Mecánica Computacional Vol 20. pp. 100 — 107
Compilado por Fernando G. Flores
Córdoba, 30 de octubre - 2 de noviembre de 2001

A PROGRAM FOR THE SOLUTION OF TRANSIENT THREE DIMENSIONAL FLOW

Luis Cardón
INENCO - Instituto UNSa-CONICET
Universidad Nacional de Salta, Buenos Aires 177, 4400 Salta, Argentina
cardon@unsa.edu.ar

ABSTRACT

This work describes a code developed for the simulation of transient laminar flow in three dimensional cartesian domains. The code is intended to be applied to incompressible indoor environmental flows on domains of moderate complexity. It is expected to serve as well as a platform to test LES turbulent modeling in geometries of moderate complexity under general boundary conditions.
Explicit fractional time steps methods were implemented. Control volumes and staggered meshes were used for spatial discretization. Central difference schemes were applied to all terms with the explict scheme.

INTRODUCTION

The purpose of the work reported here is to develop a computer program capable of simulate turbulent flow in the indoor building environment. It has been shown by Kuhen, [4], that many features of the turbulent flow in offices can not be captured by RANS models, so the objetive of this work is to try Large Eddy Simulation (LES).

Large Eddy Simulation attemps to solve the large scale features of the flow while modeling turbulence effects only on the smaller scales. The space filtered equations for a LES are similar to the Navier Stokes equation and to implement a LES solver we need a transient Navier Stokes solver first. Here we report this basic solver.

The transient solution of this set of equations in three dimensions is a numerically expensive task, both in memory and CPU time. Efficiency is then a main concern in selecting the solution algorithm and during the code development. Considering this and the previous experience of the author, we have selected to work with the control volume method over structured meshes and explicit time integration. Multigrid aceleration was used for the solution of the Poisson equation, [1].

Nevertheless the explicit actual implementation of the code, the coeficient values for the momentum equation are stored in such a way that it will be easy to develop an implicit version of the code in a latter time.

Explicit two fractional step method

We proceed here to describe the solution method. For simplicity we write Navier-Stokes equations in the following form

$$\frac{\partial \rho u_i}{\partial t} = \mathcal{H}_i - \frac{\partial p}{\partial x_i} \tag{1}$$

where $\mathcal{H}_i$ stands for the convective, the diffusive, and the body force terms,

$$\mathcal{H}_i = -\frac{\partial(\rho u_i u_j)}{\partial x_j} - \frac{\partial(2\mu S_{ij})}{\partial x_j} + \rho f_i \tag{2}$$

where S_{ij} is the deformation tensor,

$$S_{ij} = \frac{1}{2}(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}) \tag{3}$$

Equation 1 is subject to the continuity constraint

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{4}$$

A general explicit time discretization of this equation is

$$\frac{(\rho u_i)^{n+1} - (\rho u_i)^n}{\Delta t} = \frac{\partial p}{\partial x_i} + H_i(u^n, u^{n-1}, ...) \tag{5}$$

where H_i is the spatial discretization of $\mathcal{H}_i$, calculated in terms of the flow field variables at time n and previous time steps according to the time integration scheme. We will discuss later the time at which the pressure is to be evaluated, selection that leads to different solution procedures.

The present approach is based on the fractional two-step method of Chorin. In the first step an approximate (possible divergent) velocity field is calculated, to be corrected in the second step by subtraction of the gradient of a pressure like field constructed in a way to ensure that the corrected velocity field satisfies continuity.

The distinctive feature of the fractional step or projection approach is that the pressure gradient term is not taken into account in the first evaluation of the momentum equations. The procedure has been used by many others and, in particular, by Voke and Potamitis [6].

Third order explicit time integration of the approximated (without pressure term) momentum equation leads to

$$\frac{(\rho u_i)^* - (\rho u_i)^n}{\Delta t} = H_i(u_i^n, u_i^{n-1}, u_i^{n-2}) \tag{6}$$

The approximate velocity field does not satisfy the continuity equation and will be corrected later. Subtracting equations 6 from 5 the correction follows

$$u_i^{n+1} = u_i^* - \Delta t \frac{\partial p}{\partial x_i} \tag{7}$$

although p is not known at this time. Taking the divergence of 7, and considering that we want to enforce continuity for the u^{n+1} field, we obtain the pressure-like equation

$$\frac{\partial^2 p}{\partial x_i^2} = -\frac{1}{\Delta t}\frac{\partial u_i^*}{\partial x_i} \tag{8}$$

For each time step, approximates of the velocity components are advanced in time by solving in succession the momentum equations in each direction. Each time the discretization equations are calculated with the most recently evaluated set of variables.

Equation 1 under incompressible flow conditions can be solved with the following algorithm

- Start with a divergence-less velocity field for time n, $n-1$ and $n-2$.
- For each i, with i=1,2,3 do
  - Calculate explicitly $H_i(u^n, u^{n-1}, u^{n-2})$. If u_i^* is available, use it instead of u_i^n.
  - Calculate u_i^* according equation 6.
- Calculate the divergence of the provisional velocity field u_i^*.
- Solve the equation 8 for the "pressure field ".
- Calculate the pressure gradient in each direction.
- Correct the provisional velocity u_i^* to obtain the divergence-free velocity field u_i^{n+1}, equation 7.

Boundary condition for the pressure equation

The boundary condition for the pressure is obtained by projecting the correction equation 7 over the boundary Γ,

$$\left(\frac{\partial p}{\partial n}\right)_\Gamma^{n+1} = \frac{1}{\Delta t}(u_\Gamma^{n+1} - u_\Gamma^*).n \tag{9}$$

It can be shown that the solution of the Poisson pressure equation is independent of u^* and then, by choosing it, we can cancel the right hand side of equation 9, obtaining an homogeneous or Neumann condition for the pressure.

A discussion and comparison of the boundary conditions for the pressure equation for various schemes was recently published by Williams and Baker [7].

The compatibility condition is also needed

$$\int_\Gamma \nabla p.nd\Gamma = \int_\Omega \frac{1}{\Delta t}\nabla.u^* d\Omega \tag{10}$$

Voke and Potamitis, [6], remark the need of a zero surface integral of u^* over the boundaries to ensure uniqueness.

Spatial discretization

The domain is discretized with control volumes. Pressure nodes are located midway between the faces (Patankar's practice B, Patankar (1981), [5]). A staggered mesh is used for the velocities in the way of Harlow and Welsh [3]. The equivalent of second order finite differences is used on viscous and convective terms. The last choice imposes severe restrictions to the cell Peclet number, but the purpose of the calculation is to investigate fine scales features of the solution. Thus we are not expecting to use a coarse mesh here.

The diffusive component of the coefficients are calculated as

$$a_p\phi_p = \sum_{nv} a_{nv}\phi_n v + b \tag{11}$$

$$a_E = k_e\frac{A_e}{(\delta x)_e}, a_O = k_o\frac{A_o}{(\delta x)_o}, a_N = k_n\frac{A_n}{(\delta y)_n} \tag{12}$$

$$a_S = k_s\frac{A_s}{(\delta y)_s}, a_U = k_u\frac{A_u}{(\delta z)_u}, a_D = k_d\frac{A_d}{(\delta z)_d} \tag{13}$$

with

$$a_P = \sum_{np} a_{np} \tag{14}$$

$$b = a_P^0 \phi_P^0 \tag{15}$$

$$a_P^0 = \frac{\rho \Delta V}{\Delta t} \tag{16}$$

A linear velocity profile between nodes for evaluating the convective fluxes is used, practice which leads to central-finite- diference like coeficients, only shown for the x direction

$$ae_c = -\rho * u_e * \lambda_e \tag{17}$$

$$aw_c = -\rho * u_w * \lambda_w \tag{18}$$

Applying boundary conditions

Let us write the discretization equations for a control volume P for a general variable ϕ (velocity component or pressure). For the three momentum equations we use the explicit treatment. The pressure equation is not a parabolic equation and thus we do not need to advance it in time and we solve implicit.

The discretization equations are

$$a_P^0 \phi_P^{n+1} = a_P^0 \phi_P^n + \sum_{np} a_{np} \phi_{np}^* - (\sum_{np} a_{np}) \phi_P^* \tag{19}$$

If $* = n$, the scheme is explicit and if $* = n+1$, the scheme is implicit.

The same equation can be rewritten to show in more detail what it looks like near a boundary control volume B where the value of the neighbour node, denoted ϕ_B, may or may not be known

$$a_P^0 \phi_P^{n+1} = a_P^0 \phi_P^n + \sum_{np/B} a_{np} \phi_{np}^* + a_{nB} \phi_B^* - (\sum_{np/B} a_{np} + a_B) \phi_P^* \tag{20}$$

where the notations $\sum_{np/B}$ means summation over nodes nb not including node B.

On an implicit scheme (pressure)

$$a_P^0 \phi_P^{n+1} = a_P^0 \phi_P^n + \sum_{np/B} a_{np} \phi_{np}^{n+1} + a_{nB} \phi_B^{n+1} - (\sum_{np/B} a_{np} + a_B) \phi_P n + 1 \tag{21}$$

on an explicit scheme (velocity components)

$$a_P^0 \phi_P^{n+1} = a_P^0 \phi_P^n + \sum_{np/B} a_{np} \phi_{np}^n + a_{nB} \phi_B^n - (\sum_{np/B} a_{np} + a_B) \phi_P^n \tag{22}$$

Dirichlet boundary conditions

In the explicit case, knowing ϕ_B makes no difference with the other neighboring nodes, that are also considered as known, so nothing special is required to implement Dirichlet B.C..

On the implicit case, the term $a_{nB} \phi_B^{n+1}$ is known and is incorporated to the forcing term b together with $a_P^0 \phi_P^n$. Thus what we have now is

$$a_P^0 \phi_P^{n+1} - \sum_{np/B} a_{np} \phi_{np}^{n+1} + (\sum_{np/B} a_{np} + a_B) \phi_P^{n+1} = a_P^0 \phi_P^n + a_{nB} \phi_B^{n+1} \tag{23}$$

equation that, written in a more usual notation, is

$$a_P\phi_P^{n+1} - \sum_{np/B} a_{np}\phi_{np}^{n+1} = b \tag{24}$$

In a computer program the coefficients and other quantities like the forcing term b are computed without considering the boundary conditions. That means that the subroutine that generates the coefficient matrix does not distinguish between $\sum_{np/B} a_{np}\phi_{np}^{n+1}$ and $\sum_{np} a_{np}\phi_{np}^{n+1}$.

Thus in applying the boundary conditions one needs to correct previously calculated values. The forcing term correction is

$$bb + a_{nB}\phi_B^{n+1} \tag{25}$$

then one needs to substract term $a_{nB}\phi_B^{n+1}$ from $(\sum_{np} a_{np}\phi_B^{n+1}$ and makes zero the coefficient a_{nB}.

Neumann boundary conditions

Neumann boundary conditions are applied following Patankar, [5]. No flux on the boundaries implies no influence on the near boundary node P of the boundary node value at node B. Patankar's derivation leads to a zero value for the coefficient a_{nB}. Equations 20 reads then

$$a_P^0\phi_P^{n+1} = a_P^0\phi_P^n + \sum_{np/B} a_{np}\phi_{np}^* - (\sum_{np/B} a_{np})\phi_P^* \tag{26}$$

On an explicit scheme its goes

$$a_P^0\phi_P^{n+1} = a_P^0\phi_P^n + \sum_{np/B} a_{np}\phi_{np}^n - (\sum_{np/B} a_{np})\phi_P^n \tag{27}$$

Again the Euler solver programed doesn't take into account in any special way the boundaries. Thus, a $\sum_{np} a_{np}\phi_{np}^{n+1}$ cycle is done instead of what is needed: a $\sum_{np/B} a_{np}\phi_B^{n+1}$ cycle. The way to solve this problem is to make $a_{nB} = 0$ before starting the solution.

On an implicit scheme

$$a_P^0\phi_P^{n+1} = a_P^0\phi_P^n + \sum_{np/B} a_{np}\phi_{np}^{n+1} - (\sum_{np/B} a_{np})\phi_P^{n+1} \tag{28}$$

As always, the coefficient a_P was calculated as

$$a_P = (a_P^0 + \sum_{np} a_{np}) \tag{29}$$

so it needs to be corrected

$$a_P = a_P - a_{nB} \tag{30}$$

Then the coefficient a_{nB} is zeroed taking the unknown ϕ_B out of the equations.

Time integration for H_i

We have used first, second and third order Adams-Bashforth time explicit integration for the H_i term, so

$$H_i(n) = H_i^n \tag{31}$$

$$H_i(n, n-1) = (\frac{3}{2}H_i^n - \frac{1}{2}H_i^{n-1}) \tag{32}$$

$$H_i(n, n-1, n-2) = (\frac{23}{12}H_i^n - \frac{16}{12}H_i^{n-1} + \frac{5}{12}H_i^{n-2}) \quad (33)$$

Program organization

The most important procedure calculates the discretization coefficient. It is done in two stages (subroutines). In one, the coefficients are calculated over the whole domain including near boundaries control volumes. In the second, the coefficients are modified according to boundary conditions. The other important procedure is the solver itself.

Without the pressure term, the momentum equations in each direction are convection-diffusion type equations, and do not differ in more but in the variables names for the equation corresponding to the convection diffusion of a scalar. But, because of staggering, for each direction a different set of coefficients is required, as they are calculated on a different (staggered) mesh. For any additional scalar, the coefficients are calculated over the basic mesh.

Momentum equation coefficients are calculated by the three subroutines for staggered meshes. Pressure equation coefficients are caculated appart, over the basic mesh.

The boundary coefficients' modifications are done respectively in a set of subroutines. The explicit solver works on each staggered grid for the transient explicit calculation. The Adams-Bashford method have been implemented in a way that it is easy to use it of any order. CGSTAB and SIPSOL methods were used to solve the pressure equation. Other methods can be used as well. A subroutine that implements a variety of multigrid methods, including the full multigrid method, was written (Cardón, 2000), and incorporated to the program. The use of the multigrid method in conjunction with the present program was not yet fully tested.

The program flow is like follows

- main program
  - open imput an output files, read data files, read mesh
  - compute nodes positions and other geometrical cuantities
  - selected algorithm
    - explicit
      - first order
      - second order
      - third order
    - implicit (no functional yet)
      - two steps
      - four steps

The solution algorithm flow is like follows

- explicit main algorithm
  - compute pressure matrix coefficients
  - do over time steps
    - do over each direction
      - compute momentum equation coefficients
      - introduce boundary conditions
      - solve
      - correct boundary values over Neumann boundaries (for graphics)
      - correct boundary values to satisfy global mass balance
      - end for
  - compute the pressure source term
  - solve the Poisson equation
  - correct the velocities

– end do

TEST PROBLEM AND VALIDATION RESULTS

The two dimensional entrance length problem in a channel was solved as the first test case. Two dimensionality is imposed through the boundary conditions while the calculation is done on a three dimensional domain.

Let's define the $-z$ as the streamwise direction, y as the transversal direction, and x as the spanwise direction. Two dimensional flow over the $z-y$ plane will be imposed, therefore no change in the spanwise directions is allowed. In particular over the lateral planes at $x = 0$ and $x = L_x$, the x derivatives are zero. With the same effect it is possible to set $u = 0$.

The inflow and outflow sections of the channel are the cross sections normal to the streamwise direction at $z = L_z$ and $z = 0$ respectively. On the inflow plane we set $w = -1$ and on the outflow plane we require that $\frac{\partial w}{\partial z} = 0$. The same requirement should be enough for the other two velocities, but a stronger condition setting $v = 0$ and $w = 0$, was preferred.

Non slip and inpenetrability is applyed over the planes at $y = 0$ and $y = L_y$.

Results shown for the test problem were done over an $40 \times 40 \times 40$ grid, with $dt = 0.0001$, 1000 time steps (not steady state reached), and solve with CGSTAB method for small Reynolds number. Calculations were done on a AMD-K6-2 processor PC with 256MB of RAM, with aproximately 3.3 sec per time step.

The plate shows the 3D profiles of the three components of the velocity near the outflow plane. The projection of isolines of the streamwise componet of the velocity over the transversal plane is also shown as well as the projection over the same plane of the velocity vector.

From them it is clear that the streamwise component of the solution is perfectly symmetric and that the tranversal component of the velocity is antisymmetric, as they are meant to be. The order of the u velocity is $\sim 10^{-8}$, and can be considered zero.

Other features of the flow are shown in the plate; the development of the boundary layer is clearly shown in the figure as well as the inner invicid core.

To asses the correctness of the numerical solution, the steady state outflow velocity profile a over a central plane (not shown) is compared with the analytic solution for the fully developed channel flow, e.i. the Poiseuille solution with a good fit.

REFERENCIAS

[1] L. Cardón. Solving the poisson equation with multigrid methods. *Mecánica Computacional*, XIX:91–96, 2000.

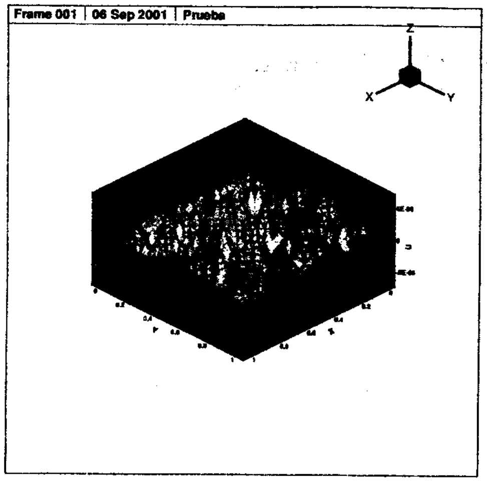
[2] F.H. Harlow and J.E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids*, 12(8):2182–2189, 1965.

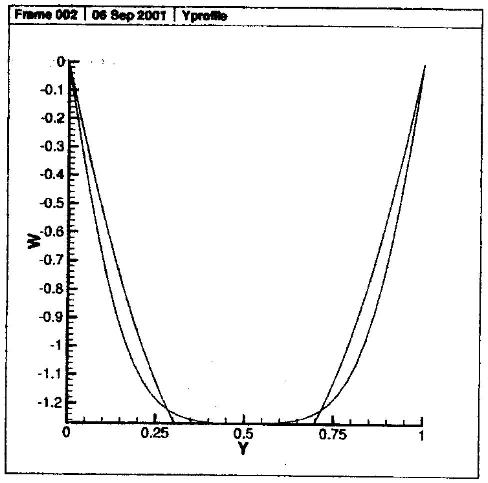
[3] T. Kuhen. Personal comunication.

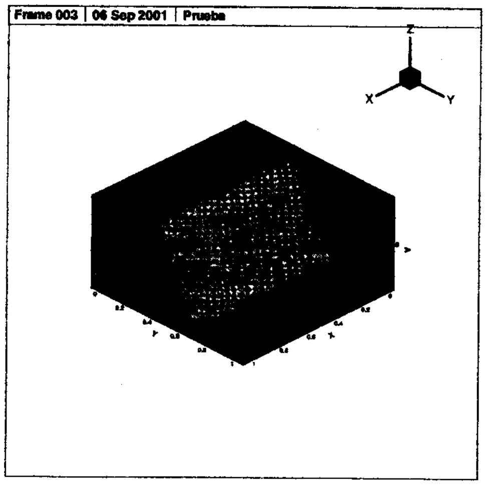
[4] S. V. Patankar. A calculationprocedure for two-dimensional elliptic situations. *Numerical Heat Transfer*, 4:409–425, 1981.

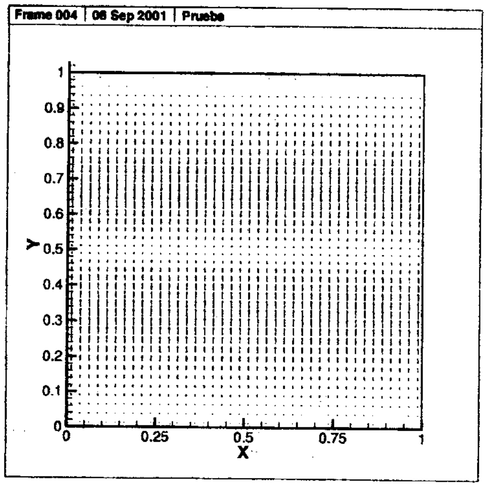
[5] P.R. Voke and S. G. Potamitis. Numerical simulation of a low-reynolds-number turbulent wake behind a flat plate. *International Journal for Numerical Methods in Fluids*, 19:377–393, 1994.

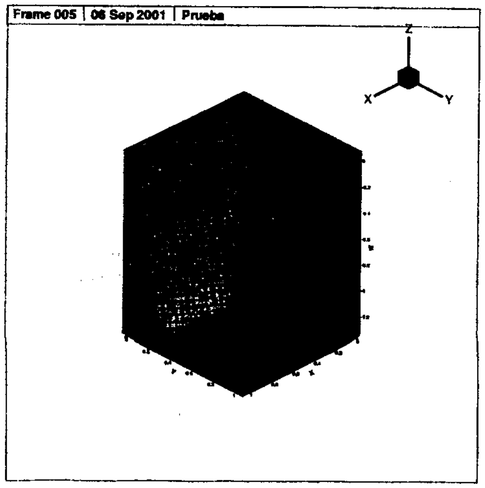
[6] P.T. Williams and A.J. Baker. Incompressible computational fluid dynamics and the continuity constraint method for the three-dimensional navier-stokes equations. *Numerical Heat Transfer, Part B*, 29:137–273, 1996.

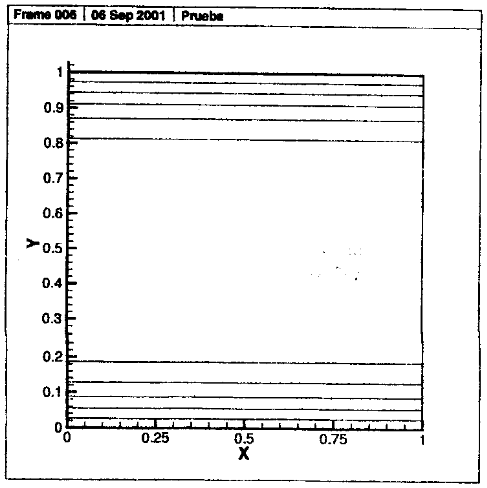

Frame 001 | 06 Sep 2001 | Prueba
Z
X
Y


Frame 002 | 06 Sep 2001 | Yprofile
W
Y
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
-0.8
-0.9
-1
-1.1
-1.2
0
0.25
0.5
0.75
1


Frame 003 | 06 Sep 2001 | Prueba
Z
X
Y


Frame 004 | 06 Sep 2001 | Prueba
Y
X
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.25
0.5
0.75
1


Frame 005 | 06 Sep 2001 | Prueba
Z
X
Y


Frame 006 | 06 Sep 2001 | Prueba
Y
X
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.25
0.5
0.75
1