Cartesian refinement grid generation and numerical calculation of flows around Naca0012 airfoil

Christine. G. Georgantopoulou, Nikolaos S. Vasilikos, George A. Georgantopoulos and

Sokratis G. Tsangaris

Abstract*—This study examines the use of Cartesian grids with block refinement in space for steady flow computations around domains of irregular shape bounds, and especially around symmetric airfoils. In order to avoid the complexity of the body fitted numerical grid generation procedure, we use a saw tooth method for the curvilinear geometry approximation. 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 laminar flows and based on a cell-centre approximation projection. We present the numerical simulation around a symmetric airfoil NACA0012. The utility of the algorithm is tested by comparing the convergence characteristics and accuracy to those of the standard single grid algorithm. The Cartesian block refinement algorithm can be used in complex curvilinear geometries and airfoils simulation, to accomplish a reduction in memory requirements and the computational time effort.*

***Keywords*—** *airfoil simulation, block nested refinement, cartesian grids, incompressible flows, N-S equations*

I. Introduction

The rapid evolution of computational fluid dynamics has been driven by the need for faster and simpler methods for the numerical calculation of flow fields around bodies. That is why, recently has became a great development of Cartesian grids. The use of them was almost abandoned when the body-fitted structured curvilinear (BFC) grid approach came in, because 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 a complex multiply connected domain with sharp boundaries it is difficult to automatically generate a grid of good quality. Most of the algorithms in BFC are still strongly dependent on the problem to be solved and required a lot of computational and human time effort. So in order to avoid these partial problems, we try to apply Cartesian grid generation and numerical estimation for flows around and inside geometries with curvilinear or complex body contours. By using Cartesian grids, the specification of the geometry description needed, is easier than the other methods because it involves only a set of cells of co-dimension one with respect to the problem domain and also the numerical grid is generated automatically containing simplified data structures and formulations for the numerical fluxes. The Cartesian grid generation was used by Clarke [2] and Falle and Giddings [3] to calculate steady compressible flows [4]. Coirier and Powell [5] used a Cartesian methodology for steady transonic solutions Euler's equations and in [6] performed accuracy and efficiency assessments of the method. It's a cell-centred method with an interesting treatment of boundary conditions. Smith and Johnston [7] develop a grid generation procedure that uses Cartesian embedded unstructured approach for complex geometries.

Adaptive mesh refinement algorithms have been used extensively to solve a variety of problems in hypervolic conservation lows and have more recently been extended to incompressible flows [8, 9, 10]. Wang [11] develops a quadtree-based adaptive Cartesian/Quadrilateral grid generator and flow solver based on cell cutting-[12, 13, 14], and Deister [15] presents a refined Cartesian grid based in octa-tree.

The use of Cartesian grids in solving the air flow around airfoils demands special treatment, because the airfoils are "thin bodies". You have to face their small thickness and to find a solution in order to create a Cartesian grid which will allow the numerical estimation algorithm convergence. That's why the application of a uniform Cartesian grid in order to solve the flow around airfoils is not recommended. On the other hand, if a refinement Cartesian grid method is

Chr. G. Georgantopoulou was with Department of Mechanical Engineering, National Technical University of Athens, Zografou, Greece. She is now with Geography Department, Harokopio University of Athens, El. Venizelou 70, 17671, Athens, Greece (corresponding author, phone: 0030 210 9549179; e-mail: chgg@hua.gr)

N. S. Vasilikos is with Department of Mechanical Engineering Educators, School of Pedagogical and Technological Education, N. Hrakleio, 14151, Greece (e-mail: vasnikoler@yahoo.gr)

G. A. Georgantopoulos is with Hellenic Air Force Academy, Dekeleia Attikis, 14510, Greece (e-mail: gageorgant@yahoo.gr)

S. G. Tsaggaris is with Department of Mechanical Engineering, National Technical University of Athens, Heroon Polytechniou 9, Zografou-Athens, Greece (e-mail: sgt@fluid.mech.ntua.gr)

used, computations of both steady and unsteady problems involving airfoils show a very satisfied accuracy. Liao [16], examines the use of embedded Cartesian grids for computations around airfoils with the small-perturbation boundary condition approach. Yang [17,18], used stationary body-conforming grids, which did not require the assumption of thin geometry, and Kirshman [19] develops a Cartesian grid method by the use of a set of shape functions over a cloud of gridless nodal points.

In this paper we present a Cartesian grid approach based on a saw-tooth method for the curvilinear geometries bounds approximation. This technique is based on Chen, Lee and Patakar [20], where they present the saw-tooth Cartesian method for heat transfer problem on a complex geometry. We apply a nested refinement algorithm based on that of Jesse [21] and Martin and Collela [22], 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. The block refinement is automatic and it can be applied in any complex curvilinear geometry. It's applied to steady, incompressible flow fields for Navier Stokes numerical simulation [23]. The flow solver is based on a pseudo-compressibility technique by Pappou and Tsangaris [24].

II. CARTESIAN 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 one as possible. The new approximate bounds are parted only by the use of grid lines, on x or z-axis either. The method used, is called saw-tooth and has been chosen as the most appropriate for the finite volume cell centered numerical simulation of flow fields. This method provides independence and automation of grid generation for problems with complex boundaries, with or without existence of an analytical function. The main advantage of saw-tooth is that you can create any approximate Cartesian geometry, if you only have a set of data points, and so you can simulate any flow field even its geometry analytical function is unknown.

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. In many cases a huge grid size is needed- as numerical simulation and calculation around airfoils-, and this is unproductive. In order to overcome this problem we create a block refinement grid wherever the flow domain demands. We define the block's bounds and we analyze the way of variable's value transfer between coarse and fine interfaces. It will be shown that the method is stable and accurate, because it provides satisfying results and minimizes the computational memory.

A. Cartesian grid approach

In order to finally create a Cartesian approximate bound of the original geometry 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 x or z-axis either. We have to control if the contour segment between two neighbour data points varies monotonically with respect to both x or z directions. If we discover that this rule doesn't occur we have to cluster the Cartesian grid. 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 (figure 1). If an original data point is on x-axis, we calculate the distance between this and its neighbouring grid nodes in the same direction (x). According the smallest distance we choose the corresponding grid node as the Cartesian approximated point.[23,25] We create a contour by connecting these points and the complex bound has been transformed to a Cartesian geometry bound (figure 2). In this work we mainly use the rule of minimum distance in relation to the final grid and the resulting numerical simulation of the flow field.

B. Refinement mesh method

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 (figure 3). The whole domain is a rectangle whose sides lie in the coordinate directions. We simulate the domain based in as many refine grids as we need. Although the discrete solution must be independent of how the refine grids are composed, we have to follow some rules, in order to succeed grid hierarchy and properly nested grids. [26]

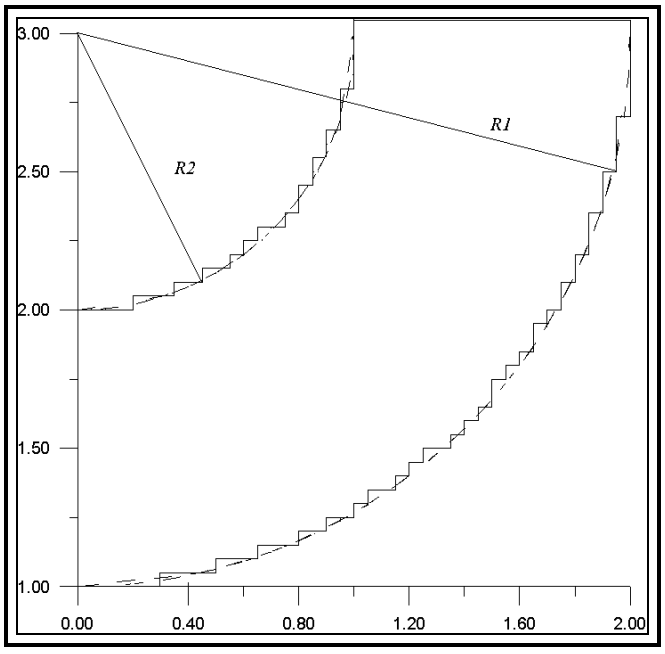


Fig. 1: Cartesian geometry approximation by saw-tooth method

So, a fine grid starts and ends at the corner of a cell in the next coarser grid [22]. Also all the sub-grids must be rectangular. The numerical simulation of any flow field is

started by the coarsest grid and follows to the next level. We have to create the neighboring girds to be only one level up or down.

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

The proposed nested algorithm contains several levels of grids. We create a coarse level at

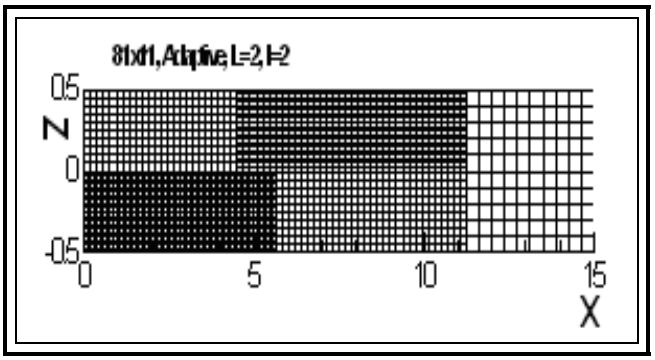


Fig. 2: Block refinement grids, I=2, m=2.

the beginning and we solve the domain. We name this coarse level *m=0* and each next refine sub – grid is named *m+1*. The coarsest grid is uniform on x and z direction respectively. We define an integer refinement factor, like [9], $I = dx_m / dx_{m+1} = dz_m / dz_{m+1}$. For convenience the above factor should be a second power.[27]

As we have created the coarse grid we simulate the flow field and calculate the variables. At this time the coarse-fine interfaces are neglected since no information from the finer level is available yet. Of course the geometry approximation error is quite big but this is not a problem, as we have just a prediction for the fluxes near the geometry bound. We have already defined the limits of the refinement levels and we proceed 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 neighbor 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. As we resolve in *m-1* levels, all of them have to be rectangular. 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's why we give the velocity values, but we

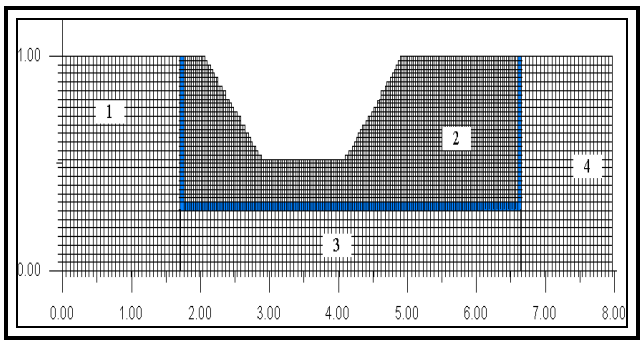


Fig. 3: Part of a used block nested numerical grid, m=1, I=2

solve for pressure. With nested grids, each grid is separately defined and has its own solution vector, so that a grid can be advanced independently of other grids, except for the determination of its boundary values. The information exchange between two successive levels is described in the next section.

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, (imposition of proper boundary conditions) and solve the flow field.
- Transfer the solution to the next grid level (m+1) by using the appropriate boundary conditions.
- 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.

III. 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 coarser level condition or by the physical boundary condition.

For a grid level m, the bordering cell values are provided using values from adjacent level, where they are available, or from physical boundary conditions. The data transfer can be done either to a coarse –fine interface, either to a fine-coarse one. For both of these cases, we can linearly interpolate linearly or bilinearly. In the present paper we interpolate linearly as described below. As we have already

mentioned, the sub-grid bounds are absolutely adjacent. The pseudo-cell of each sub-grid belongs to the boundary cells of the previous grid level. So when we solve in a refined level (m) we neglect the 'pseudo-cells' of the coarse level (m-1), and we use for the refined boundary transfer, the boundary cells by level m-1. That's very important because any other option will provide inaccurate solutions at whole flow field.

Let's 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. The $u^m(l,n)$ and $w^m(l,n)$ are the corresponding coarse grid values into the physical domain. Every interpolation takes place either on x either on y- axis. If we consider that we apply the new velocity values on x-axis, (figure 4), interpolation is applied as follows:

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

and

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

Also,

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

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

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

The relation between i and l is:

$$l=2*i-1 \quad (4)$$

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:

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

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

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, u and v the axial and the vertical velocity components respectively.

The derivatives discretization is applied by one-sided difference formula, either forward or backward. It depends on the position of each sub-grid in relation with the previous level one.

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:

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

where, I is the refinement factor.

So, we interpolate for the velocity components and we solve for pressure. Although, this isn't necessary, we prefer it because we want to maintain accurate and stable solutions. We agree with Collela [9], that if you want to obtain a robust algorithm solving for pressure is needed. Incidentally the results of both ways of simulation are good enough. [28,29]

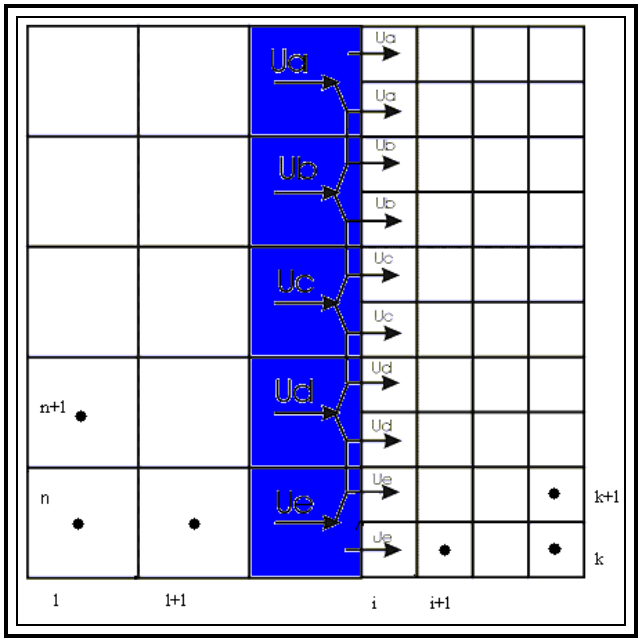


Fig. 4: Linear interpolation in order to transfer the velocity values to a coarse- fine interface.

IV. Numerical Simulation

The incompressible equations after the addition of the pseudocompressibility term, take on a hyperbolic character with pseudo-pressure waves propagating with finite speed. In such types of problems "the information" inside the flow field is transmitted along its characteristic curves. In this sense we can relate the sign of eigenvalues with the upwind representation of the flow variables at the cell faces. The upwinding of the inviscid fluxes gives more freedom in devising implicit algorithms (Steger and Kutler [30] and Thomas and Walters [31]), since it loads up the diagonals of the implicit factors. Upwind differencing (Hartwich et al. [32]), also, alleviates the necessity to add and to tune the

numerical dissipation for numerical stability and accuracy as the schemes with central differencing that belong to the family of Beam and Warming Schemes (Beam and Warming [24]).

The upwind scheme of the hyperbolic problem, in this paper, is based on the extended by the method of pseudocompressibility Flux Vector Splitting method. FVS is a shock-capturing upwind method, well known for solving compressible high speed (transonic, supersonic and hypersonic) flows. Here, we extend FVS method of Steger and Warming for solving incompressible flow fields implicitly [33]. In such flow fields the splitting of the convective flux vectors has to change sense because of their non-homogeneous property. This is a very important element of the present study. The values of the flux vectors at the cell faces are approached by upwind schemes up to third order of accuracy. The unfactored discretized Navier-Stokes equations are solved by an implicit second order accurate in time scheme, using Gauss-Seidel relaxation technique.

V. Numerical Results

The main purpose of the test cases is to demonstrate the capability of the developed overall algorithm, grid generation and flow solver for the solution of steady incompressible flow fields through domains of arbitrary shaped boundaries. In the following paragraphs we present the numerical simulation of flow field around a symmetric airfoil NACA0012 in order to examine the accuracy of the above method. We chase out the accuracy of the results comparing them to the correspondence of Cartesian uniform grid, with the same base grid size.

TABLE I
CPU TIME AND NUMERICAL VOLUMES

Grid size	***CPU time***	***Number of numerical cells***
Re=100, α=0.		
61x51, L=2, I=4	2323,76	25586
320x315, Uniform Cartesian grid	>1day	10426
976x816, Uniform Cartesian grid		796416

We simulate the flow field around a symmetric airfoil NACA0012, (figure 6). It is inconvenient to solve the airfoil domain using a uniform Cartesian grid for, because in order to achieve the desired geometry bound approximation a huge number of Cartesian grid cells would be needed. As we have already mentioned, we create the approximate cartesian bound of the airfoil (figure 5). 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 4 (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 refine grid decreases the CPU time by 93%. (table I)

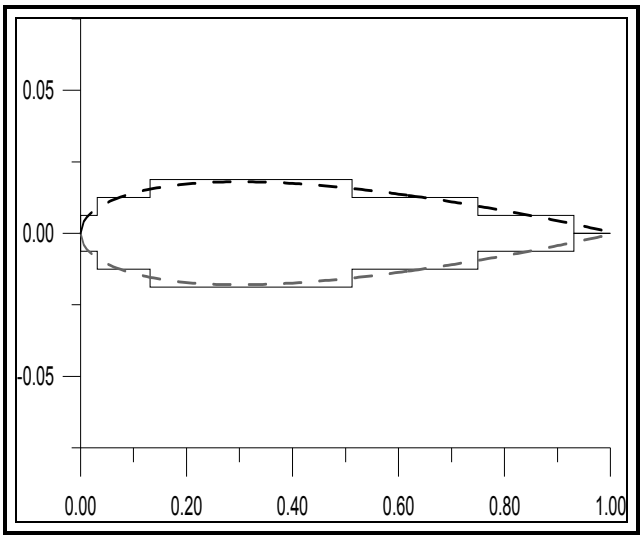


Fig. 5: Cartesian approximation of the NACA0012 airfoil bound

We applied free stream conditions and we gave the velocity value for all the boundary limits except of the [CD] limit (figure 6), where we the pressure value is given. The boundary conditions are presented below (table II)

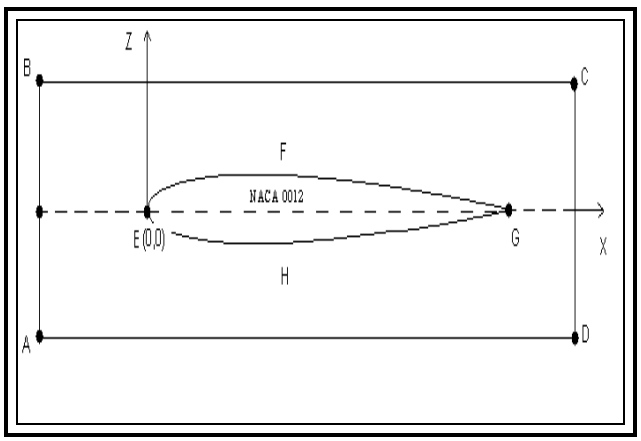
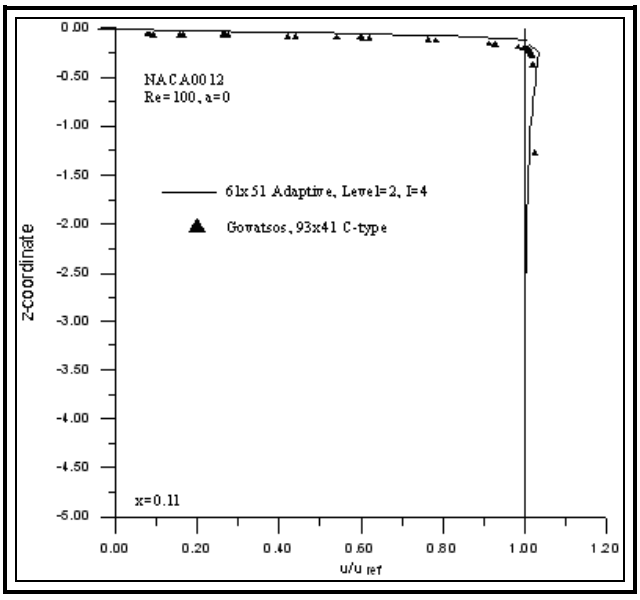


Fig. 6: Physical domain around airfoil NACA0012

TABLE II
BOUNDARY CONDITIONS

Airfoil's bound [EFGH]:	$u = w = 0, \frac{\partial p}{\partial z} = 0$
Lower bound [AD]:	$u = V\cos a, w = V\sin a, \frac{\partial p}{\partial z} = 0$
Upper bound [BC] :	$u = V\cos a, w = V\sin a, \frac{\partial p}{\partial z} = 0$
Inlet:	$u = V\cos a, w = V\sin a, \frac{\partial p}{\partial z} = 0$
Outlet:	$\frac{\partial u}{\partial x} = 0, w = 0, p = const$



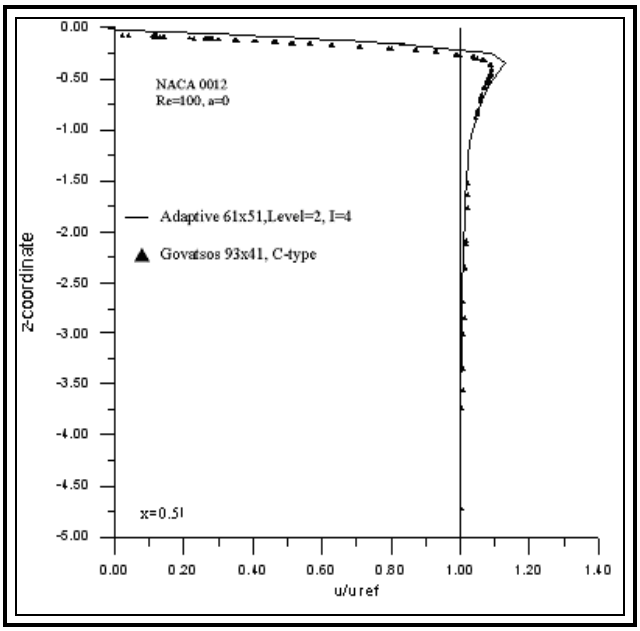
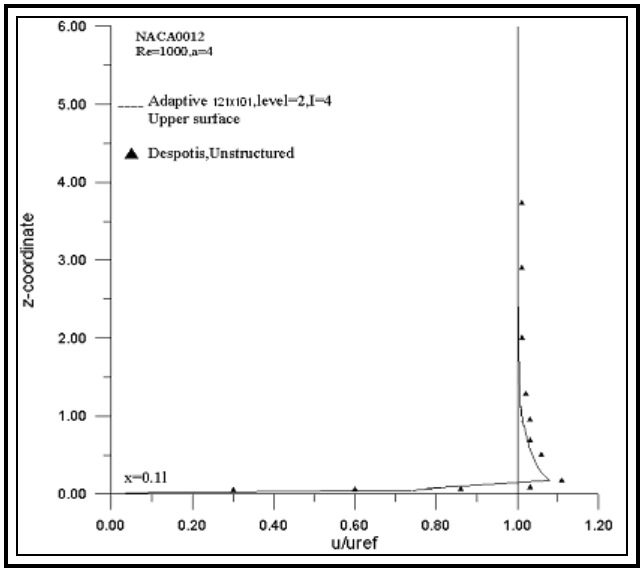


Fig. 7: Axial velocity profiles on two different positions of fluid flow around airfoil, Re=100, a=0

The angle of attack is equal to 0 and 4 and the Reynolds number is 100 and 1000. We present two axial velocity profiles along the flow field for each case, (figure 7,8) . The comparison took place by corresponding results by bibliography [27,34].

VI. 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 apply saw-tooth method in combination with a grid block refinement technique. We use a cell center discretization and the boundary transfer is demonstrated in the interfaces by the use of interpolation.



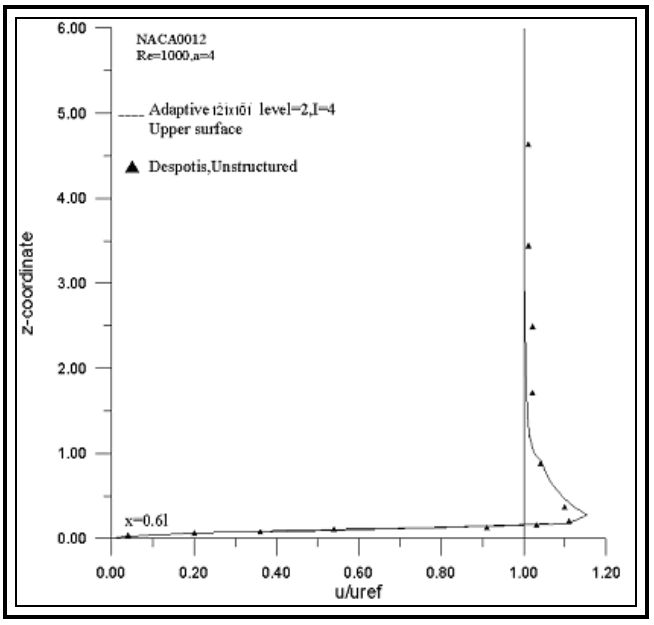


Fig. 8: Axial Velocity profiles on two different positions of fluid flow around airfoil, Re=1000, a=4

We examined and present the numerical simulation and calculation of the flow field around a NACA0012 airfoil. We created the approximate Cartesian geometry by a saw-tooth method and we applied a block-nested grid in order to achieve an approximate Cartesian bound close enough to the initial curvilinear bound using a lesser number of Cartesian cells. By the use of the block nested grid we succeed to improve the result's accuracy toward the corresponded of uniform Cartesian grid. Also, airfoils are 'thin' bodies and a use of a uniform Cartesian grid is very unprofitable and some times 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 due to the different simulation methods, types of grid, residuals and certainly to digitization error. By the use of the block nested grid we succeed in improving the converging time results and sometimes decreasing them over 90%. It's important to be mentioned that the flow rate is concerned, 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 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, accomplishing also to reduce CPU memory and the simulation's computing time effort. With appropriate choice of local block refinement multilevel solutions computed with this algorithm can attain the accuracy of the equivalent uniform fine grid at less computational cost.

VII. 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 Computational 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 International Conference on Numerical grid Generation in Computational Fluid Dynamics and related Fields, 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 Computational 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 Computational 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 Computational 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 numerical Fluid Dynamics, Vieweg, Braunschweig/Wiesbaden, 1998

[16] Liao W., Koh E., Tsai H. and Liu F., *"Euler calculation with embedded Cartesian grids and small perturbation boundary conditions"*, J. of Comput. Physics, Vol. 229, pp. 3523-3542, 2010

[17] Gao G., Yang S., Luo S., Liu F. and Schuster D., *"Calculation of airfoil flutter by an Euler method with approximate boundary condition",* AIAA Journal, Vol. 43, pp. 295-305, 2005

[18] Yang S., Kiu F., Luo S., and Tsai H., *"Three dimension aerolastic computation based on stationary body conforming grids with small perturbation boundary conditions",* AIAA , 34th AIAA Fluid Dynamic Conference, Protland, Oregon, 2004

[19] Kirchman D. and Liu F., *"Flutter prediction by an Euler method on non moving Cartesian grid with gridless boundary conditions",* Computers and Fluids, Vol. 35, pp. 571-586, 2006

[20] 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

[21] 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.

[22] Berger M.J. and Collela P., "*Local adaptive mesh refinement for shock hydrodynamics*", J. of Comput. Physics, Vol. 83, pp.64-84, 1989.

[23] Georgantopoulou Chr.G., Pappou Th.J., Tsangaris S.G., "*Cartesian grid generator for N-S numerical simulation of flow fields in curvilinear geometries"*, Proceedings of the 4th GRACM congress on comput. Mechanics, pp. 526-534, 2002

[24] Pappou, Th. and Tsangaris, S., *"Development of an artificial compressibility methodology using Flux Vector Splitting*", International J. for Numerical Methods in Fluid, Vol. 25, pp.523-545, 1997

[25] Chr.G. Georgantopoulou and S. Tsangaris, "Block mesh refinement for incompressible flows in curvilinear domains", Applied Mathematical Modeling,Vol.31, pp2136-2148.

[26] Chr.G.Georgantopoulou, G.A.Georgantopoulos, Th. Pappou, S. Tsangaris "Block nested refinement technique for the Navier Stokes equations", , 4th HEFAT, September 2005, Cairo, Egypt.

[27] Georgantopoulou C., Georgantopoulos G., and Tsangaris S., *"Mesh refinement technique for incompressible flows in complex geometries using Cartesian grids",* WIT Trancations on Modelling and Simulation, Vol.48, pp. 369-378

[28] Mavromatidis L., Michel P., Mankibi M. and Santamouris M., *"Investigation of the sensibility of Multi-Foil insulations using the guarded hot plate and the guarded hot box test methods",* Palenc 2010, 5th Eur. Conf. on energy Performance (EPIC2010), 29 Sep.-1Oct, Rhodes island, Greece, 2010

[29] Mavromatidis L., Michel P., Mankibi M. and Santamouris M., *"Investigation of the contribution of multifoil insulation in the veduction of energy consumption of buildings",* 9th Geographical Conf., 4-6Nov, Athens, Greece, 2010

[30] Steger, L. and Kulter, P. "*Implicit finite-difference procedures for the computation of vortex wakes*", AIAA J, Vol.15, pp. 581-590, 1977

[31] Thomas, L. and Walters, R.W., "*Upwind relaxation algorithms for the Navier-Stokes equations*", AIAA paper, 85-1501-CP, 1999

[32] Hartwich, M., Hsu, C.H. and Liu, C.H., "*Vectorizable implicit algorithms for the flux-difference split, three-dimensional Navier-Stokes equations*", J. of Fluid Engineering, Vol. 110, pp. 297-305, 1988

[33] Beam, R.M. and Warming, R.F., "*An implicit finite-difference algorithm for hyperbolic system of conservation-law form*", International J. of Computational Physics, Vol. 22, pp. 87-110, 1976

[34] 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