Chapter 6

Validation and Verification of the Navier-Stokes Flow Solver

Before proceeding to extensively use the Navier-Stokes flow solver for our calculations, it must be first validated. In this chapter, a qualitative and quantitative validation and verification of the proposed flow solver against experimental and numerical results is carried out in order to assess its numerical accuracy.

6.1 Flow Solver Validation and Verification. General Issues

Software Verification and Validation (V&V) is the process of ensuring that the code being developed or changed (in our case a flow solver) is: a) able to model with accuracy a real world problem, that is, *"it solves the right equations"* (validation) and b) it yields the right results or in other words, *"it solves the equation right"* (verification). For our purposes, the differences between validation and verification are unimportant and are just of interest to the theorist. Hereafter, the term V&V or just validation will be used to refer to all the qualitative and quantitative comparisons done to assess the accuracy of the proposed tools.

The proposed flow solver is Overture[1] together with PETSc[2]. Overture, is an object-oriented code framework for solving partial differential equations (PDEs) in serial and parallel environments. It is implemented as a collection of C++ libraries that enable the use of finite difference approximations to solve the governing PDEs in structured and overlapping structured grids. PETSc [11] is a suite of data structures and routines that includes a large series of linear and nonlinear equation solvers, preconditioners and Krylov subspace methods for the scalable (parallel) solution of large-scale scientific applications modeled by PDEs. Both tools used together provide a portable, scalable and flexible software development environment for applications that involve the simulation of physical processes in complex fixed or moving $2\mathbb{D}$ and $3\mathbb{D}$ geometries.

[1] https://computation.llnl.gov/casc/Overture/
[2] http://www-unix.mcs.anl.gov/petsc/petsc-as/

6.2 Numerical Results and V&V

6.2.1 The Method of Manufactured Solutions or Forced Solutions

As a first test to check the numerical accuracy of the flow solver, we use the method of manufactured solutions (MMS) or forced solutions [154]. The basic idea behind this method is to simply manufacture an exact solution to the governing equations without being concerned about its physical realism. This solution also defines the boundary conditions to be applied in any forms, *i.e.*, Dirichlet, Neumann, Robin, etc., and the initial conditions. In the MMS, the governing equations are modified through the addition of a source term so that the manufactured solution is an exact solution to the governing equations with this additional source term. The particular form of the source term depends on the manufactured solution selected. This form is found by applying the governing equations operators to the manufactured solution to obtain an analytic formula for the source terms. The source terms are then added to the original equation set to balance it. Then the discrete solutions produced by the code can be compared to the manufactured solution to determine the discretization error. A comparison of the discretization error on a series of uniformly refined meshes will either verify that the observed order of accuracy matches the theoretical order of accuracy, or it will not. In the latter case, it may indicate a coding mistake or improper formulation. A useful set of guidelines for the effective design and application of the MMS are given by Roache [154] and Knupp and Salari [102].

In general, when constructing manufactured solutions, the following guidelines should be observed [102]. First, manufactured solutions should be composed of smooth analytic functions like polynomials, trigonometric or exponential functions. Second, the solution should be general enough that it exercises every term in the governing equations. Third, the solution should have a sufficient number of non-trivial derivatives. Finally, solution derivatives should be bounded by a small constant, this ensures that the solution is not a strongly varying function of space and/or time.

The MMS is much easier and more general than looking for analytical solutions to real problems. When this systematic procedure is used, we are testing for

- All the transformations used (*i.e.*, transformation of the governing equations to generalized curvilinear coordinates).
- The order of the discretization (spatial and temporal).
- The matrix solution procedure.
- Correctness of the numerical discretization.
- Interpolation between the overlapping grids.

Hereafter, we use the MMS to check the numerical accuracy of the flow solver. Here, the incompressible Navier-Stokes equations around a circle in a square with slip walls boundary conditions are solved. In two space dimension we use the following trigonometric functions as the manufactured solutions

$$
\begin{aligned}
u &= \frac{1}{2}\cos(\pi\omega_0 x)\cos(\pi\omega_1 y)\cos(\pi\omega_3 t) + \frac{1}{2} \\
v &= \frac{1}{2}\sin(\pi\omega_0 x)\sin(\pi\omega_1 y)\cos(\pi\omega_3 t) + \frac{1}{2} \\
p &= \cos(\pi\omega_0 x)\cos(\pi\omega_1 y)\cos(\pi\omega_3 t) + \frac{1}{2}
\end{aligned}
\tag{6.1}
$$

when $\omega_0 = \omega_1$ it follows that $\nabla \cdot \mathbf{u} = 0$ (the solution is divergence free).

In table 6.1, the results of this convergence test are presented. In this table, the maximum error in $\mathbf{u}$, p and $\nabla \cdot \mathbf{u}$ are shown. The estimated convergence rate p is also presented. In figure 6.1, the overlapping grids and the solutions for three refinement levels are illustrated.

Grid	h_1/h_g	$\|u - u_{exact}\|_\infty$	$\|v - v_{exact}\|_\infty$	$\|p - p_{exact}\|_\infty$	$\|\nabla \cdot \mathbf{u}\|_\infty$
$\mathbb{G}_1 : 32 \times 32 \cup 34 \times 8$	1	0.0205	0.0235	0.0473	0.0958
$\mathbb{G}_2 : 64 \times 64 \cup 68 \times 16$	2	0.00487	0.0037	0.0128	0.0200
$\mathbb{G}_3 : 128 \times 128 \cup 136 \times 32$	4	0.00120	0.000698	0.00453	0.00635
Order of convergence p		2.24	2.72	2.06	2.47

Table 6.1: *Maximum error at t=1.0 and $\nu = 0.1$ for a trigonometric analytic solution ($\omega_0 = \omega_1 = \omega_3 = 1$). The estimated convergence rate p is also shown. The column entitled as h_1/h_g denotes the ratio of the grid spacing on grid 1 to the spacing on grid g.*

A Navier-Stokes flow solver uses a numerical algorithm that will provide a theoretical order of convergence; however, the boundary conditions, numerical models, non-linearities in the solution, presence of shocks, grid refinement (or coarsening) and perhaps other factors, will reduce this order so that the observed order of convergence p will likely be different than the theoretical one [154, 170]. As outlined by Roache [154], if a grid refinement r is performed with constant refinement ratio r (not necessarily r=2) between all the grids, the observed order of convergence p can be obtained directly from three grid solutions as follows

$$
p = \frac{\ln\left(\dfrac{f_3 - f_2}{f_2 - f_1}\right)}{\ln(r)} \tag{6.2}
$$

where f is the approximate solution or the value of an observed quantity, with the sub-index $_1$ being the solution on the finest grid, $_2$ in the intermediate grid and $_3$ in the coarser grid. If one generates a finer or coarser grid and is unsure of the value of grid refinement ratio r used, one can compute the equivalent effective grid refinement ratio $r_{effective}$ as

$$
r_{effective} = \left(\frac{\mathcal{N}_1}{\mathcal{N}_2}\right)^{\frac{1}{\mathbb{D}}} \tag{6.3}
$$

where $\mathcal{N}$ is the total number of grid points used for the grid and $\mathbb{D}$ is the dimension of the flow domain; here again, the sub-index $_1$ corresponds to the solution on the finest grid, the sub-index $_2$ the solution in the intermediate grid and the sub-index $_3$ the solution in the coarser grid.

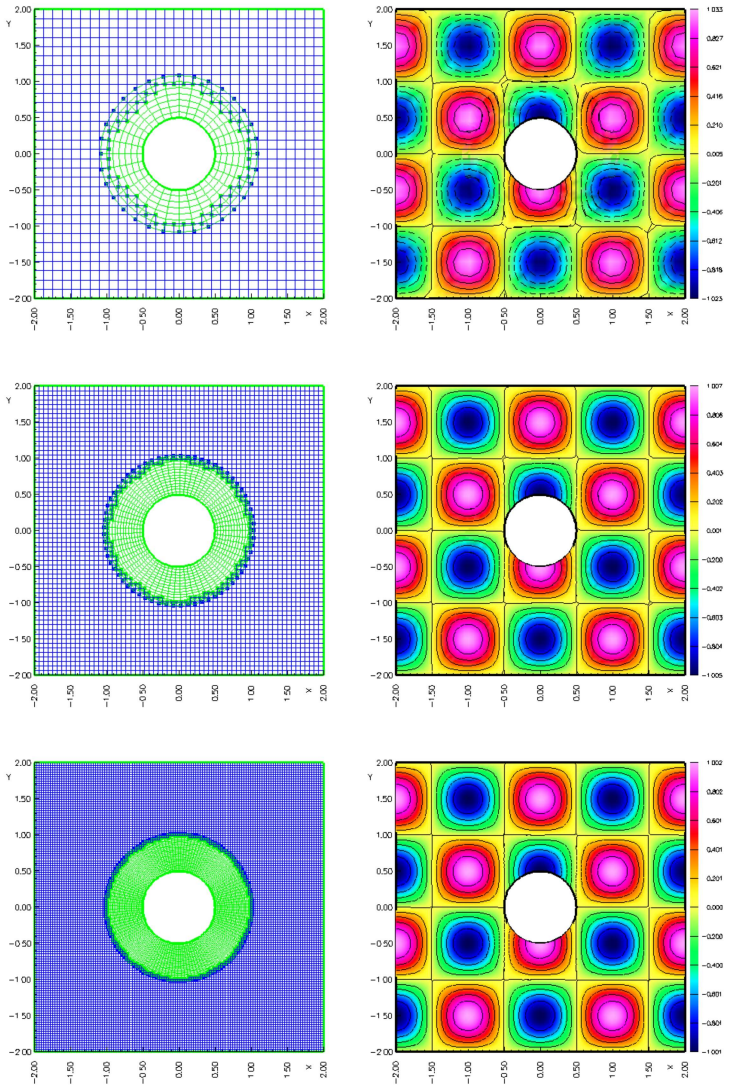

Figure 6.1: *Forced solutions of the incompressible Navier-Stokes equations around a circle in a square with slip wall boundaries and* $\omega_0 = \omega_1 = \omega_3 = 1$*. Top-to-bottom left column, grid system from coarser grid to finer grid. Top-to-bottom right column, corresponding grid level velocity u contours. Notice how the quality of the solution improves as the grid is refined.*

The results presented in table 6.1, show that although the method is converging at the expected convergence rate p (second order accuracy), the errors are significantly reduced when the grid is refined. Note that as the overlapping grids are refined, the positions of the interpolation points will change since the effective overlap decrease. As a result, the reduction in the error is not always as uniform as that from a single grid [76].

6.2.2 Flow Past a Stationary Cylinder at Various Reynolds Number Values

Hereafter, the results from the computation of a flow past a stationary cylinder in an overlapping grid system are shown. Simulations were performed at Reynolds numbers equal to 20, 40, 100, 200 and 400. For the cases of Re equal to 20, 40, 100 and 200, the obtained results were compared with other numerical and experimental data published in the literature. The case of Re equal to 400 is used as a benchmarking case in order to compare the performance of different direct and iterative solution methods.

In figure 6.2, the domain used for the cases where Re is equal to 20, 40 , 100 and 200 is illustrated. Here, the cylinder is located at the origin and has a radius of 0.5. The outer rectangular computational domain extends from $[x_a, x_b] \times [y_a, y_b] = [-5.0, 20.0] \times [-5.0, 5.0]$ and the inner cylindrical computational domain extends from $[x_{origin}, y_{origin}] \times [radius_{inner}, radius_{outer}] = [0.0, 0.0] \times [0.5, 1.0]$ (see figure 6.2). The initial conditions for all cases are those of a uniform flow with $(u, v, p) = (1.0, 0.0, 1.0)$ all over the domain. The top and bottom boundaries of the rectangular domain are slip walls. The left boundary of the rectangular domain is inflow and the right boundary is outflow. The cylinder's wall has a no-slip boundary condition. The Reynolds number based on the cylinder diameter, the kinematic viscosity and the inflow velocity $(u, v) = (1.0, 0.0)$ is controlled by changing the kinematic viscosity.

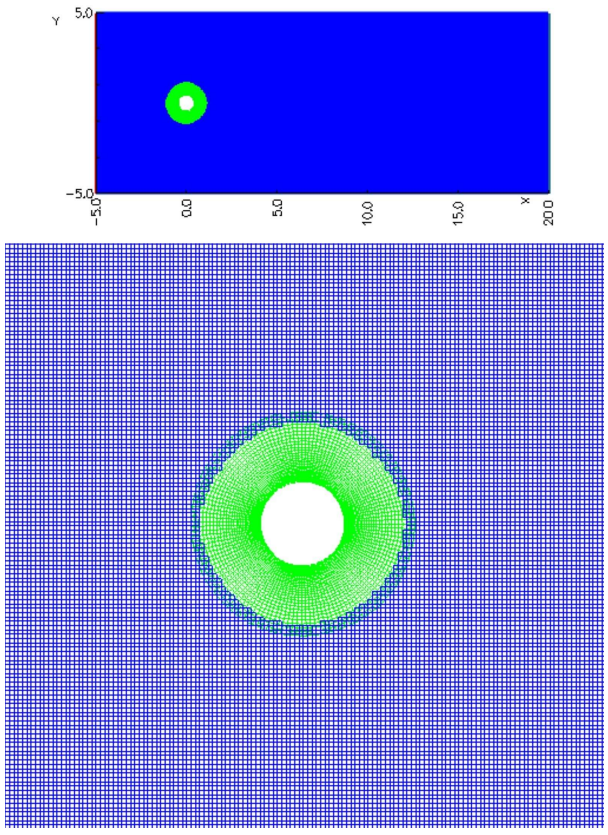

Figure 6.2: *Domain and overlapping grid system of the unsteady flow past a cylinder case. Top view: overall domain. Bottom view: close-up of the grid around the cylinder.*

For the cases where Re is equal to 20 and 40, the wake behind the cylinder shows a steady symmetric behavior as shown in figure 6.3, these solutions are consistent with the well established result that the wake behind the cylinder consist of a steady recirculation region of two symmetrically placed vortices on each side of the wake and is stable to perturbations below a Reynolds number value approximately equals to 46 ± 1 [90, 150, 180, 211]. In table 6.2, a comparison between the values obtained and other numerical and experimental results is presented. Here, despite the fact that the results found in the literature vary by as much as 10% from one another, it is found that the current results compare well with the other numerical simulations and experiments, falling within the range of the reported values.

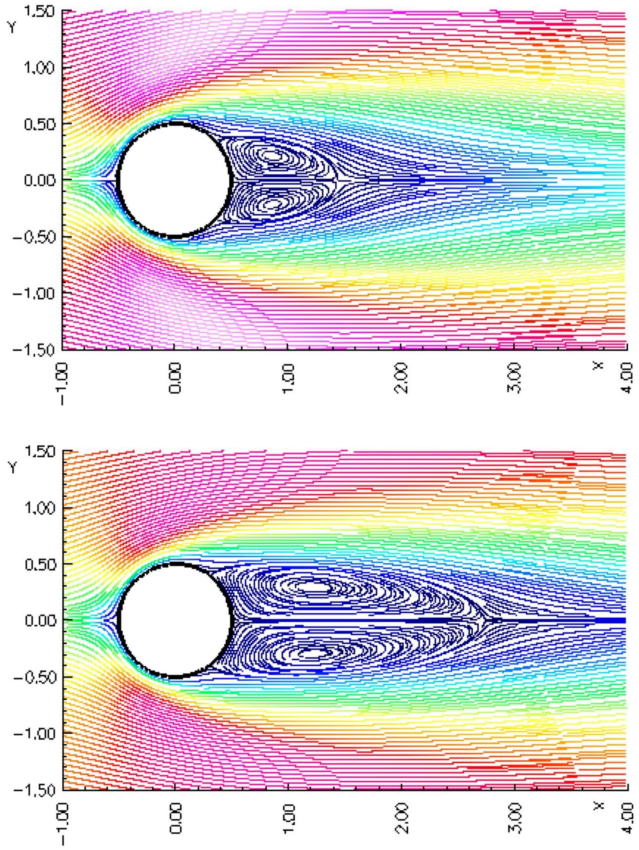

Figure 6.3: *Streamlines for* $Re = 20$ *(top figure) and* $Re = 40$ *(bottom figure) for a nondimensional time* $t = 400$.

It is generally accepted that the wake behind a cylinder first becomes unstable at a critical Reynolds number value of about $Re = 46 \pm 1$ [90, 150, 180, 211], as predicted by the linear theory of stability. Above this critical Re value, a small asymmetric perturbation in the near wake starts to grow in time and leads to an unsteady wake, known as von Karman vortex street, which is indeed what we found for the simulations where Re is equal to 100 and 200 (see figure 6.4). Figure 6.5, shows the variation of lift and drag coefficient with nondimensional time for the case where $Re = 100$. In this unsteady flow regime, the present results are compared

Reference	c_d for $Re = 20$	L for $Re = 20$	c_d for $Re = 40$	L for $Re = 40$
Tritton [194][E]	2.22	-	1.48	-
Coutanceau and Bouard [42][E]	-	0.73	-	1.89
Russel and Wang [157][N]	2.13	0.94	1.60	2.29
Calhoun [30][N]	2.19	0.91	1.62	2.18
Mittal *et al.* [215][N]	2.03	0.92	1.52	2.27
Fornberg [54][N]	2.00	0.91	1.50	2.24
Present results	2.2013	0.929	1.6208	2.216

[E] Results from experiments
[N] Results from numerical computations

Table 6.2: *Drag coefficient (c_d) and length of wake bubble (L) for $Re = 20$ and $Re = 40$.*

in table 6.3 against other numerical data. For $Re = 100$, the computed drag coefficient and lift coefficient were very close to those reported by Russel and Wang [157]; for the case of $Re = 200$, the present results compare favorably with those obtained by Braza *et al.* [24]. In general, it was found that our results compare well with the other numerical simulations and experiments. In table 6.4, a summary of the overlapping grid system used for all the previous cases is presented.

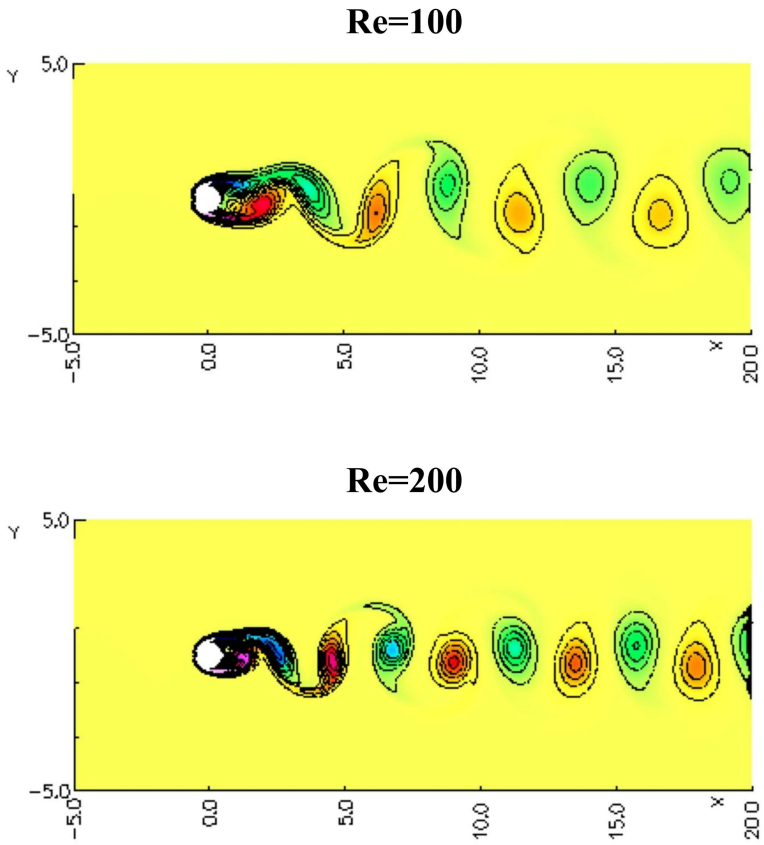

Figure 6.4: *Vorticity contours for* $Re = 100$ *and* $Re = 200$ *for a nondimensional time* $t = 500$.

The discretization of the Navier-Stokes equations leads to a large sparse non-symmetric non-linear system of equations that must be solved in order to obtain the approximate numerical solution to the initial-boundary-value-problem (IBVP). The numerical solution of the resulting system of non-linear equations is a major computational task in CFD and its accurate, robust and efficient solution is essential, especially if we want to solve larger systems (*i.e.*, finer grids).

There are several methods for solving the system of equations arising from the discretization of the governing equations, each raising the issue of the efficiency of the solution and its complexity. The increase in computing power and the introduction of parallel computing has driven the latest advances in algorithm development for the solution of large sparse systems of equations. Most of the research these days is focused on the efficient solution of these systems of equations on complex and large domains in parallel environments. For small-sized problems and even moderate-sized

Reference	c_d for $Re = 100$	c_l for $Re = 100$	c_d for $Re = 200$	c_l for $Re = 200$
Russel and Wang [157]	1.38 ± 0.007	$\pm$ 0.322	1.29 ± 0.022	$\pm$ 0.50
Calhoun [30]	1.35 ± 0.014	$\pm$ 0.30	1.17 ± 0.058	$\pm$ 0.67
Braza *et al.* [24]	1.36 ± 0.015	$\pm$ 0.25	1.40 ± 0.05	$\pm$ 0.75
Choi *et al.* [39]	1.34 ± 0.011	$\pm$ 0.315	1.36 ± 0.048	$\pm$ 0.64
Liu *et al.* [111]	1.35 ± 0.012	$\pm$ 0.339	1.31 ± 0.049	$\pm$ 0.69
Present results	1.3898 ± 0.012	$\pm$ 0.3330	1.4087 ± 0.048	$\pm$ 0.7250

Table 6.3: *Drag coefficient (c_d) and lift coefficient (c_l) for $Re = 100$ and $Re = 200$.*

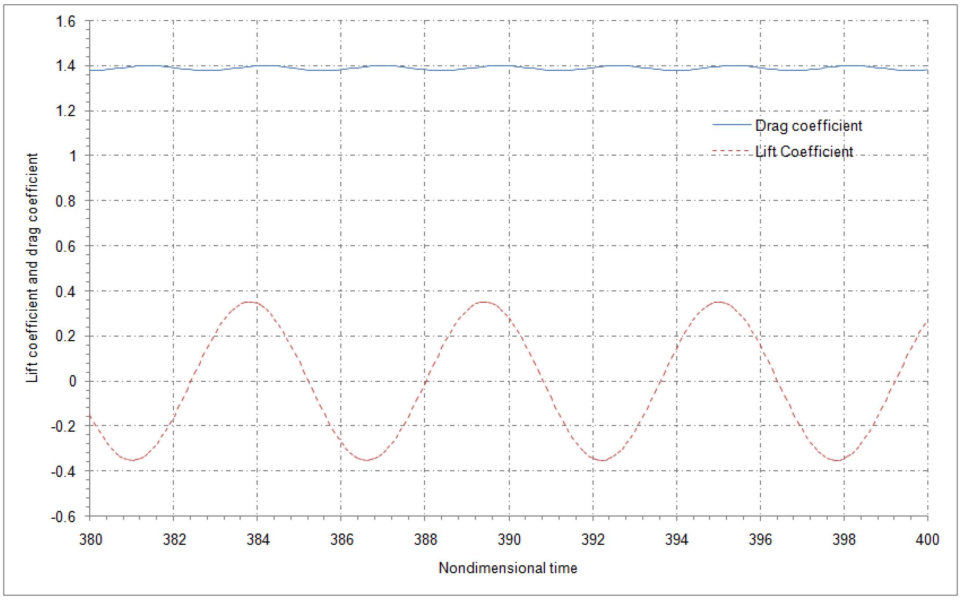

Figure 6.5: *Time dependent lift and drag coefficient for* $Re = 100$.

Number of grids	2
$\mathbb{G}_1$ dimensions (background grid)	500×200
$\mathbb{G}_2$ dimensions (annular grid)	140×80
Total number of grids points ($\mathbb{G}_1 \cup G_2$)	111200
$\mathbb{G}_1 \cup \mathbb{G}_2$, including ghost points	114752
Total number of unused points	1842
Total number of interpolation points	329
Position of the first node normal to the cylinder wall	$0.0001 \times diameter_{cyl}$

Table 6.4: *Summary of the overlapping grid system used for the cases where* $Re = 20$, $Re = 40$, $Re = 100$ *and* $Re = 200$.

problems, direct solvers are very efficient in solving the system of equations arising from the discretization, but they are not efficient for solving large problems [12, 158]. For large sparse systems of equations, Krylov iterative methods, in combination with a suitable preconditioner are the alternatives to direct solvers.

Newton-Krylov iterative methods for solving large non-linear systems have been used in CFD since the late 1980s [129, 199] and are considered an attractive and powerful approach to solve large problems due to their property of semi-quadratic convergence when starting from a good initial guess [45, 59, 158]. In Newton-Krylov methods, one applies a linearization method combined with a preconditioned Krylov subspace algorithm for solving the linear problem resulting from the linearization iteration. To enhance the efficiency and robustness of Newton-Krylov methods, it is necessary to apply preconditioning. Preconditioning is simply a means of transforming the original linear system into one which has the same solution, but which is relatively better conditioned and therefore is likely to be easier to solve with an iterative solver. The choice

of a preconditioner involves the selection of a matrix Q, called the preconditioning matrix or preconditioner, such that the preconditioned system

$$Q^{-1}Au = Q^{-1}b$$

is better conditioned than the original system, $Au = b$. Clearly, one requirement for Q is that it be easily invertible (*i.e.* the linear system having Q as the coefficient matrix can be solved with much less effort than solving the original system $Au = b$). In general, the reliability of iterative techniques, depends much more on the quality of the preconditioner than on the particular Krylov subspace solver or accelerator used. In practice, finding the best preconditioner for a given problem or class of matrices associated with a problem involves extensive testing.

Several authors [28, 29, 36, 153, 169] have studied the effect of various preconditioning methods on the convergence of Newton-Krylov iterative solvers. Their studies suggest that incomplete lower-upper ILU(k) factorization is a very efficient preconditioning strategy for a variety of Newton-Krylov solvers. The parameter k in ILU(k) denotes the level of fill-in that is allowed in the factorization, k equal to zero means no fill-in is permitted during ILU decomposition. In ILU(0) the factorized matrix and the original preconditioning matrix built from direct neighbors have the same graph (*i.e.*, same location for non zero-elements). Choosing k larger than zero would allow some additional fill-in in the factorized matrix which normally increases the accuracy of factorization and quality of the preconditioner. However, increasing the fill-in level would be at the expense of memory usage and increasing computing cost.

In the following simulation, we proceed to set the Reynolds number to 400 and we use the overlapping grid system described in table 6.5. In this case, the outer rectangular computational domain extends from $[x_a, x_b] \times [y_a, y_b] = [-2.5, 15.0] \times [-3.5, 3.5]$ and the inner cylindrical computational domain extends from $[x_{origin}, y_{origin}] \times [radius_{inner}, radius_{outer}] = [0.0, 0.0] \times [0.5, 1.0]$. The initial conditions and boundary conditions are the same as for the previous cases.

As mentioned before, this simulation will be used as a benchmarking computation in order to compare the performance of different direct and iterative solution methods. Here, we also compare the convergence behavior of various fill-in level k in ILU(k) preconditioning and others well known preconditioners such as additive Schwarz (AS), block Jacobi (BJ) and successive over relaxation (SOR) for different Krylov iterative solvers (see [11, 12, 45, 59, 158] for a detailed discussion on Krylov subspace methods and preconditioners).

Number of grids	2
$\mathbb{G}_1$ dimensions (background grid)	350×140
$\mathbb{G}_2$ dimensions (annular grid)	140×80
Total number of grids points ($\mathbb{G}_1 \cup G_2$)	60200
$\mathbb{G}_1 \cup \mathbb{G}_2$, including ghost points	114752
Total number of unused points	1120
Total number of interpolation points	295
Position of the first node normal to the cylinder wall	$0.0001 \times diameter_{cyl}$

Table 6.5: *Summary of the overlapping grid system used for the benchmarking computations.*

In table 6.6, the performance of various direct and iterative solvers used during theses benchmarking computations is compared. The implementation of the different direct and iterative solvers is based on the PETSc library, which was interfaced with Overture.

The computations carried out and presented in table 6.6, show that the GMRES + ILU(0) solver converges faster than the other methods in terms of CPU time for this specific problem. The timing for each case was carried out by measuring the CPU time from the beginning of the simulation until the moment when the wake instability behind the cylinder has its onset (see figure 6.6). We can also see that among the different preconditioner used (ILU, AS, BJ and SOR), the ILU(0) preconditioner lead to the fastest convergence closely followed by the AS preconditioner. The BICGSTAB solver with the additive Schwarz (AS) preconditioner, also shows a good convergence performance in terms of CPU time, although it is not as fast as the GMRES + ILU(0). From these results, it is also evident that the use of direct solvers for large sparse matrices becomes time and cost prohibitive; here, the direct solver is almost 15 slower than the fastest GMRES or BICGSTAB method (in terms of total CPU time) and about 5 times slower than the worst of the iterative solvers used for this benchmarking computation. All the computations were executed using a reverse-Cuthill-McKee (RCM) matrix reordering algorithm, which clusters the non-zero terms along the diagonal reducing in this way the bandwidth of the sparse matrix [16].

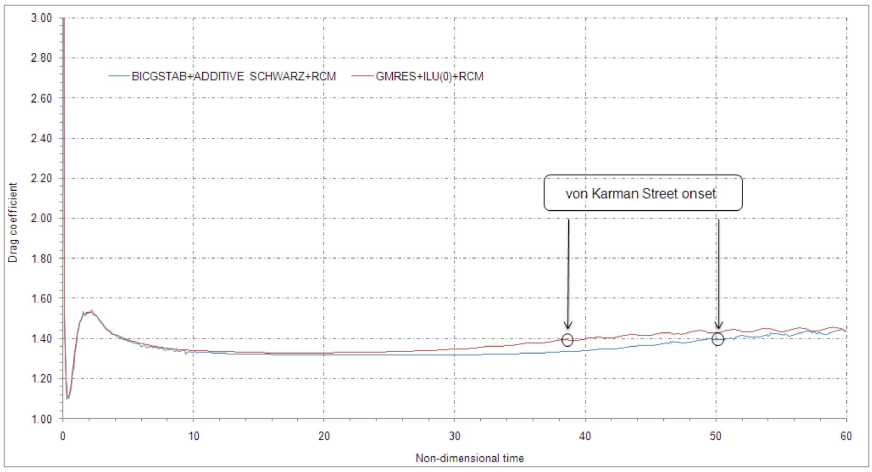

Figure 6.6: *Von Karman street onset (stopping criteria for solver benchmarking).*

These benchmarking computations clearly illustrate a case where the preconditioned GMRES solver performs very well, followed closely in performance by the preconditioned BICGSTAB solver. These two preconditioned iterative solvers will be used as the basic solvers to carry out further benchmarking computations with overlapping moving grids.

6.2.3 Comparison of Fixed Body Solution vs. Moving Body Solution

In this case, the numerical solution of a cylinder moving in quiescent air is compared against the numerical solution of the equivalent case of a flow past a fixed cylinder. Both cases were simulated using a similar computational domain (see figure 6.7), with same grid dimensions and

Case	Method	Iterations	Total CPU Time (seconds)[1]
B1	DIRECT SOLVER (YALE IMPLEMENTATION)	33830	47780
B2	GMRES + ILU(0) + RCM	3000	3180
B3	GMRES + ILU(5) + RCM	5400	5060
B4	GMRES + ILU(10) + RCM	8000	9420
B5	GMRES + ADDITIVE SCHWARZ + RCM	3050	3250
B6	GMRES + SOR + RCM	9000	10500
B7	GMRES NO PRECONDITIONER + RCM	2075	16500
B8	BICGSTAB + ILU(0) + RCM	4000	4220
B9	BICGSTAB + ILU(5) + RCM	8400	8800
B10	BICGSTAB + ADDITIVE SCHWARZ + RCM	3400	3550
B11	BICG + ILU(0) + RCM	3100	5130
B12	BICG + BLOCK JACOBI + RCM	3800	3930
B13	CG + SOR + RCM	6600	6960
B14	CI + BLOCK JACOBI + RCM	Diverged	Diverged
B15	RI + SOR + RCM	Diverged	Diverged
B16	TFQMR + RCM	Diverged	Diverged

GMRES = Generalized Minimal Residual method
BICGSTAB = Stabilized Biconjugate Gradient Squared method
BICG = Biconjugate Gradient Squared method
CG = Conjugate Gradient method
CI = Chebychev iterative method
RI = Richardson iterative method
TFQMR =Transpose Free Quasi Minimal Residual method
[1] INTEL EM64T ×2 @ 2.4 GHz., 4.0 GB RAM., OS LINUX OPENSUSE 64 BITS.

Table 6.6: *Comparison of the performance of different direct and iterative solvers.*

grid spacing. The outer rectangular computational domain extends from $[x_a, x_b] \times [y_a, y_b] = [-3.0, 10.0] \times [-3.0, 3.0]$. For the fixed body simulation, the inner cylindrical computational domain is centred in $[x_{origin}, y_{origin}] = [1.0, 0.0]$, with respective inner and outer radius of $[radius_{inner}, radius_{outer}] = [0.5, 1.0]$. In the case of the moving body, the inner cylindrical computational domain is initially located in $[x_{origin}, y_{origin}] = [8.0, 0.0]$ and then it start to move from right to left with an instantaneous horizontal velocity of $(u, v) = (1.0, 0.0)$. The Reynolds number for both cases, based on the cylinder diameter, the kinematic viscosity and the inflow velocity $(u, v) = (1.0, 0.0)$ or the cylinder translational velocity $(u, v) = (1.0, 0.0)$ is $Re = 500$.

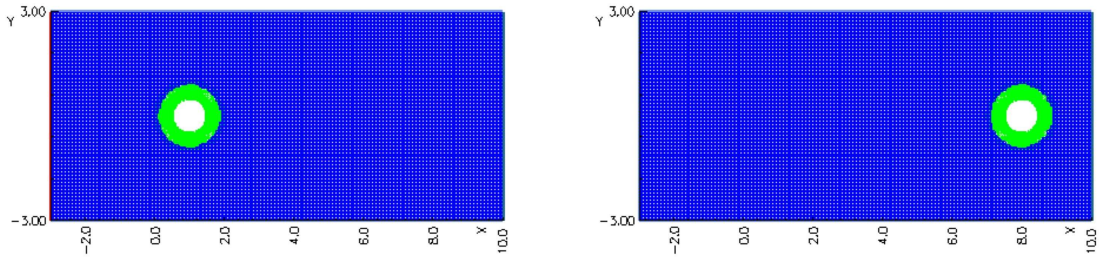

Figure 6.7: *Left: computational domain for the fixed cylinder case. Right: computational domain for the moving cylinder case (the cylinder is in the initial position and it moves from right to left).*

The pressure coefficient c_p, lift coefficient c_l and drag coefficient c_d are computed and compared for a nondimensional time of $t = 7.0$. In table 6.7, the computed c_l and c_d for both cases are shown. In figure 6.8, the c_p profiles for both cases are illustrated. In general, the match between the moving case and the stationary case is very good, which validates the moving overlapping grids framework.

Case	c_d	c_l
Fixed body	1.199795	0.000039
Moving body	1.191358	0.000038

Table 6.7: *Comparison of c_d and c_l coefficients for both cases at $t = 7.0$.*

6.2.4 Comparison to other Numerical and Experimental Results

One useful method of code validation is comparing the results of two or more flow solvers. A reasonably close agreement is encouraging, but is not sufficient to ensure the degree of accuracy, hence deeper validation must be done. Highest encouragement comes when the results from two or more codes closely agree, but they differ significantly in their approaches and algorithms (*i.e.*, finite-volume density-based method versus a finite-difference pressure-based method). Hereafter, we compare the numerical and experimental results obtained by other authors with the results obtained with the proposed flow solver.

Pedro *et al.* [140], numerically studied the propulsive efficiency of a flapping hydrofoil at a Reynolds number of 1100. They used a cell-centred pressure-based finite-volume flow solver with

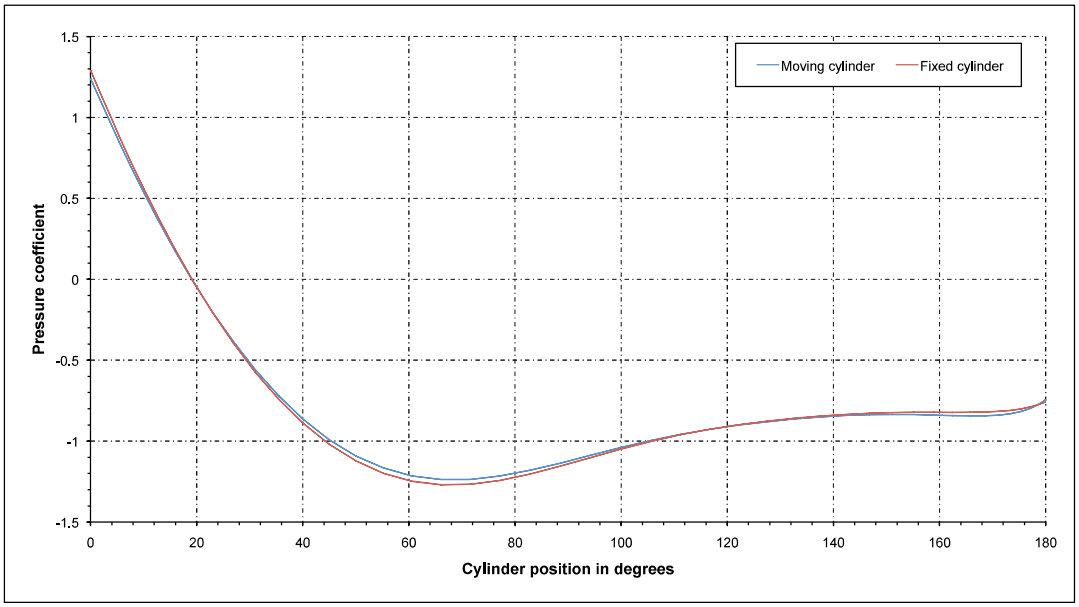

Figure 6.8: *Moving cylinder vs. fixed cylinder, pressure coefficient c_p comparison at a nondimensional time $t = 7.0$.*

Case number	$f_\alpha(Hz)$	k	$\alpha_a(^\circ)$
P1	0.0	0.0	0.0
P2	0.6366	2.0	5.0
P3	1.2732	4.0	5.0
P4	1.9098	6.0	5.0
P5	2.5464	8.0	5.0
P6	3.1830	10.0	5.0
P7	3.8196	12.0	5.0
P8	4.4562	14.0	5.0
P9	5.0928	16.0	5.0
P10	5.7294	18.0	5.0

Table 6.8: *Kinematics parameters for the pitching airfoil case.*

an explicit time-stepping on structured grids. The flow field was discretized by using central differences and to account for the mesh movement, an arbitrary Lagrangian-Eulerian (ALE) formulation was used [83]. In their work, Pedro *et al.* [140] studied airfoils undergoing pure pitching motion and combined heaving-and-pitching motions.

The first scenario considered by Pedro *et al.* [140], was the case of pure pitching. In table 6.8, the parameters governing the pitching motion are shown, where f_α is the pitching frequency, k the reduced frequency and α_a is the maximum pitch angle in degrees. In this case, it is expected that thrust will increase with an increase in either the maximum pitching angle or the frequency of oscillation. The summary of results is presented in tabular format in table 6.9. As it can be

seen in table 6.9, the average thrust coefficient ($\overline{c_t}$) and maximum lift coefficient ($\widehat{c_l}$) for cases P1 to P6, all compare favorably with the results obtained by Pedro *et al.* [140] and less favorably for cases P8, P9 and P10; this difference between both studies may be attributed to dynamic stall phenomena, grid quality issues and the highly irregular nature of the flow at such high oscillating frequencies.

	Pedro *et al.* [140]		Present results	
Case number	$\overline{c_t}$	$\widehat{c_l}$	$\overline{c_t}$	$\widehat{c_l}$
P1	-0.0581	0.0000	-0.1036	0.0000
P2	-0.1132	0.7107	-0.1280	0.6689
P3	-0.0904	2.3600	-0.1021	2.3389
P4	-0.0168	5.4341	-0.0204	5.2650
P5	0.0964	9.8144	0.0386	9.3430
P6	0.2174	15.5948	0.1779	14.9362
P7	0.4543	23.4162	0.4163	20.9512
P8	0.8624	34.1262	0.7356	29.3190
P9	1.2855	45.5064	1.0937	39.2653
P10	1.7467	57.5248	1.4418	49.3107

Table 6.9: *Average thrust coefficient $\overline{c_t}$ and maximum lift coefficient $\widehat{c_l}$ comparison for the pitching airfoil case.*

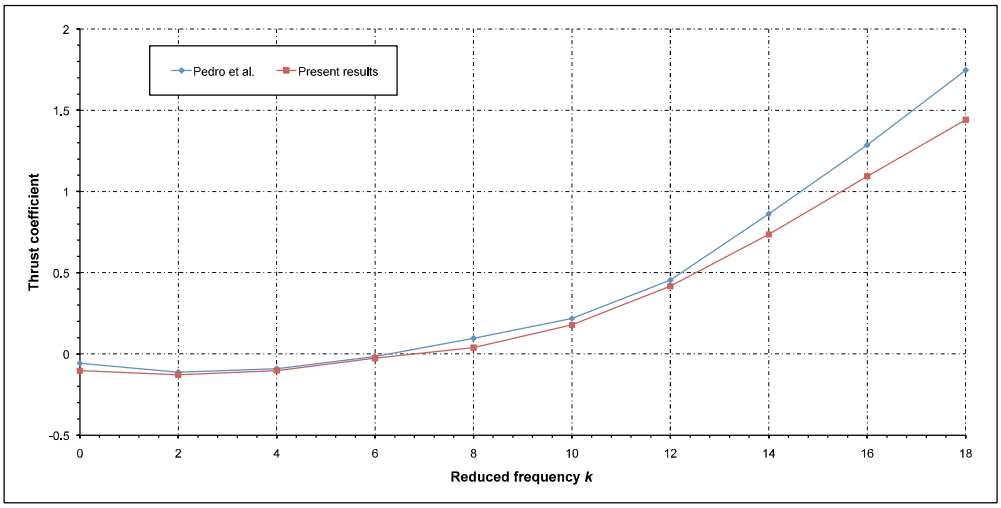

Figure 6.9: *Comparison of average thrust coefficient $\overline{c_t}$ results for the pitching airfoil case (negative values indicate drag production).*

The second scenario considered by Pedro *et al.* [140], is that of a combined heaving-and-pitching motion. The goal is to study the effect that the motion variables have on the values of thrust and lift coefficients. In table 6.10, the parameters governing the airfoil heaving-and-pitching kinematics are presented, where f is the frequency (for both, heaving-and-pitching motion), h_a is the maximum heaving amplitude, α_a is the maximum pitching angle in degrees, φ is the phase angle

between the pitching motion and the heaving motion in degrees, k is the reduced frequency and St is the Strouhal number.

Case number	f(Hz)	h_a	$\alpha_a(^\circ)$	$\varphi(^\circ)$	k	St
F1	0.225	1.0	5.0	90.0	0.7096	0.45
F2	0.225	1.0	10.0	90.0	0.7096	0.45
F3	0.225	1.0	15.0	90.0	0.7096	0.45
F4	0.225	1.0	20.0	90.0	0.7096	0.45
F5	0.225	1.0	25.0	90.0	0.7096	0.45
F6	0.225	1.0	30.0	90.0	0.7096	0.45
F7	0.225	1.0	40.0	90.0	0.7096	0.45
F8	0.255	1.0	50.0	90.0	0.7096	0.45

Table 6.10: *Kinematics parameters for the heaving-and-pitching airfoil case.*

Once again the results are presented in tabular form in table 6.11. In this table, it can be observed that the results obtained compare favorably with those obtained by Pedro *et. al.* [140], except for cases F6, F7 and F8 where the computed values are underpredicted with respect to the values obtained by Pedro *et al.* [140], nevertheless the trend is similar to that of the study carried on by Pedro *et al.* [140] (see figure 6.10). Once again, this differences can be attributed to the dynamic stall phenomena and grid quality issues. In flapping airfoils studies, the understanding of the vortical pattern created by the oscillating airfoil, responsible for the drag production in certain cases, but also for the thrust production in other cases, is a crucial issue. In figure 6.11 and figure 6.12, the comparison between the vorticity contours obtained by Pedro *et al.* [140] and the vorticity contours obtained in the current study for the heaving-and-pitching cases F1 and F6 (see table 6.10) are presented; as it can be seen the qualitative agreement is very satisfactory.

	Pedro *et al.* [140]		Present results	
Case number	$\overline{c_t}$	$\widehat{c_l}$	$\overline{c_t}$	$\widehat{c_l}$
F1	0.4324	8.3333	0.4245	8.0828
F2	0.6511	7.4834	0.6576	7.1699
F3	0.8226	6.6307	0.8360	6.5435
F4	0.9337	5.8176	0.9389	6.1133
F5	1.0046	5.0558	0.9601	5.6080
F6	1.0166	4.3721	0.9771	4.5964
F7	0.7404	3.2278	0.6964	3.6271
F8	0.2953	2.7345	0.2211	3.1785

Table 6.11: *Average thrust coefficient $\overline{c_t}$ and maximum lift coefficient $\widehat{c_l}$ comparison for the heaving-and-pitching airfoil case.*

Wang [207], motivated by the interest in the unsteady aerodynamics of insects flight, devised a computational tool to solve the Navier-Stokes equations around two dimensional moving airfoils. Wang, used a fourth order essentially compact finite difference scheme (EC4) with explicit time-stepping for solving the incompressible Navier-Stokes equations on structured grids; the method

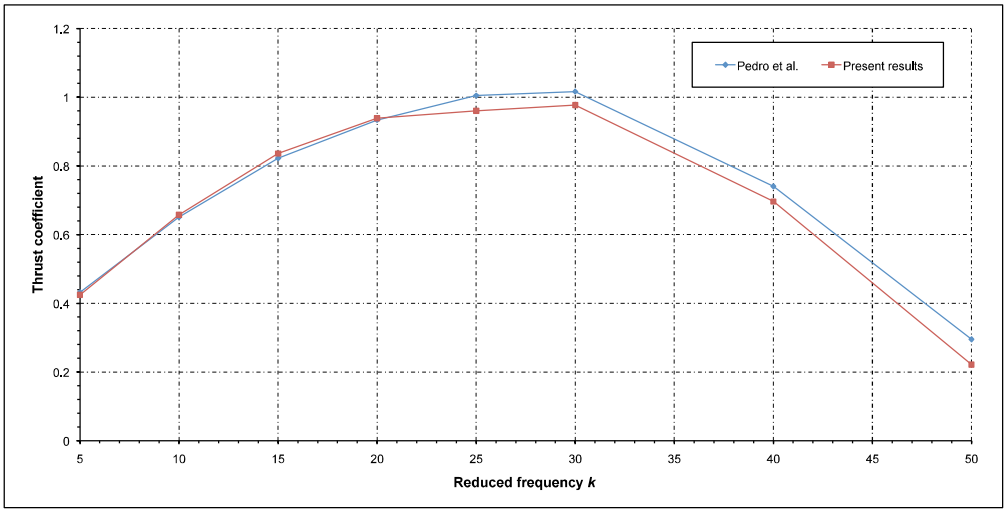

Figure 6.10: *Comparison of average thrust coefficient $\overline{c_t}$ results for the flapping airfoil case (negative values indicate drag production).*

was originally develop by Weinan and Liu [209]. The scheme uses the vorticity-stream function formulation. The governing equations are solved in the inertial frame of reference (hence the flapping motion is implemented by moving the airfoil and the computational domain at the same rate). The focus of this work was to study the frequency selection in forward flapping flight.

k	4.0	2.0	1.0	0.5
$\overline{c_t}$ Wang [207]	0.18	-0.06	-0.24	-0.08
$\overline{c_t}$ Present results	0.1094	-0.0528	-0.2164	-0.0714

Table 6.12: *Average thrust coefficient $\overline{c_t}$ comparison between the present results and the results obtained by Wang [207].*

To validate the code we computed the flow past a heaving ellipse and compare the results to those obtained by Wang [207]. In this numerical experiment the Reynolds number is set to $Re = 1000$, the forward flight velocity to $u = 1.0$, the airfoil chord to $c = 2.0$ and the Strouhal number to St=0.16 (a number based on forward dragonfly flight with $f = 40$ Hz, amplitude $A = 2.0$ cm and forward velocity $u = 5.0$ ms^{-1} [134]), as outlined by Wang [207]. Then we proceed to compute the average thrust coefficient $\overline{c_t}$ for different values of reduced frequency k=0.5, k=1.0, k=2.0 and k=4.0. In table 6.12, we compare the results obtained with those obtained by Wang [207]. As it can be seen, the agreement of the computed values with those obtained by Wang [207] for the cases where $k = 0.5$, $k = 1.0$ and $k = 2.0$ is acceptable, although the agreement is less acceptable for the case where $k = 4.0$, where the computed value is underpredicted. Despite this, the present computations show a similar trend in comparison with the results obtained by Wang [207].

Ramamurti and Sandberg [151], used a finite element incompressible flow solver based on unstructured grids for the study of the unsteady flow past oscillating airfoils. They simulated the viscous flow past a NACA 0012 airfoil at various pitching frequencies and combination of heaving-and-

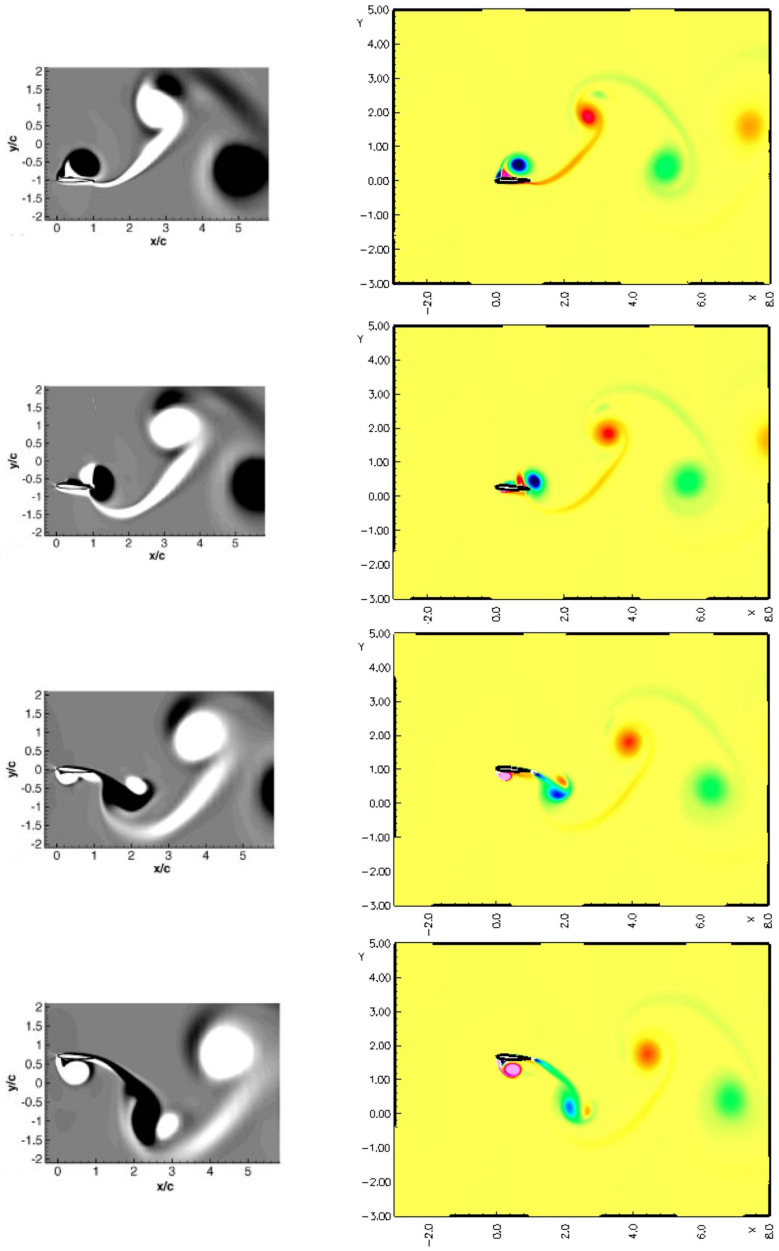

Figure 6.11: *Comparison of vorticity contours for the heaving-and-pitching airfoil case (case F1 in table 6.10). Left column: vorticity contours obtained by Pedro et al. [140]. Right column: present results. The first row is the beginning of one period, the second row is 1/8 of the period, the third row is 1/4 of the period and the last row is 3/8 of the period.*

pitching motion. To tackle the problem of moving bodies, they solved the governing equations using an ALE formulation. To assess the accuracy of the flow solver, they compared their results to those from the experiments carried by Koochesfahani [103]. In figure 6.13, the average thrust coefficient $\overline{c_t}$ values for the pitching airfoil case as described by Koochesfahani [103], are

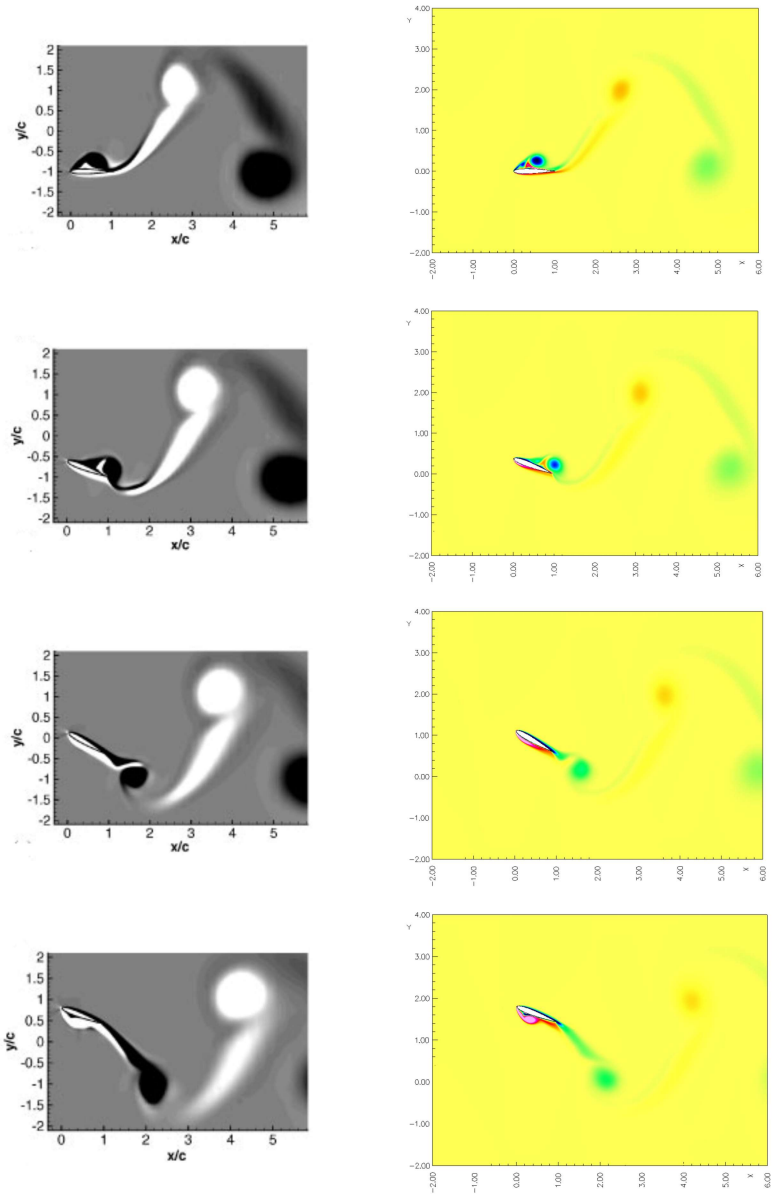

Figure 6.12: *Comparison of vorticity contours for the heaving-and-pitching airfoil case (case F6 in table 6.10). Left column: vorticity contours obtained by Pedro et al. [140]. Right column: present results. The first row is the beginning of one period, the second row is 1/8 of the period, the third row is 1/4 of the period and the last row is 3/8 of the period.*

illustrated. In this figure, the results obtained by Ramurti and Sandberg [151], Koochesfahani [103] and the current results are shown. As it can be seen in figure 6.13, the agreement between the numerical results obtained by Ramamurti and Sandberg [151] and the current results is acceptable and any difference in the solution can be attributed to the discretization scheme and

the grid resolution. In figure 6.13, the agreement between the numerical results (Ramamurti and Sandberg [151] and the current results) and the experimental results (Koochesfahani [103]) for values of reduced frequency above $k = 4$ is discouraging. The difference in these prediction, may be attributed to the method and experimental setup used by Koochesfahani [103] when measuring the forces.

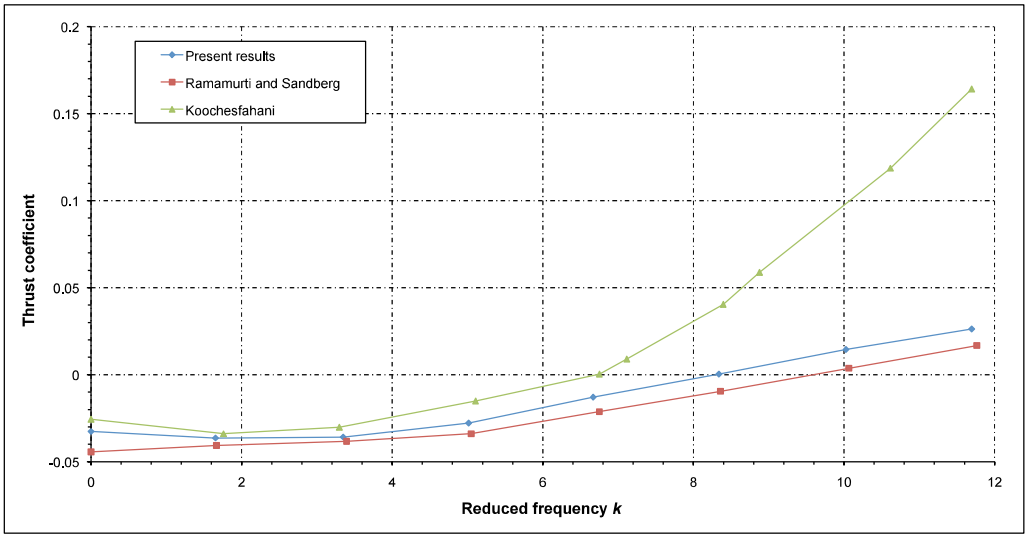

Figure 6.13: *Comparison of variation of thrust coefficient with reduced frequency (negative values indicate drag production).*

In his experimental investigation, Koochesfahani [103] computed the drag or thrust generated by the oscillating airfoil by measuring the momentum deficit or surplus downstream of the body. Usually the assumptions are made that at the cross-section where velocities are measured the flow is parallel, the pressure is freestream and the time-fluctuating quantities are small. Hence, the thrust can be computed from the following integral

$$T = \rho_\infty \int_{-\infty}^{+\infty} u\left(y\right)\left[u\left(y\right) - U_\infty\right] dy$$

If the velocity measurements are made sufficiently far downstream of the airfoil, such that the wake eddies are essentially diffused, then this method yields reasonable results, but if the measurements are made in a region where the eddies are still coherent, then the assumptions will not hold. It is not clear from Koochesfahani's article how far downstream the measurements were made for the thrust calculation, but velocity profiles presented in his article were made at one chord-length downstream of the trailing edge and at that distance the eddies are still present and coherent. Similar discrepancies with this experiment have been also documented by Jones and Platzer [97] and Liu and Kawachi [113].

Ramamurti and Sandberg [151], further validate their code by comparing their numerical results with the experimental results obtained by Anderson *et al.* [7]. The first case that was selected by them was that of a heaving-and-pitching airfoil at Reynolds number $Re = 1100$, Strouhal number

$St = 0.3$, maximum heaving amplitude $h_a/c = 1.0$, maximum pitching angle $\alpha_a = 15°$ and phase angle $\varphi = 90°$. In table 6.13, a comparison between the present results, the numerical results presented by Ramamurti and Sandberg [151] and the experimental results obtained by Anderson *et al.* [7] is shown. As it can be seen the agreement between both numerical results results is very satisfactory, even despite the fact that in both computations the results are overpredicted in comparison to the experimental results.

$\overline{c_t}^1$	$\overline{c_t}^2$	$\overline{c_t}^3$
1.3045	1.35	1.20
[1] Present results [2] Ramamurti and Sandberg [151] [3] Anderson *et al.* [7]		

Table 6.13: *Comparison of average thrust coefficient* $\overline{c_t}$.

Guglielmini and Blondeaux [62], computed the dynamics of the vortex structures generated by a foil in steady forward motion, plus a combination of harmonic heaving and pitching oscillations, by means of the numerical solution of the vorticity stream function equations. In their work, they compare their numerical solution with the experimental results obtained by Anderson *et al.* [7]. Qualitatively, Guglielmini and Blondeaux [62] found that the numerical vorticity field was in good agreement with the experimental vorticity visualizations obtained by Anderson *et al.* [7]. From the quantitative point of view, Guglielmini and Blondeaux [62] found that the forces measured experimentally were not in good agreement with the forces computed numerically and they attributed the discrepancy between the numerical and experimental values of c_t to the inaccurate procedure used by Anderson *et al.* [7] to evaluate c_t and to the two-dimensional approximation used by the numerical simulations.

In table 6.14., the average thrust coefficient $\overline{c_t}$ values for a heaving-and-pitching airfoil obtained by Guglielmini and Blondeaux [62] (numerical results), Anderson *et al.* [7] (experimental results) and in the present dissertation are shown. The experiments were conducted at Reynolds number $Re = 1100$ and Strouhal number $St = 0.32$. The maximum heaving amplitude h_a/c, maximum pitching angle α_a (in degrees) and phase angle φ (in degrees), are shown in table 6.14. As it can be seen, the agreement between the present results and the experimental results is acceptable. The difference between the present results and the results obtained by Guglielmini and Blondeaux [62], can be attributed to grid quality issues and some approximations of the flapping parameters done by them.

As a concluding remark, it can be said that from the qualitative and quantitative point of view, the agreement between all the numerical studies is quite satisfactory and any difference between these studies and the present results can be attributed to the discretization method, solution method, problem setup (initial conditions, boundary conditions, motion kinematics parameters and so on) and grid quality issues.

Case number	St	h_a/c	α_a	φ	$\overline{c_t}^1$	$\overline{c_t}^2$	$\overline{c_t}^3$
G1	0.32	0.75	30	90	0.6770	0.48	0.70
G2	0.32	0.75	30	75	0.4528	0.54	0.34
G3	0.32	0.75	30	105	0.7213	0.49	0.77
[1] Present results [2] Guglielmini and Blondeaux [62] [3] Anderson *et al.* [7]							

Table 6.14: *Comparison of average thrust coefficient* $\overline{c_t}$.

6.2.5 Comparison of Sequential Vs. Parallel Computations

Several simulations were performed in order to determine the correctness of the MPI[3] parallelization of the code on distributed memory parallel computers, by comparing the simulations performed in the parallel environment with the one performed in the serial environment. These simulations were also used to determine the relative speedup gained by the code parallelization. It is important to mention that the distributed memory parallel simulations were only performed for fixed bodies as currently the capability to simulate moving bodies in parallel is not fully implemented and still needs to be debugged and validated. Here, the unsteady flow past a cylinder at $Re = 400$ (as in section 2.2) is simulated in a distributed memory parallel computer (see table 6.21). The only difference with the serial environment case is that in this case, the grid (see table 6.5) is refined by a factor of 2.

Case	**Number of processors**	$\overline{c_d}$ $(100 < t < 120)$	**CPU Time (seconds)**	**Speedup**
P1	1[1]	1.7033	57560	-
P2	1[2]	1.7033	68670	0.83
P3	2	1.7033	25900	2.21
P4	4	1.7033	14480	3.97
P5	8	1.7033	7572	7.60
P6	16	1.7033	4255	13.52
P7	32	1.7033	3120	18.44
[1] Serial environment [2] Parallel environment				

Table 6.15: *Parallel computations benchmarking results.*

In table 6.15, average drag coefficient $\overline{c_d}$ values for various parallel simulations with different number of processors are presented, it can be seen that there is no difference between any of the benchmarking cases, indicating that the parallel implementation generates identical results for each one of the parallelization levels and the serial implementation. In figure 6.14 the vorticity magnitude contours around the cylinder are displayed, again it can be seen that there is no discernible difference between the solutions, giving confidence in the parallel code implementation.

Figure 6.15 shows the speedup factor gained by using the parallel code. For the parallel envi-

[3] http://www-unix.mcs.anl.gov/mpi/

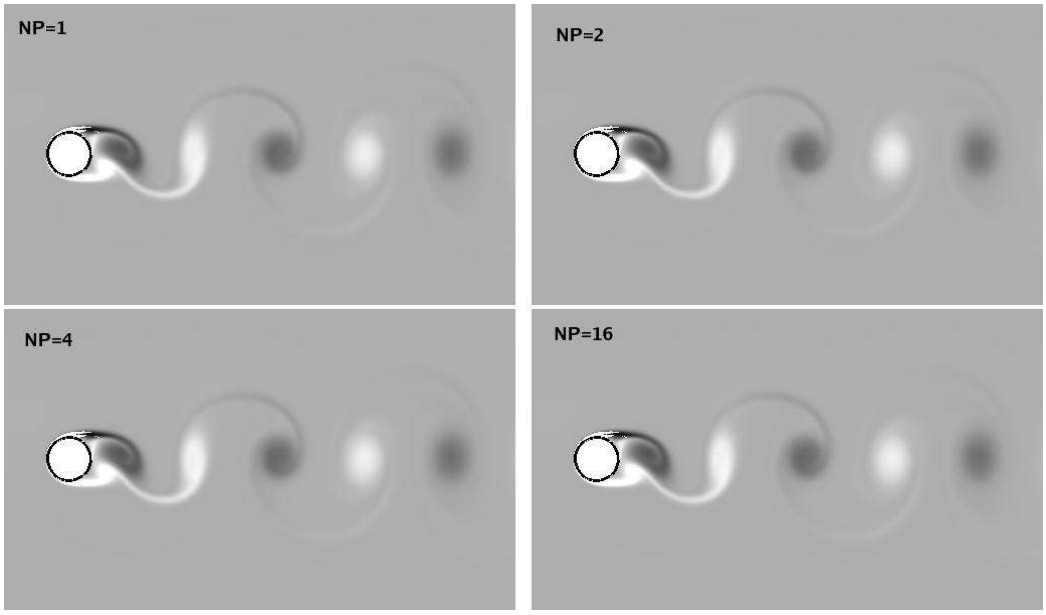

Figure 6.14: *Comparison of vorticity contours between the serial case and the parallel case for a nondimensional time $t = 100$. In the figure, NP stands for number of processors.*

ronment case using a single processor, the speedup factor (or in this case the speedown factor) is about 0.83, indicating that the parallelization added approximately 17% overhead to the serial code. A flattening of the curve in figure 6.15 with increasing processors would indicate that the maximum theoretical speedup has been reached, according with Amdahl's law [35]. At this point, inter-processor communication time will dominate over computation time.

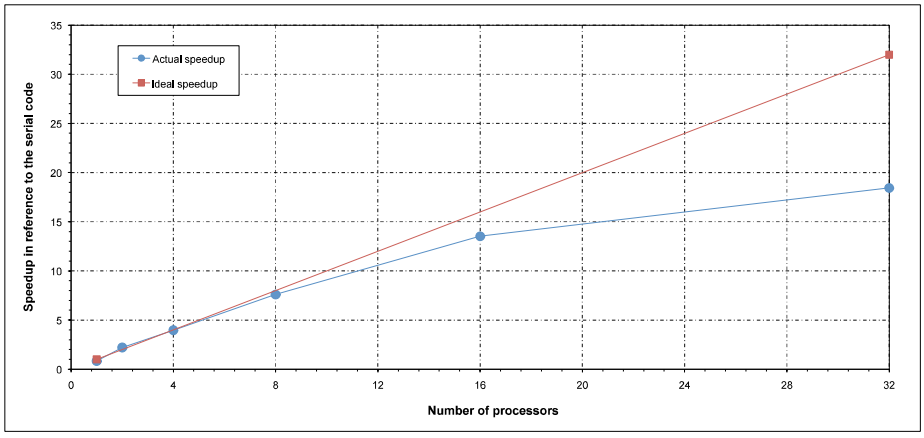

Figure 6.15: *Parallel speed up.*

Despite the fact that the distributed memory parallel implementation of the code still can not be used for moving body simulations, it was effectively used to obtain solutions in fine grids, which

were used later as initial conditions for the moving problem in coarser grids. Nevertheless, certain level of parallelization was reached when running moving body simulations by using OpenMP[4] on shared memory parallel computers, but we were limited by the number of threads (processors) and physical memory available.

6.3 Grid Refinement Study

A consistent numerical analysis will provide a solution which approaches the actual results as the grid resolution gets closer to zero. Thus, the discretized equations will approach the solution of the continuum equations. One significant issue in numerical computations is what level of grid resolution is appropriate. This is a function of the flow conditions, type of analysis, geometry, numerical methods, computational resources and other variables. One is often left to start with a grid resolution and then conduct a series of grid refinements to assess the effect of grid resolution, this is known as a grid refinement study or grid dependency study. In general, a grid refinement study, is a method used for determining the ordered discretization error in numerical simulations and involves performing the simulations on two or more successively finer grids. The method results in an error band on the computational solution which indicates the possible difference between the discrete and continuum value.

Roache [154] suggests a grid convergence index (GCI) to provide a consistent manner in reporting the results of grid refinement studies and perhaps provide an error band on the grid convergence of the solution. The GCI can be computed using two levels of grid; however, three levels are recommended in order to better estimate the order of convergence and to check that the solutions are within the asymptotic range of convergence. The basic idea behind the GCI is to approximately relate the results from any grid refinement study to the expected results from a grid doubling using a second-order method. The GCI is based upon a grid refinement error estimator derived from the theory of the generalized Richardson extrapolation. The object is to provide a measure of uncertainty of the grid convergence. The GCI is a measure of the percentage the computed value is away from the value of the asymptotic solution. It indicates an error band on how far the solution is from the asymptotic range. It also shows how much the solution would change with a further refinement of the grid. A small value of GCI indicates that the computation is well within the asymptotic range.

In practice, wherein the exact solution is not known, we perform at least three grid solutions and calculate two GCI, from the fine grid to the intermediate grid (GCI_{12}) and from the intermediate grid to the coarse grid (GCI_{23}). Then, the GCI on the fine grid is expressed as

$$\mathrm{GCI}_{fine} = F_s \frac{|\varepsilon|}{r^p - 1} \tag{6.4}$$

where ε is defined as

$$\varepsilon = \frac{f_2 - f_1}{f_1} \tag{6.5}$$

In eq. 6.5, f is the approximate solution or the value of an observed quantity and the sub-index $_1$

[4] http://openmp.org/wp/

represents the solution on the finest grid. In eq. 6.4, F_s is a factor of safety. The factor of safety is recommended to be $F_s = 3.0$ for comparisons of two grids and $F_s = 1.25$ for comparisons over three or more grids [154, 170].

It is important that each grid level yields solutions that are in or close to the asymptotic range of convergence for the computed solution. Then the constancy of

$$\mathrm{GCI}_{23} = r^p \mathrm{GCI}_{12} \tag{6.6}$$

indicates that the asymptotic range has been reached. If the desired accuracy level is known and results from the grid resolution study are available, one can then estimate the grid resolution required to obtain the desired level of accuracy,

$$r^* = \left(\frac{\mathrm{GCI}^*}{\mathrm{GCI}_{23}} \right)^{1/p} \tag{6.7}$$

where the superscript * denotes the desired level.

Finally, one must recognize the difference between a numerical result which approaches an asymptotic numerical value and one which approaches the true solution. It is expected that, as the grid is refined and resolution improves, the computed solution will not change much and will approach the asymptotic value (*i.e.*, the true solution). There still may be error between this asymptotic value and the true physical solution to the equations.

The method aforementioned will be used in the following sections in order to conduct the grid refinement study and to determine the most suitable grid in terms of computing time and solution accuracy.

6.3.1 Quantitative Study - Force Measurements

It is a very well known fact that the effect of mesh size is an important factor to consider when assessing the quality of a numerical solution. Here, we conduct a grid refinement or mesh dependency study in order to determine the most suitable grid in terms of computing time and solution accuracy from a quantitative point of view.

To conduct this study, we used the GCI method previously outlined. Here, a heaving airfoil is considered with different grid sizes, layouts and clustering. Several simulations were run at a Reynolds number equal to $Re = 1500$, with a maximum heaving amplitude of $h_a = 0.25$ and Strouhal number equal to $St = 0.5$ (thrust production regime). In each grid, there are typically up to 20 normal points in the direction normal to the airfoil surface (boundary layer area), mesh clustering is also used towards the leading and trailing edge, since we expect the vortices to be generated in these areas. All oscillating flow calculations were started from a fully converged stationary airfoil solution. In our calculations, unsteadiness is observed to disappear typically after 6 cycles of airfoil motion and further calculations show negligible non-periodicity. Thrust coefficient c_t and lift coefficient c_l are computed for each grid and their instantaneous and time average values are compared.

In table 6.16, the parameters for the three grids used for the GCI study are presented. Table 6.16

Grid $\mathbb{G}_g$	Background grid dimensions (BG)	Wake grid dimensions (WG)	Airfoil grid dimensions (AG)	$\frac{h_{\mathrm{AG}_1}}{h_{\mathrm{AG}_g}}$	Position of first node normal to the airfoil surface
$\mathbb{G}_1$	240×120	400×150	600×240	1	$0.000025 \times c$
$\mathbb{G}_2$	200×100	400×150	300×120	2	$0.00005 \times c$
$\mathbb{G}_3$	200×100	360×120	150×60	4	$0.0001 \times c$

Table 6.16: *Description of grids used for the grid refinement study (for force measurements).*

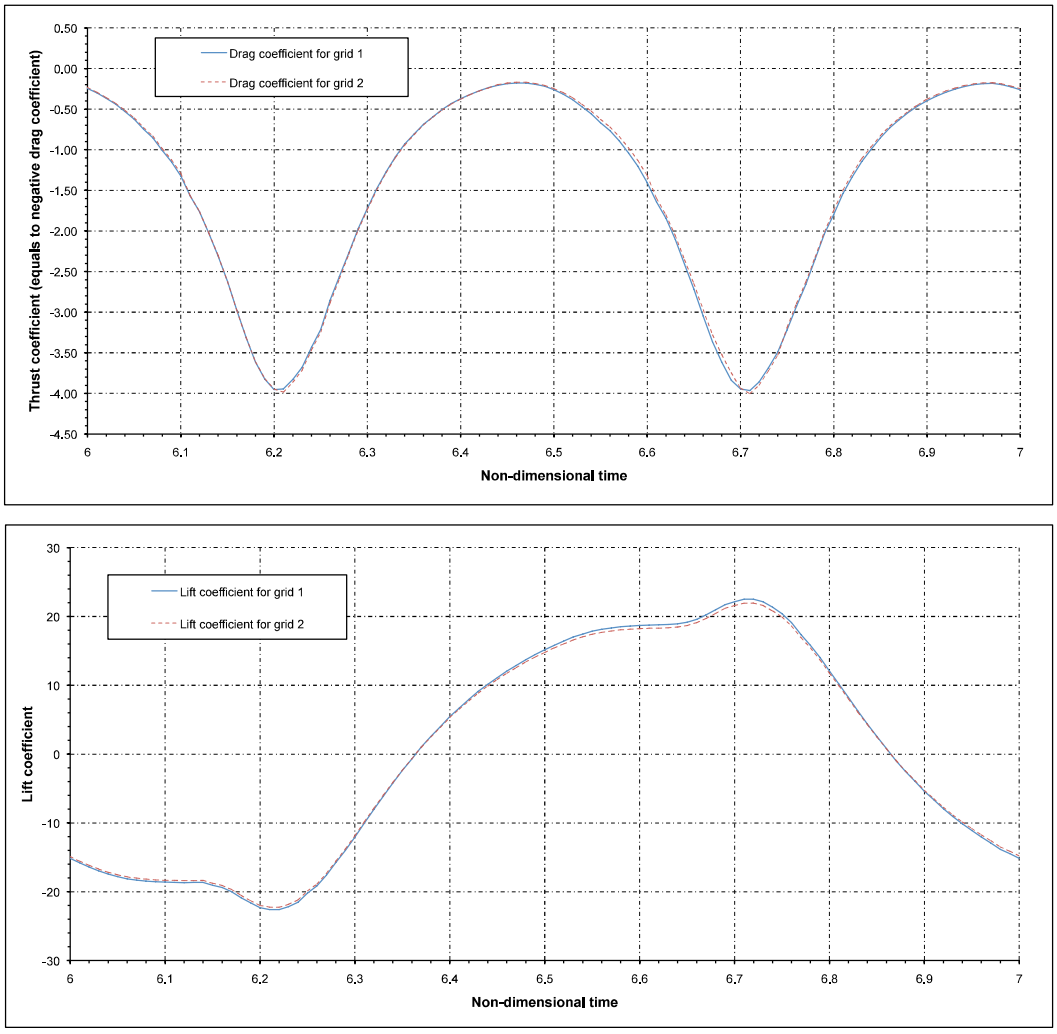

Figure 6.16: *Top: instantaneous drag coefficient c_d (negative values indicate thrust generation). Bottom: instantaneous lift coefficient c_l. Both quantities are shown for an interval equal to $6 < t < 7$.*

shows the grid dimensions, grid spacing refinement ratio (from the finest grid to the coarser grid), position of the first node normal to the airfoil surface and total number of nodes. The airfoil grid (AG) was used as a reference grid to conduct the GCI study and a grid spacing refinement ratio of $r = 2$ was used. Each simulation was checked for acceptable iterative convergence.

In table 6.17, the observed values of the average thrust coefficient $\overline{c_t}$ for each grid are presented; these values are used to calculate the observed order of convergence according to eq. 6.2 and the observed quantity value at zero grid spacing $f_{h=0}$ according to eq. 6.8. In figure 6.16, the instantaneous values of thrust coefficient c_t for $6 < t < 7$ for grids $\mathbb{G}_1$ and $\mathbb{G}_2$ are plotted. Also, the instantaneous values of lift coefficient c_l are shown in figure 6.16; here we can see how c_l varies symmetrically about the zero mean, so that the average vertical force is zero as expected from the symmetric heaving motion. The little bumps in the c_l curve result from the dynamics of the leading-edge vortices.

Grid $\mathbb{G}_g$	$h_{\mathrm{AG}_1}/h_{\mathrm{AG}_g}$	$\overline{c_t}$
$\mathbb{G}_1$	1	1.5267
$\mathbb{G}_2$	2	1.5195
$\mathbb{G}_3$	4	1.4905
Order of convergence p		2.009986
$f_{h=0}$		1 .529078

Table 6.17: *Observed values of the average thrust coefficient $\overline{c_t}$ for each grid. The observed order of convergence and the equivalent zero grid spacing values are also shown.*

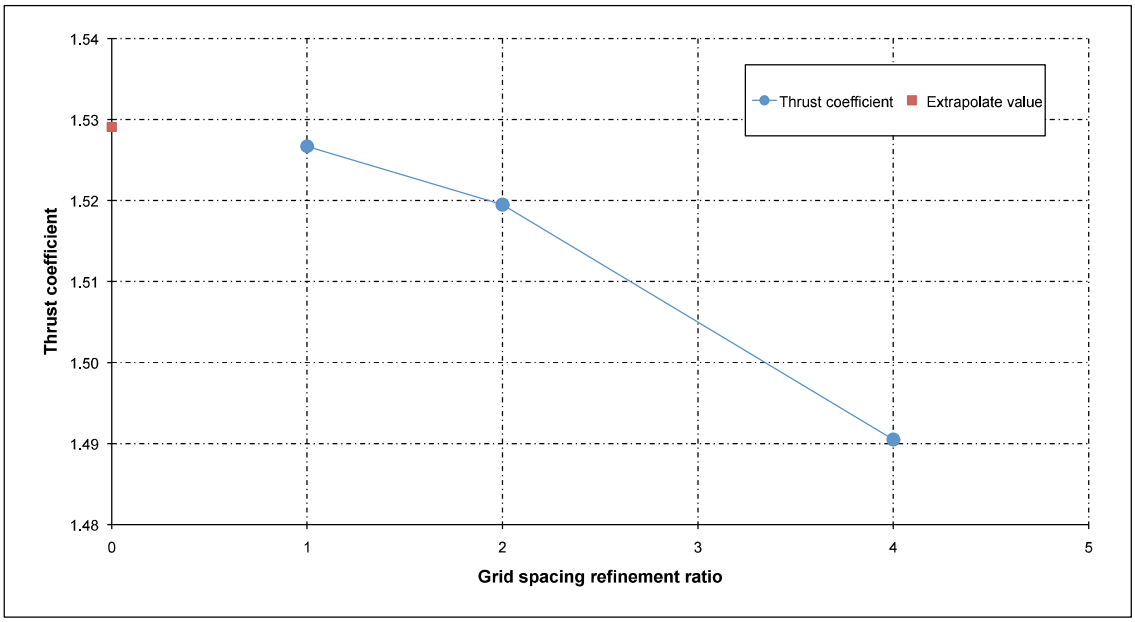

Figure 6.17: *Plot of observed quantity values (average thrust coefficient $\overline{c_t}$) for each grid. The equivalent zero grid spacing value is also plotted.*

We now apply Richardson's extrapolation

$$f_{h=0} \approx f_1 + \frac{f_1 - f_2}{r^p - 1} \tag{6.8}$$

using the two finest grids in order to obtain an estimate of the value of the observed quantities at an equivalent zero grid spacing ($f_{h=0}$) [154, 170]. Richardson extrapolation, eq. 6.8, will provide a fourth-order estimate of $f_{h=0}$ if f_1 and f_2 are computed using second order methods, otherwise it gives a third-order estimate [154, 170]. In figure 6.17, the observed values and $f_{h=0}$ are plotted.

The GCI for the fine grid solution was computed by means of eq. 6.4, here we set $Fs = 1.25$ as suggested by Roache [154] and Slater and Dudek [170]. Hereafter we proceed to calculate the

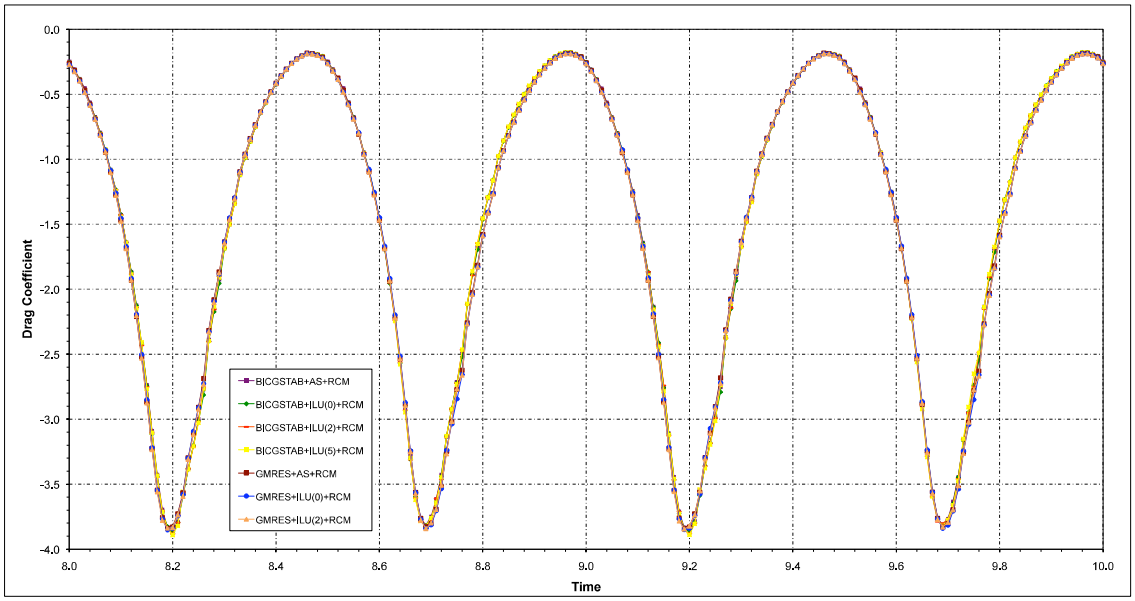

Figure 6.18: *Instantaneous drag coefficient c_d iterative convergence comparison for the heaving airfoil benchmarking case (negative values indicate thrust production).*

GCI_{fine} for grids 1 and 2 and grids 2 and 3 as follows

$$\mathrm{GCI}_{12} = 1.25 \times \frac{(1.5195 - 1.5267)}{1.5267} \times \frac{1}{2^{(2.009986)} - 1} \times 100\% = 0.194699\%$$
$$\mathrm{GCI}_{23} = 1.25 \times \frac{(1.4905 - 1.5195)}{1.5195} \times \frac{1}{2^{(2.009986)} - 1} \times 100\% = 0.787922\%$$

By using eq. 6.6, we now proceed to check that the solutions are in the asymptotic range of convergence. For the three grids we have

$$\frac{0.787922}{0.194699 \times 2^{2.009986}} = 1.004739$$

which is approximately one and indicates that the solution for the three grids are well within the asymptotic range of convergence.

Hereafter, we continue the solver benchmarking study. From the benchmarking case of an unsteady flow past a cylinder (see section 2.2), we found that the preconditioned GMRES solver, followed by the preconditioned BICGSTAB solver showed the best performance (see table 6.6). Here, we proceed to test both solvers, but for the case of a moving body. The case is exactly the same as the one used for the grid refinement study previously carried and the grid used for this benchmarking study is $\mathbb{G}_3$ (see table 6.16). In table 6.18, the results for the benchmarking computations are shown. As it can be seen, cases B3 and B7 show a small difference in terms of CPU time, however, the GMRES method with AS preconditioner seem to be the most promising solution method; hence, it will be used as the basic solution method for all the computations that will be carried on in the following chapters. In figure 6.18, the iterative convergence of the

Case	Method	Iterations	Total CPU Time (seconds)
B1	GMRES + ILU(0) + RCM	37120	22930
B2	GMRES + ILU(2) + RCM	37098	26960
B3	GMRES + ADDITIVE SCHWARZ + RCM	37096	19180
B4	BICGSTAB + ILU(0) + RCM	36742	22430
B5	BICGSTAB + ILU(2) + RCM	36780	22330
B6	BICGSTAB + ILU(5) + RCM	36754	32070
B7	BICGSTAB + ADDITIVE SCHWARZ + RCM	36762	19770

Table 6.18: *Comparison of the performance of different direct and iterative solvers.*

instantaneous drag coefficient c_d for $8 < t < 10$ is shown. In this figure, each case shown in table 6.18 is plotted. As it can be seen, the curves almost exactly collapse into one another, indicating this that each method is converging to the same solution.

6.3.2 Qualitative Validation - Wake Structures Resolution

In the previous section, we performed a quantitative study to determine the best suited grid for force measurement on the airfoil surface in terms of computing time and solution accuracy. In this section, we perform a grid refinement study but from the qualitative point of view (wake structures resolution). The grids used are summarized in table 6.19 and as for the quantitative study, there are up to 20 points in the direction normal to the airfoil surface and mesh clustering is used towards the leading and trailing edge. In figures 6.19 to 6.23, the wake structures resolution for each grid shown in table 6.19 is illustrated. From these results, it can be stated that the wake structures becomes approximately grid independent starting at the grid level case $\mathbb{G}_3$ towards $\mathbb{G}_1$ (see table 6.19).

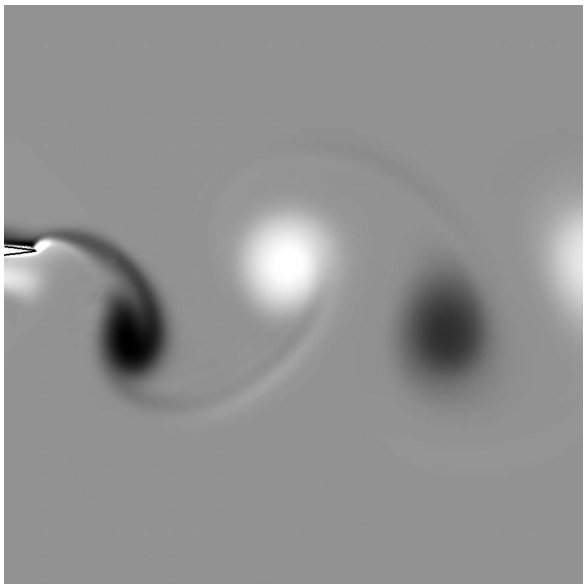

Figure 6.19: *Grid refinement study of the wake structures resolution for the heaving airfoil benchmarking case. Vorticity contours corresponding to* $\mathbb{G}_5$ *(see table 6.19) are shown.*

6.3.3 Summary of the Quantitative and Qualitative Grid Refinement Study

A quantitative and qualitative grid refinement study has been conducted and hereafter we shortly summarize the results obtained. From the grid convergence index study, it is found that grids $\mathbb{G}_1$ and $\mathbb{G}_2$ (see figure 6.17) are well in the asymptotic range of convergence. Based on the GCI index values found, we could say that the average thrust coefficient $\overline{c_t}$ is estimated to be 1.529078 with an error band of 0.194699% for grid $\mathbb{G}_1$ (see table 6.16) and within an error band of 0.787922% for grid $\mathbb{G}_2$ (see table 6.16), both well within the asymptotic range of convergence. From the qualitative study, it is found that for $\mathbb{G}_3$, $\mathbb{G}_2$ and $\mathbb{G}_1$ (see table 6.19), the wake structure becomes grid independent. In table 6.20, some of the grids used for the qualitative and quantitative

Grid $\mathbb{G}_g$	Background grid dimensions (BG)	Wake grid dimensions (WG)	Airfoil grid dimensions (AG)	$\frac{h_{\mathrm{AG}_1}}{h_{\mathrm{AG}_g}}$	Position of first node normal to the airfoil surface
$\mathbb{G}_1$	240×120	400×150	600×240	1	$0.000025 \times c$
$\mathbb{G}_2$	240×120	400×150	300×120	2	$0.00005 \times c$
$\mathbb{G}_3$	200×100	360×120	300×120	2	$0.00005 \times c$
$\mathbb{G}_4$	200×100	360×120	150×60	4	$0.0001 \times c$
$\mathbb{G}_5$	200×100	360×120	150×60	4	$0.0002 \times c$

Table 6.19: *Description of grids used for the grid refinement study (wake structures resolution).*

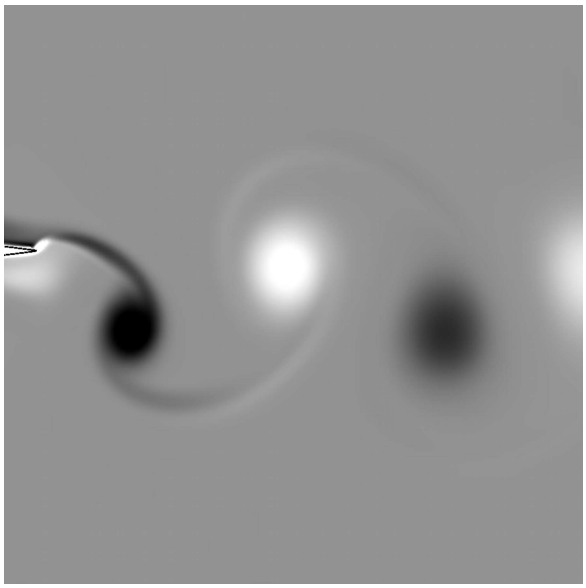

Figure 6.20: *Grid refinement study of the wake structures resolution for the heaving airfoil benchmarking case. Vorticity contours corresponding to* $\mathbb{G}_4$ *(see table 6.19) are shown.*

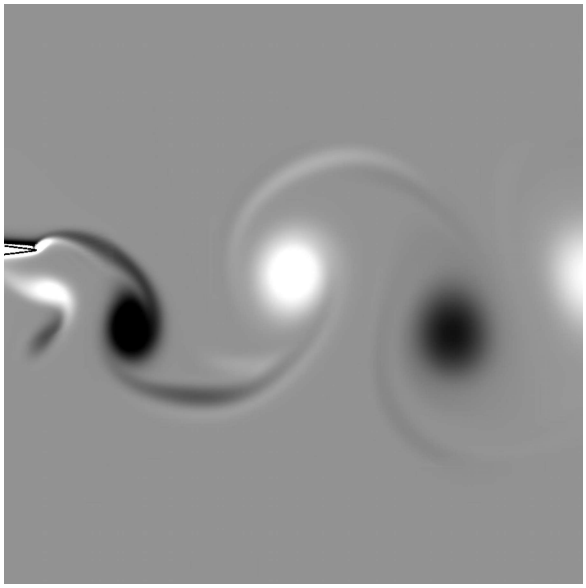

Figure 6.21: *Grid refinement study of the wake structures resolution for the heaving airfoil benchmarking case. Vorticity contours corresponding to* $\mathbb{G}_3$ *(see table 6.19) are shown.*

grid refinement study are summarized. Grids $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ and $\mathbb{G}_4$ from the quantitative (force measurements) and qualitative (wake structure resolution) standpoint provide grid independent results, but taking into account the computational resources available, CPU time restrictions and solution accuracy, grid $\mathbb{G}_3$ will be used as the base grid to perform all the $2\mathbb{D}$ computations.

Grid $\mathbb{G}_g$	Background grid dimensions (BG)	Wake grid dimensions (WG)	Airfoil grid dimensions (AG)	Position of first node normal to the airfoil surface	Total number of nodes $\mathbb{G}_1 \cup \mathbb{G}_1 \cup \mathbb{G}_1$
$\mathbb{G}_1$	240×120	400×150	600×240	$0.000025 \times c$	232800
$\mathbb{G}_2$	240×120	400×150	300×120	$0.00005 \times c$	124800
$\mathbb{G}_3$	200×100	400×150	300×120	$0.00005 \times c$	116000
$\mathbb{G}_4$	200×100	360×120	300×120	$0.00005 \times c$	99200
$\mathbb{G}_5$	240×120	400×150	150×60	$0.0001 \times c$	97800
$\mathbb{G}_6$	200×100	360×120	150×60	$0.0001 \times c$	72200
$\mathbb{G}_7$	200×100	360×120	150×60	$0.0002 \times c$	72200
$\mathbb{G}_8$	200×100	360×120	150×60	$0.0004 \times c$	72200

Table 6.20: *Description of grids used for the grid refinement study.*

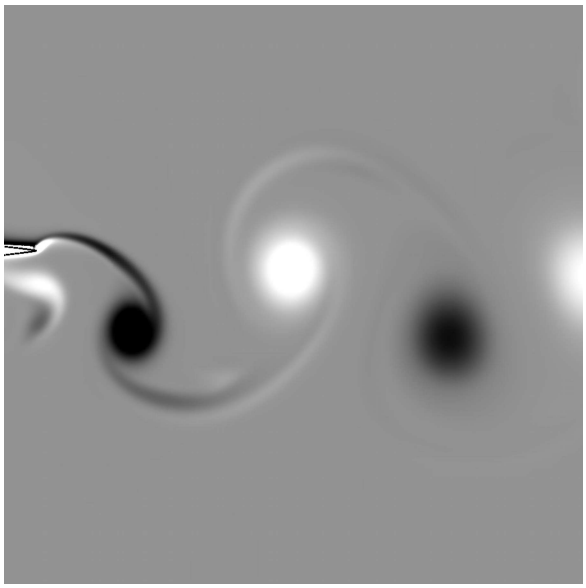

Figure 6.22: *Grid refinement study of the wake structures resolution for the heaving airfoil benchmarking case. Vorticity contours corresponding to* $\mathbb{G}_2$ *(see table 6.19) are shown.*

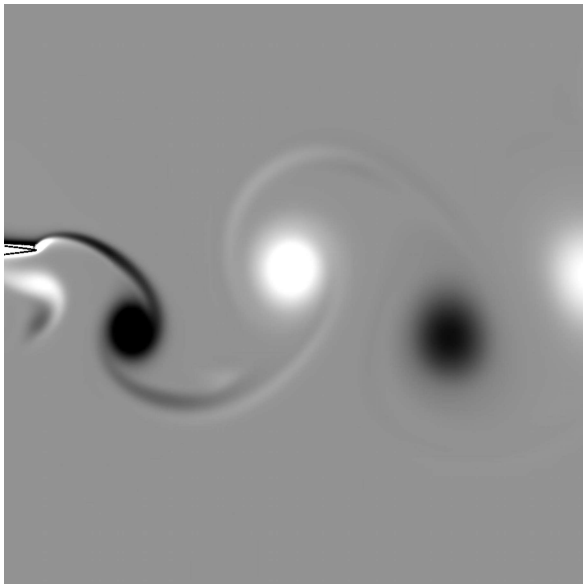

Figure 6.23: *Grid refinement study of the wake structures resolution for the heaving airfoil benchmarking case. Vorticity contours corresponding to* $\mathbb{G}_1$ *(see table 6.19) are shown.*

The overlapping grid system layout used for the qualitative and the quantitative grid refinement study is shown in figure 6.24. In both studies, the background grid (BG) extends from $[x_a, x_b] \times [y_a, y_b] = [-3.0, 9.0] \times [-2.75, 3.25]$. The wake grid (WG) extends from $[x_a, x_b] \times [y_a, y_b] = [-1.0, 7.0] \times [-1.25, 1.75]$. The airfoil grid (AG) which is an hyperbolic grid [75], is marched a distance equal to $0.5 \times c$ from the airfoil surface, with the airfoil leading edge centred at

Benchmarking case description	Computational resources
Unsteady flow past a cylinder (serial environment)	AMD64 ×2 @ 1.6 GHz., 4.0 GB RAM., OS LINUX OPENSUSE 64 BITS.
Heaving airfoil	INTEL EM64T ×2 @ 2.4 GHz., 4.0 GB RAM., OS LINUX OPENSUSE 64 BITS.
Unsteady flow past a cylinder (parallel environment)	NEC Xeon EM64T Cluster with 200 nodes @ 3.2 GHz. 160 nodes @ 1 GB RAM + 40 nodes @ 2 GB RAM. Infiniband node-node interconnection. OS LINUX TAO 64 BITS. http://www.hlrs.de/hw-access/platforms/cacau/

Table 6.21: *Computational resources used in each benchmarking case.*

$[x_{origin}, y_{origin}] = [0.0, 0.0]$ in the initial position and where the airfoil chord c is equal to 1.0. The overlapping grid system layout dimensions are chosen in such a way that the vertical distance to the top and bottom boundaries of the BG and WG, when the airfoil is in the mean position of the flapping period, is equal to $3.0 \times c$ and $1.5 \times c$ respectively. In the case of a bigger or smaller domain, the grid dimensions are scaled in order to keep the same grid spacing as for this domain. The initial conditions used in each heaving airfoil simulation are those of the fully converged solution of the corresponding fixed airfoil case, obtained with a finer grid (with a corresponding overlapping grid system spacing refinement ratio of $r = 2$). The left boundary in figure 6.24, corresponds to an inflow boundary condition with $(u, v) = (1.0, 0.0)$ and the top, bottom and right boundaries are outflow boundaries. The airfoil has no-slip boundary condition. Finally, we mention in table 6.21 the computational resources used to run each benchmarking case.

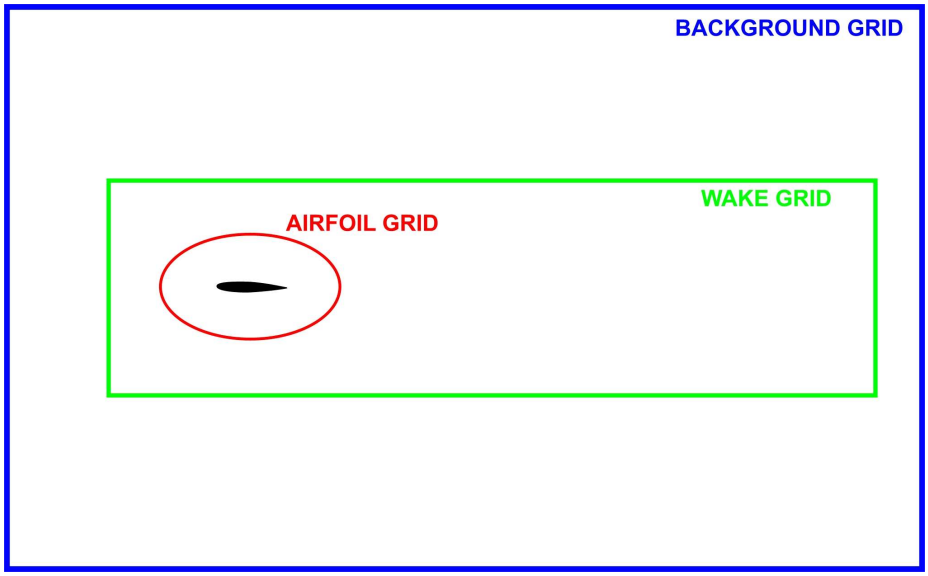

Figure 6.24: *Overlapping grid system layout.*

6.4 Closing Remarks

From the results presented in this chapter, it can be said that from the qualitative and quantitative point of view, the agreement between all the numerical studies is very satisfactory, fact that gives us confidence on the numerical accuracy of the computational tools implemented. Any difference between all the numerical studies carried out in this chapter can be attributed to the differences on the discretization method, solution method, problem setup (initial conditions, boundary conditions, motion kinematics parameters and so on) and grid quality issues.

In this chapter, it was also conducted a qualitative (wake structure resolution) and quantitative (force measurements) grid refinement study in order to determine the grid best suited for the numerical computations to be carried out. The *"optimal"* grid was chosen based on the GCI index, the wake structure resolution, CPU time restrictions and solution accuracy.