Mathematical modelling for the solution of incompressible flow through channels using block structured grids

CHRISTINA G. GEORGANTOPOULOU, GEORGE A. GEORGANTOPOULOS† and
NIKOLAOS S. VASILIKOS‡

School of Engineering
Bahrain Polytechnic
PO BOX 33349, Isa Town
KINGDOM OF BAHRAIN
christina@polytechnic.bh

†Hellenic Air Force Academy
Dekeleia Attikis
GREECE
gageorgant@yahoo.gr

‡Dept. of Mechanical Engineering Educators
School of Pedagogical and Technological Education
Irakleio Attikis, 14121
GREECE
vasnikoler@yahoo.gr

Abstract---One of the exigent issues in CFD has been the simulation and handling of complex geometries in order to estimate the fluid-solid interaction. Additionally special treatment is required at the pipelines’ simulation where in most of the cases a huge number of grid cells is needed according to their length and complexity. Our proposed method is a Cartesian grid generation using upwind numerical schemes for the computation and the prediction of flows inside or around domains of complex shaped bounds in combination with a block-nested refinement technique. The method is applied for the Navier Stokes equation solution, for incompressible, steady and laminar flows. We present analytical steps and calculations of the flow inside a backward-facing step channel as well as around a curvilinear domain as the cascade circular arch flow, trying to insure the accuracy of our block-nested grid generation as well as our numerical approach. The utility of the algorithm is tested by comparing the convergence characteristics and accuracy to those of the standard single grid algorithm. We finally conclude that the Cartesian block refinement algorithm is appropriate for the channel flows, even if they consist of some parts of curvilinear bounds and it provides accurate results concerning the flow variables and the recirculation data. It seems that the aforementioned method can provide accurate results in pipeline flows even with high aspect ratios, accomplishing a reduction in memory requirements and the computational time effort.

Key-words---Block-structured grids, Navier Stokes equations, incompressible flows, channel flow

1. Introduction

The first step in the numerical simulation and estimation of various types of flows is the initial geometry discretization according to the desired grid generation. The most common approach in grid generation is the use of a body-fitted method structured or unstructured. The main advantage of this method is that the surface is fitted with a new coordinate 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 as well as most of the times the generation is time consuming and requires high engineering knowledge from the user. [2]. Trying to avoid the above partial problems, the Cartesian generation approach has been developed rapidly and it has been applied for any complex geometry bound
approximation. There are a lot of advantages of the Cartesian grid generation as that 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 or that 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.

Using the Cartesian grid method there is a certain issue that has to be faced by the most appropriate way; the appliance of the boundary conditions. Unfortunately, in most of the boundary Cartesian approaches it is possible that not even a single node belonging to the grid matches the physical boundary [2]. A lot of methods have been applied, but most of the times we enforce the value of one variable and we solve for the others. Additionally, most of the times we have to decide in which nodes the boundary condition will be set; this is usually dependent on the way of the initial bound approximation. A method were the velocity is enforced, is presented by Faldun [8]. The boundary condition are applied to the nearest nodes to the physical boundary. Ikeno [9] uses a pressure correction scheme, where the pressure gradient at boundary nodes can modify the enforced velocity value. Tseng also [10], presents a very interesting approach for the immersed boundary method which can be probably applied in complex curvilinear geometries with satisfied convergence of the numerical algorithm.

The development of the adaptive mesh refinement algorithms is quite important especially if the initial domain appears large scale of dimensions or the aspect ratio receive very high values. Adaptive methods have been used extensively to solve a variety of problems in hypervolic conservation lows and have more recently been extended to incompressible flows [11,12, 13]. Peng has developed a nested Cartesian algorithm using fractional – step as well as immersed boundary method [14], providing very satisfied results in vortex shedding flow behaviour. 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.

In the present paper we apply a Cartesian grid approach based on a saw-tooth method for the initial curvilinear geometries bounds approximation. This technique is based on Chen, Lee and Patakar [16], 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 [17], Peng [14], Martin and Collella [12], and Berger and Collela [18], in which refined regions are organized into unions of a small number of nested rectangular blocks. The main point is to assure the accuracy of our method in order to be the appropriate one for the simulation and estimation of pipeline flows, where huge number of cells in a uniform grid are needed. 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 [19]. The flow solver is based on a pseudo-compressibility technique Pappou and Tsangaris [20]. The present work is the sequel of our previous research, trying to extend our approach to the flow domains simulation and estimation for pipelines’ flow problems.

2. Physical Domain Discretization

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 bound are parted only by the use of grid lines, on x or z-axis either. The method is used, called saw-tooth and is 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.

According to the aforementioned methodology 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. In order to
define the new approximated points we follow the below rule: 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, (fig. 1).

2.1 Block nested structured grid generation
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 criterion, in order to succeed grid hierarchy and properly nested grids:

a) A fine grid starts and ends at the corner of a cell in the next coarser grid [24].
b) All the sub-grids must be rectangular.
c) Numerical simulation fist started by the coarsest grid and follows to the next level.
d) The neighboring grids must 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.
staggered grids and the variable values are expressed on the cell’s center, we consider pseudo – cells all around the physical domain and the sub – grids too. By 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 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:

1. Create a coarse Cartesian grid (level \( m=0 \)), simulate, (imposition of proper boundary conditions) and solve the flow field.
2. Transfer the solution to the next grid level \( (m+1) \) by using the appropriate boundary conditions.
3. Solve the flow field on the new sub-domain.
4. Transfer the solution to the next level \( (m+2) \) with new boundary conditions.
   - (Repeat the procedure for all the levels)
5. Simulate and solve the flow on the last sub-domain (level \( m_{\max} \)).
6. Transfer the solution to the coarser grid level \( (m_{\max}-1) \) as its boundary conditions.
7. Solve the sub domain with the influence of the refined grid results.
   - (Repeat the procedure for all the levels)
8. Solve the coarsest-initial sub domain (level \( m=0 \)).
9. Take the solution of the variables by the finest grid.

2.2 Boundary Conditions

If the grids are adjacent, the boundary conditions of one grid are provided by the other. If they aren’t 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 or bilinearly interpolate. In the present paper we linearly interpolate 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 3),
interpolation is applied as follows:

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

and

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

Also,

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

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

\[ u^{m+1}(i,k) = u^{m+1}(i+1,k) \]  (3)

The relation between \(i\) and \(l\) is:

\[ l = 2 \times i - 1 \]  (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:

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

\[ + n_y \left[ \frac{1}{Re} \left( \frac{\partial^2 v}{\partial y^2} + \frac{1}{y} \frac{\partial v}{\partial y} + \frac{\partial^2 v}{\partial x^2} - \frac{v}{y^2} \right) - u \frac{\partial v}{\partial x} - v \frac{\partial v}{\partial y} \right] \]  (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, \( u \) and \( v \) the axial and the vertical velocity components respectively. [21]

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:

\[ u^m(l,n) = \frac{u^m(i,k) + u^m(i+1,k) + \ldots + u^m(i+I-1,k)}{I} \]  (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 [12], that if you want to obtain a robust algorithm solving for pressure is needed. By the way the results both of the ways of simulation are good enough, but it seems that the above technique is more appropriate for a variety of numerical applications. [29,30]

3. Navier- Stokes Equations Solution

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 (Thomas and Walters [22]), since it loads up the diagonals of the implicit factors. Upwind differencing (Hartwich et al. [23]), 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]).

Fig. 4: Linear interpolation in order to transfer the velocity values to a coarse- fine interface. [27]

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 [20]. 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.

4. Results

In order to examine the accuracy of the above block nested grid development methodology, we present the numerical simulation of the flow over a backward-facing step as well as the flow around a cascade of circular arch. Although the first domain has only Cartesian bounds, it has been chosen as the most appropriate in order to prove the stability and chase out the accuracy as well as the effectiveness of the refinement technique in channel flow. Comparing the results with the correspondence of Cartesian uniform grid, with the same base grid size. Great importance in this test case is the estimation of the length of recirculation inside the cylindrical tube and the accuracy that is provided by the proposed nested algorithm. This test case will be the first approach to prove that our methodology can be applied for the numerical solution of flows inside pipes even with high aspect ratios as sometimes we meet at industrial applications. By the other hand we estimate the flow around a cascade of circular arch in order to prove that our results are satisfied enough even the domain bounds are not Cartesian and so we have to produce first an approximate domain bound.

4.1 Flow over a backward-facing step.
The simulation and estimation of the flow over a backward-facing step is a classical problem in CFD. The main reason that this test case has been chosen is not only the appearance of a complex flow but also the close relationship of the specific case with the numerical modelling of air in an urban environment which we intent to develop in our future research. However this flow appears detachments and reattachment points [25] as well as recirculation zones and boundary layers.

![Fig. 5: Physical domain of the backward – facing step channel.](image)

The expansion ratio is equal to 0.5 while all the variables are estimated with the reference length to be equal to the diameter of the cylinder. ($L_{\text{ref}}=H$) 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 the 0.25% of the inlet area. The used numerical refinement grid is level=1 and I=2, (base grid: 401x26). The Re number, that was based on the maximum inlet velocity and the diameter of the inlet, was set equal to 400 as well as to 800. The boundary conditions are summarized as above, at table I. It is worth to be mentioned that due to the above outlet boundary conditions, we need to choose the appropriate length of the cylinder; 15 dimensionless lengths for Re=400 and 30 ones for Re=800. In order to control the accuracy of the proposed method, we simulated the current flow field by the use of two uniform grids sized 161x21 and 81x11 too. Finally we develop and use a Cartesian nested grid sized 81x11 with refinement factor equal to 2 using first only one and then two grid levels. The physical domain of the backward – facing step channel is presented in figure (5) while some of the used numerical grids are presented in figures (6a) and (6b).

<table>
<thead>
<tr>
<th>Table 1: Boundary Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lower bound</strong></td>
</tr>
<tr>
<td>$u = w = 0, \frac{\partial p}{\partial z} = 0$</td>
</tr>
<tr>
<td><strong>Inlet [AB]</strong></td>
</tr>
<tr>
<td>$w = 0, \frac{\partial p}{\partial x} = 0$</td>
</tr>
<tr>
<td><strong>Upper bound</strong></td>
</tr>
<tr>
<td>$u = w = 0, \frac{\partial p}{\partial z} = 0$</td>
</tr>
<tr>
<td><strong>Outlet</strong></td>
</tr>
<tr>
<td>$\frac{\partial u}{\partial x} = 0, p = 0$</td>
</tr>
<tr>
<td><strong>Inlet [AE]</strong></td>
</tr>
<tr>
<td>$u = w = 0, \frac{\partial p}{\partial z} = 0$</td>
</tr>
<tr>
<td><strong>Outlet</strong></td>
</tr>
<tr>
<td>$\frac{\partial u}{\partial x} = 0, p = 0$</td>
</tr>
</tbody>
</table>
Two velocity profiles along the flow field, upper and lower wall pressure distribution, the streamlines along the field as well as the pressure distribution are presented in figures (7), (8) and (9).

Table 2: CPU time & number of computational cells

<table>
<thead>
<tr>
<th>Grid Type</th>
<th>Recirculation length</th>
<th>Lower Wall Recirculation</th>
<th>Upper Wall Recirculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Re=400.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>161x21, Uniform cartesian</td>
<td>5.25</td>
<td>No detection</td>
<td>No detection</td>
</tr>
<tr>
<td>81x11, Uniform cartesian</td>
<td>5.33</td>
<td>No detection</td>
<td>No detection</td>
</tr>
<tr>
<td>81x11, Nested cartesian, L=1, I=2.</td>
<td>5.27</td>
<td>No detection</td>
<td>No detection</td>
</tr>
<tr>
<td>Re=800.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>161x21, Uniform cartesian</td>
<td>6.10</td>
<td>4.87</td>
<td>10.37</td>
</tr>
<tr>
<td>81x11, Uniform cartesian</td>
<td>6.20</td>
<td>4.80</td>
<td>10.50</td>
</tr>
<tr>
<td>81x11, Nested cartesian, L=1, I=2.</td>
<td>6.10</td>
<td>4.85</td>
<td>10.40</td>
</tr>
<tr>
<td>Gartling, BFC, 400x20, [26]</td>
<td>6.10</td>
<td>4.85</td>
<td>10.48</td>
</tr>
</tbody>
</table>

It seems that the convergence between block nested algorithm results and uniform grid’s is very satisfied. It’s worth mentioned that the results depicted by the block nested grid are accurate enough, present a very satisfied convergence with the according of the uniform grids as well as demand less computational time despite of the fact that the computational cells have been increased. Additionally it was quite anticipated that the results of the block nested grid with 2 grid levels and refinement factor equal to 2 provides the best results, according to the literature ones, in an appropriate enough computational time.
4.2 Cascade of Circular Arch Flow
In the first case we examined the fluid flow inside a channel in order to ensure the independence and accuracy of our block nested algorithm. In order to explore the accuracy of our computational model according to the approximate Cartesian bound we present the flow around a cascade of circular arch where the comparison of the results will take place among Cartesian grid’s and curvilinear grid’s results, [27, 28]. The numerical grid that was used is a block nested 121x81 grid, L=1 and I=2 as well as a body fitted curvilinear one with the same size. (fig. 10). The applied boundary conditions for the numerical simulation are presented below at table 4. The fluid flow variables are calculated at the cells center and the fluxes and are estimated along the cell faces. The Re number was based on the mean inlet velocity and

Table 3: Recirculation data

<table>
<thead>
<tr>
<th>Grid size</th>
<th>CPU time</th>
<th>Number of cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>81\times 11, L=1, I=2</td>
<td>72,96</td>
<td>2172</td>
</tr>
<tr>
<td>81\times 11, uniform</td>
<td>23,01</td>
<td>891</td>
</tr>
<tr>
<td>81\times 11, L=2, I=2</td>
<td>120,03</td>
<td>4711</td>
</tr>
<tr>
<td>161\times 21, uniform</td>
<td>174,94</td>
<td>3381</td>
</tr>
<tr>
<td>Re=400.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>81\times 11, L=1, I=2</td>
<td>78,84</td>
<td>2172</td>
</tr>
<tr>
<td>81\times 11, uniform</td>
<td>32,46</td>
<td>891</td>
</tr>
<tr>
<td>81\times 11, L=2, I=2</td>
<td>131,97</td>
<td>5488</td>
</tr>
<tr>
<td>161\times 21, uniform</td>
<td>187,68</td>
<td>3381</td>
</tr>
<tr>
<td>Re=800.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5. Conclusions

This paper proposes a method for the approximation of physical domains 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. The main purpose of this paper is also to test and evaluate the above algorithm in order to be used for the numerical simulation of pipeline flow. That is why we have not only pay attention at the approximation of curvilinear bounds but also to the accuracy of our results using the block-structured grid generation technique in combination with the boundary treatment as well as with the numerical scheme.

Table 4: Boundary conditions for the cascade of circular arch flow

<table>
<thead>
<tr>
<th>Lower bound [AB]&amp;[CD]</th>
<th>( \frac{\partial u}{\partial y} = 0, w = 0, \frac{\partial p}{\partial y} = 0 )</th>
<th>In.</th>
<th>( u = 1, w = 0, \frac{\partial p}{\partial x} = 0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound BC:</td>
<td>( u = w = 0, \frac{\partial p}{\partial y} = 0 )</td>
<td>Out.</td>
<td>( \frac{\partial u}{\partial x} = 0, w = 0, p = \alpha )</td>
</tr>
<tr>
<td>Upper bound:</td>
<td>( \frac{\partial u}{\partial y} = 0, w = 0, \frac{\partial p}{\partial y} = 1 )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 9(b): Pressure distribution along the channel’s walls.

Fig. 10: The Cartesian and the BFC grid that have been used. Grid size: 121x81

the chord of the arch and was set equal to 100 for the present case. Two velocity profiles for the axial velocity component are presented in figure (11). The velocity results were calculated for both the Cartesian and the curvilinear co-ordinates and they are compared each other.

Fig. 11: Velocity profiles along cascade circular arch fluid flow

We presented the numerical simulation of two flow fields: flow inside a backward-facing step as
well as around a cascade of circular air flow. At the first test-case we don’t need to produce the Cartesian approximation of a curvilinear bound but we should evaluate the independence as well as the accuracy of our block – nested structured grid generator in combination with our numerical approach. Our results have been compared with those of uniforms Cartesian grids as well as of the bibliography with very satisfied convergence. After these results it seems that our approach can be quite appropriate for the numerical simulation and estimation of pipeline flows without presenting any problem according to the domain discretization or the numerical schemes. It is also quite encouraging that the way of appliance of the boundary conditions seems to be appropriate, between the neighboring sub-grids without causing any problems in mass conservation. The prediction finally of the recirculation lengths as well as of the detachment and reattachment points is quite satisfied according to the literature results.

On the second test case we examine the flow around a cascade of circular air flow. The reason that we chose to present this geometry was the approximation of the curvilinear bound, which will be probably quite useful to the pipeline simulation that we intent to apply to our immediate future research. In this case we compared ourblock Cartesian results with the corresponded of a body fitted curvilinear algorithm with very satisfied results.

The above numerical solution proves that the Cartesian block refinement method is stable and accurate enough, and it can probably provide accurate modelling, simulation and results to a pipeline flow domain of industrial usage. The block Cartesian method is simple and gives a convergent and grid independent solution for high aspect ratio as well as 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.

References:


[25] Manzan W., Vilela C., Mariano F., “Experimental and computational simulations of the flows over backward facing step”, Proceedings of 22nd Int. COBEM2013, SP, Brazil