Mesh block refinement technique for incompressible flows in complex geometries using Cartesian grids

C. Georgantopoulou[1], G. Georgantopoulos[2] & S. Tsangaris[1]
[1]*Fluids Section, School of Mechanical Engineering, National Technical University of Athens, Greece*
[2]*Aerodynamic Lab, Hellenic Air Force Academy, Greece*

Abstract

The present study performs a block refinement technique for the simulation and computation of flows inside domains of arbitrary shaped bounds. The discretisation of the physical domains is achieved by the use of Cartesian grids only. The curvilinear geometries are approached in Cartesian co-ordinates by Cartesian grid lines. In order to achieve the best approach of the original contour, we choose the saw tooth method to determine the appropriate approximated Cartesian points. The refinement method is based on the use of a sequence of nested rectangular meshes in which numerical simulation is taking place. The method is applied for the solution of the incompressible Navier–Stokes equations, for steady and laminar flows, based on a cell centre approximation projection. We present the numerical simulation of internal and external flows for different values of a Reynolds number. The utility of the algorithm is tested by comparing the convergence characteristics and accuracy to those of the standard single grid and BFC grid algorithms.
Keywords: grid generation, incompressible flows, nested grids, subgrids, numerical simulation.

1 Introduction

When the body-fitted structured curvilinear (BFC) grid approach came into solving flow problems, the use of Cartesian grids was almost abandoned. The benefit of BFC is that the boundary surface is fitted with a new co-ordinate line based on the body contour [1]. The main problem is that if you have to simulate


WIT Transactions on Modelling and Simulation, Vol 48, © 2009 WIT Press
www.witpress.com, ISSN 1743-355X (on-line)
doi:10.2495/CMEM090341

a complex multiple connected domain with sharp boundaries it is difficult to automatically generate a grid of good quality. Current algorithms in BFC are still strongly dependent on the problem to be solved and require a lot of computational and human time effort. So in order to avoid these partial problems, there has recently been a great development in Cartesian grids. In the Cartesian grid methods the numerical grid is generated automatically, containing simplified data structures and formulations for the numerical fluxes. The Cartesian grid generation was used by Clarke et al. [2] and Falle and Giddings [3] to calculate steady compressible flows [4]. Coirier and Powell [5] used a Cartesian methodology for steady transonic solution Euler's equations and in [6] performed accuracy and efficiency assessments of the method. It is a cell-centred method with an interesting treatment of boundary conditions. Smith and Johnston [7] develop a grid generation procedure that uses a Cartesian embedded unstructured approach for complex geometries.

Adaptive mesh refinement algorithms have been used extensively to solve a variety of problems in hyperbolic conservation laws and have more recently been extended to incompressible flows [8–10]. Wang [11] develops a quadtree-based adaptive Cartesian/Quadrilateral grid generator and flow solver based on cell cutting [12–14], and Deister et al. [15] presents a refined Cartesian grid based in octa-tree.

In the present paper we present a Cartesian grid approach based on a saw-tooth method for the curvilinear geometries bounds approximation. This technique is based on Chai et al. [16], where they present the saw-tooth Cartesian method for the heat transfer problem on a complex geometry. The emphasis is placed to present an improved accurate Cartesian approximation of curvilinear geometries and the corresponding fluid solution in comparison with those of the body fitted structured curvilinear method. We apply a nested refinement algorithm based on that of Jesse et al. [17], Martin and Collela [9], and Berger and Collela [18], in which refined regions are organized into unions of a small number of nested rectangular blocks. Refinement is performed in space and the method is cell-centred finite volume, which allows the use of a single set of cell-centred solvers. It is applied to steady, incompressible flow fields for the Navier–Stokes numerical simulation [19]. The flow solver is based on a pseudo-compressibility technique (Pappou and Tsangaris [20]).

2 Grid generation

The main problem in Cartesian grid generation for a curvilinear geometry is that we have to use a technique to create an approximate Cartesian bound as close to the initial curvilinear bound as possible. The new approximate bound is parted only by the use of grid lines, on either the x or z-axis. The method used is called saw-tooth and it has been chosen as the most appropriate for the finite volume cell-centered numerical simulation of flow fields.

In order to apply the saw tooth approximation we project the original contour of the curvilinear geometry onto a Cartesian grid. This complex contour is described by a set of data points on either the x or z-axis. The second step of the

procedure is the specification of the approximated Cartesian points for the representation of the geometry by using the saw tooth method. If an original data point is on the x-axis, we calculate the distance between this and its neighbouring grid nodes in the same direction (x). According to the smallest distance we choose the corresponding grid node as the Cartesian approximated point [21].

2.1 Mesh refinement technique

The main problem of the above method is that if you want to decrease the approximation error with the initial curvilinear geometry, you have to cluster the used uniform grid and in many cases a huge grid size is needed. In order to overcome this problem we choose a block refinement technique by the use of a hierarchical structured grid approach. The method is based on using a sequence of nested rectangular meshes in which numerical simulation is taking place. We simulate the domain based in as many refined grids as we need.

A physical domain's point can be contained in several grids. The solution of the variables in this point will be taken from the finest grid containing the point.

2.2 Block-nested refinement algorithm

The proposed nested algorithm contains several levels of grids. We create a coarse level at the beginning and we solve the domain. We name this coarse level *m=0* and each next refined sub-grid is named *m+1*. The coarsest grid is uniform in the x and z direction respectively. We define an integer refinement factor, such as that in [9], $I = dx_m / dx_{m+1} = dz_m / dz_{m+1}$. For convenience the above factor should be a second power.

As we have created the coarse grid we simulate the flow field and calculate the variables. We have already defined the limits of the refinement levels and we proceed with the calculation to the next refinement level. The sub-grids bounds must lie on a grid line of the previous level grid. As we use staggered grids and the variable values are expressed on the cell's centre, we consider pseudo-cells all around the physical domain and the sub-grids too. In this way we estimate the variables using interpolation between pseudo-cells and their neighbour cells. The pseudo-cells of each sub-grid *m* are lying on the level *m-1*. We continue this process for all the sub-grids. As we have fulfilled the simulation in all sub-grids and we have the flow field results at $m_{\max}$ level, we resolve the problem in the coarser levels again to ensure conservation. In this step of the procedure we have to be careful because we can apply the numerical simulation only in rectangular sub-grids. We find a new solution, this time by the influence of the fine levels. In addition we must satisfy both Dirichlet and Neumann matching conditions along coarse–fine and fine–coarse interfaces. That is why we give the velocity values, but we solve for pressure.

The grid algorithm is comprised of multiple levels. As we have already created the Cartesian approximate geometry bound, the grid generation and the numerical simulation procedure is as follows:

- Create a coarse Cartesian grid (level m=0), simulate and solve the flow field.
- Transfer the solution to the next grid level (m+1).
- Solve the flow field on the new sub-domain.
- Transfer the solution to the next level (m+2) with new boundary conditions.

(Repeat the procedure for all the levels)

- Simulate and solve the flow on the last sub-domain (level mmax).
- Transfer the solution to the coarser grid level (mmax-1) as its boundary conditions.
- Solve the sub domain with the influence of the refined grid results.

(Repeat the procedure for all the levels)

- Solve the coarsest-initial sub domain (level m=0).
- Take the solution of the variables by the finest grid.

2.3 Boundary conditions

If the grids are adjacent, the boundary conditions of one grid are provided by the other. If they are not adjacent, the boundary conditions are established by either the coarser level condition or by the physical boundary condition.

Let us consider that we have already solved into the initial coarse grid and we have to continue the numerical simulation into a sub-grid. In order to specify the boundary conditions at coarse grid and sub-grid interfaces, we represent $u^{m+1}(i,k)$ and $w^{m+1}(i,k)$, the values of the velocity components, on the sub-grid pseudo-cells. $u^{m}(l,n)$ and $w^{m}(l,n)$ are the corresponding coarse grid values into the physical domain. Every interpolation takes place either on the x-axis or on the y-axis. If we consider that we apply the new velocity values on the x-axis, (figure 1), interpolation is applied as follows:

$$u^{m+1}(i,k) = \frac{u^{m}(l,n) + u^{m}(l+1,n)}{2}$$

and

$$w^{m+1}(i,k) = \frac{w^{m}(l,n) + w^{m}(l+1,n)}{2}. \tag{1}$$

Also,

$$u^{m+1}(i,k) = u^{m+1}(i+1,k) = \ldots = u^{m+1}(i+I-1,k). \tag{2}$$

Therefore, if the refinement factor is set to be equal to 2, (I=2), the above relation becomes as below:

$$u^{m+1}(i,k) = u^{m+1}(i+1,k). \tag{3}$$

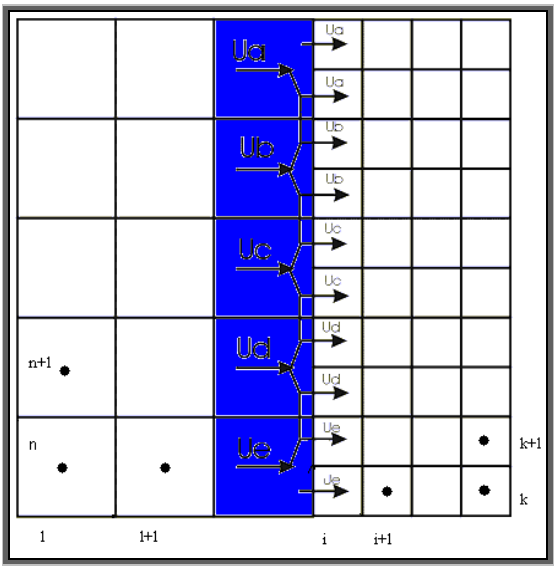

Figure 1: Linear interpolation in order to transfer the velocity values to a coarse–fine interface.

The relation between i and l is:

$$l = 2 * i - 1 . \tag{4}$$

As we have assigned the velocity values on the boundary bounds, we must apply a condition for the pressure. Assuming that we simulate for an axisymmetric flow, the pressure vertical derivative at the interface is estimated as follows:

$$\frac{\partial p}{\partial n} = n_x \left[\frac{1}{\mathrm{Re}} \left(\frac{\partial^2 u}{\partial y^2} + \frac{1}{y} \cdot \frac{\partial u}{\partial y} + \frac{\partial^2 u}{\partial x^2} \right) - u \cdot \frac{\partial u}{\partial x} - v \cdot \frac{\partial u}{\partial y} \right] + \\ + n_y \left[\frac{1}{\mathrm{Re}} \left(\frac{\partial^2 v}{\partial y^2} + \frac{1}{y} \cdot \frac{\partial v}{\partial y} + \frac{\partial^2 v}{\partial x^2} - \frac{v}{y^2} \right) - u \cdot \frac{\partial v}{\partial x} - v \cdot \frac{\partial v}{\partial y} \right] \tag{5}$$

where $\frac{\partial p}{\partial n}$ is the pressure vertical derivative, Re the Reynolds number, n_x and n_y the components of the unit normal vector and u and v the axial and the vertical velocity components respectively.

In order to transfer the boundary values through a fine–coarse interface, we once more apply interpolation and we estimate the pressure vertical derivative as above. With the same symbols, interpolation between the velocity values is:

$$u^m(l,n)=\frac{u^{m+1}(i,k)+u^{m+1}(i+1,k)+\ldots+u^{m+1}(i+I-1,k)}{I}, \tag{6}$$

where I is the refinement factor.

3 Results

In order to examine the accuracy of the above method, we present the numerical simulation of the flow field inside a stenosed tube and around a symmetric airfoil NACA0012. We chase out the accuracy of the results comparing them with the correspondence of a Cartesian uniform grid with the same base grid size.

3.1 Steady flow inside a stenosed tube

In stenosed tubes, the appearance of large recirculating zones, the high gradients of the wall shear stresses and the strong increase of the pressure drop are the main effects depending on the severity of the stenosis.

The grid generation and the numerical method that was described above were used for the calculation of steady flow inside a stenosed tube. The stenotic area is 0.25% of the inlet area. The numerical refinement grid used is level=1 and I=2, (base grid: 401x26).

The Re number, which was based on the maximum inlet velocity and the radius of the inlet, was set equal to 100. The boundary conditions are summarized as above in table 1. In order to control the accuracy of the proposed method, we simulated the current flow field by the use of two uniform grids: a curvilinear one sized 501x31 and a uniform Cartesian grid sized 401x26. A part of the block nested numerical grid is presented in figure 2, where it is shown that the whole domain is parted into four sub-grids. Two velocity profiles along the flow field are presented in figure 3 and the pressure distribution along the stenosed tube is presented in figure 4.

It seems that the convergence between the block nested algorithm results and the curvilinear grid results is very satisfied, while the Cartesian uniform grid results present greater relative error. It is important to note that we managed to approve the numerical results by the use of only one sub-grid with a refinement factor equal to two, around the stenotic area.

Table 1: Boundary conditions.

Lower bound:	$\frac{\partial u}{\partial y}=0, w=0, \frac{\partial p}{\partial y}=0$	*Inlet:*	$u=1-y^2, w=0, \frac{\partial p}{\partial x}=0$
Upper bound:	$u=w=0, \frac{\partial p}{\partial y}=0$	*Outlet*	$\frac{\partial u}{\partial x}=0, w=0, p=const$

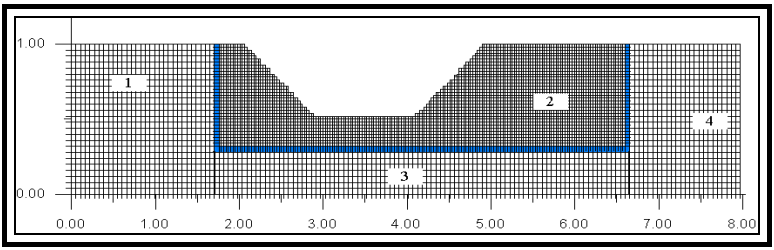

Figure 2: Part of the used block nested numerical grid 401x26, L=1, I=2.

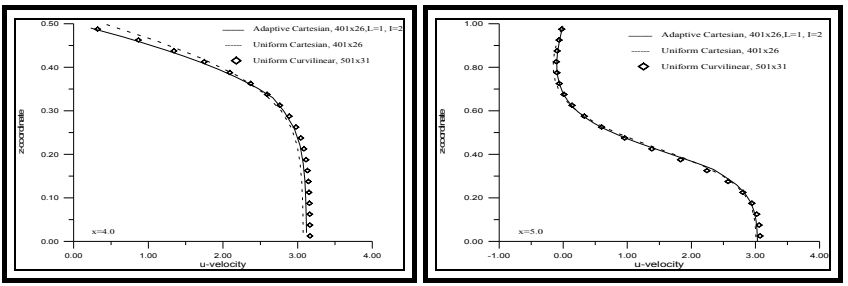

Figure 3: Velocity profiles along the flow field.

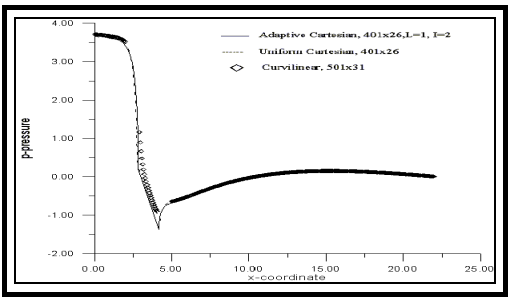

Figure 4: Pressure distribution along the stenosed tube.

3.2 Flow around airfoil NACA0012

In the second test case we simulate the flow field around a symmetric airfoil, (figure 5). It is impossible to solve the airfoil domain using a uniform Cartesian grid, because in order to achieve the desired geometry bound approximation a huge number of Cartesian grid cells would be needed. In order to avoid the above memory problem, we apply the block-nested algorithm of two grid levels (m=2), while the integer refinement factor is equal to four (I=4). The base grid size is 61x51 and the whole computational domain consists of 25586 cells. The corresponding uniform Cartesian grid comprises 796416 cells and its use for the airfoil numerical simulation is time-consuming and unprofitable. Regardless of

the time problem, we solved the fluid flow around the airfoil using a 320x315 uniform Cartesian grid and we realized that the use of the 61x51 refined grid decreases the CPU time by 93%.

We applied free stream conditions and we gave the velocity value for all the boundary limits except of the [CD] limit (figure 5), where we the pressure value is given. The angle of attack is equal to 0 and the Reynolds number is 100. We present two axial velocity profiles along the flow field (figure 6). The comparison took place by corresponding results with [22].

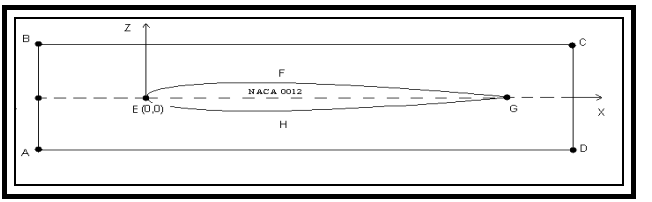

Figure 5: Physical domain around airfoil NACA0012.

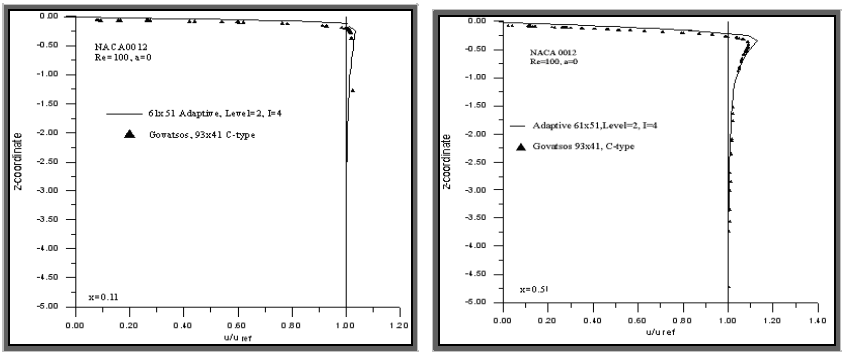

Figure 6: Axial velocity profiles of fluid flow around the airfoil, Re=100.

4 Conclusions

This paper proposes a method for the approximation of complex curvilinear geometries by using Cartesian co-ordinates only. In order to succeed the best geometry approximation close to the initial curvilinear bound, we applied the saw-tooth method in combination with a grid block refinement technique. We use a cell centre discretisation and the boundary transfer is demonstrated in the interfaces by the use of interpolation.

We presented the numerical simulation of two flow fields: both inside a stenosed tube and around a symmetric airfoil. At the stenosed tube numerical simulation, we created the approximate Cartesian geometry by a saw-tooth method and we applied a block-nested grid with one level and a refinement factor equal to two. We solved the incompressible Navier–Stokes equations into four separate sub-grids using linear interpolation in order to transfer the

boundary conditions. A comparison of the axial velocity results took place, between the block Cartesian grid, the uniform Cartesian grid and the curvilinear grid, with satisfied results. By the use of the block-nested grid we succeeded to improve the result's accuracy towards the corresponding uniform Cartesian grid.

On the second test case we examined the fluid flow around a NACA0012. Airfoils are 'thin' bodies and the use of a uniform Cartesian grid is very unprofitable and sometimes the algorithm is impossible to converge. So the use of the block-nested grid is necessary and provides a lot of advantages according to the CPU memory and converging time. A comparison of the axial velocity results took place between block Cartesian grid and bibliography results. The differences appearing between the profiles are due to the different simulation methods, types of grid, residuals and certainly to digitization error. By the use of the block-nested grid we succeeded in improving the converging time results and sometimes decreasing them by over 90%. It is important to note that the flow rate is of concern, in both of the above test cases, in spite of the differences depicted in the above velocity profiles.

The above numerical solution proves that the Cartesian block refinement method is stable and accurate enough, regardless of the fact that the produced Cartesian bound is less accurate than the curvilinear one. The block Cartesian method is simple and gives a convergent and grid independent solution for complex curvilinear geometries, also accomplishing a reduction in CPU memory and the simulation computing time effort.

References

[1] Thompson, J.F., Thames, F.C. and Mastin, C.W. "Automatic numerical generation of body fitted curvilinear coordinate system for field containing any number of arbitrary two-dimensional bodies", J. of Computational Physics, Vol. 15, pp. 299–319, 1974

[2] Clark, D.K., Salas, M.D. and Hassan, H.A., "Euler calculations for multielement airfoils using Cartesian grids", AIAA J., Vol. 24, pp.353–356, 1986

[3] Falle, S. and Giddings, J., "An adaptive multigrid applied to supersonic blunt body flow", Numerical Methods in Fluid Dynamics, 1988

[4] Pember, R.B., Bell, J.B., Collela, P., Cruthhfield, W.Y. and Welcome, M.L., "An adaptive Cartesian grid method for unsteady compressible flow in irregular regions", J. of Comp. Physics, Vol. 120, pp. 278–304, 1995

[5] Coirier, W.J. and Powell, K.G., "Solution-Adaptive Cartesian cell approach for viscous and inviscid flows", AIAA J., Vol. 34, pp. 938–945, 1996

[6] Coirier, W.J. and Powell, K.G., "An accuracy assessment of Cartesian-mesh approaches for the Euler equations", J. of Computational Physics, Vol. 117, pp. 121–131, 1995

[7] Smith, R.J. and Johnston, L.J., "A novel approach to engineering computations for complex aerodynamic flows", Proceedings of the 4th Int. Conf. on Numerical grid Generation in CFD, pp. 271–285, 1994

[8] Almgren, A.S., Bell J.B., Collela P., Howell L.H. and Welcome M.L., "A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations", J. of Comp. Physics, Vol. 142, pp. 1–46, 1998
[9] Martin D. and Collela P., "A cell-centered adaptive projection method for the incompressible Euler equations." J. of Computational Physics, Vol. 163, pp. 271–312, 2000
[10] Howell L.H. and Bell J.B., "An adaptive mesh projection method for viscous incompressible flow", SIAM J. Sci. Comput., Vol. 18, pp. 996–1007, 1997
[11] Wang, Z.J., "A Quadtree-based adaptive Cartesian/Quad grid flow solver for Navier-Stokes equation", Computers and Fluids, Vol. 27, pp.529–549, 1998
[12] Tuncer, I.H., "Two-dimensional unsteady Navier-Stokes solution method with moving overset grids", AIAA J., Vol. 35, pp. 471–476, 1997
[13] Agresar, G., Linderman J.J., Tryggvason, G. and Powell, K.G., "An adaptive, Cartesian, front-tracking method for the motion, deformation and adhesion of circulating cells", J. of Comp. Physics, Vol. 143, pp. 346–380, 1998
[14] Udaykumar, H.S., Kan, H.C., Shyy, W. and Tran-Son-Tray, R., "Multiphase dynamics in arbitrary geometries on fixed Cartesian grids", J. of Comp. Physics, Vol. 137, pp.366–405, 1997
[15] Deister F., Rocher D., Hirschel E.H. and Monnoyer F., "Adaptively refined Cartesian grid generation and Euler flow solutions for arbitrary geometries", Notes on Num. F.D., Vieweg, Braunschweig/Wiesbaden, 1998
[16] Chai, J.C., Lee, H.S. and Patakar, S.V., "Treatment of irregular geometries using a Cartesian coordinates finite-volume radiation heat transfer procedure", Numer. Heat Transfer, Vol.26, pp.179–197, 1994
[17] Jesse J.P., Fiveland W.A., Howell L.H., Collela P. and Pember R.B., "An adaptive mesh refinement algorithm for the radiative transport equation", J. of the Comput. Physics, Vol. 139, pp. 380–398, 1998.
[18] Berger M.J. and Collela P., "Local adaptive mesh refinement for shock hydrodynamics", J. of Comput. Physics, Vol. 83, pp.64–84, 1989.
[19] Georgantopoulou Chr. G., Pappou Th. J., Tsaggaris S.G., "Cartesian grid generator for N-S numerical simulation of flow fields in curvilinear geometries", Proc. of the 4th GRACM, pp. 526–534, 2002
[20] Pappou, Th. and Tsangaris, S., "Development of an artificial compressibility methodology using Flux Vector Splitting", Int. J. for Num. Methods in Fluid, Vol. 25, pp.523–545, 1997
[21] Georgantopoulou, C. and Tsangaris S., "Block mesh refinement for incompressible flows in curvilinear domains", Applied Mathematical Modeling, Vol.31, pp 2136–2148, 2006
[22] Γεωργαντοπούλου, Γ.Χ., Γεωργαντόπουλος, Α.Γ., Πάππου, Ι.Θ. και Τσαγγάρης, Σ., «Εφαρμογή καρτεσιανών πλεγμάτων στην επίλυση πεδίων ροής γύρω από καμπυλόγραμμες γεωμετρίες", Recent advances in mechanics and the related fields, Vol.1, pp.76, 2003.