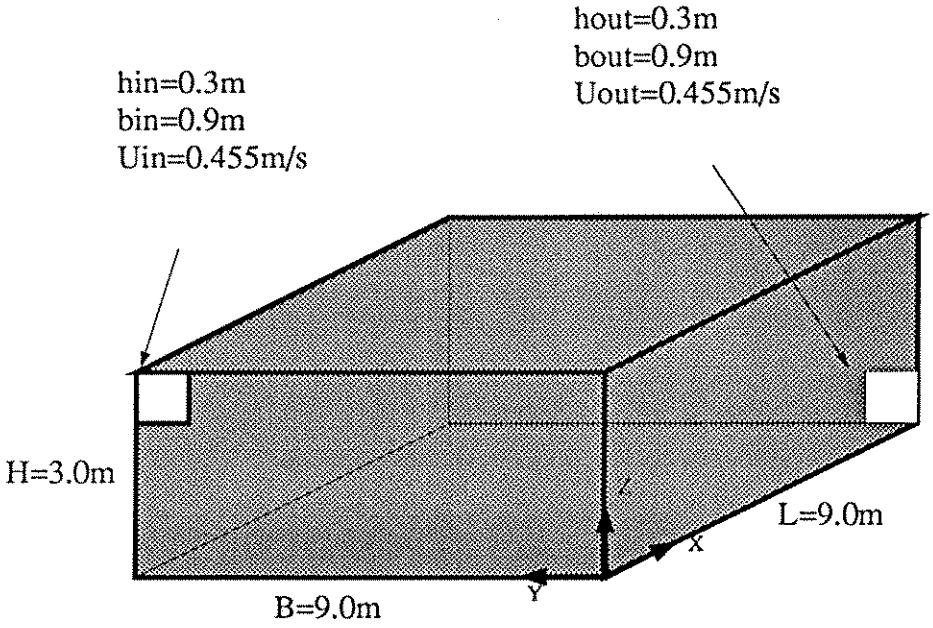

Figure 14: Three dimensional ventilated enclosure

The fact that the multigrid calculations for both 80x40 or 160x80 grids converged nicely even though corresponding single grid calculations did not indicates robustness for the multigrid method.

The oscillations in the multigrid convergence plot may seem confusing, but other workers [7] has also reported oscillations.

6.3 Three-dimensional laminar ventilated enclosure

After the two-dimensional laminar multigrid code had been tested with the two cases presented, it was extended to three dimensions. That does not imply any conceptual differencies but is tedious to implement.

The three-dimensional laminar code was tested for a ventilated enclosure, see Fig.14.

The dashed line in the convergence plot (Fig.15) shows the convergence history for single grid 40x40x40 nodes and the solid line for corresponding multigrid calculation at 40x40x40 grid and with 3 grid levels. The size of the computer memory was to small to handle the 80x80x80 grid.

Table 5 and Table 6 show the CPU-time and WU required for single and multigrid calculations. The speed up factor of the multigrid calculation compared to the singlegrid calculation is for the 40x40x40 grid 3.39. The first reflection is that this ratio is a bad result but then it should be reminded that it is the number of nodes in a characteristic direction that specifies the speed up ratio and not the number of nodes in the whole domain.

That means for a 80x80x80 grid should the speed up ratio be at least a magnitude. Note that the amplifications of the maximum residual discussed for the two-dimensional ventilated enclosure has not occured in this three-dimensional calculation.

For this problem has, as mentioned earlier, a multigrid calculation with $\alpha_p = 1.3$ and $\alpha_u = \alpha_v = \alpha_w = 0.8$ been done and it converge twice as fast as the presented one. This shows that optimizations of the relaxation factors and the V-cycle can be done but it also indicates robustness of the multigrid method to this pretty ruff change of α_p.

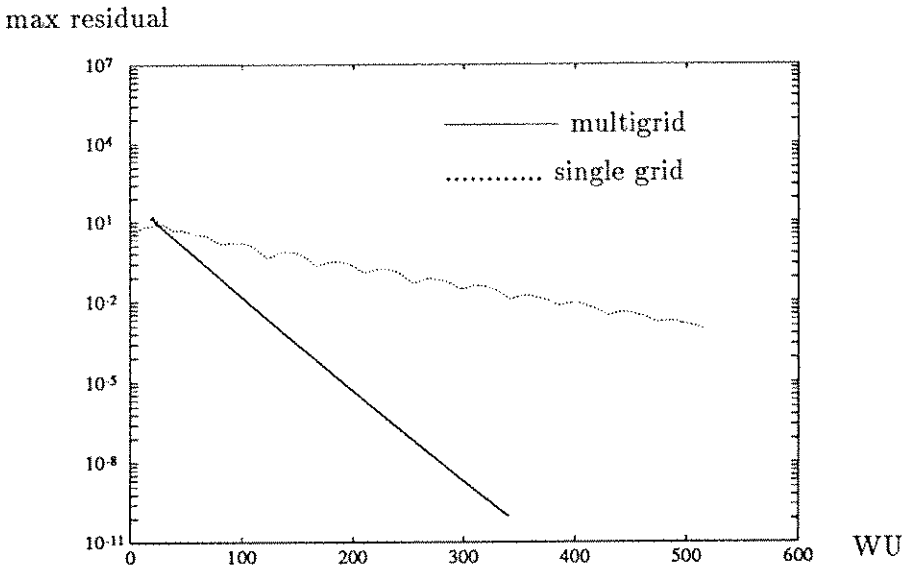

Figure 15: Convergence history plots for three dimensional laminar ventilated enclosure

GRIDSIZE	CPU-TIME	WU
40x40x40	28905	515
20x20x20	1194	182
10x10x10	82.9	106

Table 5: CPU-times for single grid calculations at the three-dimensional ventilated enclusure using a 40x40x40 grid, $\alpha_u = \alpha_v = \alpha_w = 0.7, \alpha_p = 0.3$ and max residual< 0.001

GRIDSIZE	MULTIGRID LEVELS	CPU-TIME	WU	SG/FMG CPU-TIME
40x40x40	3	8527	131	3.39
20x20x20	2	1097	141	1.09
10x10x10	1	82.9	106	1.0

Table 6: CPU-times for multigrid calculations at the three-dimensional ventilated enclosure using a 40x40x40 grid, $\alpha_u = \alpha_v = \alpha_w = 0.7, \alpha_p = 0.3$ and max residual< 0.001

7 Closure

This work has been dedicated to implement and code and therefore only a few test cases have been calculated. But even with such a few calculations some characteristica can be observed.

1. Multigrid accelleration is highly effective in 2D and 3D laminar flows.

2. Multigrid calculations on turbulent flows demands special treatment for the turbulence transport equations.

The three test cases also indicate that:

1. The effectivness is not greatly effected of grid non uniformity.

2. Neumann boundary condition can be handled as well as Dirchlet conditions.

8 Advices to further research

Some advices to further research of what has not been possible to include in this work is presented below:

1. Theoretical investigations of the coupling between the equations when turbulence is included.

2. Theoretical investigations of the smoothing properties of the SIMPLEC algorithm when turbulence is included.

3. Apply these theories or test different treatments of the turbulence and then implement the multigrid algorithm into turbulent calculations.

4. Testings of different laminar flow situations to document the prestanda of the multigrid.

5. Optimizing the relaxation factors and the V-cycle to achieve maximal convergence rate.

6. Use the multigrid algorithm in calculations with local mesh refinement.

References

[1] L. DAVIDSSON and B. FARHANIEH, A finite volume code employing colocated variable arrangement and cartesian velocity components for computation of fluid flow and heat transfer in complex three-dimensional geometries, Publication no. 91/14, Department of Thermo- and Fluid Dynamics, Chalmers University of Technology (1991)

[2] A. BRANDT, Multi-level adaptive solutions to boundary-value problems, *Math. of Comput.*, Vol.31 333-390 (1977)

[3] A. BRANDT, Multigrid techniques: 1984 Guide with applications to fluid dynamics, Computational fluid dynamics lecture notes at the von-Karman Institute, March 1984

[4] M. BARCUS, M. PERIĆ and G. SCHEUERER, A control volume based full multigrid procedure for the predictions of two-dimensional, laminar, incompressible flows, *Notes on Numerical Fluid Mechanics*, Vol.20, Vieweg, Braunschweig, 9-16 (1988)

[5] C. BECKER, J. H. FERZIGER, M. PERIĆ and G. SCHEUERER, Finite volume multigrid solutions of the two-dimensional incompressible Navier-Stokes equations, *Notes on Numerical Fluid Mechanics*, Vol.23, Vieweg, Braunschweig, 37-47 (1988)

[6] M. PERIĆ, M. RÜGER and G. SCHEUERER, A finite volume multigrid method for calculating turbulent flows, *Proc. 7th Symp. on Turbulent Shear Flows*, Stanford, CA, 7.3.1-7.3.6 (1989)

[7] M. HORTMANN and M. PERIĆ, Finite volume multigrid prediction of laminar natural convection: Benchmark solutions, *Int. J. Numer. Methods Fluids*, 11, 189-207 (1990)

[8] L. FUCHS, Multi-grid solution of the Navier-Stokes equations on non-uniform grids, *NASA-CP 2202* 83-100 (1981)

[9] L. FUCHS and H.-S. ZHAO, Solution of three-dimensional viscous incompressible flows by a multi-grid method, *Int. J. Numer. Methods Fluids*, Vol.4, 539-555 (1984)

[10] L. FUCHS, A local meshrefinement technique for incompressible flows, *Computers and Fluids*, Vol. 14, No. 1, 69-81 (1986)

[11] W. L. BRIGGS, A multigrid tutorial. Society for Industrial and Applied Mathematics Philadelphia, Pennsylvania (1987)

[12] F. S. LIEN, B. SC., M. SC., Computational modelling of 3d flow in complex ducts and passages. PhD thesis, Department of Mechanical Engineering, MIST Manchester, (1992)

[13] G. J. SHAW and S. SIVALOGANATHAN, On the smoothing properties of the SIMPLE pressure-correction algorithm, *Int. J. Numer. Methods Fluids*, 8, 441-461 (1988)

[14] S. JOHANSSON, Numerical simulation of vortex shedding past triangular cylinders, Department of Thermo- and Fluid Dynamics, Chalmers University of Technology (1991)

[15] D. S. JOSHI and P. VANKA, Multigrid calculation procedure fore internal flows in complex geometries. *Num. Heat Transf., Part B*, Vol 20, 61-80 (1991)

[16] M. C. THOMPSON and J. H. FERZIGER, An adaptive multigrid technicue for the incompressible Navier-Stokes equations, *J. Comput. Phys.* Vol.82, No.1, (1989)

[17] W. RODI, S. MAJUMDAR and B. SCHÖNUNG, Finite volume methods for two-dimensional incompressible flows with complex boundaries, *Comput. Meth. Appl. Mech. Eng.*, 75, 369-392, (1989)

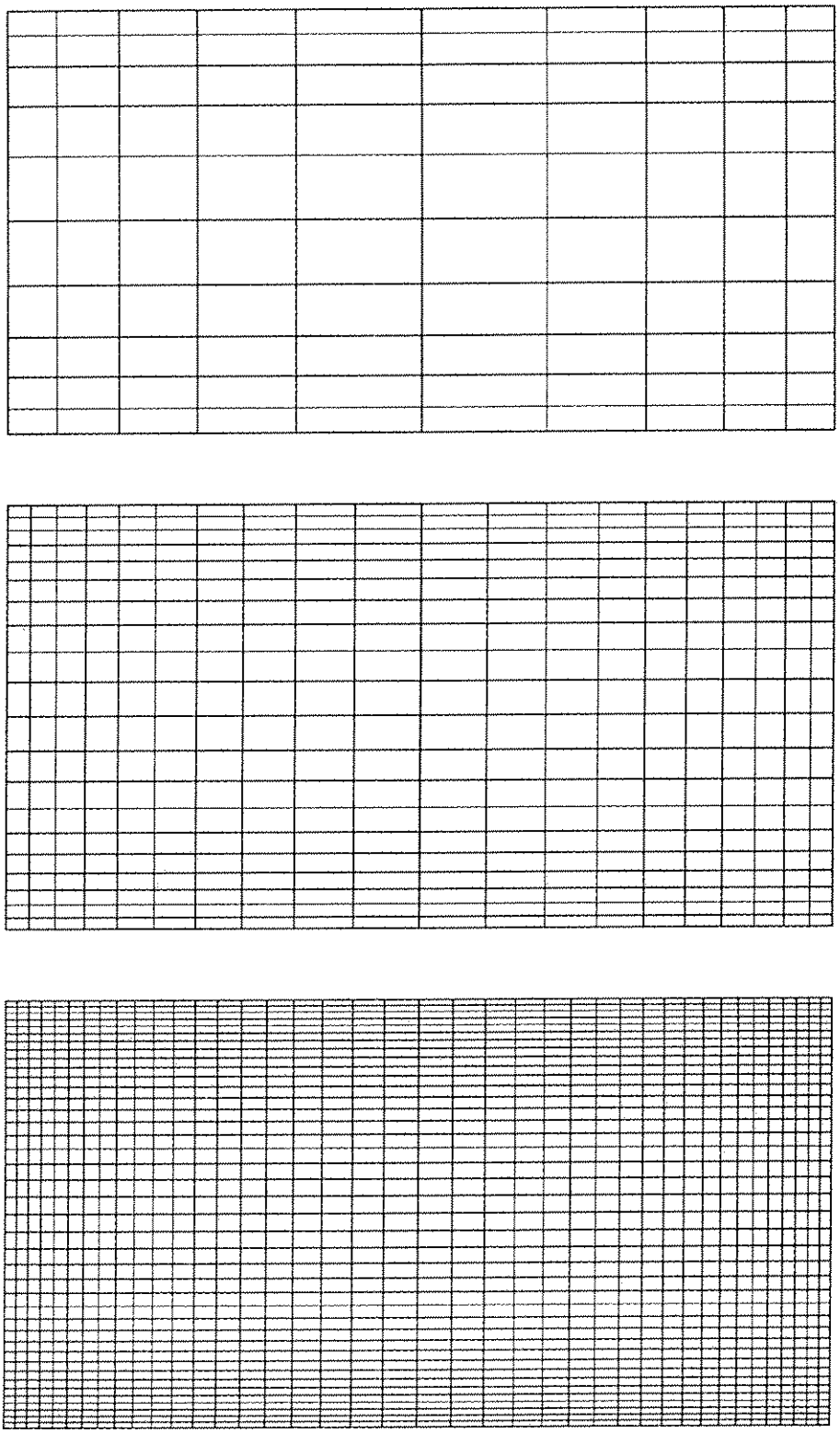

Figure 16: 10x10, 20x20 and 40x40 grids for two-dimensional laminar ventilated enclosure with non-uniform grids

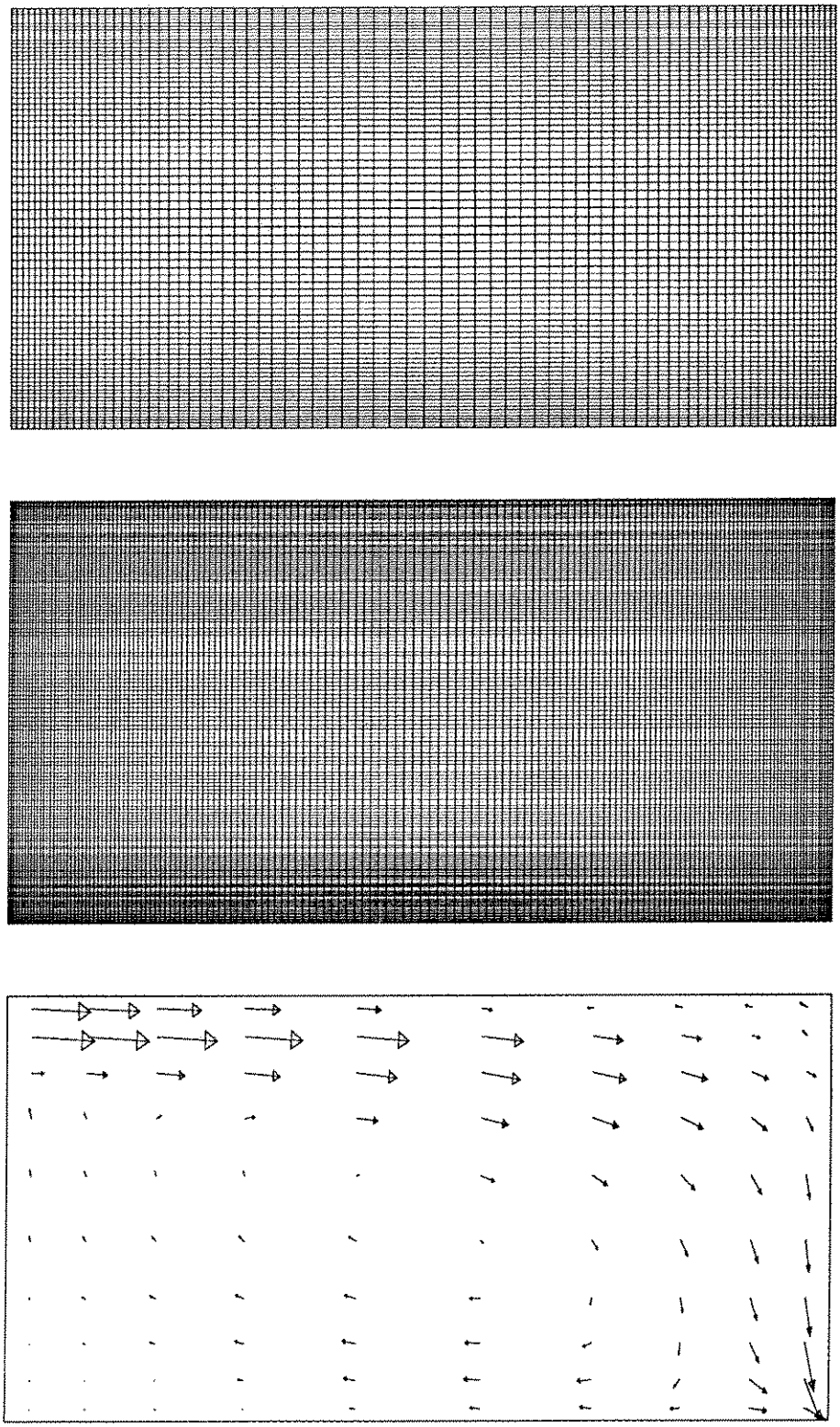

Figure 17: 80x80 and 160x160 grids for two-dimensional laminar ventilated enclosure with non-uniform grids and a vector plot at the 10x10 grid

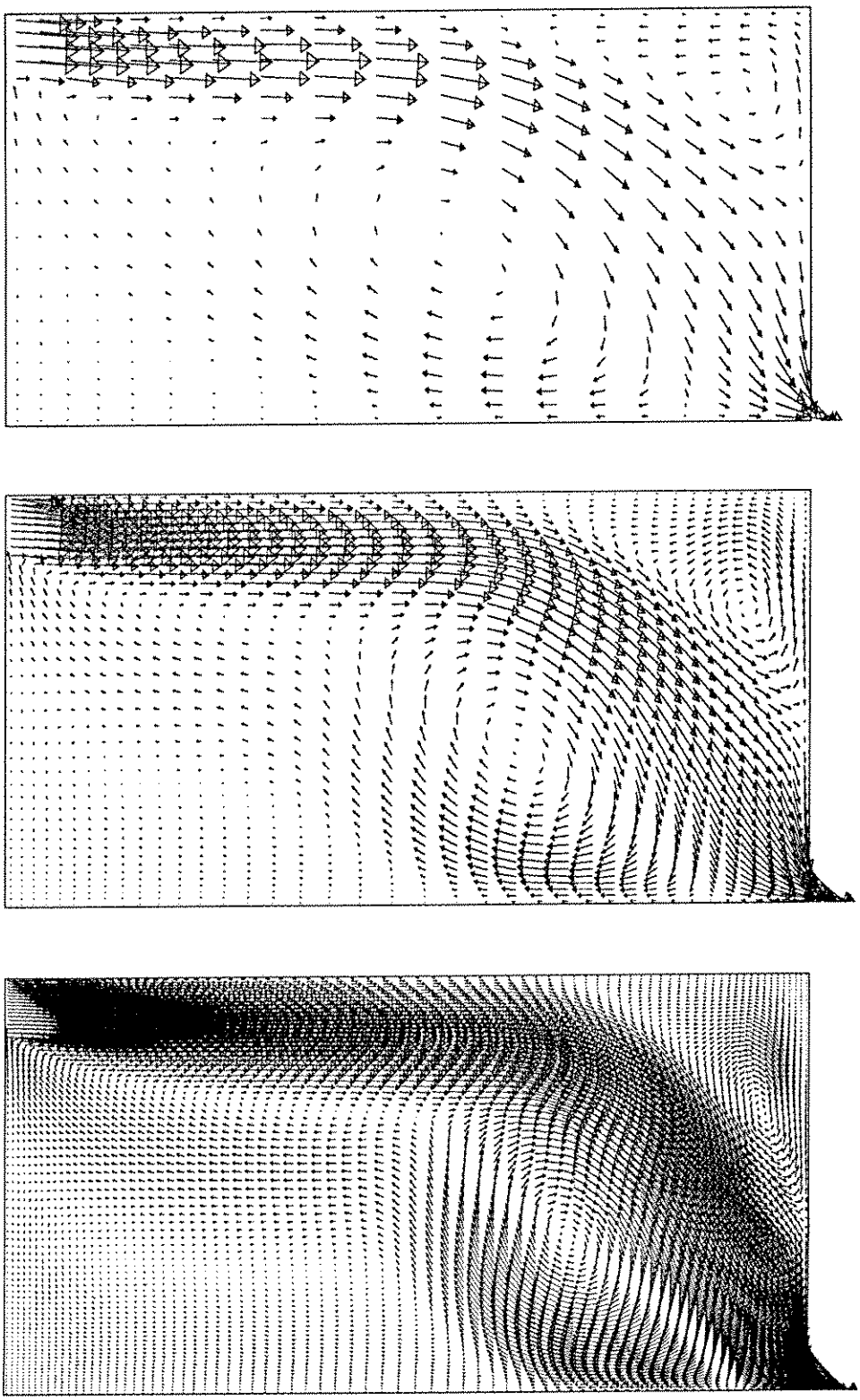

Figure 18: Vector plot for two-dimensional laminar ventilated enclosure with non uniform-grids at the 20x20, 40x40 and 80x80 grids

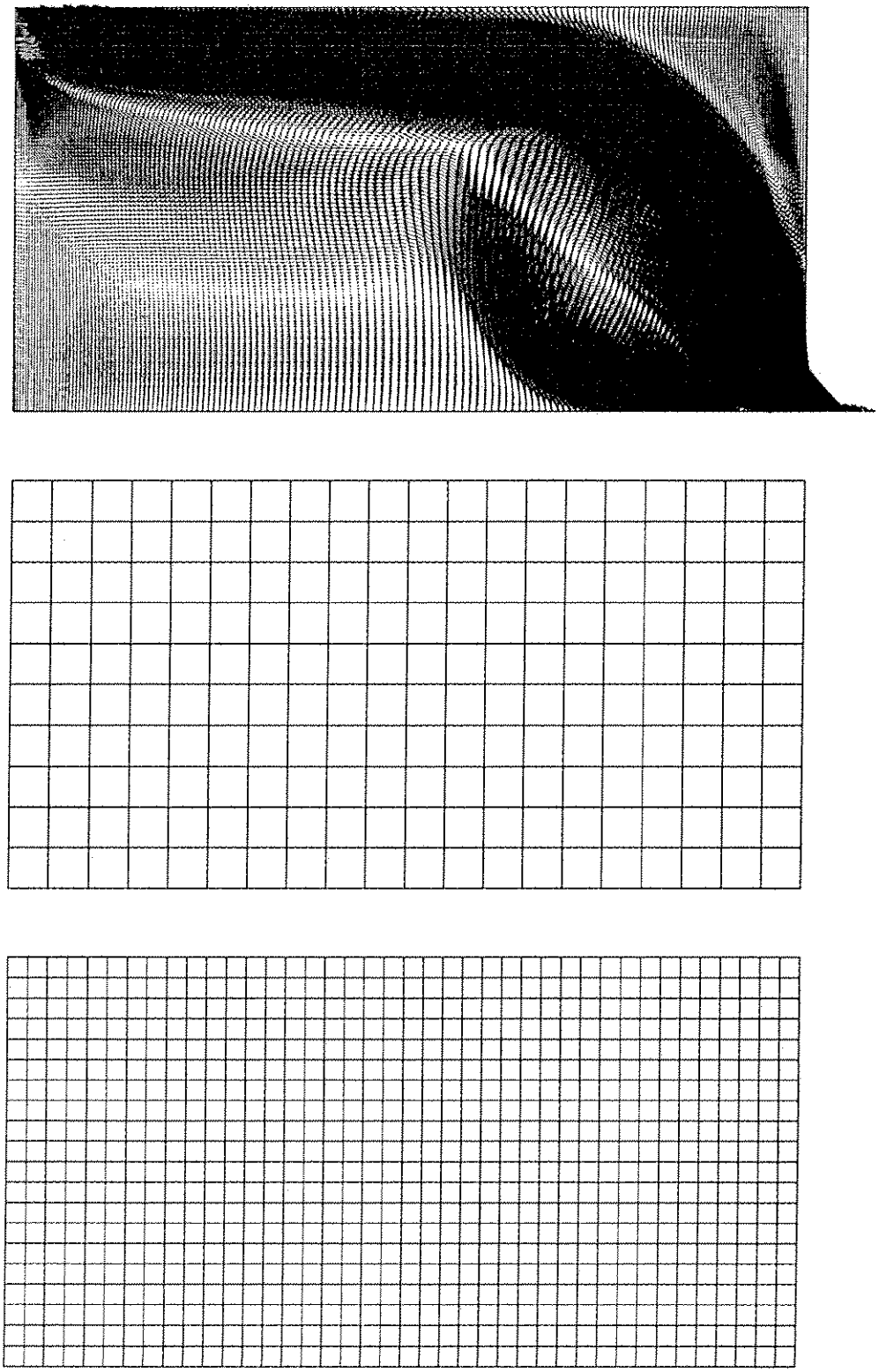

Figure 19: Vector plot for two-dimensional laminar ventilated enclosure with non-uniform grid at the 160x160 grid and grid plots for backwards facing step at 20x10 and 40x20 grid

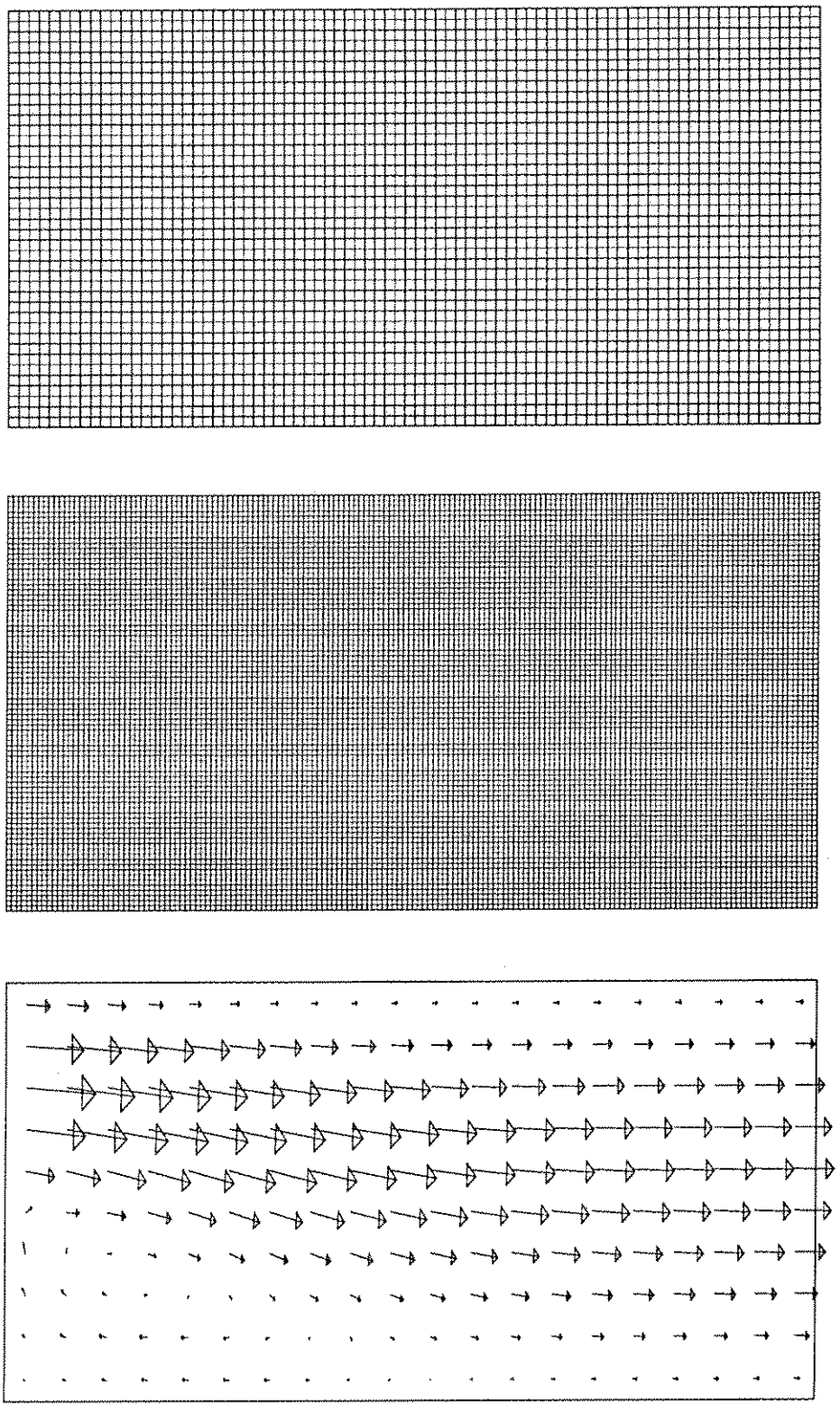

Figure 20: 80x40, 160x80 grids and a vector plot at the 20x10 grid for backwards facing step

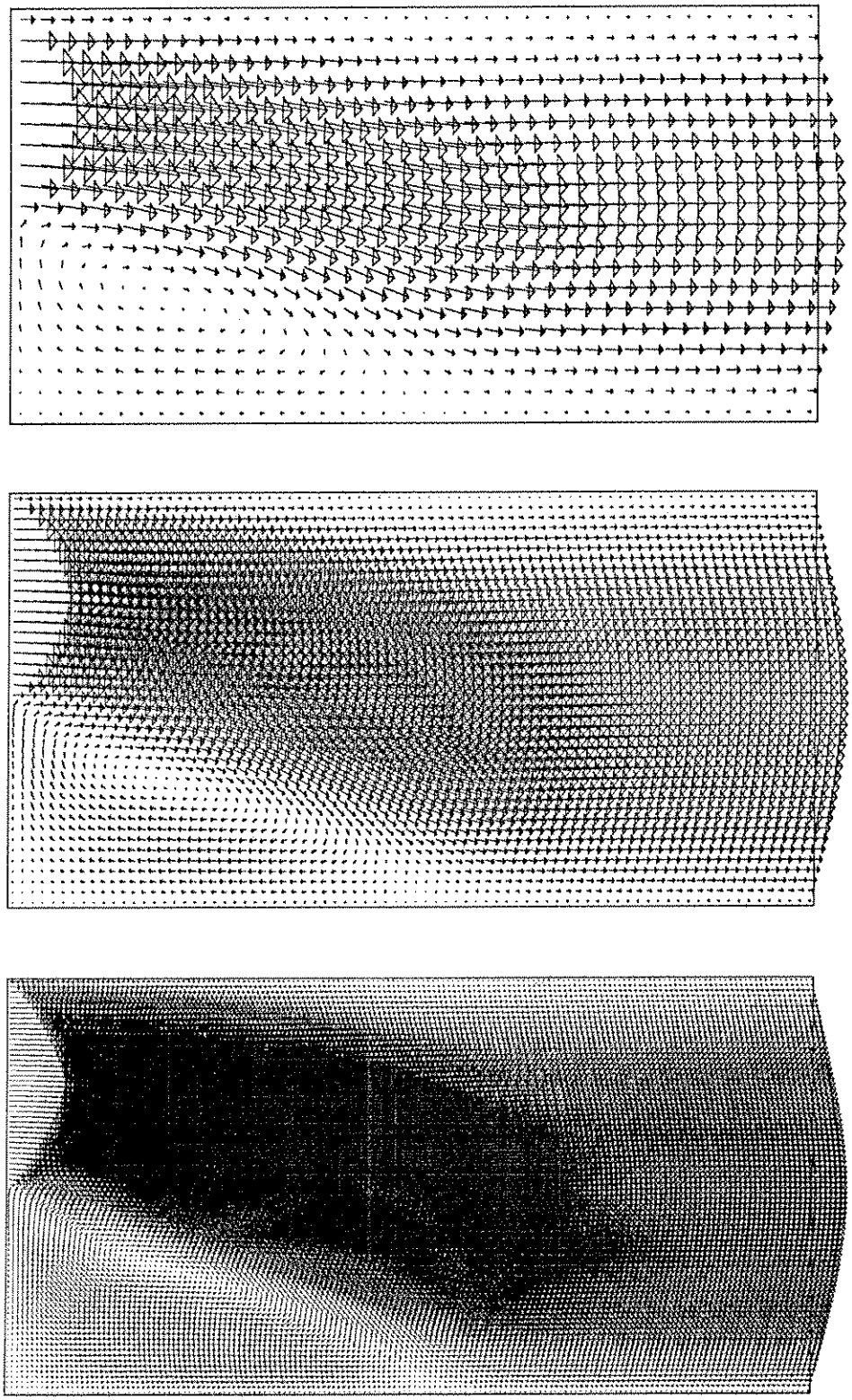

Figure 21: Vector plot for backwards facing step at the 40x20, 80x40 and 160x80 grids

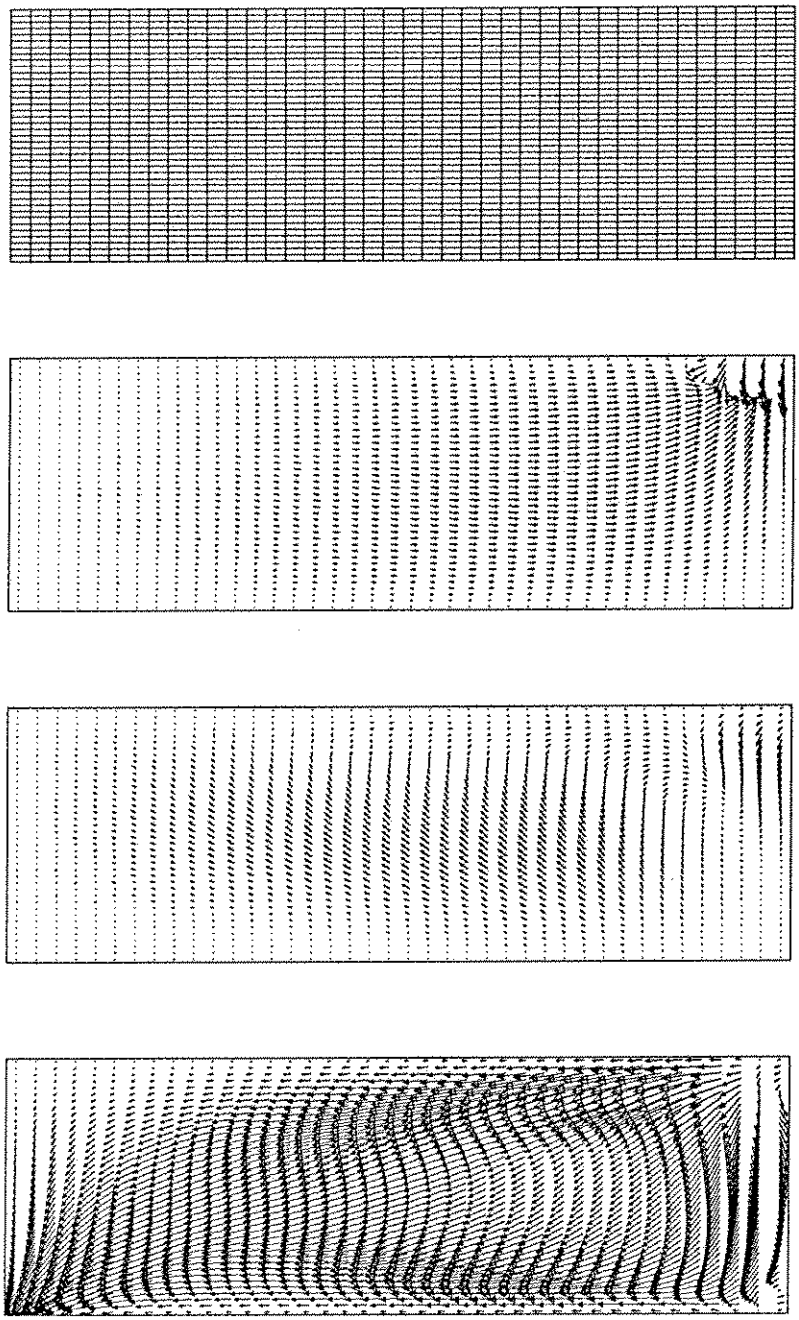

Figure 22: Grid plot for the y-z plane and vector plots for the three-dimensional laminar ventilated enclosure at x/L=0.09, x/L=0.5 and x/L=0.91

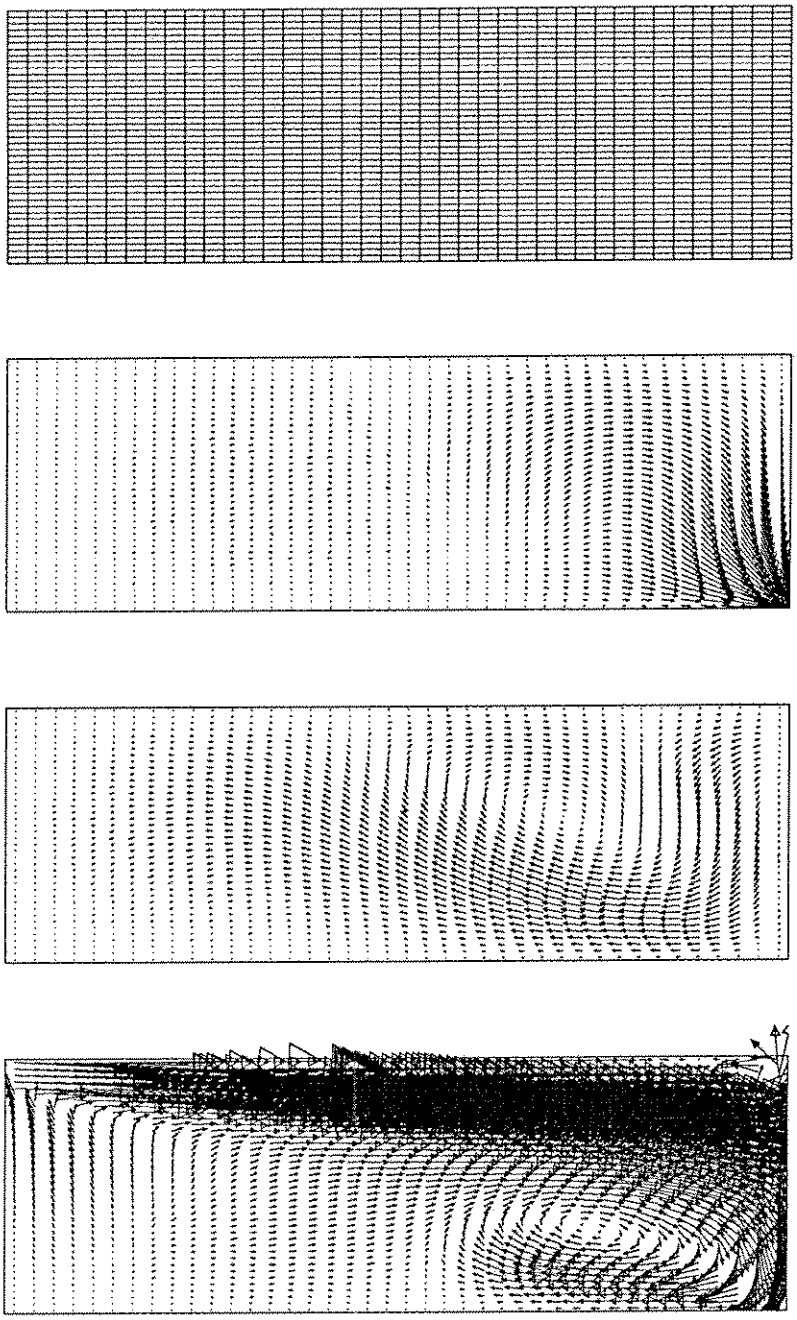

Figure 23: Grid plot for the x-z plane and vector plots for the three-dimensional laminar ventilated enclosure at y/B=0.09, y/B=0.5 and y/B=0.91

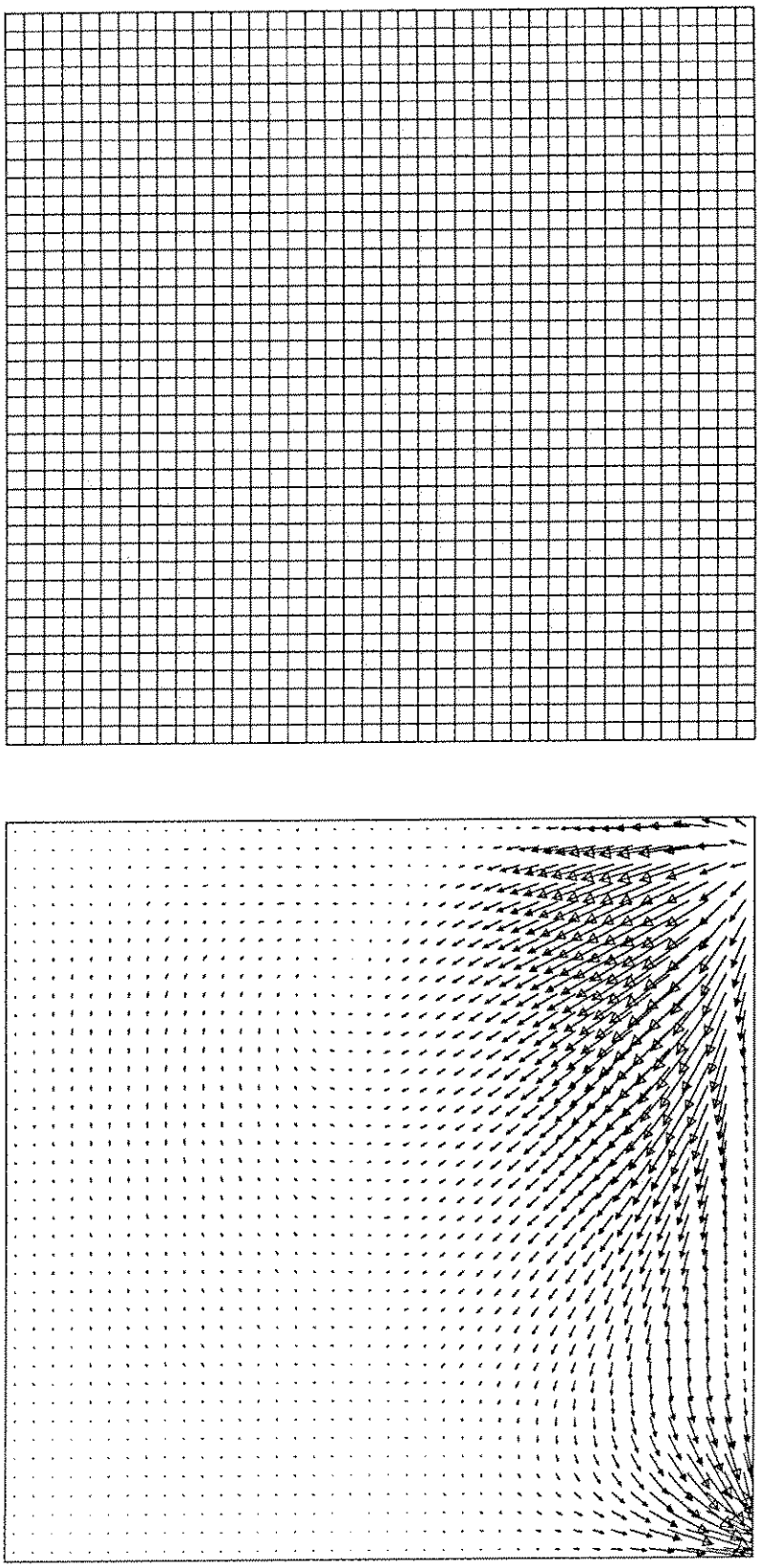

Figure 24: Grid plot for the x-y plane and vector plot for the three-dimensional laminar ventilated enclosure at z/H=0.09