ECCOMAS Congress 2016
VII European Congress on Computational Methods in Applied Sciences and Engineering
M. Papadrakakis, V. Papadopoulos, G. Stefanou, V. Plevris (eds.)
Crete Island, Greece, 5–10 June 2016

UNSTEADY ADJOINT TO THE CUT-CELL METHOD USING MESH ADAPTATION ON GPUS

K.D. Samouchos[1], S.G. Katsanoulis[1], and K.C. Giannakoglou[1]

[1]National Technical University of Athens, School of Mech. Eng., Parallel CFD & Optimization Unit, Athens , Greece
e-mail: ksamouchos@yahoo.com, skatsanoulis@gmail.com, kgianna@central.ntua.gr

Keywords: Cut-Cell Method, Moving Boundaries, Adjoint-Based Optimization, GPUs.

Abstract. *In its first part, this paper presents a cut-cell method for simulating 2D unsteady inviscid flows of compressible fluids in domains with moving boundaries. To solve shape optimization problems, the gradient of the aerodynamic shape optimization with respect to (w.r.t.) the design variables is computed via the continuous adjoint approach.*

An automatic grid adaptation method based on a quad-tree data structure allows low memory usage for storing geometric data. The no-penetration condition along the surface of the emerged bodies, is implemented using a second-order cut-cell approach. To avoid numerical instabilities during the solution of the flow and adjoint PDEs, very small cut-cells are merged with neighboring active cells to yield the finite-volume where the flow or adjoint equations are integrated.

In transonic flow simulations, the Cartesian grid is adapted not only to the solid boundaries but, also, to the evolving flow discontinuities. The refined grid follows the moving solid wall and discontinuities, by means of local refinement and derefinement processes. The paper focuses on the schemes used to interpolate the flow solution fields between successive time-steps, as the grid becomes adapted to the moving geometry. During the solution of the unsteady adjoint PDEs, since the adjoint solver marches backwards in time and the adapted Cartesian grids are continuously changing, care should be taken so as to have full access to the necessary geometric quantities at all cells, even if these did not exist at previous time-steps. Both the primal and adjoint solvers are programmed on GPUs (Graphics Processing Units), using CUDA-C, to reduce the optimization wall-clock time.

Results are presented for both the analysis and optimization problems.

1 Introduction

Flows with complex moving boundaries are of utmost importance in many engineering applications. The traditional approach involves the use of body-fitted grids and the corresponding discretization methods. Nevertheless, irrespective of the grid type (structured or unstructured), these methods require a highly delicate and often costly grid deformation process, especially when large boundary movements occur. A viable alternative encompasses the use of the Immersed Boundary Method, originally proposed by Peskin [1]. In the latter, the computation takes place on a Cartesian grid allowing for a simple and automatic grid generation.

Mittal and Iaccarino [2] discern two basic approaches stemming from the generalized term "Immersed Boundary Methods", namely the continuous and discrete forcing approaches. The continuous one makes use of an additional source term in the momentum equations to approximate the effect of solid boundaries. On the contrary, discrete forcing approaches rely on the discrete representation of the immersed boundary without altering the governing equations. Prevailing among them is the cut-cell finite volume method [3, 4] which guarantees both global and local conservation. This is achieved through the reshaping of cells that are intersected by the solid boundary, i.e., the so-called cut-cells, by making them conform to the boundary while the remaining fraction of the cut-cells as well as cells lying in the interior of solid bodies are totally discarded.

The present paper deals with the development of a cut-cell finite volume method for 2D inviscid flows of compressible fluids. In order to avoid numerical stability issues caused by the existence of small cells, a cell-merging technique is adopted. To maximize the accuracy of the flow simulation, the grid in the vicinity of both the solid boundary and flow discontinuities (e.g. shock waves) is refined. Extending this idea to moving boundaries, grid is dynamically adapted following their motion, which calls for a method to project the solution onto the grid in the next time-step. The software is developed in order to run on NVIDIA GPUs.

In addition, an optimization loop is implemented based on the continuous adjoint approach. That is, after the adjoint PDEs are derived from the flow equations, both for steady and unsteady flows, these are discretized and numerically solved utilizing the already developed method. In the course of this process, the correct computation of the derivatives of geometric quantities is crucial since the special nature of the stationary Cartesian grid must taken into account. Problems related to the maximization of lift of a stationary airfoil as well as the time-averaged lift of a pitching airfoil are showcased.

2 Grid Generation

The computational grid is dynamically generated based on criteria which make it compatible with the cut-cell approach for solving the flow and adjoint equations around stationary and moving bodies. Starting point is a uniform Cartesian grid within a rectangular domain; the initial cell volume is defined by the user (Ω_{max}). Then, cells intersected by the solid walls are recursively subdivided into four quadrants. The basic rule governing this iterative process is that cut–cells with area greater than Ω_{min} should further be subdivided; Ω_{min} is a threshold value defined by the user. This extra cell refinement may affect neighboring non–cut–cells which should also be subdivided in order to meet a second rule stating that a cell face cannot have more than two neighbors. This rule prevents the formation of tiny grid cells in contact with much greater ones, which makes the CFD solver prone to numerical inaccuracies. Two additional rules must be satisfied, see fig. 1. According to the first of them, cut-cells are not allowed to have any of their four edges intersected twice by the solid wall. The last rule does

not allow cut-cells to have their four edges intersected by solid walls. In either case, the cell must be subdivided further. In the case of a stationary isolated airfoil, the refined Cartesian grid for an inviscid flow simulation looks like that shown in fig. 2.

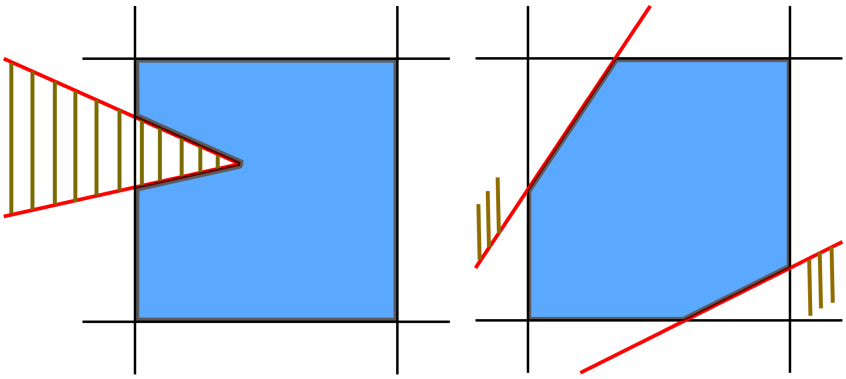
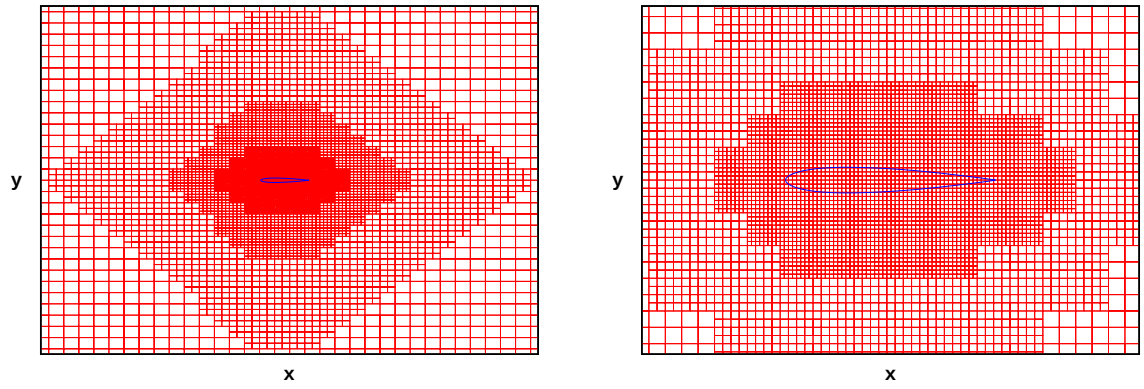


Figure 1: Visual representation of the third and fourth rule imposed during the Cartesian grid adaptation process, as described in the text. Both cases are not acceptable.



(a) (b)

Figure 2: Two views on the Cartesian grid in the vicinity of the boundary of an isolated airfoil, for an inviscid flow simulation. Further local refinement is possible if flow discontinuities appear.

Grid generation is based on a quad-tree data structure [5] which allows fast operations and ensures easy identification and access to neighboring cells, as required during the numerical solution of the flow and adjoint equations. Each and every cell is given a unique pair of integers (i, j), facilitating a lot the computation of geometric quantities such as cell volumes and barycentric coordinates. Among other, this two-index numbering system results to a noticeable reduction in memory footprint.

A special treatment of the cut–cells is needed in order to satisfy the conservation equations in the vicinity of solid boundaries with the required accuracy. For all cut–cells, their parts belonging to the non-fluid domain are discarded. This gives rise to fluid or active cells which can be shaped as triangles, quadrangles or pentagons; taking into account the aforementioned constraints, no other shape is possible in a 2D problem. By definition, the active cell area corresponds solely to the part of the cell which is within the fluid. In the course of this process, new geometric quantities such as the newly formed edges, the active area and the corresponding barycenter position appear.

A common problem in the cut-cell method is the creation of very small cells, depending on the instantaneous locations of the solid walls within the grid. It is known [7, 8] that the presence of these cells might cause convergence issues. To tackle this problem, a cell-merging approach is used, as shown in fig. 3, according to which the flow variables are stored at the barycenter of the merged cell.

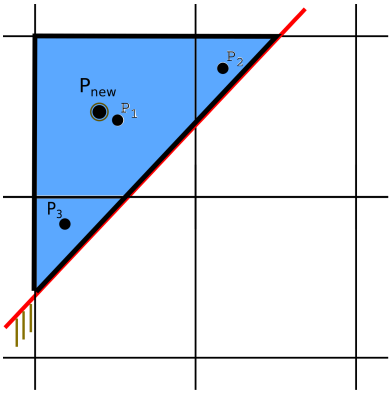


Figure 3: Three cut–cells with barycenters P_1, P_2 and P_3. Since P_2 and P_3 are considered as very small cut-cells, these are merged into a single cell with barycenter P_{new}. Flow variables for the newly formed cell are all stored at P_{new}.

3 Governing Equations & Discretization

The Euler equations for unsteady compressible flows in an inertial reference frame are written as

$$\frac{\partial \overrightarrow{U}}{\partial t} + \frac{\partial \overrightarrow{f}_x}{\partial x} + \frac{\partial \overrightarrow{f}_y}{\partial y} = \overrightarrow{0} \tag{1}$$

where $\overrightarrow{U} = \begin{bmatrix} \rho & \rho u & \rho v & E \end{bmatrix}^T$ is the vector of conservative variables and $\overrightarrow{f}_x = \begin{bmatrix} \rho u & \rho u^2 + p & \rho uv & u(E+p) \end{bmatrix}^T$, $\overrightarrow{f}_y = \begin{bmatrix} \rho v & \rho uv & \rho v^2 + p & v(E+p) \end{bmatrix}^T$ are the inviscid fluxes in the x and y direction, respectively, ρ the fluid density, u and v the Cartesian velocity components, p the static pressure and E the total energy per unit volume.

The integration of equations (1) is based on the cell-centered finite volume scheme. The integration volume consists of the part of the cell that lies in the fluid. For the merged cells, this volume is defined as the fluid part of each of the constituent cells (e.g. the blue area in fig. 4). According to this, if Ω_P^{k+1} is the control volume of a cell with centroid P (to be referred to as cell P) at time-step $k+1$, S_P^{k+1} is its boundary and $\overrightarrow{n}$ is its normal unit vector, the corresponding residual of the flow equations is

$$\overrightarrow{R}_P := \int\limits_{\Omega_P^{k+1}} \frac{\partial \overrightarrow{U}}{\partial t} d\Omega + \int\limits_{S_P^{k+1}} \overrightarrow{f}_i \cdot n_i \, dS = \overrightarrow{0} \tag{2}$$

The discretization of eqs. (2) is based on the Roe [9] scheme. The no-penetration condition is imposed along the solid walls, that is, $(\overrightarrow{u} - \overrightarrow{u}_w) \cdot \overrightarrow{n} = \overrightarrow{0}$, where $\overrightarrow{u}_w$ is the wall velocity. Based on the latter, at all cut-cells' boundary edges that correspond to solid walls the flux vector becomes $\overrightarrow{f}_w = \begin{bmatrix} \rho u_{w,n} & \rho u_{w,n} u + p n_x & \rho u_{w,n} v + p n_y & u_{w,n}(p+E) \end{bmatrix}^T$ where $u_{w,n} = \overrightarrow{u}_w \cdot$

$\overrightarrow{n}$. This flux corresponds to the center B of the solid boundary segment of the cut–cell as shown in fig. 4(a). Variables appearing in $\overrightarrow{f}_w$ are computed at point B using the Taylor expansion formula $\overrightarrow{U}_B = \overrightarrow{U}_P + \overrightarrow{PB} \cdot \left(\nabla \overrightarrow{U}\right)_P$ where $\left(\nabla \overrightarrow{U}\right)_P$ is computed using a weighted least–squares method. The same process can be extended in the case of merged cells, as shown in fig. 4(b)

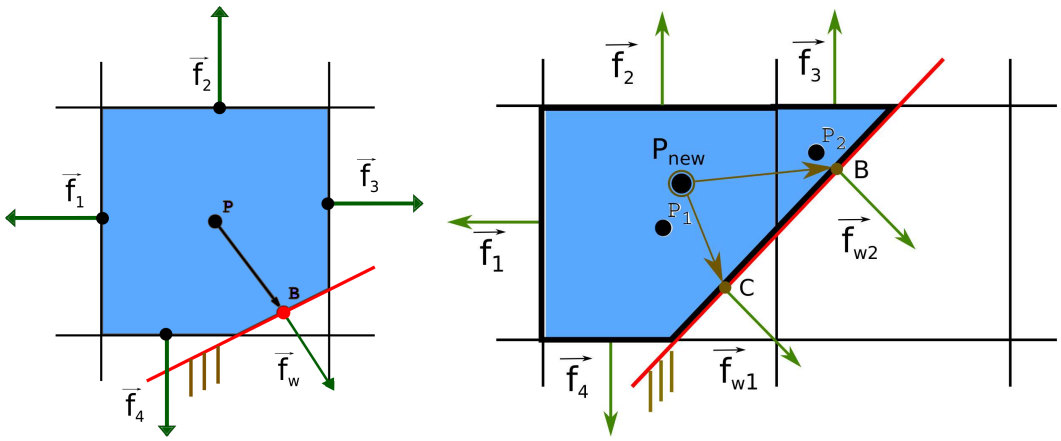


Figure 4: (a): The no-penetration condition is imposed via $\overrightarrow{f}_w$ at point B using the extrapolated variables from barycenter P. (b): Two cut–cells with barycenters P_1 and P_2 are merged into a single cell with barycenter P_{new}. Values at P_{new} are extrapolated to B and C in order to compute $\overrightarrow{f}_{w1}$ and $\overrightarrow{f}_{w2}$.

Regarding the discretization of the temporal term in eq. (2) when the immersed body moves inside the Cartesian grid, four different cases might occur. The simplest case is that of fluid cells at time-step t^k which remain intact at t^{k+1}. Thus, the integration volume is unchanged and a second-order backward differentiation scheme is readily applied. The second case concerns all cells being intersected by the solid boundary at any of the three time-steps (t^{k-1}, t^k, t^{k+1}). To account for changes in Ω_P, the Reynolds transport theorem [10] is applied,

$$\int\limits_{\Omega_P^{k+1}} \frac{\partial \overrightarrow{U}}{\partial t} d\Omega = \frac{\partial}{\partial t} \int\limits_{\Omega_P^{k+1}} \overrightarrow{U}\, d\Omega - \int\limits_{S_P^{k+1}} \overrightarrow{U}\, u_{w,n}\, dS \tag{3}$$

For the first term on the r.h.s., a second-order backward scheme is used

$$\frac{\partial}{\partial t} \int\limits_{\Omega_P^{k+1}} \overrightarrow{U}\, d\Omega \simeq \frac{1}{2\Delta t} \left(3\overrightarrow{U}_P^{k+1} \Omega_P^{k+1} - 4\overrightarrow{U}_P^{k} \Omega_P^{k} + \overrightarrow{U}_P^{k-1} \Omega_P^{k-1} \right) \tag{4}$$

while, for the surface term, the assumption

$$\begin{aligned} \int\limits_{S_P^{k+1}} \overrightarrow{U}\, u_{w,n}\, dS &\simeq \overrightarrow{U}_w^{k+1} \int\limits_{S_P^{k+1}} u_{w,n}\, dS \\ &= \overrightarrow{U}_w^{k+1} \left(\frac{\partial \Omega_P}{\partial t} \right)^{k+1} \simeq \frac{\overrightarrow{U}_w^{k+1}}{2\Delta t} \left(3\Omega_P^{k+1} - 4\Omega_P^{k} + \Omega_P^{k-1} \right) \end{aligned} \tag{5}$$

is made, where $\overrightarrow{U}_w^{k+1}$ is the vector of flow variables computed at the wall face.

According to the third case, a cell lying entirely in the solid boundary at t^k, becomes fluid cell at t^{k+1}. In such a case, Ω_P^k and Ω_P^{k-1} are equal to zero and eqs. (4) and (5) are properly transformed. Finally, cells that are solidified should transfer their conservative values to their neighboring cells in the fluid region via a merging process.

4 Dynamic Grid Adaptation

In case of moving solid boundaries, the grid must continuously be adapted to the new boundary shape. During the dynamic adaptation, which embodies both coarsening and refinement tasks, the resulting grid at each new time-step must comply with the four rules mentioned in section (2). The grid adaptation from one time-step (t^k) to the next (t^{k+1}) takes place as follows:

Initially, the grid undergoes a coarsening process, according to which every cell with volume less than Ω_{max} is merged with its neighbors. Then, the solid boundary is moved to its new position and, starting from the just coarsened grid, cells split anew in the vicinity of the displaced wall. Through this procedure, grid refinement in the vicinity of the new position of the solid boundary is achieved while the no longer necessary refined grid around the previous wall position is canceled. This process is exemplified in fig. 5. Along with the grid adaptation, interpolation should be employed to transfer the flow solution from the previous time-step (t^k) to the new one (t^{k+1}).

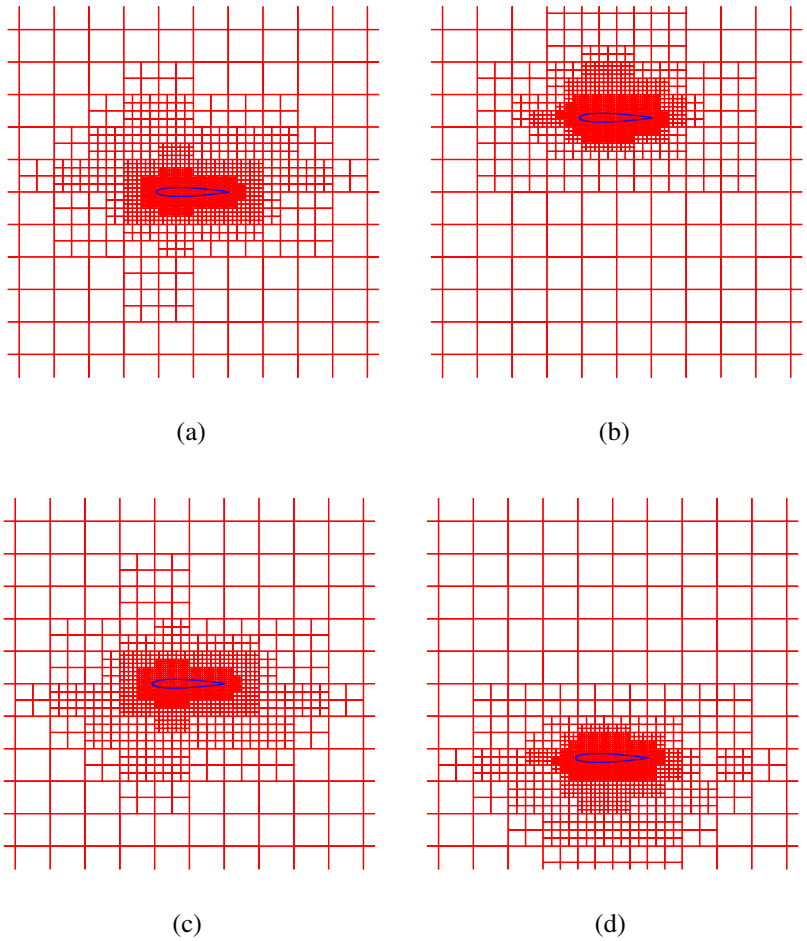


Figure 5: Translational periodic, in the vertical direction, motion of an airfoil. Adapted grid at four time-steps within the same period.

In order to attribute values to the conservative variables at the barycenter of the newly formed cells at t^{k+1}, an intermediate step (t^*) at which the solid boundary is considered to its previous position (t^k) and, then, immersed in the new grid (t^{k+1}), is used. The flow variables $\overrightarrow{U}^*$ at the intermediate step represent the time history of the newly created cells at t^{k+1}. The values transferred from t^k to t^* must ensure the satisfaction of the conservation laws.

For the sake of clarity, two simple cases will be demonstrated. According to the first one, due to the movement of the solid boundary, a cell at t^k is decomposed into four smaller cells at t^{k+1}, fig. 6. Considering that each of the newly formed cells P_i should share the same $\overrightarrow{U}$ values

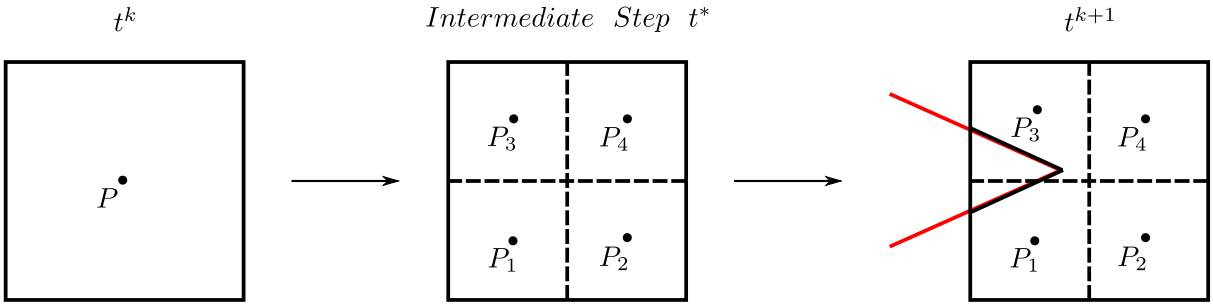


Figure 6: Cell P at step t^k, decomposed into four smaller cells P_i at step t^{k+1}, makes use of an intermediate step (t^*) in order to attribute values to the flow variables $\overrightarrow{U}$ to each of the newly formed cells at step t^{k+1}.

at t^k, it is

$$\overrightarrow{U}^*_{P_i} = \overrightarrow{U}^k_P \,, \quad i = 1, \ldots, 4 \tag{6}$$

where $\overrightarrow{U}^k_P$ is the flow variables array of cell P, at t^k, and $\overrightarrow{U}^*_{P_i}$ is the unknown vector of conservative variables for each newly formed cell P_i at t^*.

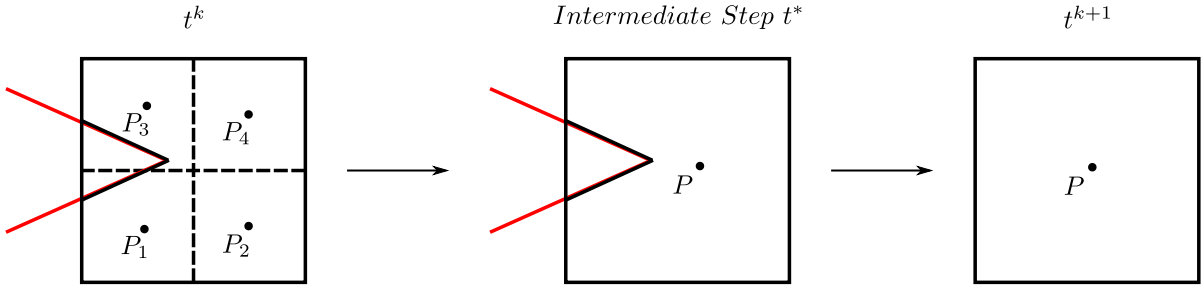


Figure 7: The process of associating values of the flow variables $\overrightarrow{U}$ to the newly created cell barycenter P at t^{k+1}.

In the second case, four cells P_i at t^k are merged to form a single new cell, fig. 7. In this case,

$$\overrightarrow{U}^*_P = \frac{\sum_{i=1}^{4} \overrightarrow{U}^k_{P_i} \, \Omega^k_{P_i}}{\Omega^*_P} \tag{7}$$

The values $\overrightarrow{U}^*_P$, computed by eqs. (6) and (7) and Ω^*_P, are considered as the flow fields and volumes, respectively, at t^k and are referred to as $\overrightarrow{U}^k_P$ and Ω^k_P in eq. (4). The same process must be followed in order to compute $\overrightarrow{U}^{k-1}_P$ and Ω^{k-1}_P.

5 Validation of the flow solver

In this section, the use of the flow solver in steady and unsteady flows is demonstrated and validated against other results. The steady flow case encompasses grid refinement due to the formation of a shock wave over the suction-side of an airfoil with non-zero incidence, while the unsteady case showcases the grid refinement according to both the airfoil motion and the formation of the shock wave.

The software was programmed in CUDA-C and runs on NVIDIA GPUs by taking advantage of the superior hardware characteristics they exhibit compared to CPUs. In order to exploit the advantages of GPUs, special care must be taken in memory handling. Moreover, an appropriate solution method should be chosen, which can be efficiently parallelized. Here, the discretized equations are solved in each pseudo-time step using the Jacobi method. The simulation was carried out on a single computational node with two NVIDIA Tesla K40 GPUs with 12 GB of GPU memory each.

The NACA0012 airfoil is investigated at transonic flow conditions ($M_\infty = 0.8$ and $a_\infty = 1.25\ deg$). The Mach number field around the airfoil with the adapted grid in the vicinity of the shock wave is presented in fig. 8. The same case was studied using several Euler-flow solvers on body-fitted grids in [11] and, as shown in Table 1, the results of the present cut–cell solver are in good agreement with those solvers.

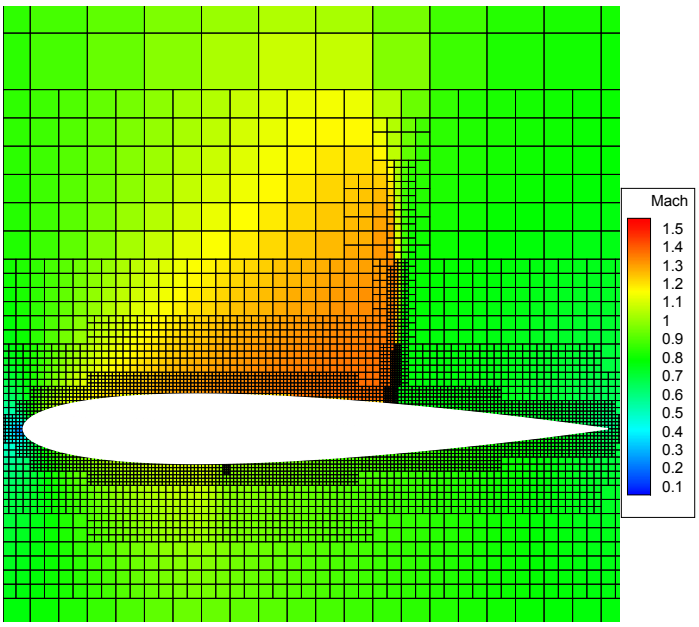


Figure 8: Transonic flow around NACA0012 airfoil-Steady flow: Computed Mach number field.

	C_L
Present cut–cell method	0.3492
Body–fitted Min	0.3482
Body–fitted Max	0.3562

Table 1: C_L comparison between the present cut-cell method and other solvers using body-fitted grids where Min and Max denote the spread of C_L values found in [11].

As far as the unsteady flow solver is concerned, its validation is carried out using the same airfoil undergoing a periodic pitching motion about $c/4$ (c is the chord) with 2.51^o amplitude and 0.016^o mean angle of attack. The Mach number is 0.755 and the reduced frequency $k := \frac{\omega}{2V_\infty}$ is 0.0814, where ω is the angular velocity of the airfoil. The computed Mach number fields at the two extreme positions of airfoil motion are shown in fig. 9.

This case has been investigated experimentally, by Landon [12]; over and above, an in-house Euler code [13] running on a body-fitted grid was used to get CFD results to compare with. Comparisons are presented in fig. 10.

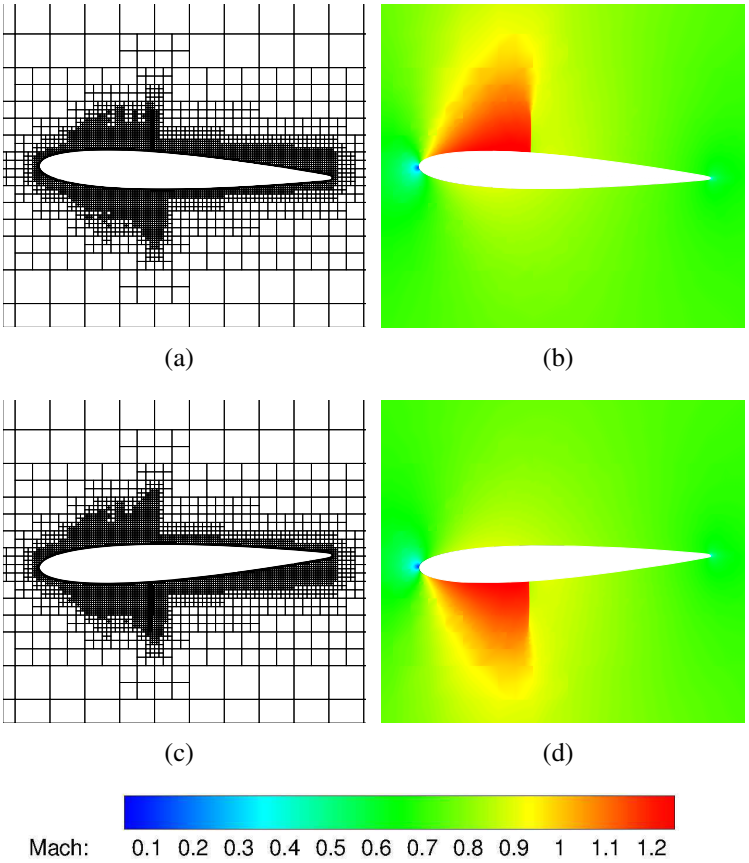


Figure 9: Transonic flow around the pitching NACA0012 airfoil: (a),(c): Dynamically adapted grid with further refinement in the vicinity of the shock wave and (b),(d): Mach number fields at the two extreme positions of airfoil motion.

6 Formulation of the Steady Adjoint Method

The objective function to be maximized is the lift of an airfoil, namely

$$F = \int_{S_w} p n_k r_k dS \tag{8}$$

where r_k are the components of the unit vector normal to the freestream velocity. The airfoil is parameterized using Bézier curves, the coordinates of the control points of which are the design variables (b_q) of the optimization process. The necessary derivative of F is computed by

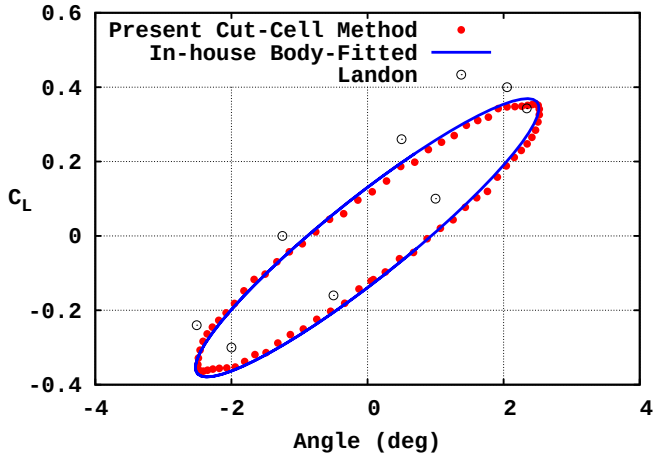


Figure 10: Time-evolution of the lift coefficient, in terms of the angle formed between the airfoil chord and the horizontal axis. Present results are contrasted with CFD results obtained using an in-house Euler solver for body-fitted grids [13] and measurements by Landon [12].

expanding it as an augmented objective function $F_{aug} = F + \int_{\Omega} \Psi_n R_n d\Omega$, where $\Psi_n,\ n = 1,4$ are the adjoint variable fields. By differentiating F_{aug} w.r.t. to the design variables and setting the multipliers of the variations in all flow variables equal to zero, the steady adjoint equations

$$-A_{nmk}\frac{\partial \Psi_n}{\partial x_k} = 0 \tag{9}$$

arise [6]. Boundary conditions to be imposed along the solid wall are $\Psi_{k+1} n_k + n_k r_k = 0$ and, in the farfield, $\Psi_n = 0,\ n = 1,4$. Then, the gradient of F w.r.t. the design variables b_q becomes

$$\begin{aligned}\frac{\delta F}{\delta b_q} &= \int_{S_w} p\frac{\delta (n_k r_k dS)}{\delta b_q} + \int_{S_w} (\Psi_{k+1} p - \Psi_i f_{ik})\frac{\delta n_k}{\delta b_q} dS \\ &- \int_{S_w} \Psi_i \frac{\partial f_{ik}}{\partial x_l} n_k \frac{\delta x_l}{\delta b_q} dS + \int_{S_w} \Psi_i \frac{\partial f_{il}}{\partial x_l}\frac{\delta x_k}{\delta b_q} n_k dS\end{aligned} \tag{10}$$

Eq. (10) requires the computation of $\frac{\delta x_k}{\delta b_q}$ which stands for the variation in the airfoil's surface w.r.t. the design variables. In contrast to body-fitted grids, the Cartesian grid used in the cut-cell method is stationary even if the geometry changes during the optimization process. This must be taken into consideration and, thus, the above term is computed at the intersection points of the geometry with the Cartesian grid. Each intersection point can move only along the corresponding cut-cell's edge. Consequently, it is the projection of $\frac{\delta x_k}{\delta b_q}$ to the edge direction that must only be taken into account in order to compute the derivatives at the mid-point of each cut-cell's solid edge, as shown in fig. 11.

7 Steady Adjoint Results

Below, the shape optimization of a symmetric airfoil is examined. The airfoil ($M_\infty = 0.44$ and $a_\infty = 0\ deg$) pressure and suction sides are both parameterized using Bézier curves. The design variables are the coordinates of six control points marked with empty circles, fig. 12.

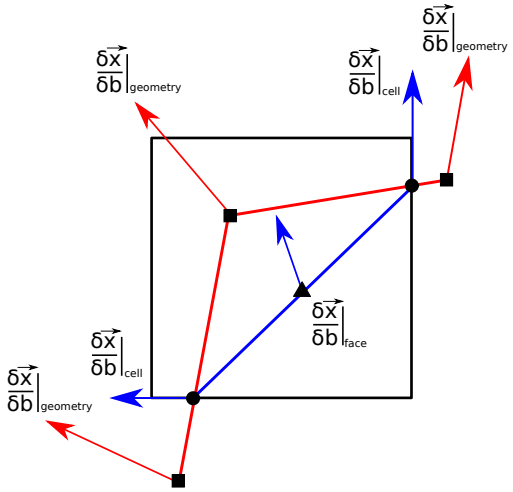


Figure 11: The red polyline denotes the surface of the geometry which cuts the cell in two points. The blue line segment is part of the boundary of the cut-cell. The derivatives of the boundary w.r.t. the design variable b (red vectors) are initially computed at the points marked with squares by differentiating the geometry's parameterization. These derivatives are used to compute $\frac{\delta\vec{x}}{\delta b}$ at the intersection points marked with circles which are taken into account to compute the derivative at the mid-point of the line segment.

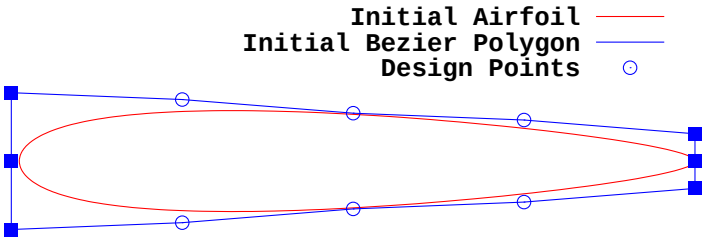


Figure 12: Bézier control polygon generating the initial shape of the airfoil. Points marked as empty circles signify control points, the coordinates of which are used as design variables.

The maximization of lift, defined by eq. (8), is chosen as the target of the optimization process.

After ten optimization cycles, the suction side is cambered enough, causing an increase in the lift coefficient from zero to approximately 0.36, as shown in figs. 13 and 14.

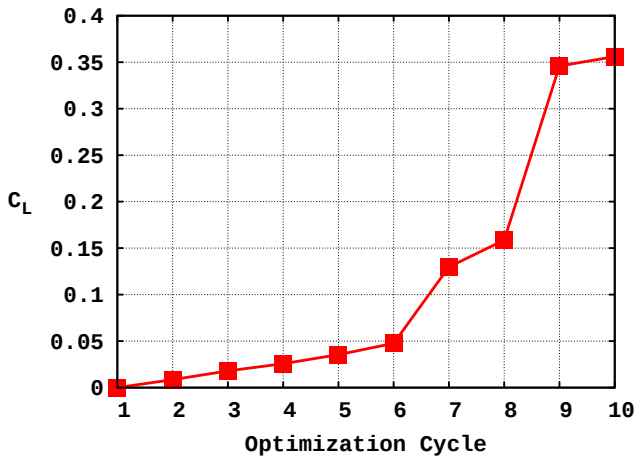


Figure 13: Subsonic flow around the airfoil parameterized as in fig. 12. Adjoint-based optimization for lift maximization. Lift coefficient evolution during the optimization cycles.

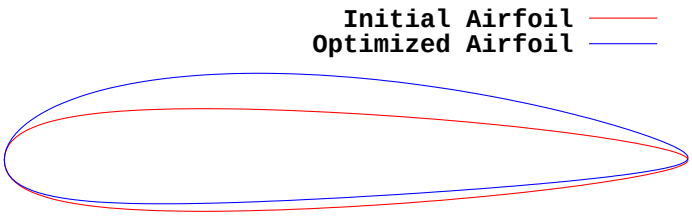
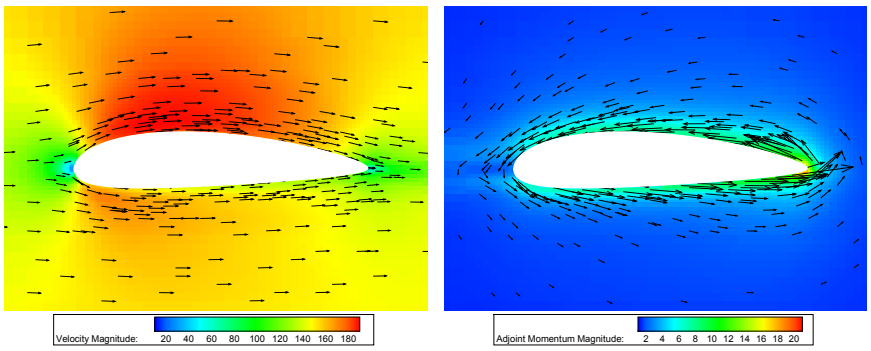


Figure 14: Subsonic flow around the airfoil parameterized as in fig. 12. Adjoint-based optimization for lift maximization. Initial and optimized airfoils.

For the objective function defined in eq. (8) with r_k pointing along the y-axis, the imposed adjoint conditions at the wall can be simplified as $\Psi_n = -n_y$, where Ψ_n is the normal adjoint momentum and n_y is the ordinate of the normal unit vector over the airfoil, pointing towards the solid. Along the pressure side, n_y is positive enforcing the adjoint momentum flux to exit the airfoil. In contrast, along the suction side, n_y is negative so the adjoint momentum flux points to the opposite direction. The aforementioned conditions result in the adjoint momentum magnitude field presented in fig. 15(b) while the corresponding flow velocity field is shown in fig. 15(a).



(a) (b)

Figure 15: Subsonic flow around the airfoil parameterized as in fig. 12. Adjoint-based optimization for lift maximization. (a): Velocity magnitude field and velocity vectors. (b): Adjoint momentum magnitude field and vectors. Both figures correspond to the optimized geometry.

8 Formulation of the Unsteady Adjoint Method

The unsteady adjoint method is tailored to the flow around of pitching airfoil, with period T. The objective function to be maximized is defined as

$$F = \frac{1}{T}\int_0^T \int_{S_w} p n_k r_k dS dt \tag{11}$$

The derivative of F is computed by differentiating the augmented objective function $F_{aug} = F + \int_0^T \int_\Omega \Psi_n R_n d\Omega dt$, where Ψ_n, $n = 1,4$ are the unsteady adjoint variable fields. The resulting unsteady adjoint equations are

$$-\frac{\partial \Psi_m}{\partial t} - A_{nmk}\frac{\partial \Psi_n}{\partial x_k} = 0 \tag{12}$$

The imposed boundary conditions along the moving solid wall are $\Psi_{k+1}n_k+\Psi_4 u_{n,w}+\frac{1}{T}n_k r_k=0$ and, at the farfield, $\Psi_n=0,\ n=1,4$. Then, the gradient of F w.r.t. the design variables b_q is

$$\begin{aligned}\frac{\delta F}{\delta b_q} &= \frac{1}{T}\int_0^T\int_{S_w} p\frac{\delta(n_k r_k dS)}{\delta b_q}dt+\int_0^T\int_{S_w}\left(\Psi_{k+1}p-\Psi_i f_{ik}\right)\frac{\delta n_k}{\delta b_q}dSdt \\ &- \int_0^T\int_{S_w}\Psi_i\frac{\partial f_{ik}}{\partial x_l}n_k\frac{\delta x_l}{\delta b_q}dSdt+\int_0^T\int_{S_w}\Psi_i\frac{\partial f_{il}}{\partial x_l}\frac{\delta x_k}{\delta b_q}n_k dSdt \\ &+ \int_0^T\int_{S_w}\Psi_i\frac{\partial U_i}{\partial x_l}\frac{\delta x_l}{\delta b_q}u_{n,w}dSdt+\int_0^T\int_{S_w}\left(\Psi_i U_i+p\Psi_4\right)\frac{\delta u_{n,w}}{\delta b_q}dSdt \end{aligned} \tag{13}$$

As described in section 4, the grid is dynamically adapted to the motion of the solid boundary. Due to the fact that the unsteady adjoint equations are solved backwards in time, each and every adapted grid must be stored. The quad-tree data structure allows for the minimum amount of data to be kept in memory, i.e. the pair of integers (i,j) (see section 2) and information needed to transfer values from one grid to the next one, in order to retrieve the proper grid at every time-step during the unsteady solution. Regarding the adjoint variables' projection to the grid at the next time-step, eqs. (6),(7) are used.

9 Unsteady Adjoint Results

An optimization loop is implemented for the pitching airfoil with amplitude 2.5 *deg*, at fixed infinite flow conditions of $M_\infty=0.44$ and $a_\infty=0$ *deg*. The optimization aims at maximizing the lift integral given by eq. (11). Starting from a symmetric airfoil, (fig. 12), where $F=0$ and after ten optimization cycles, the airfoil became cambered and the objective function increased to almost 0.32 (figs. 16, 17).

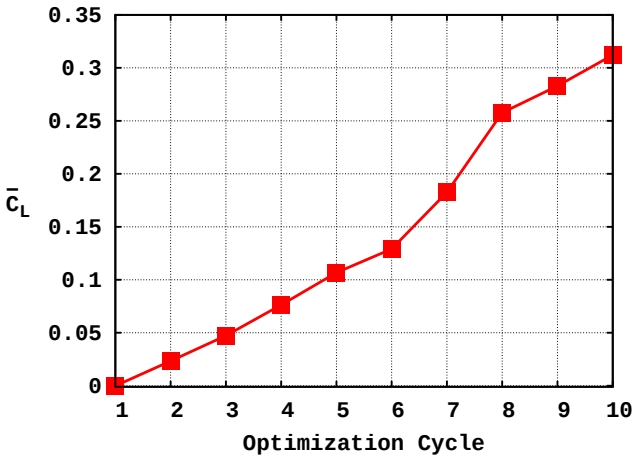


Figure 16: Unsteady subsonic flow around the pitching airfoil parameterized as in fig. 12. Adjoint-based maximization of the time-averaged lift. Variation in the objective function in terms of the optimization cycles.

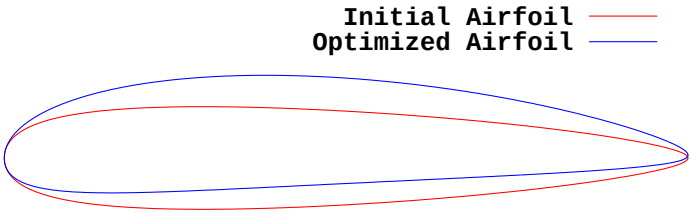


Figure 17: Unsteady subsonic flow around the pitching airfoil parameterized as in fig. 12. Adjoint-based maximization of the time-averaged lift. Initial and optimized airfoils.

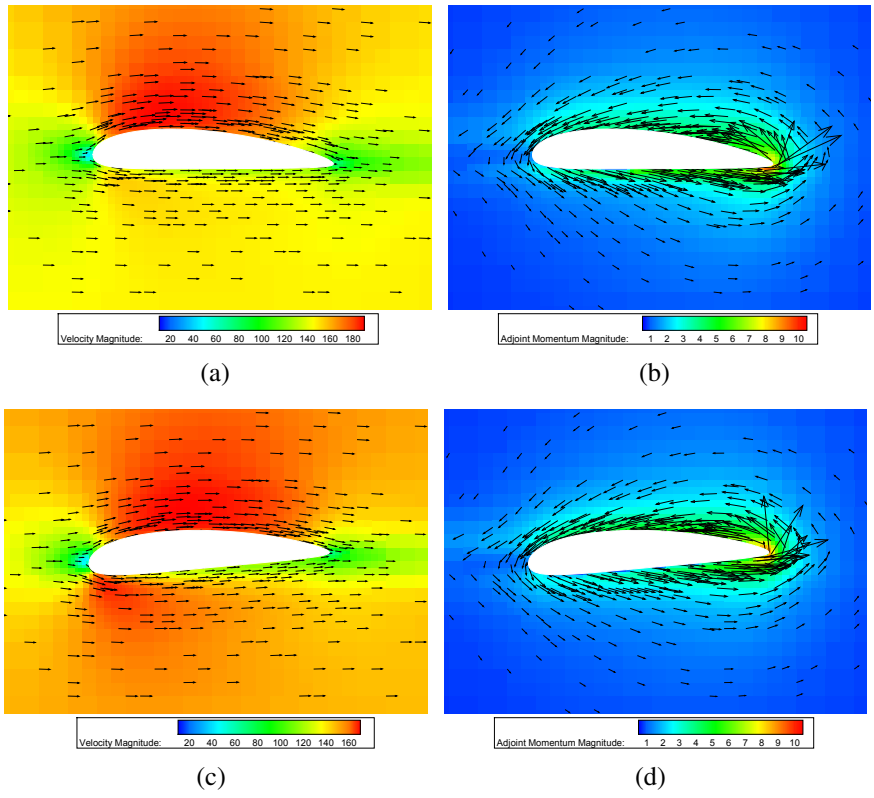


Figure 18: Unsteady subsonic flow around the pitching airfoil parameterized as in fig. 12. Adjoint-based maximization of the time-averaged lift. Flow and adjoint momentum magnitude fields at the two extreme positions of the optimized airfoil's oscillation. Vectors represent the velocity and adjoint momentum for flow and adjoint fields, respectively.

10 Conclusions

The cut-cell method for inviscid flows of compressible fluids and the corresponding continuous adjoint method were developed and programmed on GPUs. These were used in steady and unsteady analysis and optimization problems. Both steady and unsteady analysis results were validated against measurements and CFD results obtained using body-fitted grids. A technique for merging very small cut-cells is employed to avoid numerical instabilities. The Cartesian grid is adapted to the stationary solid wall boundaries and the emerging flow discontinuities. This conforms to the quad-tree data structure, ensuring a significant reduction in memory requirements. The proposed scheme for transferring the flow and adjoint variables between successive time steps, in case refinement or coarsening locally applies, proved to be accurate and capable of supporting unsteady problems. The developed software was used to solve steady and unsteady lift maximization problems in external aerodynamics but this could be extended to other

objective functions with minor modifications.

REFERENCES

[1] C.S. Peskin. Flow patterns around heart valves: a numerical method. *J. Comp. Phys.*, **10** (2): 252-271, 1972.

[2] R. Mittal, G. Iaccarino. Immersed boundary methods. *Annu. Rev. Fluid Mech.*, **37** (1): 239-261, 2005.

[3] D.K. Clarke, H. Hassan, M. Salas. Euler calculations for multielement airfoils using Cartesian grids. *AIAA J.*, **24** (3): 353-358, 1986.

[4] R. Gaffney Jr., H.A. Hassan. Euler calculations for wings using Cartesian grids. *25th AIAA Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics*, Reston, Virginia, 1987.

[5] H. Ji, F. Lien, E. Yee. A new adaptive mesh refinement data structure with an application to detonation. *J. Comp. Phys.*, **229**: 8981-8993, 2010.

[6] D.I. Papadimitriou, K.C. Giannakoglou. Aerodynamic shape optimization using first and second order adjoint and direct approaches. *Arch. Comput. Methods Eng.*, **15**: 447-488, 2008.

[7] J. Hee Seo, R. Mittal. A sharp-interface immersed boundary method with improved mass conservation and reduced spurious pressure oscillations. *J. Comp. Phys.*, **230** (19): 7347-7363, 2011.

[8] H. Ji, F. Lien, E. Yee. Numerical simulation of detonation using an adaptive Cartesian cut-cell method combined with a cell-merging technique. *Computers & Fluids*, **39**: 10411057, 2010.

[9] P. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *J. Comp. Phys.*, **43** (2): 357-372, 1981.

[10] O. Reynolds. Papers on mechanical and physical subjects. *Cambridge University Press*, Vol. 3, 1903.

[11] J. Vassberg, A. Jameson. In pursuit of grid convergence for two-dimensional Euler solutions. *AIAA J. Aircraft*, **47** (4): 1152-1166, 2010.

[12] R. H. Landon. NACA 0012 oscillatory and transient pitching. *AGARD Report 702*, Dataset 3, January 1982.

[13] I.C. Kampolis, X.S. Trompoukis, V.G. Asouti, K.C. Giannakoglou. CFD-based analysis and two-level aerodynamic optimization on graphics processing units. *Comput. Methods Appl. Mech. Engrg.*, **199**: 712-722, 2010.