Blucher Mechanical Engineering Proceedings
May 2014, vol. 1 , num. 1
www.proceedings.blucher.com.br/evento/10wccm

10th World Congress on Computational Mechanics
8-13 July 2012 • São Paulo • Brazil

ADAPTIVE NUMERICAL SIMULATIONS OF A TURBULENT JET

P. C. Calegari[1], A. M. Roma[1], G. C. K. Filho[2], L. C. C. Santos[1]

[1] Department of Applied Mathematics, Mathematics and Statistics Institute at the University of São Paulo (priscila@ime.usp.br)

[2] Department of Mechanics, Polytechnic School at the University of São Paulo

Abstract. *This work is concerned with assessing the performance of a numerical method, which combines an adaptive mesh refinement technique, an implicit-explicit time stepping strategy, and a linear multilevel-multigrid methodology, when applied to a challenging real-life problem: a three-dimensional turbulent jet flow. Typically, whenever a moving fluid emerges from a narrow opening into an otherwise quiescent fluid, shear is created between the entering and the ambient fluids, causing fluid instabilities, turbulence, and mixing at downstream. Turbulent jets represent an important class of fluid flow phenomena which occurs in many instances both in environmental and in industrial applications such as waste water discharges into rivers, plumes from smokestacks, and flames on combustion nozzles. Mathematically, the fluid dynamics is modeled by the non-steady Navier-Stokes equations for a three-dimensional incompressible flow whose material properties vary. The turbulence modeling is given by the large eddy simulation approach for which a careful selection of the Smagorinsky constant is performed. To resolve accurately and efficiently sharp gradients, vorticity shedding, and localized small length scale flow features (e.g. the ones present in high turbulence regions), dynamic adaptive mesh refinements are employed which form a level hierarchy composed by a set of nested, Cartesian grid patches (block-structured grid). That spatial adaptation is used in conjunction with a variable time step, linearly implicit time integration scheme, based on a semi backward difference formula (SBDF), especially designed to work with the non-linear diffusive term arising from the turbulent viscosity. The NS solver is based on an increment-pressure projection method. Information on how often the mesh adapts itself, on the number of computational cells in use, on the stability and size of the integration time step, and on the behavior of the multilevel-multigrid solvers is collected, showing the performance and testing the capabilities of the overall methodology.*

Keywords: *Adaptive mesh refinement, Implicit-explicit scheme, Large eddy simulation, Multilevel-multigrid method, Projection method.*

1. INTRODUCTION

Adaptivity is an important component to be considered for efficient numerical solutions of partial differential equations. Many techniques appeared through the years and, nowadays, the term "adaptive mesh refinement" (or simply AMR) embraces an entire collection of approaches which spread to a variety of different application fields [12]. However, the key idea behind of all those approaches is still the same: to concentrate computational power where it is most needed by increasing the resolution in space on regions of special interest in the computational domain. As an example, in Computational Fluid Dynamics, one needs more grid resolution where there are immersed interfaces/bodies to describe accurately intricate geometry details, where there is high vorticity/turbulence, around boundary layers, and in the vicinity of other flow features of special interest (and with highly localized span).

In the present work, a combination of an AMR strategy [2,3,4,16] with an implicit-explicit (IMEX [1]) projection method [10,14,17] have been applied to solve the equations modeling an incompressible turbulent jet. Section 2 presents the mathematical model which describes the problem, Section 3 the numerical methodology in use, and Section 4 presents the numerical results for several performance tests designed to expose the strengths and weaknesses of the approach adopted. Section 5 concludes the text with remarks on the pros/cons, pointing to directions to improve even further the efficiency of the overall methodology.

2. MATHEMATICAL MODEL

Mathematically, the fluid dynamics is modeled by the non-steady Navier-Stokes (NS) equations for a three-dimensional incompressible flow whose material properties may vary,

$$\frac{\partial u_j}{\partial x_j} = 0, \tag{1}$$

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j}\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + f_i, \quad 1 \leq i \leq 3, \tag{2}$$

where ρ is the specific mass, the vector (u_1, u_2, u_3) is the fluid velocity, p is the pressure, τ_{ij} is the viscous tensor,

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \quad 1 \leq i, j \leq 3, \tag{3}$$

with μ the dynamic viscosity (molecular plus turbulent terms, detailed next). In (2), f_i represents the sum of all external forces acting on the fluid. Note that the summation convention is in use. The mathematical model (1)-(2) requires boundary and initial conditions which will be detailed in Section 4. From here on, ρ (but not μ, needed in the turbulence modeling) will be assumed to be a constant.

The turbulence modeling is given by the large eddy simulation (LES) approach, which explicitly compute the largest structures of the flow (typically, structures larger than the finest computational mesh size), modeling the influence of the smaller scales [13]. The state variables in (1)-(2) are filtered, that is, decomposed into a sum of a resolved component plus a residual (or subgrid scale (SGS)) component. The filtered equations can be written in the standard form, with (2) containing the *residual-stress tensor* that arises from the residual motions. The closure of the set of equations is obtained by modeling the residual-stress tensor by *Smagorinsky model* [13], for which the turbulent viscosity is

$$\mu_t = (C_s \Delta)^2 \mathcal{S}, \tag{4}$$

where $\Delta = (\Delta x_1 \Delta x_2 \Delta x_3)^{\frac{1}{3}}$ is the filter size, $\mathcal{S}$ is the characteristic filtered rate of strain given by

$$\mathcal{S} = \sqrt{2 S_{ij} S_{ij}}, \text{ with } S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{5}$$

and C_s is known as *Smagorinsky constant*. Its value is chosen between 0.1 to 0.18 [13].

3. NUMERICAL METHODOLOGY

3.1. Discretization in time

Discretization in time is performed by a variable time step, linearly-implicit time integration scheme based on a second-order, two-step Gear's Method [10,17] which is capable of handling the non-linear diffusive term arising from the turbulent viscosity. The idea is to rewrite (2) as

$$\frac{\partial u_i}{\partial t} = \frac{\lambda}{\rho}\frac{\partial^2 u_j}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial p}{\partial x_i} + \frac{\tilde{f}_i}{\rho}, \quad 1 \le i \le 3, \tag{6}$$

where $\tilde{f}_i = \frac{\partial \tau_{ij}}{\partial x_j} - \lambda \frac{\partial^2 u_j}{\partial x_j^2} - \rho\left(u_j \frac{\partial u_i}{\partial x_j}\right) + f_i$, and λ is an *ad hoc* constant chosen through numerical experimentation. Based only on Gear's Method, all terms in the right hand side of (6) should be treated implicitly. However, here, $\tilde{f}_i$ is extrapolated in time. The resulting time integration scheme is

$$\frac{\partial u_j^{n+1}}{\partial x_j} = 0, \tag{7}$$

$$\frac{a_2 u_i^{n+1} + a_1 u_i^n + a_0 u_i^{n-1}}{\Delta t} = \frac{\lambda}{\rho}\frac{\partial^2 u_j}{\partial x_j^2}^{n+1} - \frac{1}{\rho}\frac{\partial p}{\partial x_i}^{n+1} + \frac{b_1 \tilde{f}_i^n + b_0 \tilde{f}_i^{n-1}}{\rho}, \quad 1 \le i \le 3, \tag{8}$$

where $a_0 = \Delta t^2/(\Delta t_0 \Delta t_1)$, $a_1 = -\Delta t_1/\Delta t_0$, and $a_2 = (\Delta t_0 + 2\Delta t)/\Delta t_1$, $b_0 = -\Delta t/\Delta t_0$ and $b_1 = \Delta t_1/\Delta t_0$, with $\Delta t = t^{n+1} - t^n$, $\Delta t_0 = t^n - t^{n-1}$, and $\Delta t_1 = \Delta t_0 + \Delta t$.

The pressure-velocity coupling present in (7)-(8) is treated with an increment-pressure projection method [7,10,14], such that one solves in turn the equations

$$\frac{a_2 u_i^{\star,k} + a_1 u_i^n + a_0 u_i^{n-1}}{\Delta t} = \frac{\lambda}{\rho}\frac{\partial^2 u_j}{\partial x_j^2}^{\star,k} - \frac{1}{\rho}\frac{\partial p}{\partial x_i}^{n+1,k-1} + \frac{b_1 \tilde{f}_i^n + b_0 \tilde{f}_i^{n-1}}{\rho}, \quad 1 \le i \le 3, \tag{9}$$

$$u_i^{\star,k} = u_i^{n+1,k} + \frac{\Delta t}{a_2 \rho}\frac{\partial q}{\partial x_i}, \tag{10}$$

$$\frac{\partial u_j^{n+1,k}}{\partial x_j} = 0, \tag{11}$$

where $u_i^{\star,k}$ is a "preliminary" velocity field obtained from the diffusion equation (9), and q is the *pressure increment* obtained from (10)-(11). Together, they give rise to an elliptic equation for q, being responsible for enforcing the incompressibility constraint. Once q is determined, updates for the pressure and velocity fields are given by $p^{n+1,k} = p^{n+1,k-1} + q$, and by $u_i^{n+1,k} = u_i^{\star,k} - (\Delta t/a_2\rho)\,\partial q/\partial x_i$. Note that the iteration in k is needed to split the solution into two parts, one for the velocity and one for the pressure increment. The number of iterations per time step is fixed to two (that is, $k = 1, 2$), which is enough to get a second-order scheme in time, and $p^{n+1,0} = p^n$ is the initial approximation taken for the pressure.

To complete the description of the integration in time, a word is needed on the time-step stability constraints. From numerical experimentation, λ is taken in (6) such that one succeeds in relaxing the parabolic stability constraint induced by the viscous dissipation term (for the jet considered here, $\lambda = 2||\mu||_\infty$ has proven to be sufficient). In this context, the only stability constraint left comes from the explicit discretization of the advection term,

$$\Delta t \leq 0.8 \min \left\{ \frac{\Delta x_i}{||u_i||_\infty} \right\}, 1 \leq i \leq 3. \tag{12}$$

3.2. Discretization in space

Discretization of the flow domain follows closely the adaptive mesh refinement technique first proposed by Berger and Colella [2] with grid adaptation given by Berger and Rigoutsos' Algorithm [4]. Such technique is based on a grid structure given by a set of nested, Cartesian grid patches which forms a level hierarchy, usually referred to as *composite grid*. Grid patches obey certain rules, targeting for easiness in their construction and efficiency in their use:

1. a fine grid starts and ends at the corner of a cell in the next coarser grid, and
2. all fine grid cells at level l must be surrounded either level l cells or by level $l-1$ cells except when it touches the border of the physical domain.

Note that, differently from [2], in the implementation being used here [10], grid patches are *disjoint* from each other, that is, given two grid patches belonging to the same hierarchical level, they do not share interior computational cells. Figure 1 shows an example a three-dimensional composite grid and Figures 2(a)-2(b) show the grid patches that compose this three-dimensional mesh. It is important to highlight that it is an *essential part* of this technique the existence of *ghost* computational cells, which form layers around each grid patch (see a simple 2D example given by Figure 3). Ghost cells furnish boundary conditions from polynomial interpolations of values at neighboring coarse/fine levels. Another important difference regarding the original work by Berger and Colella [2] is that, here, there is no time-step refinement: the numerical solution of the problem, on all grid patches, in all levels, evolves with the same time step from the finest level. The reason is that, for incompressible flows, discontinuities have no finite speed of propagation since the incompressibility constraint couples the solution (through the pressure field) at every point of the domain at every instant of time (hence, no subdomain may evolve in time separately).

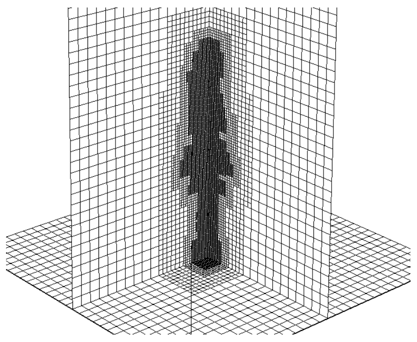
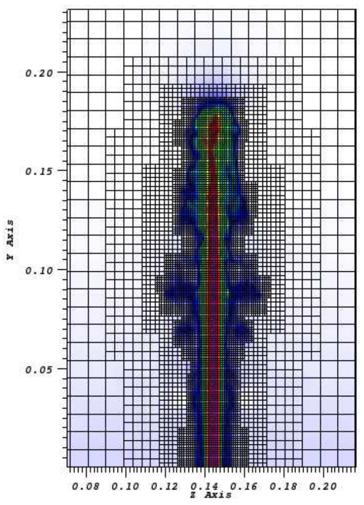
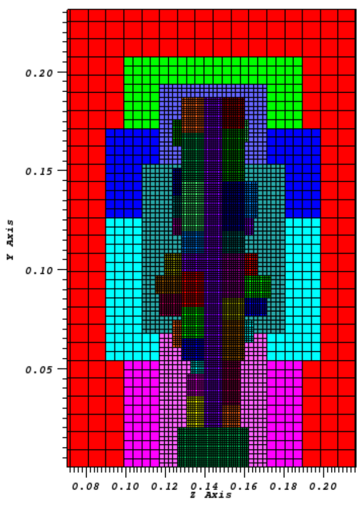


Figure 1. 3D-mesh.



(a) Levels.



(b) Patches.

Figure 2. Refinement mesh details.

On the composite grids, variables are placed in a "marker and cell" (MAC) fashion: scalars at the cell centers (e.g. specific mass and viscosity), and vectors have their components at cell faces. Standard second-order finite difference operators are employed in the discretization of gradient, divergent, and stress tensor differential operators, including for the nonlinear transport term [10,11].

Composite grids have been employed in several contexts [2,3,8,10,14,16,17]. In the present work, a variety of performance tests employing an available computer implementation of (9)-(12) [10], reveal efficiency strengths and weaknesses for the application considered.

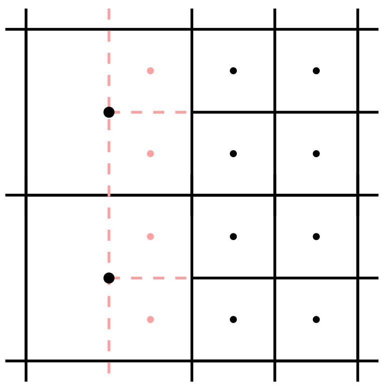


Figure 3. Ghost cells 2D-mesh.

3.3. Linear multilevel-multigrid method

Once the discretizations in space have been introduced, the pressure-increment projection method in use requires the solution of eight linear systems per time step (four for $k = 1$ and four for $k = 2$): six of them are of parabolic type (for the velocity components), and two are of elliptic type (for the pressure-increment). To solve these linear systems, an *in house implementation* of a multilevel-multigrid method is employed [6,10,14,15]. Here, "multilevel" refers to the fact that refinement levels belonging to the grid structure are also considered to be multigrid levels. It is important to recall that refinement levels differ from usual multigrid levels in that they do not necessarily cover the entire domain of computation. The multilevel-multigrid used was the V-cycle with convergence criterium $O(\Delta x^2)$ where $\Delta x = \min\{\Delta x_i\}$ is the mesh size, with $1 \leq i \leq 3$. Details of implementation can be found in [10,11].

4. NUMERICAL RESULTS

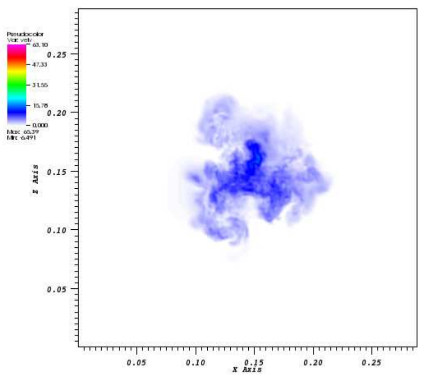
The flow domain is given by the Cartesian product $[a_1, b_1] \times [a_2, b_2] \times [a_3, b_3]$, a parallelepiped, where $a_1 = a_2 = a_3 = 0$, $b_1 = b_3 = 0.288\ m$, and $b_2 = 0.576\ m$. Initially, the fluid is at rest with null pressure, and the inflow boundary contains in its center a main jet with diameter $d_0 = 7.2\ mm$ which is surrounded by a secondary jet whose diameter is $d_1 = 18.2\ mm$. More specifically, the inflow boundary conditions for the velocity are given by $u_{1,in} = u_{3,in} = 0$, and $u_{2,in} = u_{d_0} + u_{d_1}$, with

$$\begin{aligned} u_{d_0} &= \frac{v_2 + v_1}{2} - \left(\frac{v_2 - v_0}{2}\right) \tanh\left[25\left(\frac{r_j}{r_0} - \frac{r_0}{r_j}\right)\right], \\ u_{d_1} &= \frac{v_1 + v_0}{2} - \left(\frac{v_1 - v_0}{2}\right) \tanh\left[25\left(\frac{r_j}{r_1} - \frac{r_1}{r_j}\right)\right], \end{aligned} \tag{13}$$

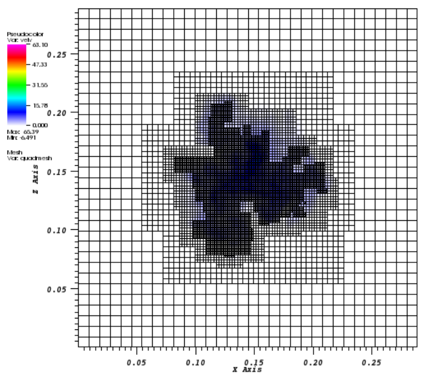
where $r_j = \sqrt{(x_3 - 0.5 \cdot b_3)^2 + (x_1 - 0.5 \cdot b_1)^2}$, $v_0 = 0.9\ m/s$, $v_1 = 11.4\ m/s$, and the main jet velocity $v_2 = 49.6\ m/s$, where r_1 e r_0 are, respectively, $d_1/2$ and $d_0/2$.

At the inflow boundary, the turbulence is modeled by $0.01\,\omega\,u_{2,in}$, with ω being a random number in $[0, 1]$ and homogeneous Neumann boundary condition is adopted for the pressure. At the other computational boundaries, one has homogeneous Neumann boundary conditions for the velocity and Dirichlet boundary conditions for the pressure increment [5]. The Reynolds number for the simulation being considered is approximately 2.0×10^4.

The computation employs a composite grid having a base level with $32 \times 64 \times 32$ computational cells and three refinement levels (four levels in total). The finest level has $\Delta x_1 = \Delta x_2 = \Delta x_3 = 1.125 \times 10^{-3}\ m$. The refinement criterium is based in the maximum norms of the viscous tensor and of the turbulent viscosity, constrained to a 85% efficiency (i.e. the ratio between the number of cells in need of refinement over the total number of cells that end up being refined to get a grid patch). A detail of the flow on the plane $x_2 = 0.432\ m$ and the refined mesh employed can be seen in Figure 4(a) and Figure 4(b), respectively. Figure 5(a) shows the component of the velocity along the flow direction on the plane $x_1 = 0.144\ m$ with a zoom of the refined mesh (Figure 5(b)).

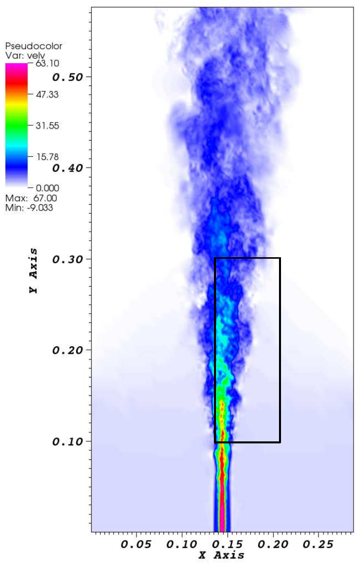


(a) Velocity in the flow direction.

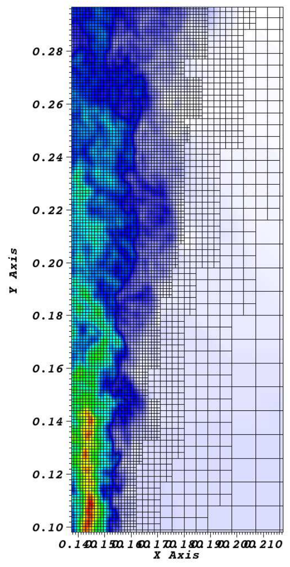


(b) Refined mesh.

Figure 4. Detail of the velocity along the flow direction with its refined mesh in the plane $x_2 = 0.432\ m$.



(a) Velocity in the direction flow.



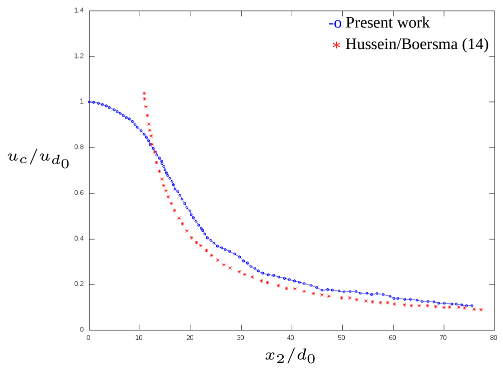
(b) Zoom of the box in the Figure 5(a).

Figure 5. Velocity along the flow direction with the refined mesh on the plane $x_1 = 0.144\,m$.

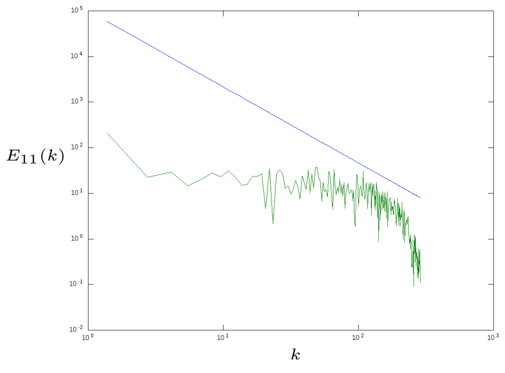
The velocity decay along the centerline of the computational domain for turbulent jets [5,9] is given by

$$\frac{u_c}{u_{d_0}} = \frac{B_u d_0}{x_2 - x_0}, \tag{14}$$

where u_c is the velocity in the flow direction along the centerline, u_{d_0} is the velocity of the main jet, and the parameters B_u and x_0 are 5.8 and $4d_0$, respectively [5]. The expression above is obtained from velocity measurements in a turbulent jet [9]. Figure 6(a) compares the decay obtained in the present work with the decay given by (14). Figure 6(b) shows the turbulent spectrum and the decay at $-\frac{5}{3}$ slope. In the turbulence model, the Smagorinsky constant was set to be $C_s = 0.15$.



(a) Velocity decay in the x_2-centerline.



(b) The turbulent energy cascade in the point $(0.144\,m, 0.324\,m, 0.144\,m)$.

Figure 6. The velocity decay in the centerline and the turbulent energy cascade.

Next, several performance tests aiming to expose strengths and weaknesses of the methodology adopted, applied to the turbulent jet flow described above, are reported below. Tests are performed to evaluate the linear multilevel-multigrid method, the AMR and time-step strategies, and other implementation aspects.

Regarding the multigrid methodology, by timing individual parts of the code, it is observed that 69.5% of the computational time in each time step is spent in resolving the pressure correction equation, an ill conditioned elliptic equation. Typically, 15 V-cycles are performed for elliptic equation. Relaxing the multilevel-multigrid convergence criterium to O(Δx), 12 V-cycles are performed. By the few difference it was decided to maintain the convergence criteria of O(Δx^2), as described in Section 3.3.

Regarding the discretization in space, to fairly compare the AMR approach to the uniform mesh approach, one must estimate how the performance on a uniform mesh would be, if that uniform mesh had spacing equal to the finest level in the AMR grid (to achieve the same accuracy as on the AMR grid). Initially, taking few integration time steps on both meshes, in the same machine, revealed that the AMR approach can be 190 times faster and in the final time steps (when the jet is well developed), 5 times faster. The advantage of the AMR approach over the uniform grid approach is directly connected to the number of computational cells used. Note that only a small percentage of processing time is spent in managing the AMR grid (e.g. switching grids, initializing). Typically, less the 2% of that time is spent. Attempts made to decrease the computational time by relaxing the CFL time-step constraint (12) (e.g. by switching from $\|u_i\|_\infty$ to $\|u_i\|_2$) led to unstable solution. In few integration time steps the residue in the multilevel-multigrid decreases slowly reaching the maximum V-cycles number. In this case the $\Delta t =$ O$(\Delta x^{0.73})$.

Regarding compilers, the computation time of a single time step employing both the **gnu** (version 4.4.4) and **intel** (version 11.0) compilers using the flag “-O3”, on an Intel based machine (Intel Core 2 Quad processor 2.33GHz and 16Gb RAM) are measured. The best performance is achieved by the **gnu** compiler, which has proven to be 1.2 times faster than the **intel** compiler, for the test case considered. Profiling the code using the flag “-pg” and executing *gprof*, it is possible to identify the parts of the current implementation which spend most of the processing time. The results show that the set of subroutines/functions involved in the computation of ghost cell values is the one called the most, spending about 43% of the cpu time. One reason is the large number of grid patches needed when the jet is well developed (see the Figure 2(b)). To increase efficiency, by decreasing the number of patches while keeping the jet region well resolved, a grid patch composed by a single parallelepiped along the flow direction (a “tube”) is employed to replace a large set of smaller grid patches, thus decreasing the number of calls needed to set up ghost cell values. Table 1 contains the number of cells used on AMR, on AMR (with a tube), and on uniform meshes. The column “Total” counts all kinds of cells employed (that is ghost, interior, on multigrid levels below base level, and underneath finer levels), the column “Refinement levels” excludes those at multigrid levels below base level, and the column “Visible” also excludes those covered cells, underneath refinement patches. Employing the tube strategy, the time spent in a single time step is 1.27 times faster than the original strategy, spending about 63% in solving for the elliptic equation (increment pressure equation).

Table 1. Number of cells on several grid situations.

Mesh	Total	Refinement levels	Visible
AMR	8.69×10^6	3.44×10^6	3.01×10^6
AMR (with a *tube*)	7.41×10^6	3.39×10^6	2.92×10^6
Uniform	41.5×10^6	33.5×10^6	33.5×10^6

5. CONCLUSION

The performance of a numerical methodology based on an adaptive mesh refinement technique coupled with a semi-implicit time integration scheme is assessed by conducting a series of numerical experiments while solving a turbulent jet flow. Aspects regarding the multilevel-multigrid methodology employed, the use of AMR and time step scheme, and other implementation aspects are explored. One of the strongest sides is the capability of achieving high grid resolution locally, only around regions which bares special interest. If those regions grow in time, the number of grid patches increases and, beyond a certain point, that same capability becomes one of its downsides: the overload of computer time at coarse-fine grid interfaces (interpolation at ghost cells). Numerically, the multigrid methodology employed requires stop criteria on the order of O(Δx^2), and typically 15 V-cycles are performed for elliptic equation (pressure increment) and 4 V-cycles, for parabolic equations (velocity components). The time spent in the solution of the linear system of the elliptic equation is bottleneck of the code. At least that solution should be performed in parallel. The linear CFL time step constraint results, typically, $\Delta t = \mathrm{O}(\Delta x^{1.65})$. Both **intel** and **gnu** compilers have been tried out, with **gnu** performing better for the current code implementation and for the test under consideration.

Acknowledgements

For their valuable discussions and helpful insights, the authors are in debt with Dr. Aristeu da Siveira-Neto, Dr. Millena M. Villar, and their group from the MFLab/FEMEC-UFU, and Dr. Rudimar L. Nós. Financial support was provided in part by CAPES, and by CNPq grants # 142071/2007-2, # 307348/2008-3, and # 309433/2011-8. Simulations were performed at the Applied Mathematics Laboratory (LabMAP) at IME-USP.

6. REFERENCES

[1] Ascher U. M., Ruuth U. M., Wetton B. T. R., "Implicit-explicit methods for time-dependent partial differential equations". *SIAM Journal on Numerical Analysis.* 797-823, 1995.

[2] Berger M. J., Colella P., "Local adaptive mesh refinement for shock hydrodynamics". *Journal of computational physics.* 82(1), 64-84, 1989.

[3] Berger M. J., Oliger J., "Adaptive mesh refinement for hyperbolic partial differential equations". *Journal of computational Physics.* 53, 484-512, 1984.

[4] Berger M. J., Rigoutsos I., “An algorithm for point clustering and grid generation”. *Systems, Man and Cybernetics, IEEE Transactions on.* 21(5), 1278-1286, 1991.

[5] Boersma B. J., Brethouwer G., Nieuwstadt F. T. M., “A numerical investigation on the effect of the inflow conditions on the self-similar region of a round jet”. *Physics of fluids.* 10(4), 899-909, 1998.

[6] Briggs W. L., McCormick S. F., “A multigrid tutorial”. 2000.

[7] Chorin A. J., “Numerical solution of the Navier-Stokes equations”. *Math. Comp.* 22, 745-762, 1968.

[8] Griffith B. E., Hornung R. D., McQueen D. M., Peskin C. S., “An adaptive, formally second order accurate version of the immersed boundary method”. *Journal of computational physics* 223, 10-49, 2007.

[9] Hussein H. J., Capp S. P., George W. K., “Velocity measurements in a high-Reynolds-number, momentum-conserving, axisymmetric, turbulent jet”. *Journal of Fluid Mechanics* 258(1), 31-75, 1994.

[10] Nós R. L., Ceniceros H. D., Roma A. M., “Three-dimensional, fully adaptive simulations of phase-field fluid models”. *Journal of computational physics.* 229(17), 6135-6155, 2010.

[11] Nós R. L., “Simulações de escoamentos tridimensionais bifásicos empregando métodos adaptativos e modelos de campo de fase”, PhD thesis (in Portuguese), University of São Paulo, 2007.

[12] Plewa, T., Linde, T. J., Weirs, V. G., “Adaptive mesh refinement, theory and applications: proceedings of Chicago Workshop an Adaptive Mesh Refinement Methods”. 2005.

[13] Pope S. B., “Turbulent flows”. 2000.

[14] Roma A. M., Peskin C. S., Berger M. J., “An adaptive version of the immersed boundary method”. *Journal of computational physics.* 153(2), 509-534, 1999.

[15] Trottenberg U., Oosterlee C. W., Schüller A., “Multigrid”. 2001.

[16] SAMRAI *Structured Adaptive Mesh Refinement Application Infrastructure* https://computation.llnl.gov/casc/SAMRAI/index.html

[17] Villar M. M., Ceniceros H. D., Roma A. M., Silveira-Neto A. “A robust, fully adaptive hybrid level-set/front-tracking method for two-phase flows with an accurate surface tension computation”. *Communications in Computational Physics* 8(1), 51-94, 2010.