Implicit Euler computations for steady and unsteady transonic flows

C. Chalons
Lehr- und Forschungsgebiet fur Mechanik,
RWTH Aachen, 52056 Aachen, Germany
chalons@lufmech.rwth-aachen.de

Abstract

In this note, we address the issue of the numerical simulation of unsteady flows involving inviscid compressible fluids. In this context, we propose a new hybrid scheme for solving at each time iteration the non linear systems coming from implicit discretizations. Our scheme is built on the basis of heuristic arguments from the so-called Newton's method and dual time step approach, and tested against a fluid structure interaction of engineering interest.

1 Introduction

This note aims at summarizing a study which has been done during the four months stay[1] of the author in Aachen, more precisely in the RWTH laboratory of mechanics named Lufmech and headed by Prof. Ballmann. During this period, the author also defended its PhD thesis at the Ecole Polytechnique in France, but here the topic is actually different.

This work fits in the general framework of the so-called Quadflow program, which is being developed in the collaborative research center SFB401 (RWTH Aachen). Quadflow solves the Euler as well as the Navier-Stokes equations around complex aerodynamic configurations when using unstructured meshes. It aims at dealing with unsteady compressible fluid flows and structure interactions, and uses grid adaptation (see [1], [2], [3]). The later feature is performed by means of a multiresolution analysis using wavelets and threshold techniques (see [4], [5], [6], [7], [8], [9], [10]).

To go further in details, this study is devoted to the time integration of the governing equations of an inviscid fluid flow during the unsteady process of a fluid/structure interaction. The bodies under consideration are always assumed to evolve according to a rigid motion, modelled by introducing a rigidly moving frame of reference. In order to allow the use of time steps greater than the ones given by the classical explicit CFL restriction, the temporal evolution is treated when using implicit schemes. We propose to deal with several implicit strategies when studying their behaviours and stability properties against test cases of engineering interest. The underlying accuracy of such time discretisations plays an important role in these problems. Indeed, since we consider unsteady computations, it is necessary to propose at least second order schemes in time. Several techniques in the literature provide us with such schemes, but their associated cost (namely evaluation, computing resources, ...) sometimes actually differs. Here we would like to get a stable one step method, that is to say a stable method involving only two times steps (namely t^n and t^{n+1}), which obeys in

[1]From October 2002 till January 2003

addition the second order accuracy property. While the second expected property has already been motivated above, the first one should be of interest and very pleasant in order to save memory and simplicity in the code. This is more particularly true in the context of mesh adaptation and multiresolution analysis. The main idea of this paper is to propose a modified Cranck-Nicholson implicit time integration we find out to be more stable than the original one, while still ensuring the second order accuracy property. Some heuristic arguments will be given in order to understand the behaviours put forward along this study.

The paper is organized as follows. The first part presents the fluid governing equations considered in this report. Then, we do some reminders on implicit time integration techniques in the context of unsteady problems, while describing with details two strategies for solving the non linear systems coming from implicit discretizations : the *Newton's method* and the *dual time step technique*. Equipped with these procedures, we then describe a hybrid implicit technique for updating the solution at each time step. The basic motivation of such a hybrid scheme is seen to stem from heuristic arguments for ensuring the stability and second order accuracy properties. The last section proposes some numerical experiments on the NACA0012 Airfoil in order to prove the relevance of our hybrid scheme.

2 Governing equations

In this paper, we are interested in simulating unsteady fluid flows around bodies in motion. With this in mind, we introduce a frame of reference rigidly connected to the body and then moving according to its motion. In this frame of reference, we consider the two-dimensional Euler equations. In two space dimensions, they read :

$$\partial_t \mathbf{U} + \partial_x \mathbf{F}(\mathbf{U}) + \partial_y \mathbf{G}(\mathbf{U}) = 0, \tag{1}$$

with

$$\mathbf{U} := (\rho, \rho u, \rho v, \rho E),$$

and

$$\mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho U \\ \rho u U + p \\ \rho v U \\ U(\rho E + p) + x_t p \end{pmatrix}, \quad \mathbf{G}(\mathbf{U}) = \begin{pmatrix} \rho V \\ \rho u V \\ \rho v V + p \\ V(\rho E + p) + y_t p \end{pmatrix}.$$

Here, ρ, u, v and E classicaly denote the density, the cartesian components of the velocity and the total energy. The quantities U and V represent the contravariant velocities defined by

$$U = u - x_t, \quad V = v - y_t,$$

where x_t and y_t are the grid speed components. Concerning the required closure relation on the pressure law p, we consider a thermally and calorically ideal gas, so that :

$$\rho E = \frac{1}{2}\rho u^2 + \frac{p}{\gamma - 1}.$$

The adiabatic exponent γ is fixed to 1.4.

3 Implicit time integration for unsteady problems

In this section, we describe two methods often used in the literature when dealing with implicit time integration methods applied to unsteady problems. More precisely, we consider the so-called *dual time step method* and the most usual *Newton's method.* These methods take place in the computation of the uptaded value of the unknowns vector $\mathbf{U}$ at time t^{n+1}, assuming as given this vector at time t^n.

Usually, the partial differential equations (1) are first discretized in space, so that the methods we consider here aim at solving a set of equations which can be written in the following form :

$$\partial_t \mathbf{U}_K + \frac{1}{|K|}\mathbf{R}_K(\mathbf{U}) = 0, \quad K \in \mathcal{K}.$$

Here K represent any computational cell arising in the space domain discretization $\mathcal{K}$, $|K|$ its volume, while $\mathbf{R}_K$ contains all the terms arising from the spatial discretization of $\partial_x \mathbf{F}(\mathbf{U}) + \partial_y \mathbf{G}(\mathbf{U})$. In the forthcoming developments, we will more particularly rely on the Hanel/Schwane Flux Vector Splitting method described in [11]. Observe that the associated numerical flux have the pleasant property of being differentiable, which will be of interest in the next two subsections for the sake of rigour.

Then, the time discretization issue is addressed. The matter is to propose a discrete version of $\partial_t \mathbf{U}_K$ within the interval $[t^n, t^{n+1} := t^n + \Delta t[$, together with an evaluation of $\mathbf{U}$ in $\mathbf{R}_K(\mathbf{U})$ at a relevant time $t \in [t^n, t^{n+1}[$. Several such discretizations are very popular and often used in the applications. Let us quote for instance :

(i) the explicit first order Backward Euler method, which writes

$$\mathbf{R}_K^{\star,eE}(\mathbf{U}^{n+1}) := \frac{\mathbf{U}_K^{n+1} - \mathbf{U}_K^n}{\Delta t} + \frac{1}{|K|}\mathbf{R}_K(\mathbf{U}^n) = 0, \tag{2}$$

(ii) the implicit first order Backward Euler method, which reads

$$\mathbf{R}_K^{\star,iE}(\mathbf{U}^{n+1}) := \frac{\mathbf{U}_K^{n+1} - \mathbf{U}_K^n}{\Delta t} + \frac{1}{|K|}\mathbf{R}_K(\mathbf{U}^{n+1}) = 0, \tag{3}$$

(iii) the implicit Cranck-Nicholson scheme :

$$\mathbf{R}_K^{\star,\alpha}(\mathbf{U}^{n+1}) := \frac{\mathbf{U}_K^{n+1} - \mathbf{U}_K^n}{\Delta t} + \frac{1}{|K|}(\alpha\mathbf{R}_K(\mathbf{U}^n) + (1-\alpha)\mathbf{R}_K(\mathbf{U}^{n+1})) = 0, \tag{4}$$

with $\alpha \in [0, 1[$,

(iv) an implicit second-order time integration based on three time levels :

$$\mathbf{R}_K^{\star,itl}(\mathbf{U}^{n+1}) := \frac{3\mathbf{U}_K^{n+1} - 4\mathbf{U}_K^n + \mathbf{U}_K^{n-1}}{2\Delta t} + \frac{1}{|K|}\mathbf{R}_K(\mathbf{U}^{n+1}) = 0. \tag{5}$$

Let us recall that the implicit Cranck-Nicholson approach is second order time accurate if and only if we set $\alpha = 1/2$, while the implicit Backward Euler method is recovered when simply choosing $\alpha = 0$. The well-known Runge-Kutta type approximations will not be considered in this paper. Anyway, whatever the scheme we choose, we generically end up with the following system to be solved :

$$\mathbf{R}^{\star}_{K}(\mathbf{U}^{n+1}) = 0, \quad K \in \mathcal{K}, \tag{6}$$

where $\mathbf{R}^{\star}_{K}$ includes both time and space discretizations under consideration. Of course and by definition, such a system is trivially solved when considering an explicit time integration (see (2) for instance).
Let us now address the issue of the resolution of (6). In that purpose, the next two subsections briefly report the working of the so-called *Newton's* and *dual time step* methods.

3.1 The Newton's method

The Newton's method is a well-known iterative algorithm for solving systems of nonlinear algebraic equations of the form (6). Newton's method is based on a first order Taylor expansion, which reads :

$$\mathbf{R}^{\star}_{K}(\mathbf{U}^{(l+1)}) = \mathbf{R}^{\star}_{K}(\mathbf{U}^{(l)}) + J^{\star}(\mathbf{U}^{(l)})\Delta\mathbf{U}^{(l+1)}, \quad K \in \mathcal{K},$$

where with classical notations, $J^{\star}$ denotes the Jacobian matrix associated with $\mathbf{R}^{\star} = (\mathbf{R}^{\star}_{K})_{K\in\mathcal{K}}$ and $\Delta\mathbf{U}^{(l+1)} = \mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}$. Observe that each element of the Jacobian matrix $J^{\star}$ needs to be well-defined when evaluated at $\mathbf{U}^{(l)}$ in order to consider such an expansion. In that aim, remember that the numerical flux under consideration (associated with the space discretization of $\partial_x\mathbf{F}(\mathbf{U}) + \partial_y\mathbf{G}(\mathbf{U})$) is actually differentiable. Thus, time discretizations like (2), (3), (4) or (5) immediately allow the definition of $J^{\star}$.
Then, since we want $\mathbf{R}^{\star}$ to vanish, we are led to define $\mathbf{U}^{l+1}$ as a linear function of $\mathbf{U}^{l}$ according to :

$$0 = \mathbf{R}^{\star}_{K}(\mathbf{U}^{(l)}) + J^{\star}(\mathbf{U}^{(l)})\Delta\mathbf{U}^{(l+1)}, \quad K \in \mathcal{K},$$

that is to say :

$$J^{\star}(\mathbf{U}^{(l)})\Delta\mathbf{U}^{(l+1)} = -\mathbf{R}^{\star}_{K}(\mathbf{U}^{(l)}), \quad K \in \mathcal{K}. \tag{7}$$

As a consequence, each iteration in the Newton's algorithm amounts in solving a linear problem of the form $A^{Nm}X = B$ (here the superscript Nm means Newton's method). The right-hand side $B = -\mathbf{R}^{\star}_{K}(\mathbf{U}^{(l)})$ is called *explicit residual part* while the matrix A^{Nm} in the left-hand side is named *implicit matrix.* For the sake of completeness, we often consider $\mathbf{U}^{(0)} = \mathbf{U}^{n}$ as initial condition. This algorithm is proved to converge rapidly to the expected value if the initial guess is precise enough. Unfortunately, such a relevant guess is possibly "far" from $\mathbf{U}^{n}$, especially when using large time steps Δt, and as a consequence the method may fail. Nevertheless, this method is very popular.
To conclude this subsection, let us precise the matrices A^{Nm} together with the right-hand sides B involved in the linear system to be solved at each iteration, when considering the above four time integrations. With that aim, we use the notation J for denoting the Jacobian matrix of the space discretization

$\mathbf{R} = (\mathbf{R}_K)_{K\in\mathcal{K}}$. Thus, easy calculations lead to the following results :
(i) for the explicit Backward Euler method, we immediately get

$$A^{Nm} = \frac{1}{\Delta t} I, \quad B = -\mathbf{R}_K^{\star,eE}(\mathbf{U}^{(l)}), \tag{8}$$

where I represents the identity matrix,
(ii) the implicit first order Backward Euler method gives

$$A^{Nm} = J(\mathbf{U}^{(l)}) + \frac{1}{\Delta t} I, \quad B = -\mathbf{R}_K^{\star,iE}(\mathbf{U}^{(l)}), \tag{9}$$

(iii) the Cranck-Nicholson implicit scheme is associated with :

$$A^{Nm} = (1-\alpha) J(\mathbf{U}^{(l)}) + \frac{1}{\Delta t} I, \quad B = -\mathbf{R}_K^{\star,\alpha}(\mathbf{U}^{(l)}), \tag{10}$$

for all $\alpha \in [0, 1[$, and
(iv) the three time levels based second order implicitation corresponds to :

$$A^{Nm} = J(\mathbf{U}^{(l)}) + \frac{3}{2\Delta t} I, \quad B = -\mathbf{R}_K^{\star,itl}(\mathbf{U}^{(l)}). \tag{11}$$

3.2 The dual time step method

The dual time step method was first suggested by Jameson [12]. The main idea of this strategy is to consider the solution $\mathbf{U}^{n+1}$ of (6) as a steady state of a related problem evolving with respect to a fictitious time τ, the so-called dual time. More precisely, the associated problem writes as follows :

$$\partial_\tau \mathbf{U}_K + \mathbf{R}_K^\star(\mathbf{U}) = 0, \quad K \in \mathcal{K}. \tag{12}$$

In order to reach the steady state of such a problem, we may discretize the time partial derivative with any of the methods available in the literature, from explicit to implicit ones. The basic interest of such a procedure stems from the fact that all the acceleration techniques developed for steady problems may be used during the marching process in dual time. Let us quote for instance the local time stepping, the residual smoothing, the multigrid approach or the linearized implicit time integrations.
In this study, we propose a linearized implicit time-marching approach, in order to avoid too restrictive CFL restrictions on the dual time step. Since we are only interested in the steady regime of (12), only first order accuracy in fictitious time τ is necessary. We are thus led to consider the Backward Euler time discretization on the pseudo-time τ :

$$\frac{\mathbf{U}_K^{(l+1)} - \mathbf{U}_K^{(l)}}{\Delta\tau} + \mathbf{R}_K^\star(\mathbf{U}^{(l+1)}) = 0, \quad K \in \mathcal{K}, \tag{13}$$

where with clear notations, the superscript $(l+1)$ corresponds to the time level $(l+1)\Delta\tau$. We then propose to accelerate such a fully implicit method when linearizing the non linear terms, thanks to a first order Taylor expansion. This widely used technique boils down to deal with :

$$\frac{\mathbf{U}_K^{(l+1)} - \mathbf{U}_K^{(l)}}{\Delta\tau} + \mathbf{R}_K^\star(\mathbf{U}^{(l)}) + J^\star(\mathbf{U}^{(l)})(\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}) = 0, \quad K \in \mathcal{K},$$

which equivalently recast in the following convenient incremental form :

$$(J^\star(\mathbf{U}^{(l)}) + \frac{1}{\Delta\tau})\Delta\mathbf{U}^{(l+1)} = -\mathbf{R}^\star_K(\mathbf{U}^{(l)}), \quad K \in \mathcal{K}, \tag{14}$$

with as a natural initialization $\mathbf{U}^{(0)} = \mathbf{U}^n$. Here we have used the same definitions as in the previous subsection for $\Delta\mathbf{U}^{(l+1)}$ and $J^\star$. This acceleration technique just consists in performing, at each evolution in dual time, only the first iteration of the Newton's algorithm applied to (13) (see previous subsection for details). Therefore, it is not rigorously consistant with respect to the fictitious time τ but here it is not of matter. The very benefit clearly comes from the fact that only a linear system of the form $A^{dts}X = B$ has to be solved at each iteration. Here the superscript *dts* means dual time step. Again, A^{dts} and B are usually called *implicit matrix* and *explicit residual part* respectively. It is worthy to noticing that due to (14), the matrices A^{dts} involved at each iteration are related to the companion ones A^{Nm} (associated with the Newton's method) by

$$A^{dts} = A^{Nm} + \frac{1}{\Delta\tau}I,$$

whatever the space discretization we are considering. $B = -\mathbf{R}^\star_K(\mathbf{U}^{(l)})$ is still associated with the discretization operator under consideration. As an immediate consequence, the only difference between the two algorithms lies in the extra term $\frac{1}{\Delta\tau}$ on the diagonal of the implicit matrix. This simple remark will be of interest in the forthcoming developments.

3.3 About the convergence criterion

Both the Newton's and dual time step approaches are iterative algorithms for obtaining the solution $\mathbf{U}^{n+1}$ of (6). Thanks to the incremental forms (7) and (14), it is clear that the convergence of such algorithms as l tends to infinity leads to :

$$R^\star_K(\lim_{l\to\infty} \mathbf{U}^{(l)}) = 0, \quad K \in \mathcal{K}.$$

As a consequence, the limit coincides with the desired value :

$$\mathbf{U}^{n+1} = \lim_{l\to\infty} \mathbf{U}^{(l)}.$$

In order to measure how far we are from this limit, we introduce a parameter ϵ. From a numerical point of view, algorithms under study will be considered as converged if and only if :

$$||R^\star_K(\mathbf{U}^{(l)})|| \leq \epsilon, \quad K \in \mathcal{K}. \tag{15}$$

Of course, $\epsilon = 0$ is equivalent to an exact convergence.

4 A hybrid formulation

We begin this section by briefly giving some heuristic remarks on the methods described in the previous section, concerning their behaviour and stability properties and including a short discussion on the implicit time integration to be chosen. In order to reach a better stability (always in a heuristic way), we will be drived to consider hybrid methods involving two different time integrations in both the explicit residual part and the implicit matrix. In the next section, these hybrid strategies are compared with usual ones against a test case of engineering interest.

Let us first deal with the Newton's approach. Implicit techniques are well-known to be more stable than explicit ones, since they allow the use of larger time steps (*i.e.* larger CFL numbers) at each time iteration. When having a look on the matrices (8) and (9), respectively associated with the explicit and implicit Backward Euler methods, we then formally conclude that adding the Jacobian $J(\mathbf{U}^{(l)})$ in the implicit left-hand side seems to stabilize the corresponding iterative algorithm. Following this point of view, the implicit matrix given in (4) does not say anything else that the Cranck-Nicholson time integration is probably less stable than the implicit Backward Euler one when $0 < \alpha < 1$. We are thus led to consider a first hybrid algorithm, obtained on the basis of the Cranck-Nicholson time integration but when considering α as a free parameter in the left-hand side matrix of the linear system to be solved at each iteration of the Newton's algorithm. More precisely, we are tempted to consider $\alpha \in [0,1[$ "far" from 1 and possibly equal to 0. In addition, we would like the limit solution $\lim_{l\to+\infty} \mathbf{U}^{(l)}$ to obey the expected second order accuracy property in time. From now on, we then propose to fix α to $1/2$ inside the residual part $\mathbf{R}_K^{\star,\alpha}(\mathbf{U}^{(l)})$ associated with the Cranck-Nicholson time integration. The first hybrid algorithm (FHA) then reads :

$$\text{(FHA)} \quad \begin{cases} ((1-\alpha)J(\mathbf{U}^{(l)}) + \frac{1}{\Delta t}I)(\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}) = -\mathbf{R}_K^{\star,1/2}(\mathbf{U}^{(l)}), \quad K \in \mathcal{K}, \\ \\ \mathbf{U}^{(0)} = \mathbf{U}^n. \end{cases}$$

Let us now turn considering the dual time step method. First, let us recall that when compared to the Newton's method, this strategy just consists in adding the dual term $1/\Delta\tau$ in the diagonal of the implicit matrix. Put in other words, the physical time step Δt in the left-hand side implicit matrix is modified by $\overline{\Delta t}$ such that :

$$\frac{1}{\overline{\Delta t}} = \frac{1}{\Delta t} + \frac{1}{\Delta\tau},$$

and clearly becomes :

$$\overline{\Delta t} = \frac{\Delta t \Delta\tau}{\Delta t + \Delta\tau} < \Delta t. \tag{16}$$

That is a convenient way in order to understand the dual time step method. In addition, observe that the Newton's method is formally recovered when choosing $\Delta\tau = +\infty$. From a stability point of view, adding a non negative term in the implicit diagonal implies a better conditionning of the corresponding matrix. In that sense, the dual time step approach seems to be more stable than the Newton's one, of course whatever the time and space discretizations under consideration. Thus, the second hybrid algorithm (SHA) we propose is based on the first one (FHA) and aims at ensuring a better conditionning of the involved implicit matrix, when introducing a dual term in the diagonal. We eventually obtain :

$$\begin{cases} ((1-\alpha)J(\mathbf{U}^{(l)}) + (\frac{1}{\Delta t} + \frac{1}{\Delta\tau})I)(\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}) = -\mathbf{R}_K^{\star,1/2}(\mathbf{U}^{(l)}), \quad K \in \mathcal{K}, \\ \\ \mathbf{U}^{(0)} = \mathbf{U}^n, \end{cases}$$

which can be recast in the following condensed form :

$$\text{(SHA)}\quad \begin{cases} ((1-\alpha)J(\mathbf{U}^{(l)}) + \frac{\beta}{\Delta t}I)(\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}) = -\mathbf{R}_K^{\star,1/2}(\mathbf{U}^{(l)}), \quad K \in \mathcal{K}, \\ \mathbf{U}^{(0)} = \mathbf{U}^n, \end{cases}$$

with of course $\beta \geq 1$ (see (16)). Observe that Δt being fixed, the mapping $\Delta\tau \to \beta := \beta(\Delta\tau)$ is one-to-one :

$$\beta = \frac{\Delta t + \Delta\tau}{\Delta\tau}.$$

To conclude, it is easy to check that the first and the second hybrid algorithms are identical when β is chosen to be equal to 1.

5 Numerical experiments for NACA0012 Airfoil

We follow by numerically testing our hybrid procedure past a test case of interest in applications, namely a fluid flow around NACA0012 Airfoil. We aim at comparing the stability of the hybrid implicit matrix derived in (SHA) with the one of the matrix associated with the exact Cranck-Nicholson approach : namely when considering $\alpha = 1/2$ and $\beta = 1$. Both steady and unsteady flows are considered. In a first approach, we fix $\beta = 1$ so that only the influence of the parameter α in (SHA) is studied.

5.1 The steady case

In order to apply our technique to stationary computations, replacing the unsteady residual $\mathbf{R}_K^{\star,1/2}$ in (SHA) with the steady one $\mathbf{R}_K$ is clearly sufficient. We then obtain the following iterative scheme :

$$\begin{cases} ((1-\alpha)J(\mathbf{U}^{(l)}) + \frac{1}{\Delta t}I)(\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}) = -\mathbf{R}_K(\mathbf{U}^{(l)}), \quad K \in \mathcal{K}, \\ \mathbf{U}^{(0)} = \mathbf{U}^n, \end{cases} \tag{17}$$

which simply consists in applying a time marching strategy to the Cranck-Nicholson time discretization. Of course, dealing with $\alpha \neq 0$ (and more particularly with $\alpha = 1/2$) in (17) is not fully adapted for steady flows since time accuracy is not necessary during the time marching process. As a consequence, such a simulation only aims at giving encouraging results before using our strategy with unsteady flows.

With clear notations, the far field is characterized by

$$\rho_\infty = 1.28255, \quad v_\infty = 0, \quad T_\infty = 285, \quad \text{and} \quad M_\infty = 0.755. \tag{18}$$

Figure 1 presents the grid under consideration. It is composed of 9600 cells, 2400 in each of the four blocks.

In order to get the steady state associated with this problem, we propose to use the following extreme values for α : 0 and 1/2. Notice that $\alpha = 0$ is nothing else but the usual first order Backward Euler implicit matrix (see indeed (9)) while $\alpha = 1/2$ corresponds to the exact second order Cranck-Nicholson

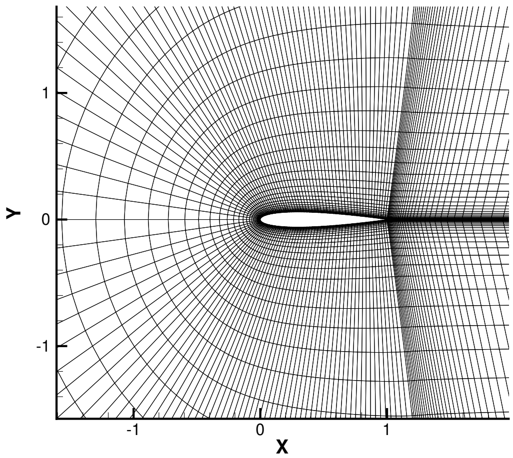

Figure 1: Grid with 9600 cells

matrix. Our numerical attempts clearly drived to the conclusion that the exact Cranck-Nicholson matrix is actually not stable, since leading to a crash even with small CFL numbers (say larger than 10). On the contrary, the implicit Backward time discretization turns out to be very stable. Indeed, it allowed us to reach the expected steady state in about 90 iterations (when the convergence criterion equals 10^{-5}), when using CFL numbers close to 10^5. Here the CFL evolution strategy consists in a multiplication by a factor 1.2. Figure 2 shows the corresponding Mach number component.

To conclude, this test case provides us with the expected encouraging conclusion, namely the implicit matrix in (SHA) is more stable when α is chosen close to 0. Put in other words, the exact Cranck-Nicholson implicit matrix seems unstable, at least for steady computations. The next section aims at extending these conclusions to unsteady flows, when dealing with a right-hand side associated with a second-order Cranck-Nicholson residual in (17).

5.2 The oscillating unsteady case

In this section, we investigate the unsteady transonic flow past a pitching NACA0012 airfoil when the Mach number still equals 0.755 (see also (18)). We assume that the airfoil rotates about the quarter chord axis according to the following law for the angle of attack :

$$\alpha(t) = 0.016^\circ + 2.51^\circ \sin(\omega t + \phi).$$

Here the reduced frequency is $k = \omega c / u_\infty = 0.1628$ where c represents the airfoil chord and u_∞ the free stream velocity. We introduce a phase shift $\alpha = 90^\circ$ so that at time $t = 0$, easy calculations show that the motion of the airfoil is accelerated but has no rotational velocity.

Remind that iterative algorithm defined by (SHA) has to converged at each time iteration for obtaining the unsteady field, implicitly updated, at time t^{n+1}.

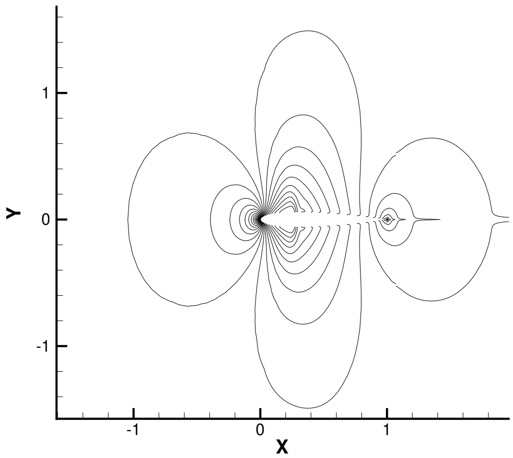

Figure 2: Steady state : Mach component

The physical time step is chosen such that 100 time steps per cycle are resolved. Figures 3-4 and 5-6 present the hysteresis curves of the lift c_L and moment c_M coefficients for the fourth cycle of oscillation. For the sake of comparison, we plot these curves for two second-order implicit time discretizations : the Cranck-Nicholson approach (see (4)) and the one based on three time levels (see (5)). Here the convergence threshold ϵ (see above (15)) is such that the unsteady residual is reduced of four orders of magnitude within each time step. It turns out that best rates of convergence do not improve the quality of the solution. In addition, Figure 7 proves that this value is sufficient to avoid some noises in the solution. Indeed, we observe that when the rate of convergence equals 10^{-3}, the moment coefficient no longer exhibits a smooth behaviour. For these computations, we have still used the mesh proposed in Figure 1.

Figure 8 shows (again for the fourth cycle) the number of iterations needed in the iterative algorithm (SHA) at each time iteration. Several values of α are considered. We note that this algorithm is cheaper when α approaches $1/2$, which is clearly expected.

We now propose to perform the same computation when using mesh adaptation techniques (see [2], [3]) available within Quadflow. With this in mind, we consider several values of the threshold ϵ for adaptation, while the maximum mesh refinement level within mesh is fixed to 4. Figures 9 and 10 show the mesh and the C_p coefficient, defined as follows :

$$C_p = \frac{p - p_\infty}{\frac{\rho_\infty}{2} u_\infty^2},$$

respectively, when the threshold value coincides with 10^{-2}. In the same way, Figures 11 and 12 represent the mesh and the C_p coefficient, respectively, but when the threshold value is 10^{-4}.

In addition, Figure 13 is devoted to the moment coefficient for both the threshold value considered above. To conclude, Figure 14 compares the moment coeffi-

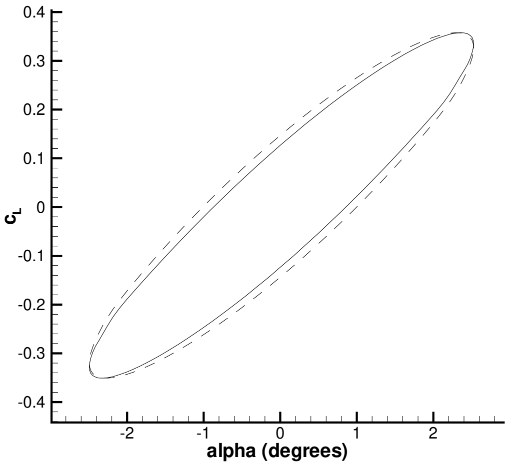

Figure 3: Lift coefficient. Solid : Cranck-Nicholson - Dashed : Three time levels

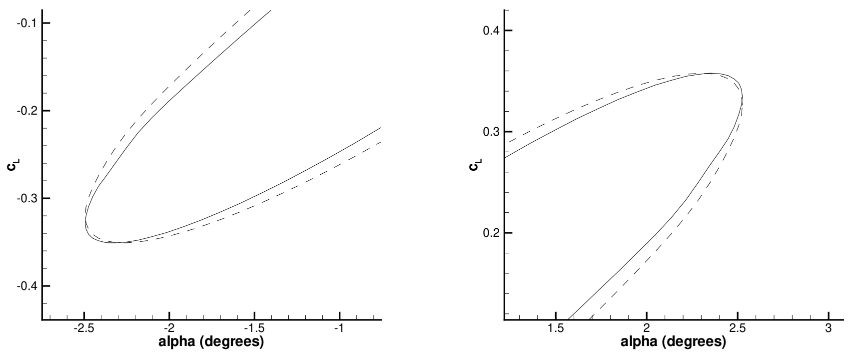

Figure 4: Close-up of the lift coefficient. Solid : Cranck-Nicholson - Dashed : Three time levels

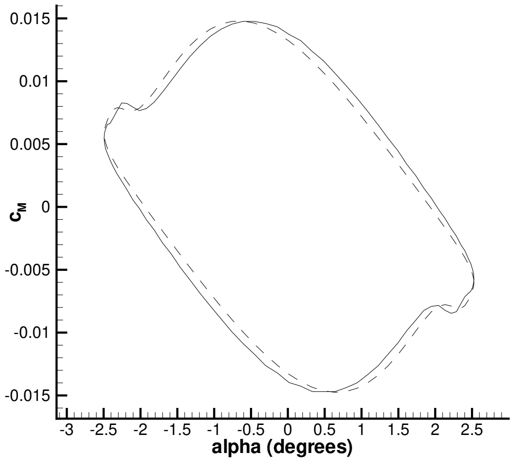

Figure 5: Moment coefficient. Solid : Cranck-Nicholson - Dashed : Three time levels

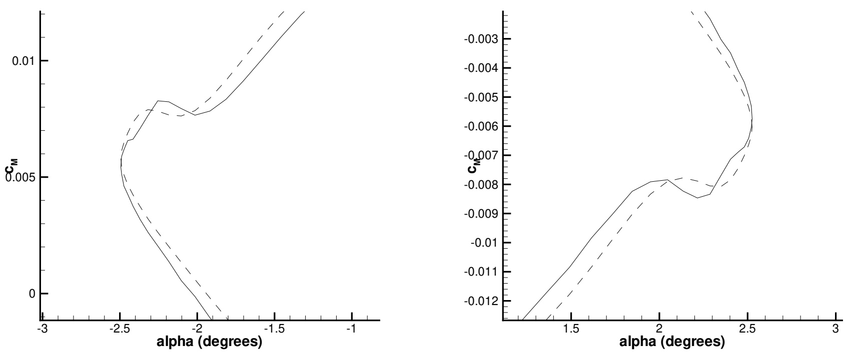

Figure 6: Close-up of the moment coefficient. Solid : Cranck-Nicholson - Dashed : Three time levels

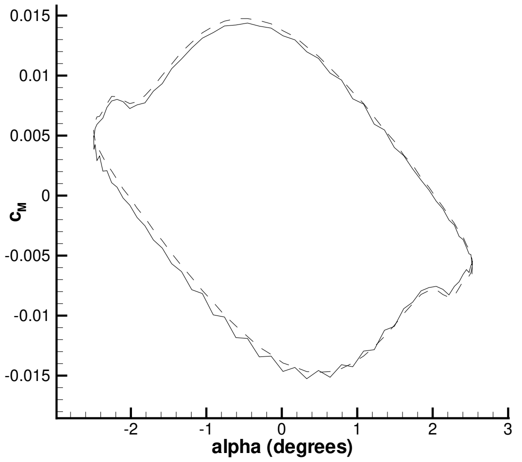

Figure 7: Moment coefficient. Solid : $\epsilon = 10^{-3}$ - Dashed : $\epsilon = 10^{-4}$

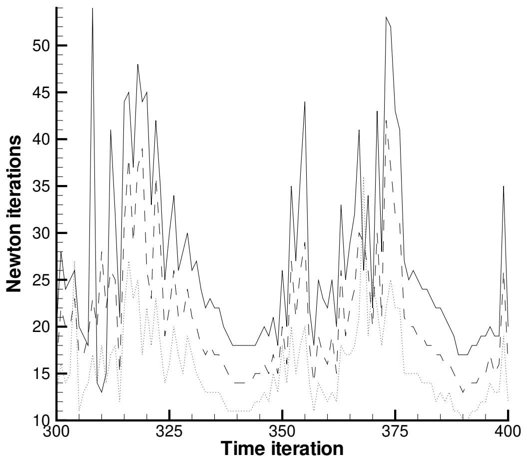

Figure 8: Number of Newton iterations needed per time iteration. Solid : $\alpha = 0$ - Dashed : $\alpha = 1/4$ - Dotted : $\alpha = 1/2$

cients, obtained with or without adaptation techniques (in the latter case, the threshold value is fixed to 10^{-4}). Of course, they do not exactly coincide since the mesh refinement level is not exactly the same.

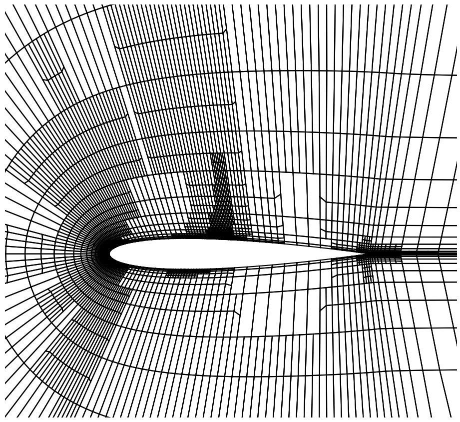

Figure 9: Typical adaptive grid when the threshold equals 10^{-2} (5465 cells)

It is worthy to noticing that for this test case, each value of α between 0 and 1/2 leads to a stable algorithm and allows to obtain the above curves. As a consequence, such a test case is not adapted for proving the good design of our hybrid scheme when compared with the exact Cranck-Nicholson one. Then, it would be of interest to propose a test case for which the Cranck-Nicholson time discretization fails when used with $\alpha = 1/2$ in the implicit matrix, while succeeds when α equals a smaller value (say 0 for instance). Turbulent simulations may provide such a test case. Then, the matter would be to choose a relevant value of α so that the number of iteration needed in the iterative algorithm at each time iteration stays reasonable. $\alpha = 1/4$ should be suitable.

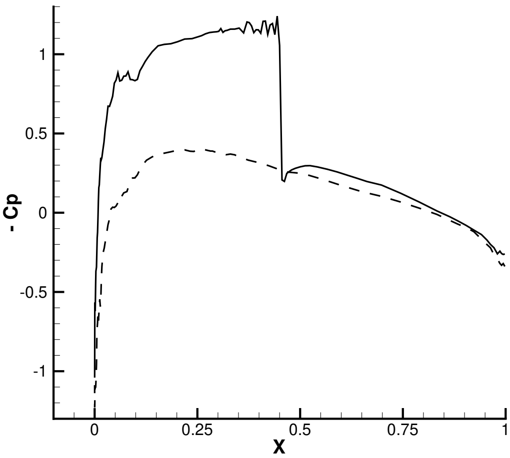

Figure 10: C_p coefficient with adaptation techniques and threshold 10^{-2}. Solid : upper surface of the airfoil - Dashed : lower surface of the airfoil

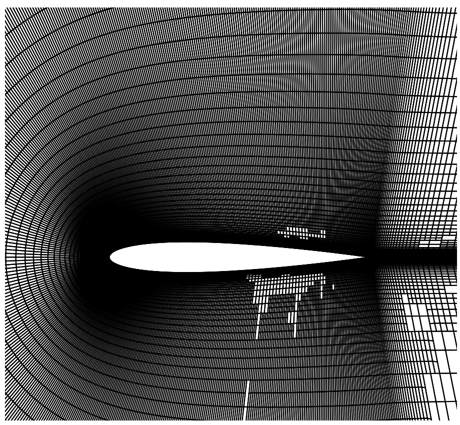

Figure 11: Typical adaptive grid when the threshold equals 10^{-4} (42401 cells)

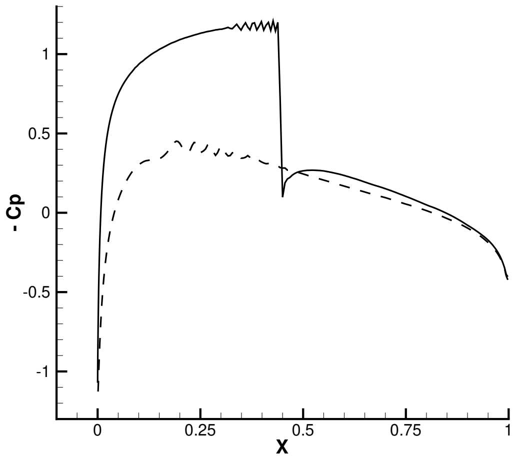

Figure 12: C_p coefficient with adaptation techniques and threshold 10^{-4}. Solid : upper surface of the airfoil - Dashed : lower surface of the airfoil

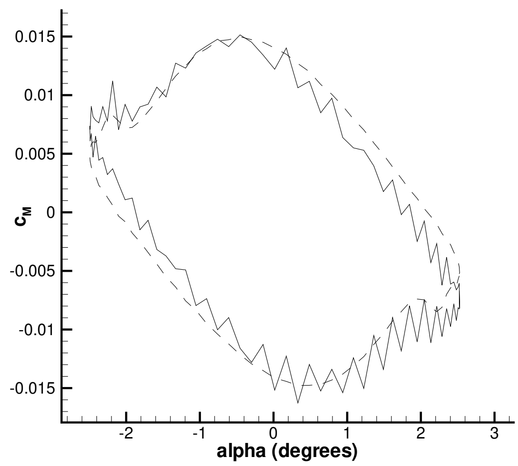

Figure 13: Moment coefficient with adaptation techniques. Solid : threshold value equals 10^{-2} - Dashed : threshold value equals 10^{-4}

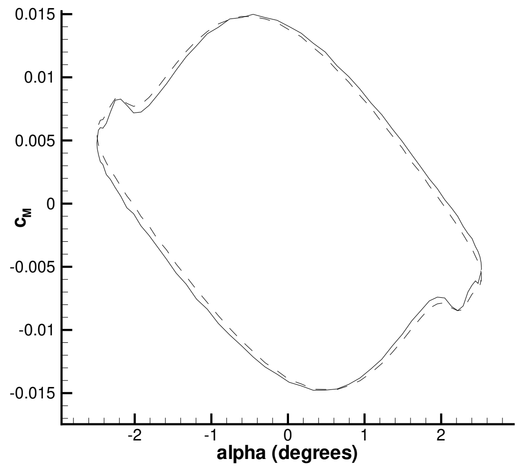

Figure 14: Moment coefficient. Solid : with adaptation techniques (threshold equals 10^{-4}) - Dashed : without adaptation techniques

Acknowledgments The author is very grateful to Prof. Ballmann for his invitation in Aachen and the corresponding financial support. The author also insists on thanking S. Andreae, J. Ballmann, F. Bramkamp, C. Brown, M. Hesse, W. Hoffmann, S. Wiertz and A. Zerouali for their very kind welcome, and F. Bramkamp for giving him an introduction to Quadflow.

References

[1] Bramkamp F. and Ballmann J., *Solution of the Euler Equations on Locally Adaptive B-Spline Grids*, NNFM, vol 77, pp 281-288, Springer (2002).

[2] Bramkamp F., Ballmann J. and Muller S., *Development of a Flow Solver Employing Local Adaptation Based on Multiscale Analysis on B-Spline Grids*, Proceedings of the 8^{th} Annual Conference of the CFD Society of Canada, Montreal, pp 113-118 (2002).

[3] Bramkamp F., Gottschlich-Muller B., Hesse M., Lamby P., Muller S., Ballmann J., Brakhage K.H. and Dahmen W., *H-adaptive Multiscale Schemes for the Compressible Navier-Stokes Equations. Polyhedral Discretization, Data Compression and Mesh Generation*, IGPM Report No. 207, RWTH Aachen (2001).

[4] Cohen A., Kaber S.M., Muller S. and Postel M., *Fully adaptive multiresolution schemes for conservation laws*, Mathematics of Computation, PII S 0025-5718(01)01391-6, in press (2002).

[5] Dahmen W., Gottschlich-Muller B. and Muller S., *Multiscale Concepts for Multidimensional Conservation Laws*, Numerische Math., vol 88 No. 3, pp 399-443 (2001).

[6] Dahmen W., Muller S. and Schlinkmann T., *Multigrid and Multiscale Decompositions*, Notes on Numerical Fluid Mechanics, vol 62, pp 18-41 (1998).

[7] Dahmen W., Muller S. and Schlinkmann T., *On an Adaptive Multigrid Solver for Convection-Dominated Problems*, SIAM Scientific Computing, vol 23 No. 3, pp 781-804 (2002).

[8] Gottschlich-Muller B. and Muller S., *Application of Multiscale Techniques to Hyperbolic Conservation Laws*, Computational Mathematics, Lecture notes in Pure and Applied Mathematics, vol 202, pp 113-138 (1998).

[9] Gottschlich-Muller B. and Muller S., *On Multiscale Concepts for Multi-Dimensional Conservation Laws*, Notes on Numerical Fluid Dynamics, Numerical Treatment of Multiscale Problems, vol 70, pp 119-133 (1999).

[10] Gottschlich-Muller B. and Muller S., *Adaptive Finite Volume Schemes for Conservation Laws based on Local Multiresolution Techniques*, Hyperbolic Problems, Theory, Numerics and Applications, pp 119-394 (1999).

[11] Hanel D. and Schwane R., *An implicit Flux-Vector Splitting Scheme for the Computation of Viscous Hypersonic Flows*, AIAA Paper 89-0274 (1989).

[12] Jameson A., *Time-Dependent Calculations using Multigrid, with Applications to Unsteady Flows past Airfoils and Wings*, AIAA Paper 91-1596 (1991).