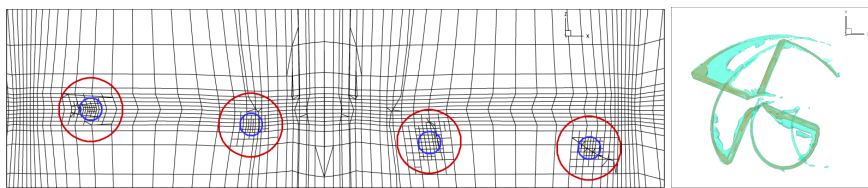




## **Executive Summary**

# **Multi-time multi-grid techniques. Application to helicopter rotor simulations.**

A VKI course



### **Problem area**

The simulation of rotorcraft aerodynamics is considerably more complex than the simulation of fixed wing aircraft aerodynamics. Rotorcraft flow is inherently dynamic, the inertial and elastic forces of the rotor blades interact with the aerodynamic forces, and aerodynamic interference of the rotor wake with the fuselage and tail rotor is important in many flight conditions.

The flow condition known as Blade-Vortex Interaction (BVI) is an important example of such interactions. Especially in low-speed descent, the rotor blades fly in their own wake. The (parallel) interaction of tip vortices and rotor blades causes strong pressure fluctuations on the blade, responsible for the typical ‘wopwop’ sound of helicopters. Prediction of BVI is challenging: the blade motion under inertial, elastic, and aerodynamic forces must be predicted correctly and the convection of the tip vortices must be accurate enough to

retain the vortices for, typically, one and a half rotor revolution (corresponding to a distance of one hundred blade chords).

This multiscale behaviour and multidisciplinary nature of rotorcraft aerodynamics has delayed the introduction of CFD techniques for rotorcraft aerodynamics. The numerical dissipation of standard CFD algorithms may destroy the vortex before the BVI event takes place. More advanced CFD algorithms, using high order discretization and/or local grid refinement, are better suited to tackle the problem. These algorithms, however, generally increase the computational complexity of the simulation and their efficiency should be improved before they can be applied to rotorcraft aerodynamics. In this paper, a solution algorithm will be presented in detail which significantly improves the efficiency by exploiting the periodic nature of rotor flows.

### **Report no.**

NLR-TP-2009-616

### **Author(s)**

H. van der Ven

### **Classification report**

Unclassified

### **Date**

November 2009

### **Knowledge area(s)**

Computational Physics en  
Theoretische Aërodynamica

### **Descriptor(s)**

multigrid  
discontinuous Galerkin  
time-periodic  
rotorcraft

**Description of work**

The basic idea is to solve the time-dependent flow equations for all time steps at once. The flow equations are discretised on a four-dimensional mesh containing all time levels. The resulting system of equations is solved by a multitime multigrid convergence acceleration technique which is an extension of standard multigrid solvers. The discretization, which already allowed local grid refinement in space, is extended to allow local grid refinement in time. Special attention is given to the issue of four-dimensional grid generation. A feature-based pre-adaptation algorithm is described which allows efficient grid generation for the simulation of rotor wakes. The algo-

rithm is applied to the simulation of a four-bladed helicopter rotor in high-speed flight.

**Results and conclusions**

A framework for four-dimensional flow simulation of time-periodic problems is presented, which significantly decreases the computational complexity of rotorcraft CFD simulations. The multitime multigrid algorithm allows efficient simulations of time-periodic aero-elastic problems with local grid refinement to capture relevant flow features.

**Applicability**

The algorithm can be used to predict complex and localised phenomena occurring in rotorcraft aerodynamics, such as BVI.

**PROJECT****Projectleider**

H. van der Ven

**Projecttitel**

ADIGMA

**Projectnummer**

2046203



NLR-TP-2009-616

## Multi-time multi-grid techniques. Application to helicopter rotor simulations

A VKI course

H. van der Ven


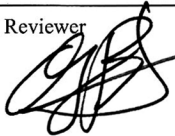

This report constitutes the notes to the contribution of NLR to the VKI 36th CFD/ADIGMA course on hp-adaptive and hp-multigrid methods, October 26-30, 2009. The work described in this paper is partially funded by the EU-project ADIGMA, and partially by NLR's programmatic research.

The contents of this report may be cited on condition that full credit is given to NLR and the author.

This publication has been refereed by the Advisory Committee AEROSPACE VEHICLES.

Customer	NLR
Contract number	----
Owner	NLR
NLR Division	Aerospace Vehicles
Distribution	Unlimited
Classification of title	Unclassified
	February 2010

Approved by:

Author  12/2/2010	Reviewer  12/2/2010	Managing department  12/2/2010
--	--	---



## Contents

<b>1</b>	<b>Introduction</b>	<b>7</b>
<b>2</b>	<b>Introduction to helicopter dynamics</b>	<b>8</b>
2.1	General	8
2.2	Relevant scales	10
<b>3</b>	<b>Numerical method</b>	<b>11</b>
3.1	Euler equations in a moving and deforming space-time domain	11
3.2	Geometry of space-time elements	12
3.3	Flow field expansion	13
3.4	Weak formulation of the Euler equations	13
<b>4</b>	<b>Multi-time multi-grid techniques</b>	<b>14</b>
4.1	Convergence to periodic solutions	14
4.2	Application to rotor flow	16
4.3	Grid generation	19
4.3.1	Simple approach	20
4.3.2	Motion representation under time refinement	20
4.3.3	Partial time levels	21
4.4	MTMG algorithm - smoother, grid levels, prolongation operator	22
4.4.1	Smoother	22
4.4.2	Restriction and prolongation operators, definition of grid levels	25
4.5	Grid design and expected speedups	30
<b>5</b>	<b>Adaptation strategies</b>	<b>32</b>
5.1	Introduction	32
5.2	Adaptation on streak lines	33
<b>6</b>	<b>Results</b>	<b>35</b>
6.1	Transonic oscillating NACA0012	35
6.2	7AD rotor in forward flight	42
<b>7</b>	<b>Conclusions</b>	<b>49</b>



## References

51

2 Tables

33 Figures

(53 pages in total)

## 1 Introduction

The simulation of rotorcraft aerodynamics is considerably more complex than the simulation of fixed wing aircraft aerodynamics. Rotorcraft flow is inherently dynamic, the inertial and elastic forces of the rotor blades interact with the aerodynamic forces, and aerodynamic interference of the rotor wake with the fuselage and tail rotor is important in many flight conditions. The flow condition known as Blade-Vortex Interaction (BVI) is an important example of such interactions. Especially in low-speed descent, the rotor blades fly in their own wake. The interaction of tip vortices and rotor blades may cause strong pressure fluctuations on the blade, responsible for the typical ‘wopwop’ sound of helicopters. Prediction of BVI is challenging: the blade motion under inertial, elastic, and aerodynamic forces must be predicted correctly and the convection of the tip vortices must be accurate enough to retain the vortices for, typically, one and a half rotor revolution.

The requirement to correctly represent the blade motion has led to the development of time-accurate flow solvers which are coupled with dynamics solvers or rotor comprehensive codes (Buchala et al. (Ref. 4), Pomin et al. (Ref. 16), Altmikus et al. (Ref. 1); the reader is also referred to the excellent review paper on rotorcraft CFD by Datta et al. (Ref. 6)). Several efforts have been undertaken to improve the vortex capturing capability of standard flow solvers, such as local grid refinement (Bottasso et al. (Ref. 3)), Chimera techniques with specific vortex grid systems (Ochi et al. (Ref. 14), Duraisamy et al. (Ref. 8)), and high order methods (Wake et al. (Ref. 29)). None of these techniques has been particularly successful or efficient, and successful BVI predictions have only been obtained by brute-force methods, using meshes of 100 million cells (Lim et al. (Ref. 12)) and time-marching several rotor revolutions before structural dynamics, trim, and aerodynamics have balanced out. Lim et al. use a series of grids of which the fine grid contains 100 million elements and apply a time step of 0.05 azimuthal degrees. Although the simulations exhibit BVI, the authors are not satisfied with the vortex resolution of the simulation as compared with experiment. They estimate a mesh containing 7 billion elements is necessary for the simulations to agree with experiment. This number is remarkably close to the estimates given by Caradonna (Ref. 5) in his review article. Clearly, such simulations cannot be run in a routine way and despite the qualitative success so far, there is a need for more efficient algorithms.

In this course an efficient solution algorithm is described which exploits the periodic nature of rotor flow. The algorithm itself, a multitime multigrid convergence acceleration algorithm, has first been introduced in (Ref. 26) and published in (Ref. 22). The dynamics are not solved time step after time step, but are solved for all time steps at once. Treating time like space, the

equations are solved on a four-dimensional mesh using multigrid techniques to accelerate convergence. Essentially, this turns a dynamic problem into a steady-state problem. This has important consequences for local grid refinement (no dynamic load balancing problems anymore) and the coupling with blade dynamics (no need for intricate time-accurate coupling schemes as in Piperno et al. (Ref. 15) and Wagner et al. (Ref. 28)). Moreover, given a suitable numerical scheme, the four-dimensional mesh can also be refined locally in time.

The underlying numerical scheme is discontinuous Galerkin space-time method for the Euler equations (Van der Vegt and Van der Ven (Ref. 20, 27)), which, using the conventional time-serial solution algorithm, has been successful in predicting rotor flow (Boelens et al. (Ref. 2)). After a short introduction to helicopter rotor dynamics in Chapter 2 and a short description of the numerical method in Chapter 3, the multitime multigrid algorithm is described in Chapter 4. The algorithm is based on the spatial multigrid algorithm of Klaij et al. (Ref. 11) which is extended to four dimensions and locally refined meshes. Details on grid generation and expected speedups with respect to the conventional time-serial approach are presented. Chapter 5 discusses feature-based grid adaptation techniques for rotor flow. Results are described in Chapter 6.

## **2 Introduction to helicopter dynamics**

### **2.1 General**

In this section a very short introduction to helicopter dynamics is presented. The interested reader is referred to the detailed text books on helicopters by Johnson (Ref. 10). The background presented in this section is sufficient for the reader to appreciate the complexity of rotor aerodynamics, structural dynamics and flight mechanics.

The helicopter rotor serves both as a lifting device and a steering device. The rotation of the blades generate a relative air flow around the blades. The blade cross sections are shaped like wing sections, and increasing the pitch settings of the blades will increase the local angle of attack and increase the lift generated by the blades (up till stall of course). When the rotor is hovering in the same place, the local Mach number varies along the span of the blade, but is independent of the azimuth angle of the blade. Due to this symmetry, the total aerodynamic force is perpendicular to the rotor plane. This changes when the helicopter moves forward: the rotor blade on the advancing side (moving in the direction of flight) experiences higher flow speeds than the rotor blade on the retreating side (moving opposite the direction of flight). If the pitch setting of the blade is unchanged and constant in azimuthal direction, the advancing side of the rotor generates more lift than the retreating side, and a rolling moment is created. In order to avoid this,



a cyclic pitch setting of the blades is introduced which increases the local angle of attack on the retreating side such that the rolling moment vanishes. Conversely, the cyclic pitch setting can be used to steer the helicopter by introducing rolling or pitching moments. The yawing moment is controlled by the tail rotor in conventional, single rotor, helicopters. The pitch settings of the main and tail rotor are controlled by the pilot who has so-called sticks to control the collective, longitudinal and lateral cyclic settings of the main rotor and the collective of the tail rotor.

For the simulation of an isolated rotor, there are three angles which describe the pitch setting: the collective pitch and the longitudinal and lateral pitch (the amplitudes of the two harmonics with frequency equal to the rotor rotational frequency). The pitch settings are the principle actuators for the rotor dynamics. The resultant aerodynamic forces are, however, not completely determined by the pitch settings. This is because the actual blade motion is determined by inertial and structural forces. The pitch settings are passed to the blades through the so-called pitch link. In conventional, fully articulated, rotors, the pitching of the blade is made possible because the blade is connected to the rotor hub by a pitching hinge. In order to reduce the transmission of dynamic moments to the rotor hub, hinges in the other directions are added between blade and hub. Within bounds, the blade is allowed free flapping and lead-lag motions under the influence of inertial and aerodynamic forces.

Rotor blades have high aspect ratio's: the typical ratio of blade span and blade chord is in the order of 15. Weight restrictions in the design of rotor blades imply that the blade structure is very flexible. Hence the blade will deform under aerodynamic and inertial forces, adding an additional complexity to the simulation of rotor dynamics.

Summarizing, the blade motion is primarily determined by the rotor rotational speed and pitch settings, but the interplay of inertial, dynamic, and aerodynamic forces together with the structural and geometrical properties of the blades and the dynamic properties of the control system determine the blade motion and blade loads. Only once these forces have balanced out, the net forces such as thrust and rolling and pitching moments are known. In the validation of a simulation, it is of the utmost importance to have the same thrust and moments as in the experiment or flight test. The process of obtaining the same forces by changing the pitch settings is called trim.

From an aerodynamic viewpoint, the flow phenomena occurring in rotor flow are challenging in their own right. Obviously, the flow is dynamic. High angles of attack on the advancing sides may lead to shocks. High angle of attacks on the retreating side may cause dynamic stall. More important are various interference effects, of which the most important effect is that rotor blades fly in the wake of preceding blades. Hence, the inflow field of a rotor blade is not necessarily

undisturbed flow, and it is important to accurately predict the wake of the blades. A well-known example of the interaction between the wake and the blade is the typical ‘wopwop’ sound made by helicopters in descent. The sound is caused by the interaction of the tip vortices with the blades, which may cause strong pressure fluctuations, and hence sound. In order to capture this phenomenon, a numerical method must be capable of resolving the tip vortex for at least one and a half rotor revolution, which corresponds to a length of hundreds of rotor blade chords. Other interference effects are the interaction between the rotor wake and the fuselage or tail of the helicopter.

So, any successful simulation of rotor flow must treat the following items:

- dynamic problem,
- multi-physics,
- trim,
- accurate wake prediction,
- multi-scale problem,
- turbulence issues: dynamic stall, shock boundary layer interaction.

All but the last item will be treated in the following. The multi-scale issue will be discussed in the next section.

## 2.2 Relevant scales

For the aerodynamics the most important scaling parameters are the rotor radius  $R$ , the tip Mach number  $M_{\text{tip}}$  the freestream Mach number  $M_\infty$ . The ratio of the latter two is called the advance ratio  $\mu = \frac{M_\infty}{M_{\text{tip}}} = \frac{u_\infty}{\Omega R}$ , where  $u_\infty$  is the freestream velocity and  $\Omega$  is the angular velocity of the rotor.

For accurate wake prediction, it is important to capture the tip vortices. According to Caradonna (Ref. 5) the size of the vortex core is  $R/50$ . A standard numerical method requires about 20 cells across the vortex core<sup>1</sup>, so the mesh width  $h$  should be  $h = R/1000$ . A uniform mesh covering a computational domain of size  $O(R^3)$  then requires one billion elements.

Accurate convection of the vortex requires a convective CFL number in the order of 1. So, for  $T = 2\pi/\Omega$  the period of one rotor revolution,

$$\frac{\Delta t}{T} = \frac{h}{uT} = \frac{h}{2\pi R} \frac{\Omega R}{u} = \frac{h/R}{2\pi\mu}. \quad (1)$$

Typical advance ratio's are between 0.1 and 0.3, hence the number of time steps per revolution is in the order of one thousand.

---

<sup>1</sup>Figures for DG are presented in Section 4.5

In a convential multi-physics coupling approach, it takes tens of revolutions before the trim and aero-elastic response have balanced out. So we end up with a simulation on a mesh with one billion elements using 10,000-100,000 time steps.

We are clearly in want of an efficient algorithm.

### 3 Numerical method

In this chapter the Euler equations in a moving and deforming flow domain and their space-time discontinuous Galerkin discretization are presented.

#### 3.1 Euler equations in a moving and deforming space-time domain

The Euler equations of gas dynamics are considered in a time-dependent flow domain. Since the flow domain boundary is moving and deforming in time, no explicit separation between the space and time variables is made and the Euler equations are considered directly in  $\mathbb{R}^4$ . Let  $\mathcal{E} \subset \mathbb{R}^4$  be an open domain. A point  $x \in \mathbb{R}^4$  has coordinates  $(x_1, \dots, x_4)$ , but the notation  $t = x_4$  is also frequently used for the time coordinate. The flow domain  $\Omega(t)$  at time  $t$ , ( $t_0 < t < T$ ), is defined as:  $\Omega(t) := \{\bar{x} \in \mathbb{R}^3 \mid (\bar{x}, t) \in \mathcal{E}\}$ , with  $t_0$  and  $T$  the initial and final time of the evolution of the flow domain. The flow domain  $\Omega(t)$  at time  $t$  is also referred to as the spatial domain at time  $t$  to distinguish it from the space-time domain  $\mathcal{E}$ . The space-time domain boundary  $\partial\mathcal{E}$  consists of the hypersurfaces  $\Omega(t_0) := \{x \in \partial\mathcal{E} \mid x_4 = t_0\}$ ,  $\Omega(T) := \{x \in \partial\mathcal{E} \mid x_4 = T\}$  and  $\mathcal{Q} := \{x \in \partial\mathcal{E} \mid t_0 < x_4 < T\}$ .

Let  $\mathbf{F} : \mathbb{R}^5 \rightarrow \mathbb{R}^{5 \times 4}$  denote the flux tensor, which is defined as:

$$\mathbf{F} = \begin{pmatrix} \rho u_1 & \rho u_2 & \rho u_3 & \rho \\ \rho u_1^2 + p & \rho u_1 u_2 & \rho u_1 u_3 & \rho u_1 \\ \rho u_1 u_2 & \rho u_2^2 + p & \rho u_2 u_3 & \rho u_2 \\ \rho u_1 u_3 & \rho u_2 u_3 & \rho u_3^2 + p & \rho u_3 \\ (\rho E + p)u_1 & (\rho E + p)u_2 & (\rho E + p)u_3 & \rho E \end{pmatrix},$$

with  $\rho$ ,  $p$ , and  $E$  the density, pressure, and specific total energy, respectively, and  $u_i$  the velocity components in the Cartesian coordinate directions  $x_i$ ,  $i \in \{1, 2, 3\}$  of the velocity vector  $u : \mathcal{E} \rightarrow \mathbb{R}^3$ . Let the vector  $U : \mathcal{E} \rightarrow \mathbb{R}^5$  denote the conservative flow variables with components:

$$U_i = \mathbf{F}_{i4},$$

then the Euler equations of gas dynamics are defined as:

$$\operatorname{div} \mathbf{F}(U(x)) = 0, \quad x \in \mathcal{E}, \quad (2)$$

together with either initial or periodic conditions,

$$\begin{cases} U(x) = U_0(x), & \bar{x} \in \Omega(t_0), \\ U(\bar{x}, T) = U(\bar{x}, t_0), & \bar{x} \in \Omega(t_0) = \Omega(T) \end{cases} \quad \text{or}$$

and boundary conditions:

$$U(x) = \mathcal{B}(U, U_w), \quad x \in \mathcal{Q}.$$

Here  $U_0 : \Omega(t_0) \rightarrow \mathbb{R}^5$  denotes the initial flow field,  $\mathcal{B} : \mathbb{R}^5 \times \mathbb{R}^5 \rightarrow \mathbb{R}^5$  the boundary operator and  $U_w : \mathcal{Q} \rightarrow \mathbb{R}^5$  the prescribed boundary flow field data. The divergence of a second order tensor is defined as:  $\text{div } \mathcal{F} = \frac{\partial \mathcal{F}_{ij}}{\partial x_j}$ , and the summation index is used on repeated indices in this article. The Euler equations are completed with the equation of state for a calorically perfect gas:  $p = (\gamma - 1)\rho(E - \frac{1}{2}u_i u_i)$ , with  $\gamma$  the ratio of specific heats.

For the design of solution strategies it is important to note that (2) is hyperbolic in space and time.

### 3.2 Geometry of space-time elements

The tessellation  $\mathcal{T}_h$  of the space-time domain  $\mathcal{E}$  is the union of four-dimensional hexahedral elements:<sup>1</sup>

$$\mathcal{T}_h := \{\mathbf{K}_j \mid \bigcup_{j=1}^N \mathbf{K}_j = \mathcal{E} \text{ and } \mathbf{K}_j \cap \mathbf{K}_{j'} = \emptyset \text{ if } j \neq j', 1 \leq j, j' \leq N\}.$$

Each element  $\mathbf{K} \in \mathcal{T}_h$  is related to the master element  $\hat{\mathbf{K}} = (-1, 1)^4$  through the mapping  $F_{\mathbf{K}}$ :

$$F_{\mathbf{K}} : \hat{\mathbf{K}} \rightarrow \mathbf{K} : \xi \mapsto x = \sum_{i=1}^{16} x_i(\mathbf{K}) \chi_i(\xi),$$

with  $x_i(\mathbf{K}) \in \mathbb{R}^4$ ,  $1 \leq i \leq 16$ , the space-time coordinates of the vertices of the hexahedron  $\mathbf{K}$  and  $\chi_i(\xi) = \frac{1}{16} \prod_{j=1}^4 (1 \pm \xi_j)$  the quad-linear finite element shape functions for hexahedra (with the combination of signs depending on the vertex index).

**Remark 1.** *This definition of the tessellation of the space-time domain allows elements which are arbitrarily oriented in space-time. In practice, grid generators are not capable of generating such meshes, and the elements are created by linearly interpolating two three-dimensional, possibly deforming, hexahedra in time (see Section 4.3.1 for details). It should be noted that the numerical algorithm described in this paper is suited for space-time meshes containing arbitrarily oriented hexahedra.*

<sup>1</sup>The correct name for a four-dimensional hexahedron is probably hyper-octahedron, motivated by the name for hypercube and the fact that the polyhedron has eight faces. For convenience sake, the elements are still called hexahedra in this paper.

### 3.3 Flow field expansion

The discontinuous Galerkin finite element discretization is obtained by approximating the flow field  $U(\bar{x}, t)$  and test functions  $W(\bar{x}, t)$  with polynomial expansions in each element  $\mathcal{K}$ , which are discontinuous across element faces, both in space and time. In the master element  $\hat{\mathcal{K}}$  the basis functions  $\hat{\phi}_m$  ( $m = 0, \dots, 4$ ) are defined which are linear in space and time:

$$\hat{\phi}_m(\xi) = \begin{cases} 1, & m = 0, \\ \xi_m, & m > 0. \end{cases}$$

The basis functions  $\phi_m^K$  in an element  $\mathcal{K}$  are related to the basis function in the master element  $\hat{\mathcal{K}}$  through the parametrization  $F_K$ :  $\phi_m^K = \hat{\phi}_m \circ F_K^{-1}$ , ( $m = 0, \dots, 4$ ). The superscript  $K$  is dropped when the element in question is clear.

Let  $V_h^1(\mathcal{T}_h)$  be the discrete broken function space defined as

$$V_h^1(\mathcal{T}_h) = \{f : \mathcal{T}_h \rightarrow \mathbb{R}^5 \mid f|_K \subset \text{span}\{\phi_m \mid 0 \leq m \leq 4\}\}.$$

A solution vector  $U_h$  in  $V_h^1(\mathcal{T}_h)$  will be written as

$$U_h|_K = \sum_{m=0}^4 \hat{U}_m^K \phi_m^K,$$

with  $\hat{U}_m^K$  the expansion coefficients. The first expansion coefficient is also written as  $\bar{U} = \hat{U}_0$ , and is equal to the cell average if the mesh is cartesian. The other expansion coefficients are referred to as gradients, closely related to the directional derivative in one of the computational directions. With this choice of basis functions the method is second-order accurate in space and time.

### 3.4 Weak formulation of the Euler equations

The weak formulation of the Euler equations is obtained by multiplying the (space-time) Euler equations with a test function  $W_h$ , integrating over a space-time element  $K$  and using Gauss' theorem to obtain face flux integrals.

In order to ensure that the weak formulation of the Euler equations is well defined, the broken space  $V(\mathcal{T}_h)$  is introduced:

$$\begin{aligned} V(\mathcal{T}_h) := \{ & U : \mathcal{T}_h \rightarrow \mathbb{R}^5 \mid (\text{grad } U^1)^T : \mathcal{F}(U^2)|_K \in L^1(K); \\ & \gamma^-(U^1) \cdot (n_K^T \mathcal{F}(\gamma^-(U^2)) + n_K^T \mathcal{F}(\gamma^+(U^3))) \in L^1(\partial K); \\ & \forall (U^1, U^2, U^3) \in V(\mathcal{T}_h), \forall K \in \mathcal{T}_h \}, \end{aligned}$$

with  $L^1$  the space of Lebesgue integrable functions,  $\gamma^\pm(U) = \lim_{\epsilon \downarrow 0} U(x \pm \epsilon n_K)$  the traces of  $U$  at  $\partial K$ ,  $n_K \in \mathbb{R}^4$  the unit outward normal vector at  $\partial K$ , and superscript  $T$  denoting the transposition of a vector. The gradient operator  $\text{grad} : \mathbb{R}^5 \rightarrow \mathbb{R}^{4 \times 5}$  is defined as:  $(\text{grad } U)_{ij} = \frac{\partial U_j}{\partial x_i}$  and the symbol  $:$  represents the dyadic product of two second order tensors and is defined for  $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{n \times m}$  as  $\mathcal{A} : \mathcal{B} = \mathcal{A}_{ij} \mathcal{B}_{ij}$ .

The weak form of the Euler equations is:

$$\sum_{j=1}^N \left\{ - \int_{K_j} (\text{grad } W_h)^T : \mathbf{F}(U_h) d\mathbf{K} + \int_{\partial K_j} \gamma^-(W_h) \cdot H(\gamma^-(U_h), \gamma^+(U_h), n_K) d\partial K \right\} = 0. \quad (3)$$

The numerical flux  $H = H(\gamma^-(U_h), \gamma^+(U_h), n_K)$  is introduced to stabilize the central flux. In this article, the extension of the HLLC flux for moving meshes is used for the flux through space-time faces. For time faces, which are parallel to the spatial directions, a pure upwind flux is used.

The DG system of equations is obtained by replacing  $W_h$  in (3) by  $\phi_l^K$  for each  $K \in \mathcal{T}_h$  and  $0 \leq l \leq 4$ . The details are of no importance in the current paper and the system of equations is summarised as

$$\mathcal{L}_h(U_h) = 0.$$

The standard way to solve the above system is to solve it time slab after time slab. A time slab is defined as  $\mathcal{E} \cap [t^n, t^{n+1}]$ , for a given time interval  $[t^n, t^{n+1}]$ . As the solution  $U_h^{n-1}$  in the previous time slab is known, the systems of equations in a given time slab is written as

$$\mathcal{L}_h(U_h^n, U_h^{n-1}) = 0.$$

Details of the system of equations and its derivation are given in Van der Vegt et al. (Ref. 20).

## 4 Multi-time multi-grid techniques

### 4.1 Convergence to periodic solutions

The flow about an isolated rotor is a time-periodic problem. Trying to obtain a periodic flow solution by marching in time is a slow process. The convergence rate is comparable to Jacobi processes, and hence is equal to  $1 - O(\Delta t/T)$ . As multigrid algorithms are specifically designed to move the convergence rate away from unity, we need multigrid in time. In order to do this, the flow solution at all time levels is required. Basically, the time-dependent flow equations are solved on a four-dimensional mesh containing all time levels. On the four-dimensional mesh, a

standard  $h$ -multigrid algorithm is applied, where the coarse grid levels are defined by coarsening in both space and time. The algorithm for multigrid in time is called a multi-time multi-grid algorithm and is described below.

Let  $\mathcal{L}_h$  be the DG discretisation operator for a given time slab,  $U_h^n$  the solution in the time slab  $[t^n, t^{n+1}]$ , which satisfies the equations

$$\mathcal{L}_h(U_h^n, U_h^{n-1}) = 0,$$

where  $U_h^{n-1}$  is the solution in the previous time slab. For a periodic problem the equations are

$$\begin{cases} \mathcal{L}_h(U_h^i, U_h^{i-1}) = 0, & 1 \leq i \leq N, \\ U_h^0 = U_h^N, \end{cases}$$

if the period is divided into  $N$  time slabs.

In the time-serial algorithm the equations are solved in pseudo-time  $\tau$  as

$$\frac{\partial \tilde{U}_h^n}{\partial \tau} + \mathcal{L}_h(\tilde{U}_h^n, U_h^{n-1}) = 0, \quad n = 1, 2, \dots, \infty$$

where  $U_h^0$  is the initial solution, upon convergence we set  $U_h^n = \tilde{U}_h^n$ , and proceed to the next time step. In the MTMG approach the equations are solved as

$$\frac{\partial \tilde{U}_h^i}{\partial \tau} + \mathcal{L}_h(\tilde{U}_h^i, \tilde{U}_h^{i-1}) = 0, \quad 1 \leq i \leq N,$$

simultaneously, with the periodic boundary condition  $\tilde{U}_h^0 = \tilde{U}_h^N$ .

The  $L_2$ -residual  $\epsilon_{\text{per}}$  of a solution obtained with the MTMG algorithm is defined as

$$\epsilon_{\text{per}}^2 = \frac{1}{N} \sum_{i=1}^N \|\mathcal{L}_h(\tilde{U}_h^i, \tilde{U}_h^{i-1})\|_2^2, \quad (4)$$

where  $\|\cdot\|_2$  is the  $L_2$ -norm in the time slab. In the time-serial approach the single time step residual  $\epsilon_s^{(n)}$  is measured:

$$\epsilon_s^{(n)} = \|\mathcal{L}_h(\tilde{U}_h^n, U_h^{n-1})\|_2, \quad (n \geq 1) \quad (5)$$

but this residual does not measure the convergence to a periodic solution. Given a series  $\tilde{U}_h^{kN+1}, \dots, \tilde{U}_h^{(k+1)N}$  of solutions in the  $k$ -th period, define the solution vectors  $\tilde{V}_h^i = \tilde{U}_h^{kN+i}, 1 \leq i \leq$

$N$ . Considering  $\tilde{V}_h$  as a periodic solution, we compute  $\epsilon_{\text{per}}$  as follows:

$$\begin{aligned}
 \epsilon_{\text{per}}^2 &= \frac{1}{N} \left( \|\mathcal{L}_h(\tilde{V}_h^1, \tilde{V}_h^N)\|_2^2 + \sum_{i=2}^N \|\mathcal{L}_h(\tilde{V}_h^i, \tilde{V}_h^{i-1})\|_2^2 \right) \\
 &= \frac{1}{N} \left( \|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2^2 + \sum_{i=2}^N \|\mathcal{L}_h(\tilde{U}_h^{kN+i}, \tilde{U}_h^{kN+i-1})\|_2^2 \right) \\
 &= \frac{1}{N} \left( \|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2^2 + \sum_{i=2}^N (\epsilon_s^{(kN+i)})^2 \right). \tag{6}
 \end{aligned}$$

When the conventional residuals  $\epsilon_s^{(n)}$  are converged to machine accuracy, the residual  $\epsilon_{\text{per}}$  is dominated by the periodic residual  $\|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2$ .

The MTMG method has been applied to two-dimensional transonic flow over an harmonically oscillating NACA0012 foil. The freestream Mach number is 0.8, the angle of attack oscillates between  $-0.5^\circ$  and  $4.5^\circ$  with a non-dimensional frequency of 0.314. The space-time mesh consists of  $256 \times 128 \times 20$  elements, where the last dimension is the time dimension. In Figure 1 the convergence of the MTMG method is compared with that of the conventional time-serial method. For the time-serial method at each time step the implicit system is solved in pseudo-time, and six time steps are shown. At each time step  $n$  the residual  $\epsilon_s^{(n)}$  is converged to  $5 \cdot 10^{-7}$ . For the MTMG method full multigrid has been applied, with 150 iterations on the coarsest mesh with  $64 \times 32 \times 5$  grid cells, 200 multigrid cycles on the next finer level, and 500 multigrid cycles on the fine mesh. For both methods V cycles with one prerelaxation and one postrelaxation have been used. The convergence rate of the MTMG method is comparable to the convergence rate of the multigrid algorithm for the space DG discretisation operator for steady state problems.

Using (6) we have

$$\epsilon_{\text{per}}^2 = \frac{1}{N} \left( \|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2^2 + (N-1)(5 \cdot 10^{-7})^2 \right). \tag{7}$$

In Table 1 this residual is shown for five consecutive periods of the time-serial iterates. Clearly, convergence in  $L_2$ -norm to a time periodic solution is slow, and the residual  $\|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2$  dominates the time step residual  $\epsilon_s^{(n)}$ . Note that based on the aerodynamic coefficients in Figure 2 one would conclude convergence in about three periods.

#### 4.2 Application to rotor flow

Apart from generating a periodic solution by construction, the main advantage of the solution algorithm lies in the fact that it transforms a time-dependent (*dynamic*) problem into a steady-state (*static*) problem. This has several advantages:



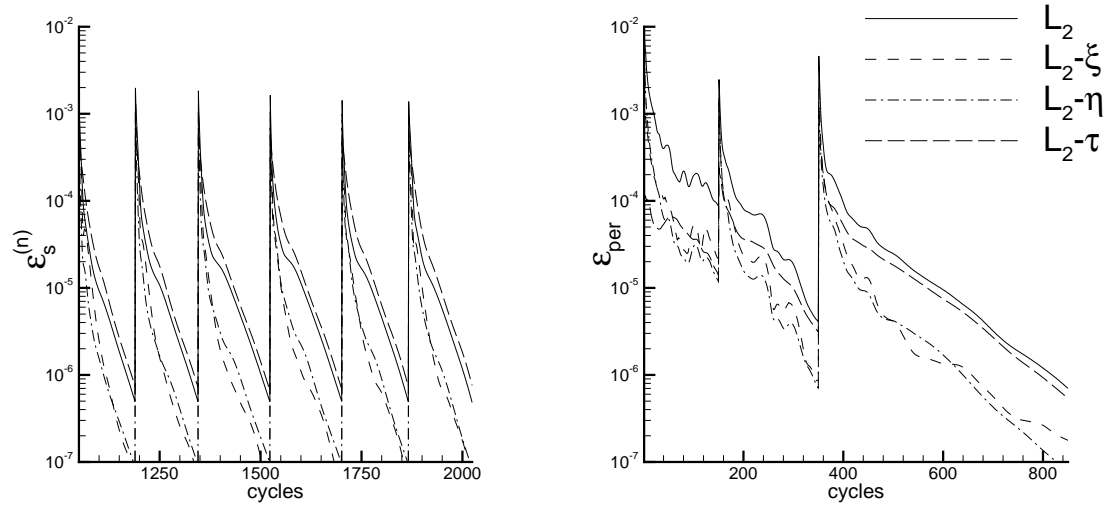


Fig. 1 Convergence history of the residual  $\epsilon_s^{(n)}$  of six time steps of the time-serial method (left) and the complete convergence history of the residual  $\epsilon_{per}$  of the MTMG method with full multigrid (right).

period	$\epsilon_{per}$
1	$0.89 \cdot 10^{-3}$
2	$0.61 \cdot 10^{-3}$
3	$0.35 \cdot 10^{-3}$
4	$0.24 \cdot 10^{-3}$
5	$0.20 \cdot 10^{-3}$

Table 1 Residual  $\epsilon_{per}$  defined in (7) of the time-serial simulation.

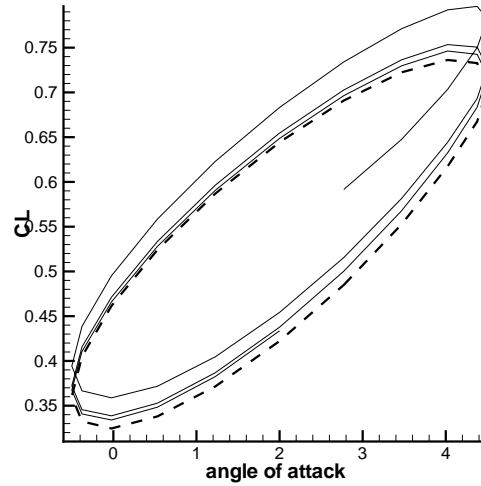
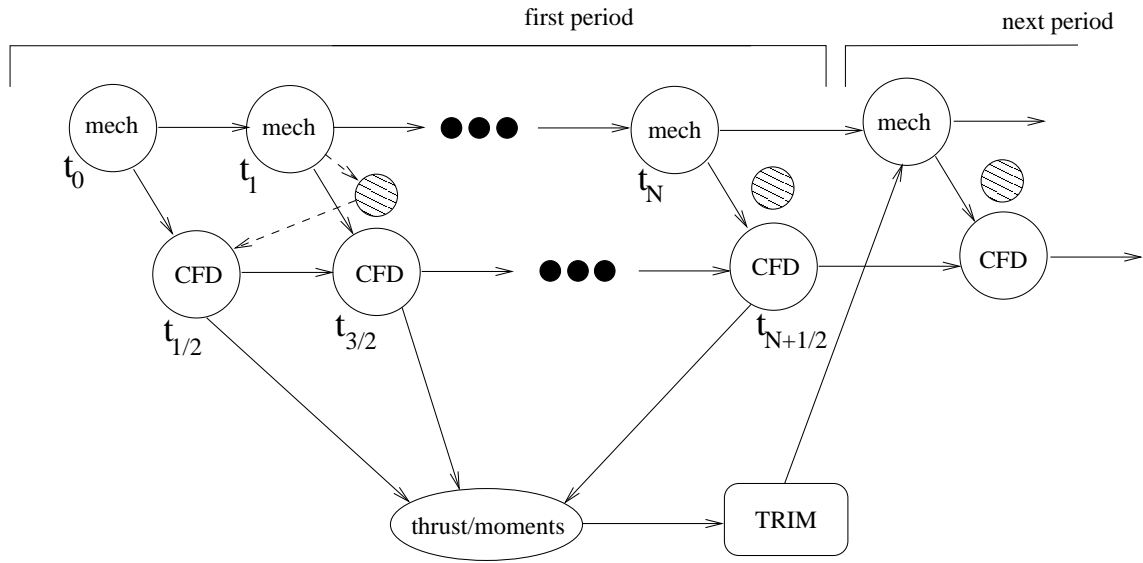


Fig. 2 Polar plot of the lift coefficient. Solid: time serial method; dashed: MTMG.

- as long as the solution process converges, the final solution is independent of the solution process. The underlying discretised equations are unmodified, and, moreover, there is no problem with the possible accumulation of numerical errors from preceding time steps;
- local grid refinement can be extended to the time dimension. Since there is no time direction in the solution algorithm, interpolation or the order of time steps in the case of hanging nodes are no issue;
- combining local grid refinement and parallel processing does not lead to dynamic load balancing problems. Since the local grid refinement no longer needs to be applied at each time step, the number of grid adaptations reduces significantly to about five times per simulation, which is the usual number for steady-state simulations. Hence the parallel efficiency on a massively parallel processors (MPP) machine is not hindered by dynamic load balancing issues. Because of the grid sizes of four-dimensional grids it is expected that the MTMG algorithm easily scales to thousands of processors;
- time-accurate coupling with other physics models is straightforward.

For the strongly coupled aeroelastic simulation of a trimmed rotor/hub/fuselage system all these benefits help to decrease the computational complexity of the simulation.

Conventionally, the solution procedure for the aeroelastic simulation is as shown in Figure 3, based on the implicit/implicit staggered scheme of Wagner et al. (Ref. 28) and Piperno et al. (Ref. 15). After each period, the thrust and moments can be computed and used to trim the rotor, after which a new period is simulated. If grid adaptation were applied, dynamic load balancing



*Fig. 3 Conventional time serial coupling procedure for the aeroelastic modeling of the rotor/hub/fuselage system, including trim. Shown is the implicit/implicit staggered scheme of Wagner et al. (Ref. 28). After each revolution the thrust and moments are collected and used to trim the rotor system, after which the mechanics/CFD iterations are restarted.*

problems would occur, effectively destroying any existing efficiency in other parts of the aerodynamics model than the adaptation algorithm.

In the MTMG approach all simulation data is available at all time steps, and the trust and moments are readily available to trim the rotor. Moreover, a modification in the pitch schedule will require only a small number of pseudo-time iterations for the flow solution to conform to the new schedule. Another consequence of having the data available at all time steps is that the coupling between the aerodynamics and mechanics modules can be made genuinely implicit, without the need of predictor-corrector mechanisms.

Since local grid refinement will not lead to dynamic load balancing issues in the MTMG approach, local grid refinement can be applied to decrease the required grid size (see Chapter 5 for more details). The coupling procedure for the aeroelastic simulation using the MTMG approach, including trim and grid adaptation, is shown in Figure 4.

### 4.3 Grid generation

Even though there are no four-dimensional grid generators, it is interesting to note that in four dimensions it is possible to generate a single, boundary conforming mesh about a fuselage-main rotor-tail rotor configuration. So MTMG allows the computation of the flow around individually

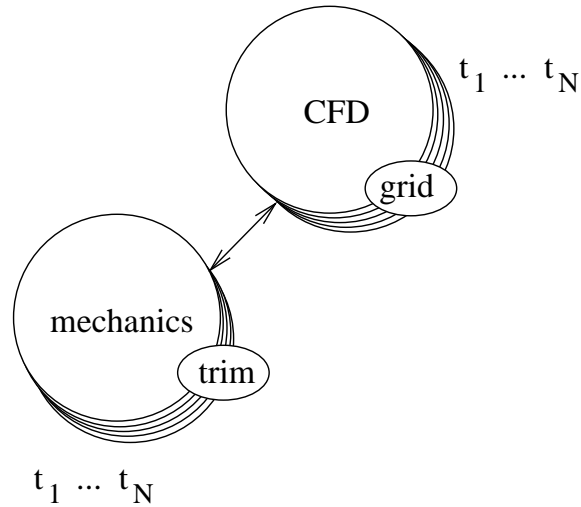


Fig. 4 Coupling procedure for the aeroelastic modeling of the rotor/hub/fuselage system, including trim and grid adaptation, using the MTMG solution algorithm.

moving geometries without the need to resort to interpolation as is necessary in Chimera and sliding grid techniques. As interpolation is a source of dissipation and dispersion errors, this is an important benefit of the four-dimensional framework of MTMG.

#### 4.3.1 Simple approach

As there are no 4D grid generators, the 4D space-time meshes must be generated ‘by hand’. The most simple way to construct a 4D mesh is collating a series of spatial meshes to form a four-dimensional mesh. The spatial meshes, including grid deformation to accommodate the geometry motion, are generated in the conventional way. An in-house block-structured grid generator called ENGRID (Ref. 19) is used to generate the original mesh. The meshes at a given time are deformed to accommodate the geometry motion. In this way each space-time element in the 4D mesh is the linear interpolation in time of a 3D element at a time  $t = t_n$  and a 3D element at time  $t = t_{n+1}$ .

#### 4.3.2 Motion representation under time refinement

The standard local refinement algorithm obtains the coordinates of new grid points by linearly interpolating between two existing grid points. Grid consistency for internal cells is ensured if the original cell is a hexahedron parametrised by a quadlinear mapping. Obviously, this refinement algorithm does not preserve the curvature of the geometry. A remedy is to project newly created grid points on the geometry onto the geometry definition. In principle, this may lead to grid folding if the original mesh does not capture the curvature sufficiently well. For most meshes the original curvature is captured well, so this is a feasible algorithm for spatial refinement.

For time refinement, however, this is not the case. In general, the physical CFL number for cells near the geometry will be large, implying a large aspect ratio as well. Projecting a newly created geometry point (due to time refinement) onto the geometry (that is, ensuring the correct grid motion) is hence much more likely to incur grid folding. On the other hand, accuracy is seriously impaired if the location of geometry points created under time refinement does not correspond to the analytically defined geometry motion. The motion is then only correctly represented on the original time resolution of the 4D mesh.

To overcome this problem, several strategies have been tried. The most obvious one is to generate a 4D mesh as described in Section 4.3.1 with sufficient time resolution as an initial mesh. For rotor simulations, where a starting spatial mesh (without local mesh refinement for vortex capturing) typically contains half a million grid cells, and a thousand time steps per revolution are required to capture phenomena as BVI, this would lead to meshes of hundreds of million elements, even before spatial refinement to capture the vortices, forfeiting the efficiency claims of the MTMG algorithm.

Another approach is to leave the geometry points created under time refinement at their linearly interpolated position, but modify the slip boundary condition so as to accommodate the actual grid motion. This boundary condition is known as the transpiration boundary (Sankar et al. (Ref. 17)). The geometry motion and grid motion need not coincide, and the difference is compensated by prescribing a transversal flow along the boundary (effectively a linearization of the correct boundary motion on the incorrect mesh). This is an efficient approach for potential methods and obviates the need for a grid deformation algorithm. However, it turned out that the linearization underlying the transpiration boundary condition was not capable of removing the non-smoothness of the linearly-interpolated grid motion.

The final approach is described in the next section.

### 4.3.3 Partial time levels

During anisotropic refinement new time levels may be created which do not cover the original computational space-domain. The ability of the numerical method to accommodate such meshes can also be exploited during grid generation: not all time levels need to cover the computational space-domain. So the idea is to limit the space-domain for selected time levels to a region near the geometry. This approach reduces the number of grid cells on the initial four-dimensional mesh, while correctly representing the geometry motion for a sufficient number of time levels.

An illustration is given in Figure 5 for a simple spatially one-dimensional flow domain, representative of a 1D piston problem. The left side of the spatial domain oscillates in time. The space-

time mesh contains three full ( $\Omega(t_0)$ ,  $\Omega(t_4)$  and  $\Omega(t_8)$ ) and six partial time levels. The spatial end points of the partial time levels are hanging nodes in the space-time mesh. In order to ensure that the space-time mesh is folding-free, the end points of the partial time levels should be located on the straight lines between the corresponding spatial points of the full time levels. This is most easily accomplished by fixing the location in space of the end points of the partial time levels.

Summarising, connected domains around the moving geometries are chosen which will comprise the spatial computational domain of the partial time levels. The grid deformation algorithm used to accommodate the geometry motion is only used in this domain. Dirichlet boundary conditions are applied for the deformation algorithm: equal to the geometry motion at the geometry and zero on ‘farfield’ boundary of the domain. For the full time levels grid points outside the spatial computational domain of the partial time levels are not allowed to move.

Geometrically, the extent of this region is restricted by the ability of the grid deformation algorithm to accommodate the geometry motion. Aerodynamically, the initial extent of this region is irrelevant, since the region can be extended during the simulation using local grid refinement in time.

The proposed algorithm on the one hand allows to start with a sufficiently fine time resolution about the geometry, while on the other hand restricting the total number of cells in the space-time mesh. A time uniform mesh of the space-time domain shown in Figure 5 would contain 64 cells, the mesh shown contains only 40 cells. For the 2D simulations in the Section 6.1, the mesh with 64 time levels, of which only 4 are full, only contains 8500 elements, whereas a time-uniform mesh would contain 32,000 elements. For rotor applications the benefit is even greater.

#### **4.4 MTMG algorithm - smoother, grid levels, prolongation operator**

##### **4.4.1 Smoother**

In compressible CFD, usually a full multigrid algorithm is applied with Runge-Kutta smoothers. The smoother for the multitime multigrid algorithm is the Runge-Kutta 5 scheme of (Ref. 20), with optimised coefficients for the DG discretisation of the advection equation.

The stability and smoothing characteristics of the Runge-Kutta scheme will be analysed for the time-dependent, one-dimensional convection equation  $\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0$ , with  $c > 0$ . The equation is discretised on a uniform space-time mesh, periodic in space and time, with mesh width  $\delta x$ , and with time step  $\delta t$ . The aspect ratio of the space-time cells is related to the physical CFL number,  $CFL_f = c\delta t/\delta x$ .

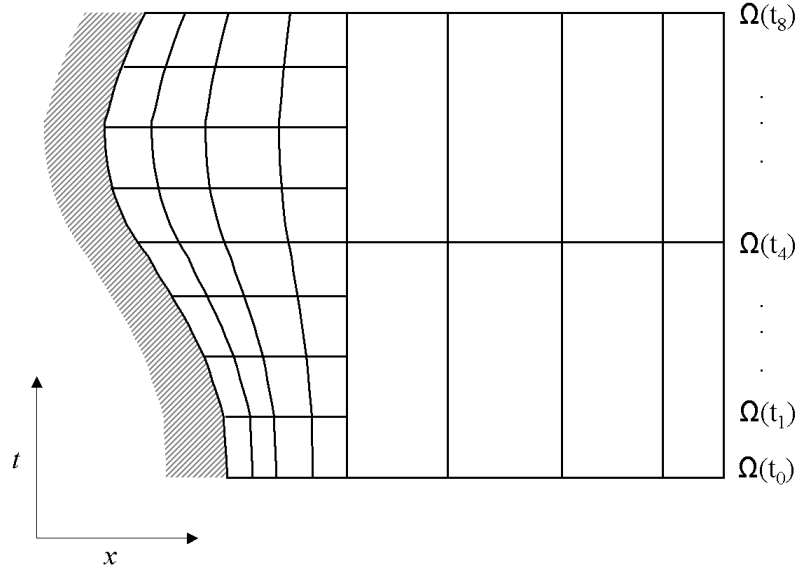


Fig. 5 Illustration of the partial time levels for a spatially one-dimensional domain with a moving boundary on the left. The space-time mesh contains three full ( $\Omega(t_0)$ ,  $\Omega(t_4)$  and  $\Omega(t_8)$ ) and six partial time levels.

For clarity of presentation, the convection equation is discretised using only the constant basis function, and the linear basis function in time (so the spatial gradient is discarded). Hence in each cell there are two variables: the cell-averaged quantity  $\bar{u}$  and the time gradient  $\hat{u}$ . Let  $\mathbf{u}_i^n = (\bar{u}_i^n, \hat{u}_i^n)^T$  be the solution in spatial cell  $i$  in time slab  $n$ . Deriving the equations from the weak form, we get the following two equations, where the first is obtained by substituting the cell-average basis function as test function in the weak form, and the second by substituting the time-gradient basis function.

$$\begin{aligned} (\bar{u}_i^n + \hat{u}_i^n - \bar{u}_i^{n-1} - \hat{u}_i^{n-1}) \delta x + c (\bar{u}_i^n - \bar{u}_{i-1}^n) \delta t &= 0, \\ (-\bar{u}_i^n + \hat{u}_i^n + \bar{u}_i^{n-1} + \hat{u}_i^{n-1}) \delta x + \frac{c}{3} (\hat{u}_i^n - \hat{u}_{i-1}^n) \delta t &= 0, \end{aligned}$$

which are abbreviated to  $\mathcal{L}'_h \mathbf{u}_h = 0$ , where  $\mathbf{u}_h = \{(\mathbf{u}_i^n) | 1 \leq i \leq N_x, 1 \leq n \leq N_t\}$  is the solution vector (with  $N_x$ , resp.  $N_t$ , the number of spatial cells, resp. time steps).

In order to solve the equations a pseudo-time derivative is added, properly scaled with the space-time volume:

$$\delta t \delta x \frac{\partial \mathbf{u}_h}{\partial \tau} + \mathcal{L}'_h \mathbf{u}_h = 0.$$

In order to make clear the relationship with the physical and pseudo time step, this equation is

rewritten as

$$\frac{\partial \mathbf{u}_h}{\partial \tau} + \frac{1}{\delta t} \mathcal{L}_h \mathbf{u}_h = 0,$$

with  $\mathcal{L}_h = \mathcal{L}'_h / \delta x$ .

The matrix  $\mathcal{L}_h$  is a matrix consisting of block  $2 \times 2$  matrices. All block diagonals are the same and equal to

$$\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} + \text{CFL}_f \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{3} \end{pmatrix}, \quad (8)$$

the first matrix comes from the time derivative, the second from the convective term.

The system is converged to steady-state using a low-storage Runge-Kutta scheme. Each stage of such a scheme consists of

$$\mathbf{u}_i^{n,s} = \mathbf{u}_i^{n,s-1} - \alpha_s \frac{\delta \tau}{\delta t} \mathcal{L}_i^n(\mathbf{u}_h^{s-1}), \quad (9)$$

where  $\delta \tau$  is the pseudo time step, and  $\mathbf{u}_h^{s-1} = \{(\mathbf{u}_i^n)^{s-1}\}$  is the solution vector at the previous stage.

Since the equations are solved for space and time simultaneously, there are two stability constraints, determined by the CFL numbers  $\text{CFL}_x = \frac{c \delta \tau}{\delta x}$  in the spatial direction and  $\text{CFL}_t = \frac{\delta \tau}{\delta t}$  in the temporal direction. The temporal CFL restriction is the most restrictive when the physical CFL number is less than one. For conventional schemes this time step restriction can be removed by treating the discretised time derivative implicitly (Melson et al. (Ref. 13)). The same approach will be taken here.

Remembering that the solution  $\mathbf{u}_i^n$  is a vector consisting of the cell average and the time gradient, implicit treatment of the diagonal term of the time discretisation will lead to

$$(1 + \alpha_s \frac{\delta \tau}{\delta t} A) \mathbf{u}_i^{n,s} = \mathbf{u}_i^{n,s-1} - \alpha_s \frac{\delta \tau}{\delta t} (\mathcal{L}_i^n(\mathbf{u}_h^{s-1}) - A \mathbf{u}_i^{n,s-1}), \quad (10)$$

where  $A$  is the matrix given in (8):

$$A = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}.$$

Equation (10) is obtained from (9) by adding  $\alpha_s \frac{\delta \tau}{\delta t} A \mathbf{u}_i^{n,s}$  to the LHS and  $\alpha_s \frac{\delta \tau}{\delta t} A \mathbf{u}_i^{n,s-1}$  to the RHS. This method will be referred to as the matrix-implicit Melson correction.



The Runge-Kutta scheme with and without matrix implicit Melson correction has been analysed for different physical CFL numbers. The amplification factor of the Runge-Kutta scheme is computed with standard Fourier analysis. The results are plotted in Figure 6. The amplification factor is plotted for different Fourier modes, in space and (physical) time. For a stable scheme all modes should have an absolute value of the amplification factor smaller than one. For convenience the unit circle is plotted in the figures. All analyses have been done with a pseudo CFL number of two ( $CFL = \frac{c\delta\tau}{\delta x}$ ), which is the stable CFL number for this Runge-Kutta scheme for spatial simulations.

From Figure 6 it is clear that both schemes are stable for  $CFL_f > 1$ . For  $CFL_f = 1$  the standard scheme without implicit terms is unstable. For  $CFL_f < 1$ , for which the implicit method is devised, the matrix-implicit scheme remains stable (note the different scaling of the axes for the unstable scheme). The instability of the Runge-Kutta scheme without the corrections is to be expected since the dominating pseudo-time step constraint is ignored.

Figure 7 compares the smoothing properties of the schemes for  $CFL_f = 0.01$ . For each scheme the maximum stable pseudo-time step is taken. The maximum eigenvalue for each Fourier mode is plotted. Five Runge-Kutta steps are taken to make the differences in damping properties more clear. It is clear from the figure that the matrix-implicit scheme has much better smoothing properties for the high spatial modes than the explicit scheme. This is of importance to the efficiency of the multigrid algorithm.

Hence the matrix implicit method improves the stability of the pseudo-time integration, by effectively making the stability of the scheme independent of the size of the physical time step. Moreover, the smoothing property of the scheme is improved.

#### 4.4.2 Restriction and prolongation operators, definition of grid levels

The multigrid algorithm is taken from Klaij et al. (Ref. 11). As the time-dependent Euler equations are hyperbolic in both space and time, the time derivative has the same behaviour as the convective operator with convection velocity equal to one. Hence there is no need to repeat the two-level analysis presented in Klaij et al. (Ref. 11) and Van der Vegt (Ref. 21). The multigrid algorithm is simply applied to four instead of three-dimensional flow simulations. It remains to extend the restriction and prolongation operators to four dimensions and to locally refined meshes.

The prolongation operator is defined as the  $L_2$ -projection of the coarse grid solution to the fine grid solution. In order of this definition to make mathematical sense, the coarse grid function space should be embedded in the fine grid function space. This embedding is satisfied for uniform meshes, but for curvilinear meshes this is in general not the case.

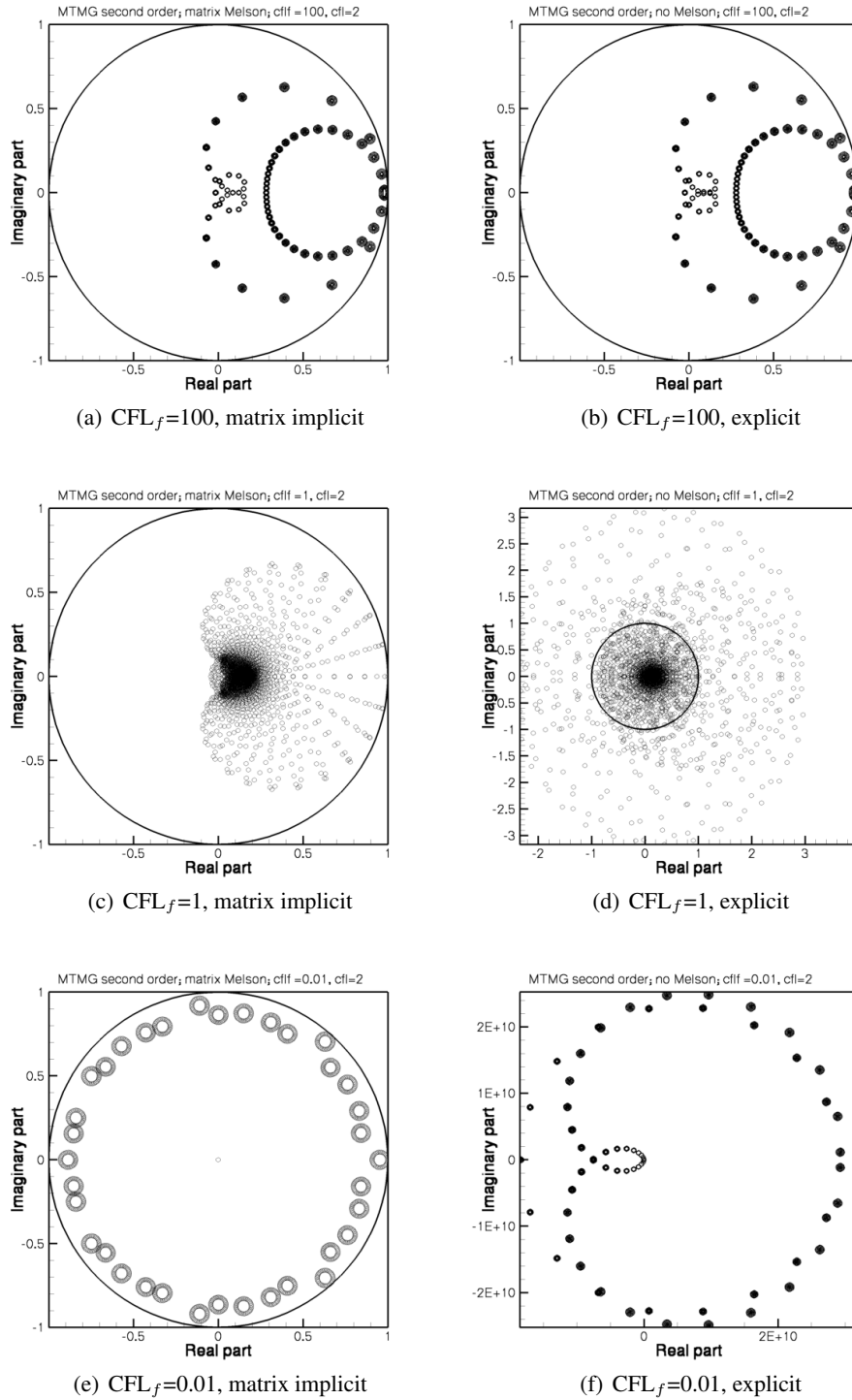


Fig. 6 Stability analysis of the space-time DG discretization of the convection equation for the RK5 scheme, with matrix-implicit or explicit treatment of the time derivative; for different physical CFL numbers. Shown is the amplification factor in the complex plane, together with the unit circle. All figures with  $CFL=2$ . Note the different scaling of the axes for the unstable methods.

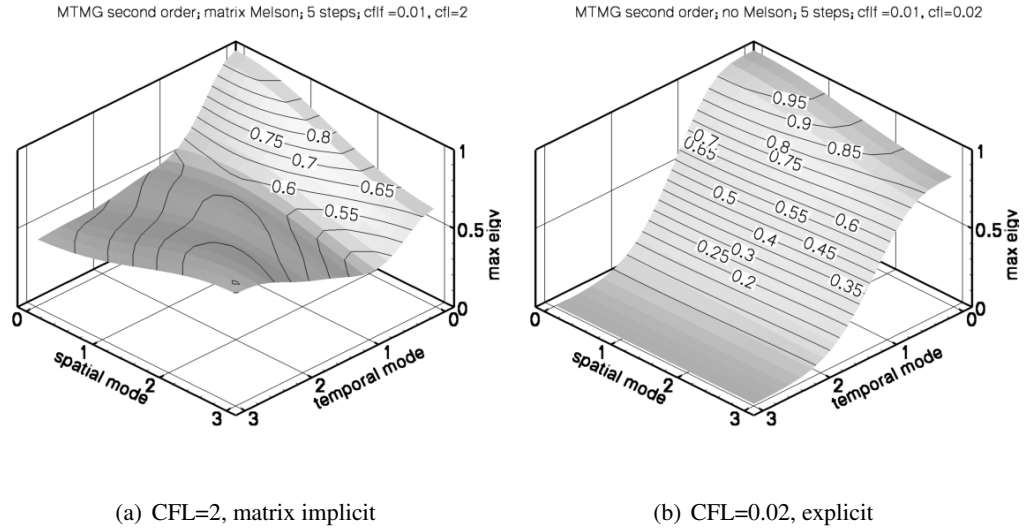


Fig. 7 Stability analysis of the space-time DG discretization of the convection equation for the RK5 scheme, with matrix-implicit, point-implicit, or no treatment of the time derivative; for physical CFL number equal to 0.01. The pseudo-CFL numbers are such that the scheme is stable.

On a one-dimensional mesh, a coarse grid cell  $K_H$  is the union of two fine grid cells  $K_h^\pm$  where the sign is determined by the coarse grid cell basis function:

$$\pm \phi_1^{K_H} |_{K_h^\pm} \geq 0.$$

The  $L_2$ -projection in one dimension on a uniform mesh is easily computed to be

$$\left( \begin{array}{cc|cc} 1 & -\frac{1}{2} & 1 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{array} \right), \quad (11)$$

which maps  $(\bar{U}^{K_H}, \hat{U}^{K_H})^T$  to  $(\bar{U}^{K_h^-}, \hat{U}^{K_h^-} | \bar{U}^{K_h^+}, \hat{U}^{K_h^+})^T$ . Note that the prolongation operator contains no geometrical details.

On locally refined meshes the coarse grid levels are obtained by agglomeration of fine grid cells using the grid refinement tree. The grid refinement tree keeps track of the grid refinements of a given cell. If that cell is refined anisotropically in one direction, two kid cells are added to the refinement tree at the cell's location. The kid cells are the newly created cells. The refinement tree also keeps track of the refinement direction. The agglomeration algorithm traverses the refinement tree backwards until a sufficiently large number of cells is agglomerated into a coarse grid cell. For the four-dimensional MTMG algorithm the number of agglomerated cells should

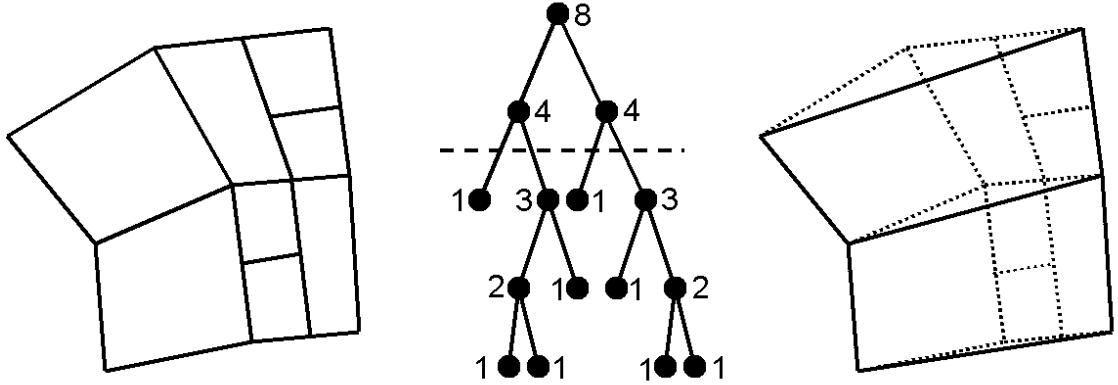


Fig. 8 Illustration of the agglomeration algorithm in two dimensions. To the left a group of 8 cells is shown, created by adaptation of an original group of four cells. The corresponding refinement tree is shown in the middle. Each node represents a cell in the refinement process, the numbers correspond to the number of leaves in the subtree starting at the node. The leaves of the tree correspond to the cells at the finest grid level. The dashed line in the refinement tree signifies the location where the refinement tree is pruned to obtain the coarse grid level. The resulting group of cells is shown to the right. Note that the orientation of the two coarse grid cells (horizontal in the figure) depends on the order of the refinements in the refinement tree.

be about sixteen ( $2^4$ ). Figure 8 illustrates the algorithm in two dimensions. Note that on locally refined meshes it is no longer ensured that a coarse grid cell is obtained by coarsening once in every direction. The advantages of this agglomeration algorithm are firstly that each agglomerated cell is a hexahedron, for which the DG discretization is defined, and secondly that the fine grid cells are defined with respect to the coarse grid cell as a sequence of local grid refinements in certain directions. This facilitates the definition of the prolongation operator: it is defined as the product of prolongation operators per refinement direction, corresponding to the sequence of grid refinements needed to obtain the fine grid cell.

To be more precise, given a coarse grid cell  $K_H$ , let  $C_l^\pm : K_H \mapsto K_h^\pm$  be the refinement operator which refines the coarse grid cell in the  $l$ -th computational direction ( $1 \leq l \leq 4$ ):

$$K_H = K_h^- \cup K_h^+ = C_l^-(K_H) \cup C_l^+(K_H), \quad \pm \phi_l^{K_H}|_{K_h^\pm} \geq 0.$$

The corresponding prolongation operator  $P_l^\pm : V_H^1(K_H) \rightarrow V_h^1(K_h^\pm)$  is defined as  $P_l^\pm(U^{K_H}) =$

$\sum_k p_{lk} \phi_k^{\mathbf{K}_h^\pm}$  where the fine grid cell expansion coefficients are given by:

$$\begin{cases} p_{l0} = \hat{U}_0^{\mathbf{K}_H} \pm \frac{1}{2} \hat{U}_l^{\mathbf{K}_H}, & k = 0, \\ p_{ll} = \frac{1}{2} \hat{U}_l^{\mathbf{K}_H}, & k = l, \\ p_{lk} = \hat{U}_k^{\mathbf{K}_H}, & \text{otherwise.} \end{cases}$$

Note that this prolongation operator is obtained by extending the operator defined in (11) to the identity for the gradients which are in the other directions than the refinement direction. For an arbitrary fine grid cell  $\mathbf{K}_h$  and a coarse grid cell  $\mathbf{K}_H$  which contains  $\mathbf{K}_h$ , there is a sequence of  $n$  refinement operators  $C_{l_i}^{s_i}$ ,  $1 \leq i \leq n$ , such that

$$\mathbf{K}_h = \left( \prod_i C_{l_i}^{s_i} \right) (\mathbf{K}_H), \quad 1 \leq l_i \leq 4, s_i \in \{\pm\}.$$

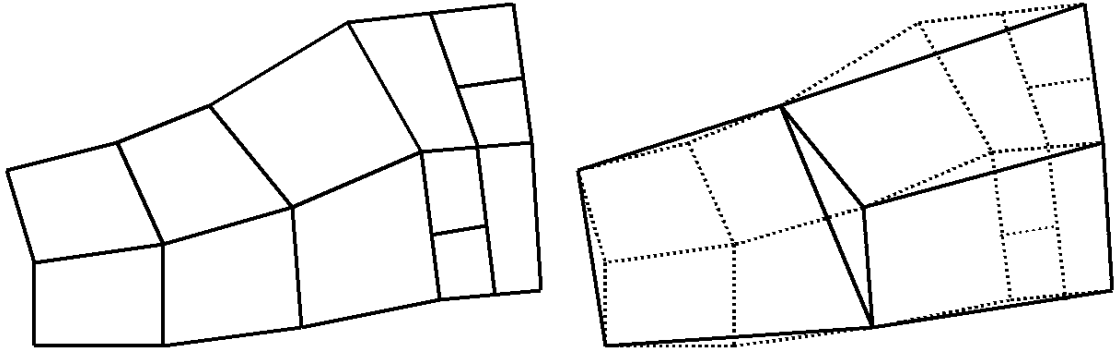
The prolongation operator  $P : V_H^1(\mathbf{K}_H) \rightarrow V_h^1(\mathbf{K}_h)$  is then defined as:

$$P = \prod_i P_{l_i}^{s_i}.$$

Since the prolongation operators in different computational directions commute, the definition of the prolongation operator is independent of the order of the refinement operators. Ignoring the geometrical details, such as skewness, in the definition of the prolongation operator is different from what is described in (Ref. 11). The main motivation to ignore the geometrical details is that multigrid algorithms concern the solution vector rather than geometry. Another motivation is that under the assumption that locally the coarse and fine meshes are geometrically similar, the prolongation operator does not contain geometrical details. Results in Section 6.1 will show that the algorithm with the simplified prolongation operators is effective on locally refined meshes.

The restriction operator for the residuals is defined as the transpose of the prolongation operator. The restriction operator for the solution vector is defined as the inverse of the prolongation operator. These definitions have been shown to be effective for the multigrid solution of the DG discretization of the laminar Navier-Stokes equations.

It should be noted that on locally refined, curvi-linear meshes the computational domain will in general not be a disjunct union of the coarse grid cells obtained from the above agglomeration algorithm. This is illustrated in Figure 9, where the agglomeration algorithm leads to coarse grid cells which do not completely cover the computational domain. The reverse, overlapping coarse grid cells, may also happen. The effect of this type of grid folding on the coarse levels is difficult to analyse (no need to mention that it would be disastrous on the fine level), but the forcing function in the multigrid algorithm contains the residual of the grid folding as well, and hence it is expected that the algorithm is robust. This will be demonstrated experimentally in Section 6.1.



*Fig. 9 Illustration of the grid folding caused by the agglomeration algorithm on curvi-linear, locally refined meshes. To the left a set of fine grid cells are shown: the fine grid cells of Figure 8 with a neighbouring group of four cells without refinement. To the right the resulting set of coarse grid cells is shown. The four cells are coarsened to a single cell, whereas the cells of Figure 8 are coarsened to two cells. The resulting hanging node is located on its original location of the curvi-linear mesh, and not on the straight edge of the coarse grid cell on its left. So the coarse grid cells do not cover the computational domain.*

#### 4.5 Grid design and expected speedups

Combination of the adaptivity of the DG algorithm and four-dimensional nature of the MTMG algorithm allows the design of grids in which the required resolution in the vortex core is obtained only when and where a vortex is present. As explained in Section 2.2, the accurate convection of the tip vortices requires a uniform mesh in both space and time within the vortex core. Let  $n$  be the number of cells which are required across the vortex core. For the third order DG scheme  $n = 7$ , and for the second order scheme  $n = 15$ , provide good estimates. With a core size of  $R/50$ , the mesh width in the vortex core is  $h = R/50n$  and the corresponding time step  $\Delta t = T/100n\pi\mu$  (see (1)).

In order to allow an accurate representation of the induced velocity field of the vortex, the region near the vortex core has to satisfy certain resolution requirements as well. The approach followed here is that in a region at a distance of  $R/25$  of the vortex core the resolution should be at least twice the resolution in the vortex core. This is repeat once more, in a region at a distance of  $R/12.5$  the resolution should be at least four times the resolution in the vortex core. The attained resolution in the latter region is thus about  $R/12n$ . Note that the time step grows proportionally with the mesh width.

The total number of grid cells in these regions is determined by the length of the tip vortex. A

good estimate of the convection velocity of the vortex is the freestream velocity  $u_\infty$ . The average distance a vortex must travel to move outside the rotor disk area is  $R$ , and hence the average time a vortex remains in the disk area is  $R/u_\infty$ . Since the vortices are created at the blade tips which travels at a velocity of  $\Omega R$ , the average length of the vortices within the rotor disk area is  $(\Omega R)(R/u_\infty) = R/\mu$ .

So the required number of cells  $N_{\text{core}}$  within the vortex core is equal to ( $V_{\text{core}}$  is the space-time volume of the core area)

$$\begin{aligned}
 N_{\text{core}} &= V_{\text{core}}/(h^3 \Delta t) = (R/\mu)(\pi(R/50)^2)T/((h^3 \Delta t)) \\
 &= \frac{\frac{R}{\mu} \pi \left(\frac{R}{50}\right)^2 T}{\left(\frac{R}{50n}\right)^3 \left(\frac{T}{100n\pi\mu}\right)} \\
 &= 2\pi^2 50^2 n^4
 \end{aligned}$$

So, for the third order scheme with  $n = 7$  the number of cells in the four-dimensional mesh in the vortex core alone is 118 million. For the second order method with  $n = 15$  it is 2.5 billion.

It is left to the reader to compute the number of cells in the two outer regions about the vortex. However, as both mesh width and time step are doubled, the number of cells is considerably less: the first region contains about 22 million cells for the third order scheme and the second region about 6 million cells.

The resolution outside these regions is much coarser. A generic spatial mesh about a four-bladed rotor with geometry refinement contains about 200,000 cells. In order to capture the blade-vortex interaction the time resolution of the geometry must be the same as the time resolution in the vortex core. So the space-time mesh outside the vortex regions but including the geometry contains  $200,000(T/100n\pi\mu)$  cells, which for  $n = 7$  is roughly between 20 and 40 million cells for realistic advance ratio's.

Although these numbers may seem staggering, they should be compared with the resolution required for a uniform mesh. The numbers are tabulated in Table 2. The mesh size of the uniform spatial mesh size is obtained by a uniform mesh with resolution  $R/50n$  in a computational domain of volume  $R^3 = (\pi R^2)(R/\pi)$ , the rotor disk with height  $R/\pi$ . Depending on the advance ratio, the four-dimensional meshes contain 20-50 times less cells than a uniform mesh contains. This is under the assumption that it is possible to generate such highly irregular meshes. This is the subject of the next chapter.

Number of cells across vortex core	7			15		
Advance ratio	0.15	0.25	0.35	0.15	0.25	0.35
Mesh width (fraction of blade span)	0.0028	0.0028	0.0028	0.0013	0.0013	0.0013
Time steps (in degrees azimuth)	1.1	0.66	0.46	0.51	0.31	0.22
MTMG mesh (in millions)	183	172	162	3130	3150	3170
Average spatial mesh size	2.0	1.3	0.93	18	11	7.7
Uniform spatial mesh size	43	43	43	422	422	422
Efficiency gain	22	34	46	24	40	55

Table 2 Estimated grid sizes and efficiency gains for a four-bladed rotor in forward flight with different advance ratio's. Mesh sizes in millions.

Although not the subject of the current notes, the benefit of a high order discretization method is clear from the above numbers. For the second order scheme, the number of expansion coefficients within a cell is five. So the MTMG mesh at an advance ratio of 0.15 contains 16 billion degrees of freedom per equation. The third order scheme has 15 expansion coefficients per cell (no tensor basis), so the MTMG mesh at the same advance ratio contains 2.7 billion degrees of freedom, six times less than the second-order mesh. A fourth order discretization has 35 expansion coefficients, and assuming  $n = 4$  for this scheme, the MTMG mesh contains 24 million cells, so only 0.8 billion degrees of freedom.

## 5 Adaptation strategies

### 5.1 Introduction

For industrial applications the most common grid adaptation strategy is to use sensors based on flow features. Shock sensors based on gradients in the flow have in general been quite effective in resolving shocks. For tip vortices in the rotor wake the effectiveness of feature-based sensors is not so clear. The straightforward vorticity magnitude sensor does not discriminate between tip vortices, vortex sheets or numerically induced boundary layers. More advanced sensors, such as the  $\lambda_2$  criterion (Jeong et al. (Ref. 9)), require significant resolution of the vortex to detect it. Such a resolution is not present on the initial coarse meshes. Because of these findings, it was concluded to opt for pre-adaptation, where the mesh is refined before the actual flow computation, based on the expected location of the tip vortices. This will be explained in detail later. In effect, the pre-adaptation strategy resembles unstructured grid generation more than conventional mesh refinement. It is comparable to the Chimera approach of Dietz et al. (Ref. 7), where



Chimera grids are constructed around the expected tip vortex locations. The benefit of our approach is that it is still based on the single-grid concept.

## 5.2 Adaptation on streak lines

By their nature, the location of the tip vortices is predominantly determined by the trajectory of the tip. So, to a very good approximation, the vortex trajectories can be taken to be the blade tip trajectories. The blade tip trajectories  $\Gamma(t)$  at time  $t$  are cycloids, described by

$$\Gamma(t) = \{(R \cos(\Omega(t - s)), R \sin(\Omega(t - s)), w(t - s)) + u_\infty s\},$$

where  $w = w(t)$  is the flapping motion of the tip.

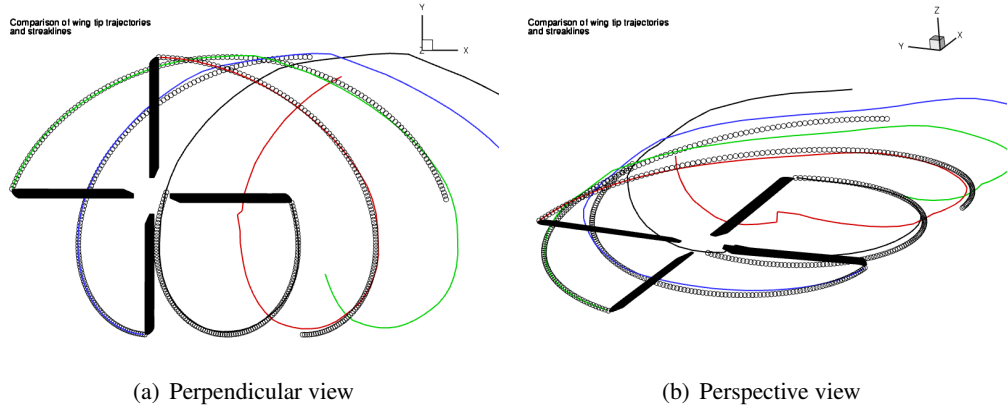
Based on this geometrical information, the mesh may be pre-adapted to increase the resolution near the tip trajectories. Whenever a cell is within a given distance of a tip trajectory and the mesh width or the time step is greater than a given threshold the cell is refined. Eventually a mesh is ‘generated’ with uniform space-time resolution in the expected vortex regions. The main benefit of this method is that there will be no refinement in other regions, for instance in the vortex sheets, hence the total number of grid cells is reduced compared to feature-based adaptation.

Of course, the assumption that the tip vortices follow the tip trajectories ignores the effects of downwash, contraction and interaction. Hence the free stream velocity in the definition of the tip trajectories  $\Gamma(t)$  is replaced by the actual, computed velocity field. That is, the blade tip trajectories are replaced by the streak lines of particles released at the blade tips.

In this way, an iterative pre-adaptive procedure is constructed, where the accuracy of the predicted tip vortex locations is increased with each iteration. The adaptation strategy has been applied to the simulation of a four-bladed rotor in forward flight. Details of the flow will be described in Section 6.2, but the adaptation strategy will be illustrated in this section for this simulation.

In Figure 10, a comparison is made between the blade tip trajectories and the streak lines of particles released at the blade tips. The difference increases with the age of the vortex. The main difference between the two methods is that the second accounts for the downwash of the rotor.

Let Mesh  $G_0$  be the initial mesh, generated according to Section 4.3.3 without local refinement for the vortices. Let  $G_1$  be the mesh the mesh obtained by pre-adaptation of  $G_0$  on the blade tip trajectories. On Mesh  $G_1$  a high-order flow solution is computed, and from this flow solution streak lines are computed which are used for the next pre-adaptation. This next pre-adaptation is



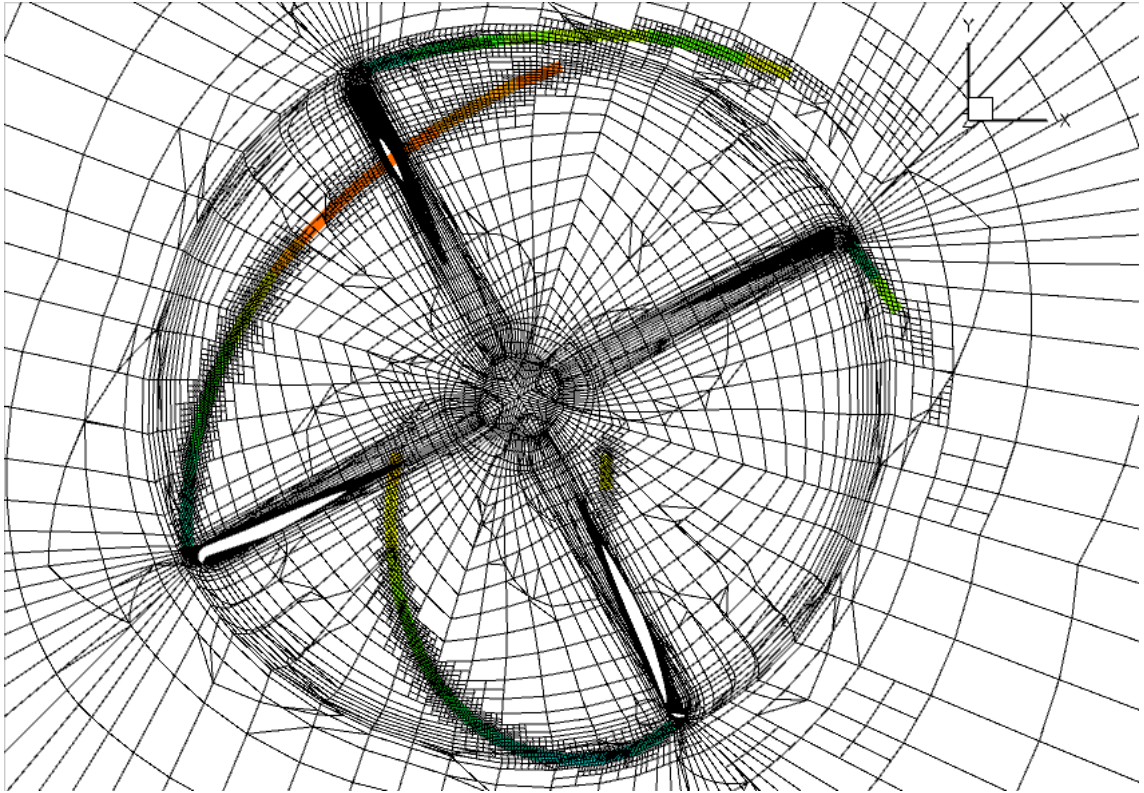
*Fig. 10 Difference between the blade tip trajectories (dots) and streak lines (coloured lines) for a four-bladed rotor in forward flight*

again executed on Mesh  $G_0$ , that is, the previous pre-adapted mesh is discarded. The mesh  $G_2$  is generated in several adaptation steps, where each step zooms in on the expected vortex location by reducing both the target mesh width and the region in which the grid is refined (consistent with the grid design described in Section 4.5). First the cells at a distance of  $0.1R$  are refined with the target mesh width  $0.02R$ . Both distance and mesh width are decreased proportionally, and the final step in the adaptation process has a target mesh width of  $0.007R$  at a distance of  $0.035R$ . In the simulation described in Section 6.2, the adaptation on streak lines is repeated once, and the final mesh  $G_3$  has 17.3 million elements. Due to memory restrictions on NLR's compute server, the resolution in the vortex core is restricted to three cells across the core. The simulation with a third-order DG scheme requires 60GB of memory.

The 17.3 million elements of Mesh  $G_3$  are equivalent to 260 million degrees of freedom per equation. Since the space-time mesh contains 64 time slabs, the average number of degrees of freedom per time slab is 4 million. This clearly is a modest number when trying to resolve the rotor wake.

It should be remarked that the target mesh width is not necessarily attained. A cell is refined whenever the mesh width is more than two times the target mesh width. Hence, on the final mesh, mesh widths will be between once and twice the target mesh width. Moreover, out of practical considerations, the mesh adaptation is stopped before saturation of the refinement criterion, and the quality of the mesh is judged by inspection.

Figure 11 and Figure 12 show Mesh  $G_3$  at two horizontal cross sections. The expected tip vortex locations are shown as curves, and blanked when the distance of the curve to the plane is greater



*Fig. 11 Illustration of the pre-adaptation strategy. An arbitrary horizontal slice of Mesh  $G_3$  is shown, together with the expected vortex locations as predicted by the streak lines. The vortex locations are coloured with their distance to the plane, and not shown if the distance is greater than  $0.035R$ . Clearly visible are the refinement regions near the expected vortex locations.*

than  $0.0375R$ , which is the user-defined distance within which cells should be refined. Figure 13 illustrates the pre-adaptation algorithm in a grid plane through the rotor axis and approximately perpendicular to the vortices.

It is clear from these figures that the mesh has been refined in a uniform way near the predicted vortex locations, and nowhere else. Hence the pre-adaptation algorithm is very effective in limiting the number of refined elements.

## 6 Results

### 6.1 Transonic oscillating NACA0012

In this section the MTMG algorithm will be applied to spatially two-dimensional simulations. The aim of the simulations is to demonstrate the feasibility of the MTMG algorithm for compu-

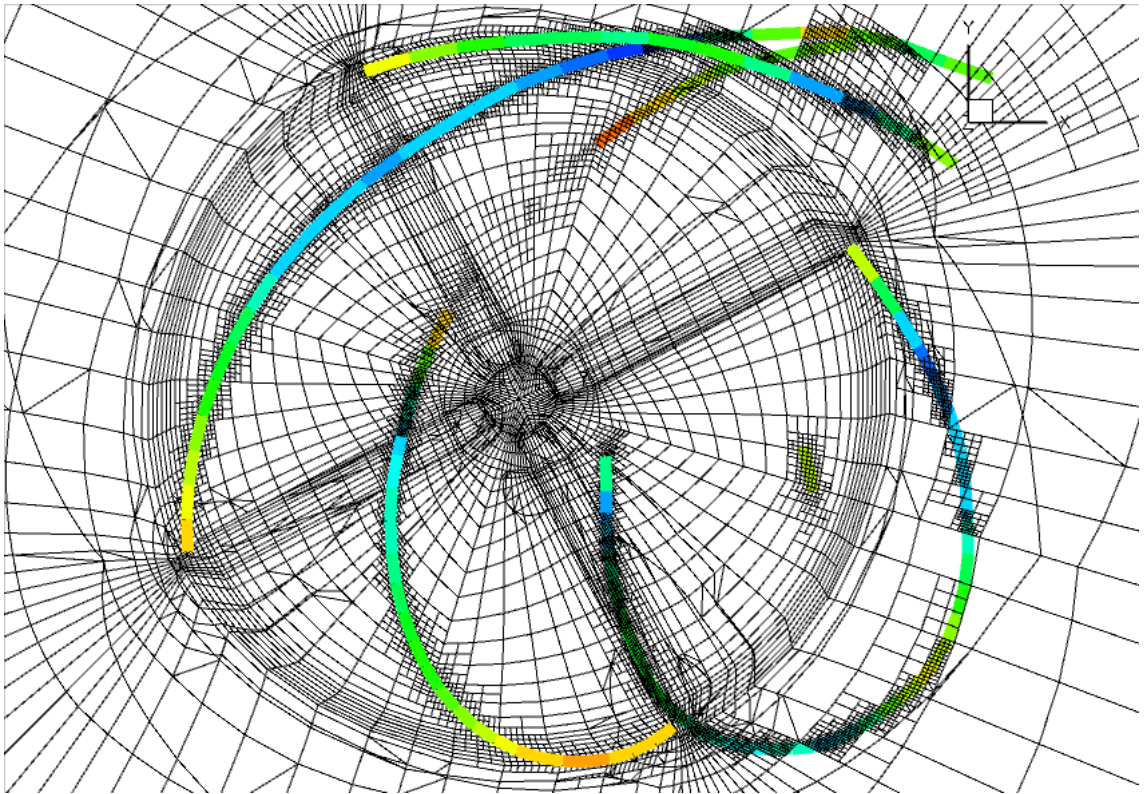


Fig. 12 As the previous figure, but a different horizontal plane, below the previous one

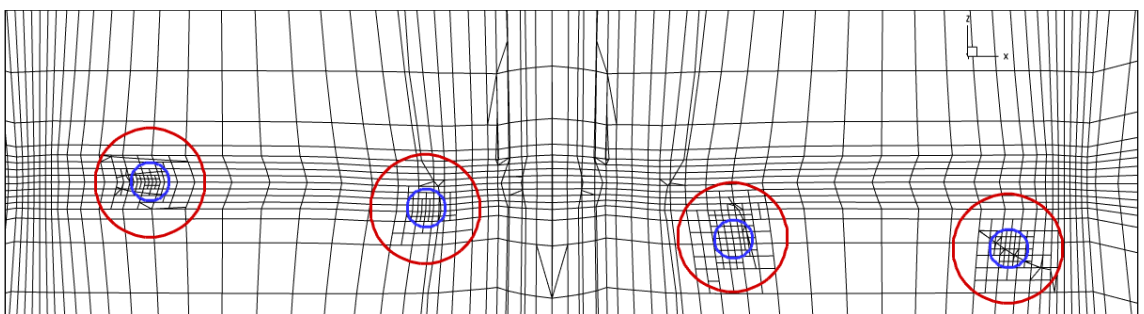


Fig. 13 Impression of the mesh obtained from pre-adaptation on streak lines. The grid plane shown is through the rotor axis (in the center of the figure) and almost perpendicular to the vortices. The red circles have a diameter of  $0.1R$  (within the circles the target mesh width is  $0.02R$ ); the blue circles have a diameter of  $0.035R$  (within the circles the target mesh width is  $0.007R$ ).

tations on locally refined meshes, such as are needed for BVI simulations. All simulations will concern an oscillating NACA0012 airfoil, either in subsonic or transonic conditions. The airfoil oscillates at a reduced frequency  $\omega^* = \frac{\omega c}{a_\infty} = 0.4917$  with an amplitude of three degrees around an angle of attack of zero degrees, where  $c$  is the chord length, and  $a_\infty$  is the freestream speed of sound.

The first simulation is a demonstration of the grid independence of the convergence rate. A three-dimensional mesh (two spatial and one temporal direction) with 32 time levels and 8192 spatial cells at all time levels is created. The mesh contains five multigrid levels with respectively 64, 512, 4096, 32768, and  $262144 = 32 \cdot 8192$  space-time cells. A full-multigrid simulation over the three finer levels has been conducted, using a V cycle with two pre-relaxations and two post-relaxations at each grid level (denoted as ‘2-2 cycles’ in the legend of the figures). At a Mach number of 0.5 the flow is subsonic.

The convergence of the four equations (cell-averaged, two spatial gradients and one temporal gradient) are shown in Figure 14. The dashed lines in the figure are parallel and it is clear from the figures that the asymptotic convergence rate obtained on the coarse mesh (over three grid levels) is also obtained on the medium mesh (over four levels) and the fine mesh (over five levels). The fact that the convergence rate is similar on all grids demonstrates that the multigrid algorithm is functioning properly. The rate of convergence itself is determined by the smoother, and improvements in the smoother, such as a diagonally implicit treatment of the linearised spatial fluxes, will be the subject of future work.

The next simulations concern the performance of the multigrid algorithm on locally refined meshes. Three three-dimensional meshes with partial time levels are generated starting from a coarse spatial mesh of 512 elements. The three grids contain 16, 32, resp. 64 time steps for a period, and three, seven, resp. fifteen partial time levels between full ones. Hence all grids contain four full time levels. The deformation region is extended half a chord length from the airfoil. Impressions of the grid are shown in Figure 15. Transonic simulations at a Mach number 0.8 are performed. As part of the simulation, local grid refinement with a standard shock sensor is applied, which measures the jumps over faces in a certain computational direction and anisotropically refines the neighbouring cells in that direction which connect to faces with the largest jumps. The time resolution is restricted to the one on the original mesh: cells may be refined in time, but not below the original time step size of 16, 32, resp. 64 to a period.

Each simulation on one of the three grids consists of a convergence to engineering accuracy on the original grid, six adaptations based on the flow solution to improve the shock capturing, and

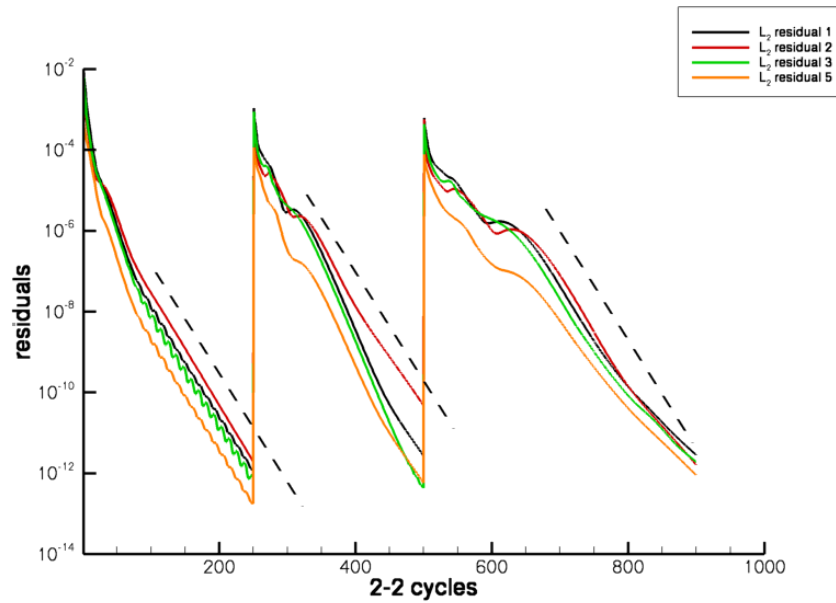


Fig. 14 Demonstration of grid independence of the convergence rate of the multigrid algorithm. Full multigrid simulation of a subsonic oscillating NACA0012 on a series of time-uniform meshes with 4096, 32768, resp. 262144 elements and three, four, resp. five multigrid levels. The dashed lines are parallel.

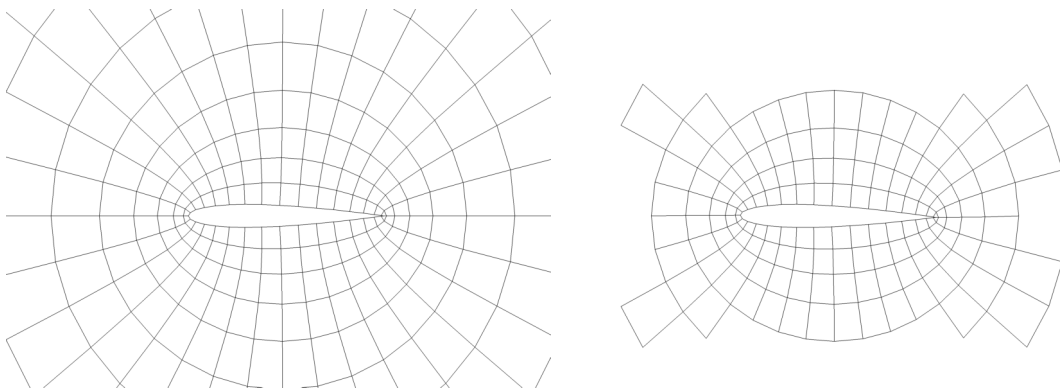
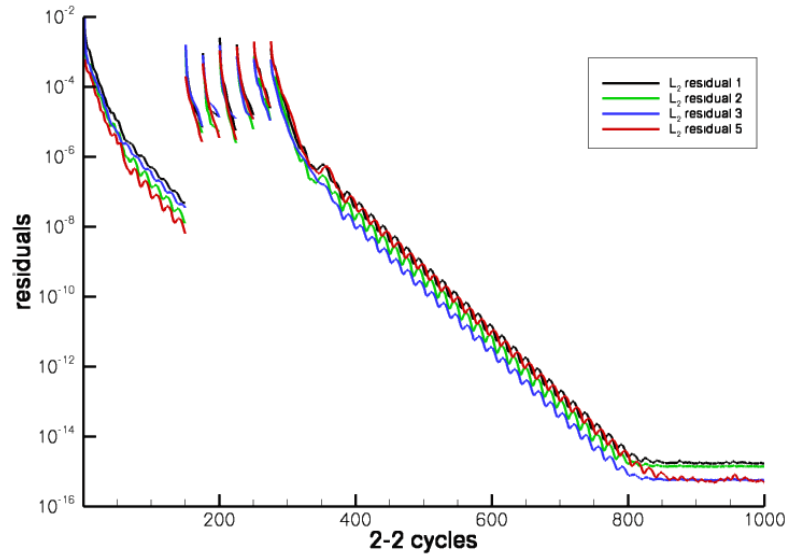


Fig. 15 Initial grid near the airfoil: full time level (left) and partial time level (right)





*Fig. 16 Convergence history of transonic oscillating NACA0012 with grid refinement. The oscillation period is divided into 16 time steps and the mesh contains three partial time levels between full ones.*

finally a convergence to machine accuracy. The convergence histories are shown in Figures 16 to 18. The histories look very similar despite the different number of time steps and partial time levels. Note that the convergence rate on the final, locally refined mesh, is the same as on the original mesh for all three grids. As remarked in Section 4.4.2 the coarse grid levels of the locally refined meshes will contain grid folding, but the convergence results show that the performance of the multigrid algorithm is unaffected by it.

Impressions of the locally refined mesh at various time levels for the case with 64 time steps are presented in Figure 19. It is clear that the shock sensor is capable of anisotropically refining the mesh in order to improve the shock capturing. Note that for the partial time levels (time step index not a multiple of 16) the greater part of the computational domain is now covered through local time refinement. Some gaps in the spatial domains are still visible. Note that local grid refinement only takes place at the shock positions. For a conventional time-serial adaptation algorithm this can only be accomplished using local refinement and de-refinement at each time step since the shock changes position. In order to simulate one period for this case, 64 grid adaptations would be required, whereas for MTMG only six adaptations are required.

The last simulation compares the performance of the multitime multigrid algorithm with the performance of a single grid algorithm and with the performance of the multigrid algorithm of Van

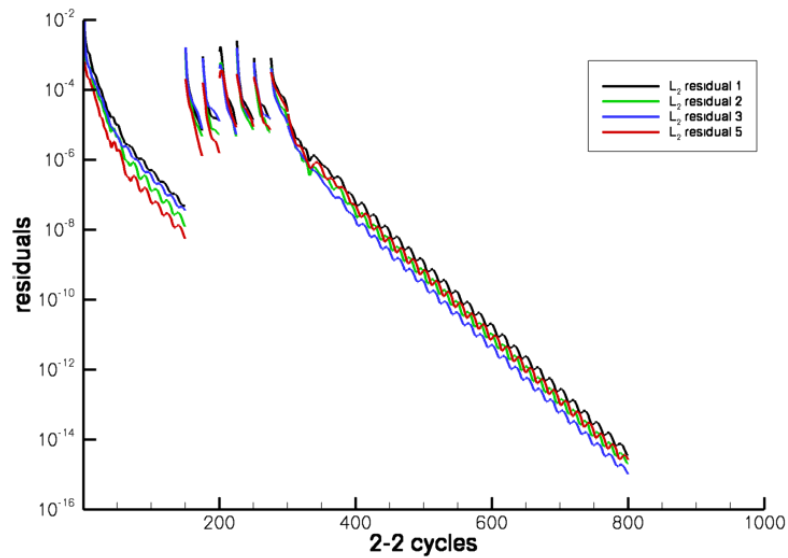


Fig. 17 Convergence history of transonic oscillating NACA0012 with grid refinement. The oscillation period is divided into 32 time steps and the mesh contains seven partial time levels between full ones.

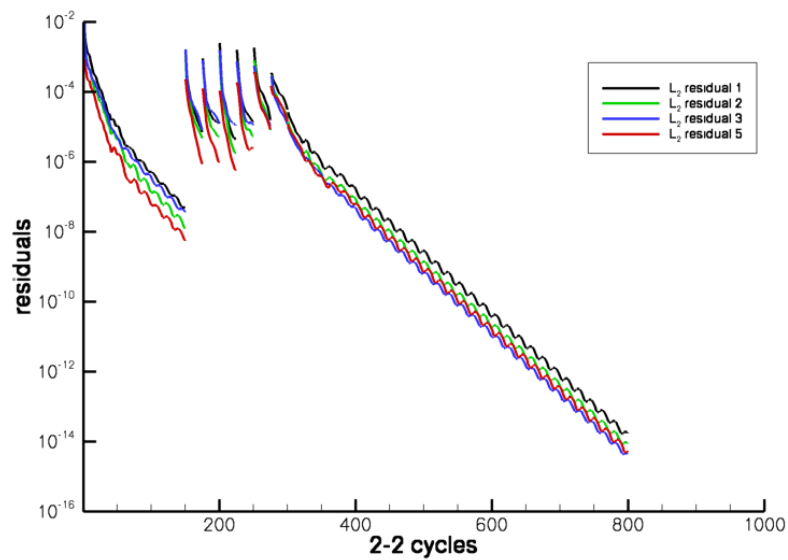
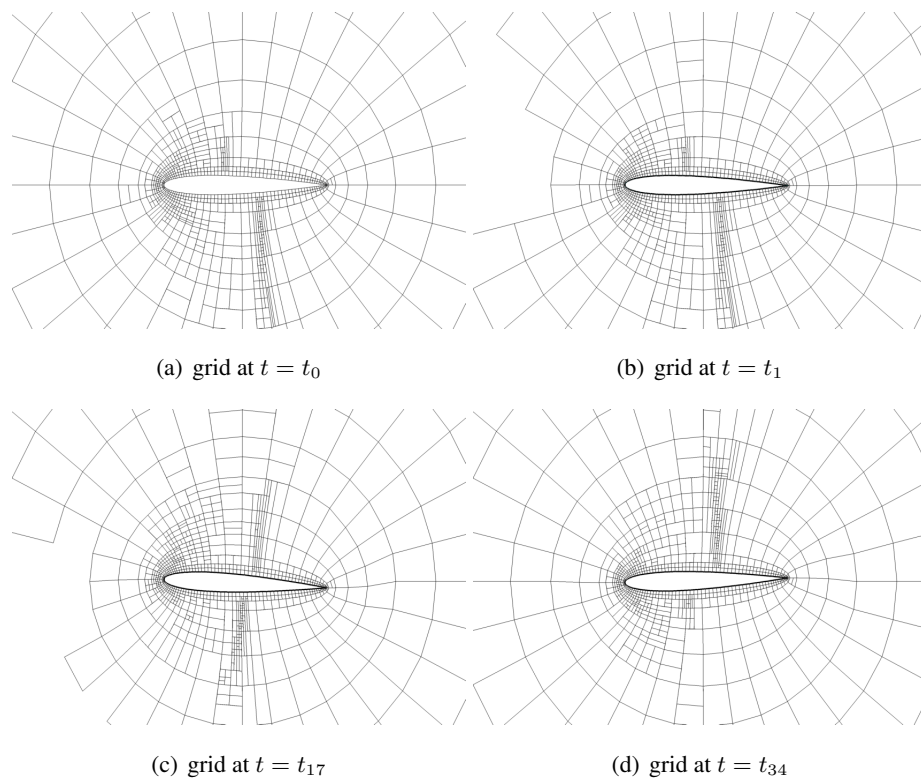


Fig. 18 Convergence history of transonic oscillating NACA0012 with grid refinement. The oscillation period is divided into 64 time steps and the mesh contains fifteen partial time levels between full ones.





*Fig. 19 Impression of grids for the transonic oscillating airfoil with 64 time steps.*

der Vegt et al. (Ref. 20) which does not solve all equations on the coarse grid levels, but only retains the equations for the cell-averaged solution. A transonic simulation is performed on the final locally refined mesh with 16 time levels obtained from the previously described simulations. This mesh is an extreme test case for any convergence acceleration algorithm.

The convergence history for the three methods is shown in Figure 20. The methods are compared with respect to the number of fine grid relaxations, which is a good metric for relative computational complexity. The work done on a coarse grid level by the multigrid algorithms is at most 1/16 of the work done on the fine levels, and as such is negligible. Clearly the multitime multigrid algorithm outperforms the other two. The MTMG algorithm is about four times faster than the single grid algorithm in obtaining a solution which is converged to machine accuracy. For practical applications, only engineering accuracy (three to four orders decrease in residual) is required. The MTMG algorithm obtains a residual level of  $10^{-5}$  in 200 fine grid relaxations, whereas the single grid computation requires 1500 fine grid relaxations. So for practical application the relative performance of the MTMG algorithm is a factor of eight. Compared to the multigrid algorithm using a first order discretization on the coarse levels the relative performance of the MTMG algorithm is more than a factor two for engineering accuracy.

## 6.2 7AD rotor in forward flight

Data point 135 of the Helishape wind tunnel program is simulated (Ref. 18). This data point concerns a high-speed level flight case at an advance ratio of 0.356 for the isolated 7AD1 rotor with parabolic and anhedral tip. The blade has a radius  $R$  of 2.1m and a chord  $c$  of 0.14m (aspect ratio  $R/c = 15$ ). In the experiment the rotor is trimmed for a thrust coefficient of  $C_T = \frac{F_z}{\pi \rho_\infty \Omega^2 R^4} = 0.0071$  and zero flapping.

The flow displays (weak) shocks at the advancing side and the vortex wake contributes to the vibratory loads at the rotor hub. Hence a CFD mesh must be generated capable of resolving shocks and vortices.

The grid design for this simulation has been described in Section 5.2. In the following the vortex system as computed by the third order DG method on Mesh  $G_3$  is described in detail. Other flow features, such as pressure distributions and forces, including validation, are described in (Ref. 22).

Figure 21 shows the vortex systems at an azimuth angle of 45 and 67.5 degrees, for second and third order solutions. Clearly, the second order simulations show little evidence of vortices at the shown vorticity levels: the vortices are present but weak. The third order solutions show much improvement over the second order solutions.

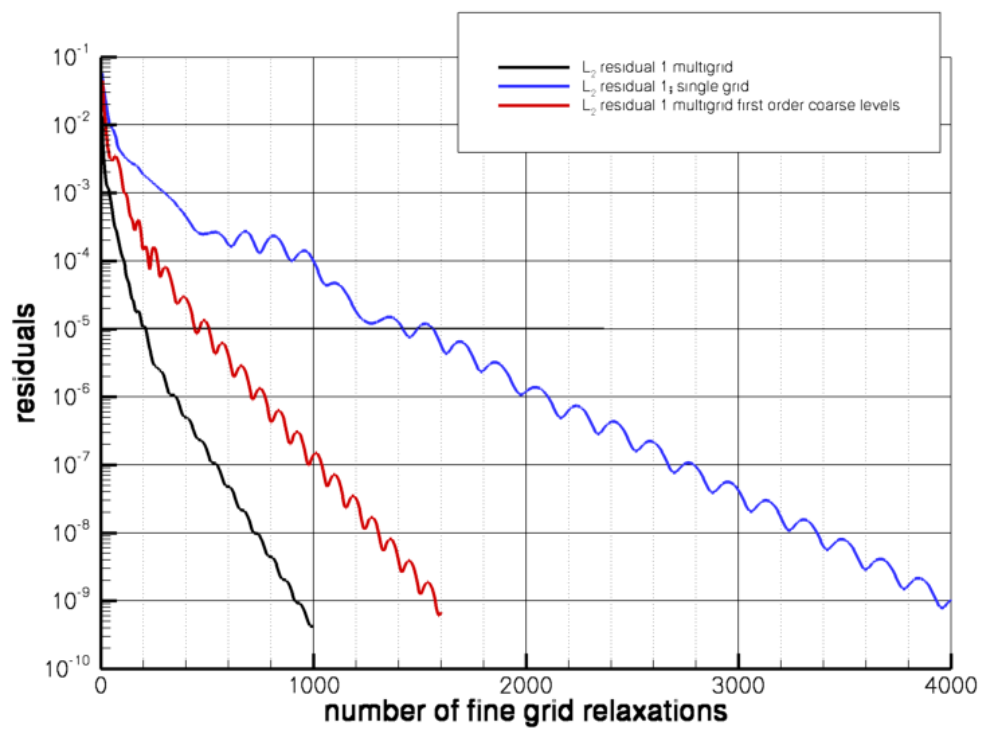
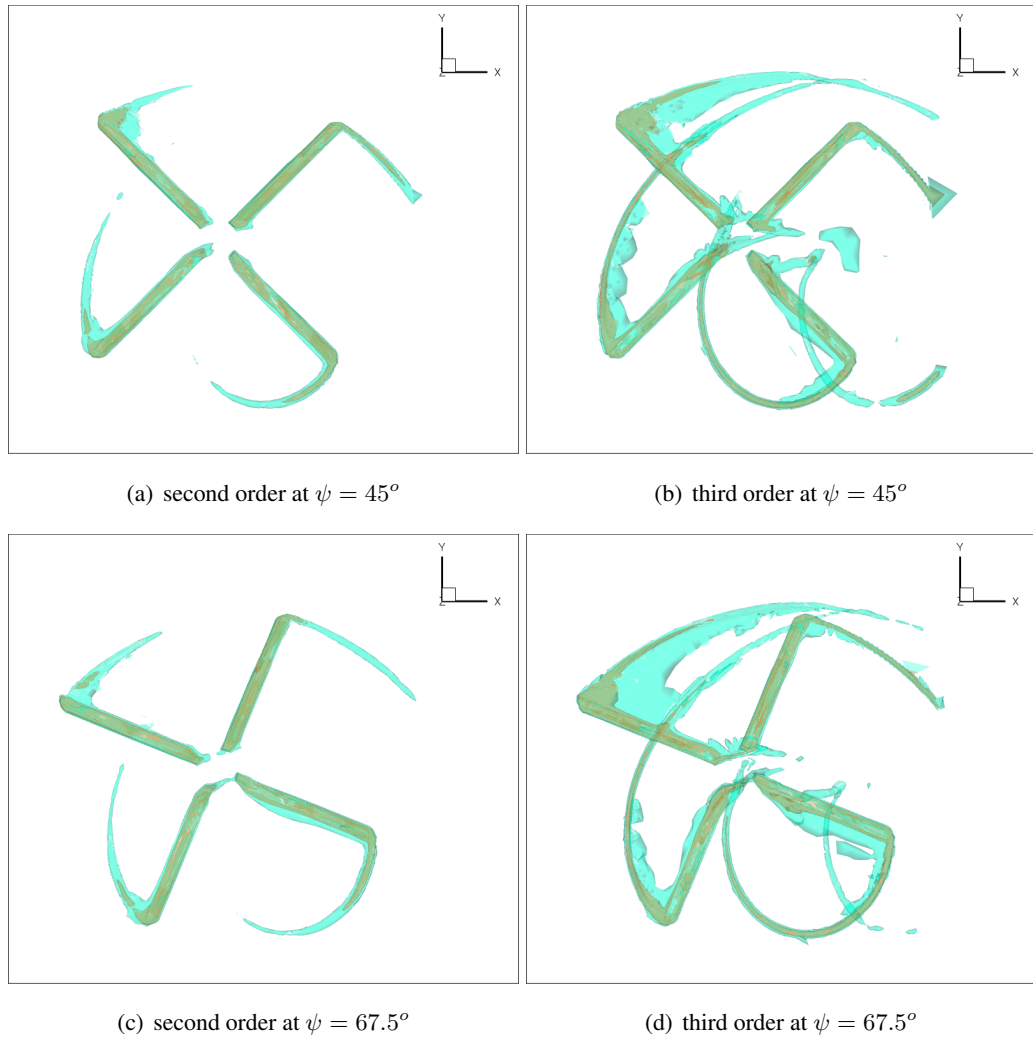
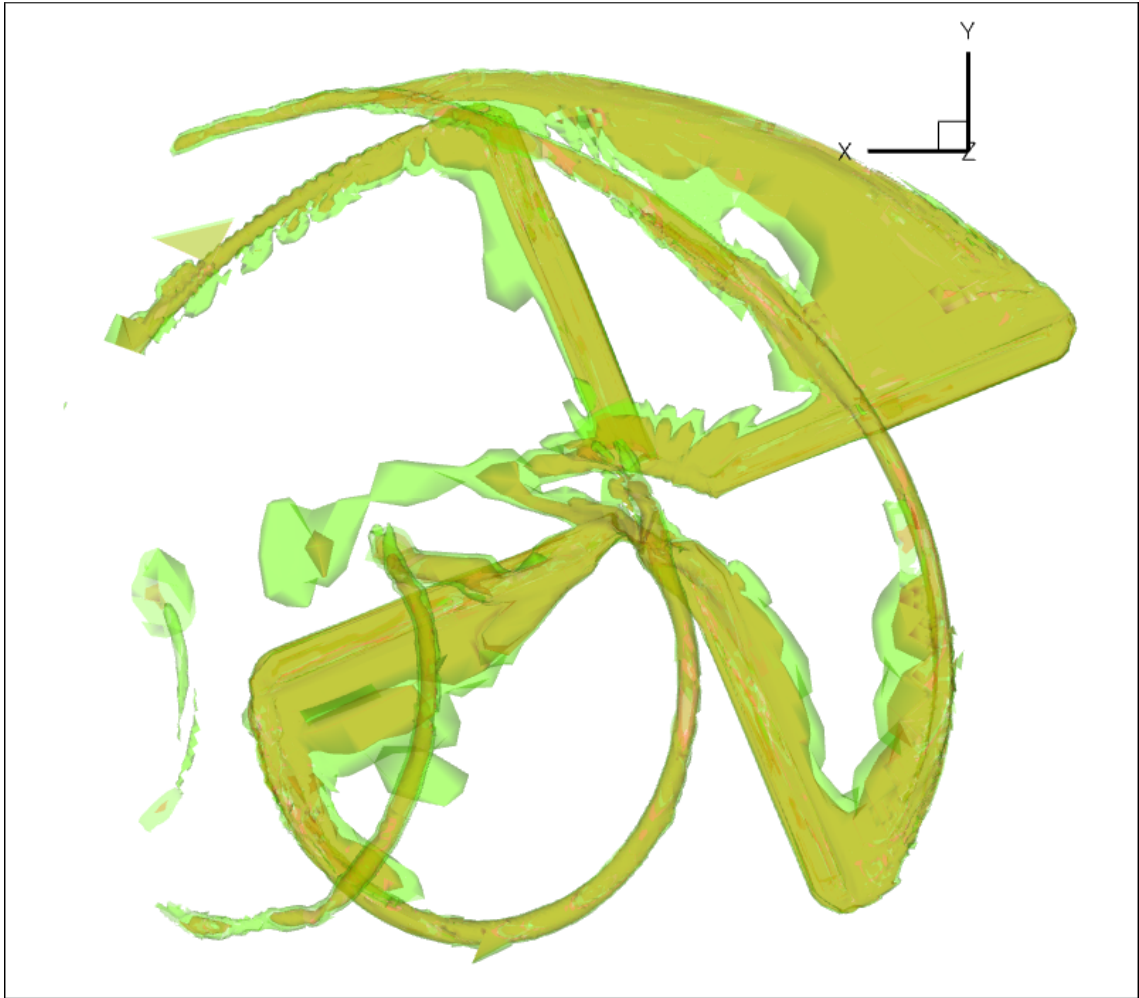


Fig. 20 Comparison of convergence history of transonic oscillating NACA0012 with grid refinement for the multigrid algorithm, the multigrid algorithm with first order discretization on the coarse grid levels and the single grid algorithm. The mesh is the final locally refined mesh containing 16 time levels.



*Fig. 21 Comparison of the vorticity contours for two orders of accuracy. Iso-contours are shown at vorticity magnitude level of  $2a_\infty/R$  and  $5a_\infty/R$ , where  $a_\infty$  is the speed of sound.*



*Fig. 22 The vortex system at  $\psi = 56^\circ$ , as obtained with the third order method, viewed from below. Iso-contours are shown at vorticity magnitude level of  $1.75a_\infty/R$  and  $1.25a_\infty/R$ , where  $a_\infty$  is the speed of sound.*

Figure 22 displays the vortex system, viewed from below, at an azimuth angle of 56 degrees for the third order simulation. Since the vortex wake is located below the rotor disk, this view presents a clear picture of the vortex system. At the retreating side (bottom of the figure), vortices of four different ages are visible. At the advancing side (top of the figure) only two vortices are visible. The advancing side vortices appear to be weaker and are destroyed by the root vortices at the hub.

In the following figures the vortex system is discussed in more detail. Each set of figures shows the vortex system from above (for example, Figure 23(b)), and in a cross-sectional vertical slice (for instance, Figure 24 and Figure 25). The figure with the vortex system viewed from above

also presents the location of the cross-sectional plane, including numbered labels at locations where the vortices intersect the plane. The numbers are repeated in figures of the cross-sectional slice, with an additional letter, signifying whether the vortex is a tip vortex ('T') or root vortex ('R').

In Figure 24, tip vortices of four ages are visible in the third order solution: T1, T2, T4, and T5. Note that these vortices are barely visible in the second order solution. Tip vortex T6 is the tip vortex with about the same age of vortex T1, and hence is much clearer. The root vortices R3 and R4 are located near the tip vortices T2 and T4, and since they are of opposite rotation they are likely to destroy the tip vortices. In reality the wake of the hub will have weaker interactional effects than the present root vortices. Comparing the vortex locations and grid resolution in Figure 25, shows that the grid has been refined in the correct locations.

Figure 27 shows the vortex system at a later stage and more on the advancing side of the rotor. The two tip vortices T1 and T2 are clearly visible in the third order simulation results, but the next tip vortex T3 is very weak. Note the different scale in the vorticity levels: the tip vortices emanating from the blades in the second quadrant are weaker than the vortices emanating from the blades in the third quadrant (compare Figure 24). As the blade loading is not significantly different for the two quadrants, this is most probably due by lack of resolution (see Figure 28).

Finally, two cross-sectional planes through the rotor blades are presented in Figure 29 at an azimuth angle of 80 degrees. The solution in the plane through the blades at  $170^\circ$  and  $350^\circ$  azimuth (Figure 30) shows a clear tip vortex T1, close to the blade, but further downstream the vortices are interacting with the root vortices R2 and R3. The solution in the plane through the blade at  $80^\circ$  azimuth (Figure 32) shows a clear vortex TI, but the next vortex TII has weakened and is located at a significant distance of the blade and will have little influence on the blade pressures.

For all the cross-sectional planes discussed above, the adaptation algorithm accurately refines the cells at the actual vortex locations (apart from the vortices emanating from the blade in the second quadrant, see Figure 33). In general one can conclude that the pre-adaptation on streak lines is an efficient and accurate way of generating meshes with sufficient resolution in the vortex regions.

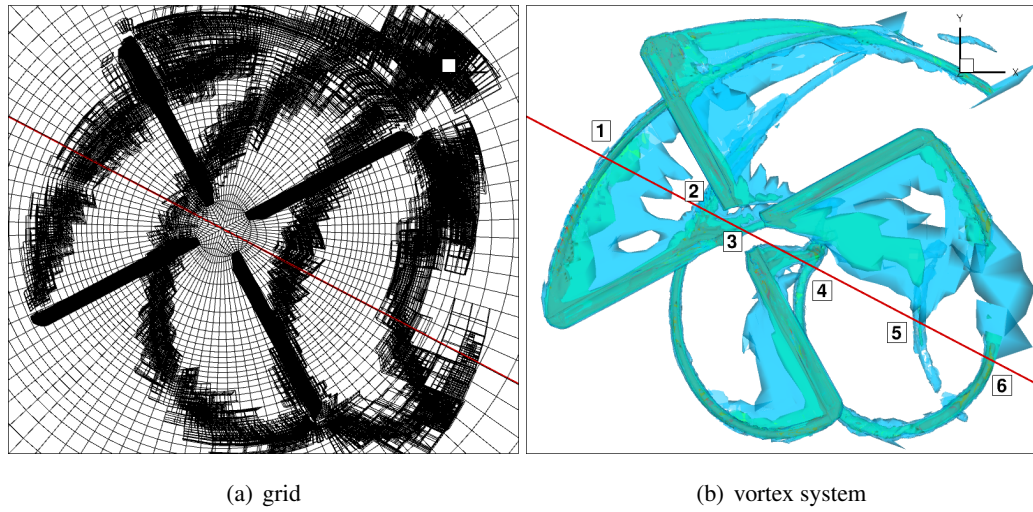


Fig. 23 Definition of the cross-sectional slice, shown in red, at  $\psi = 28^\circ$

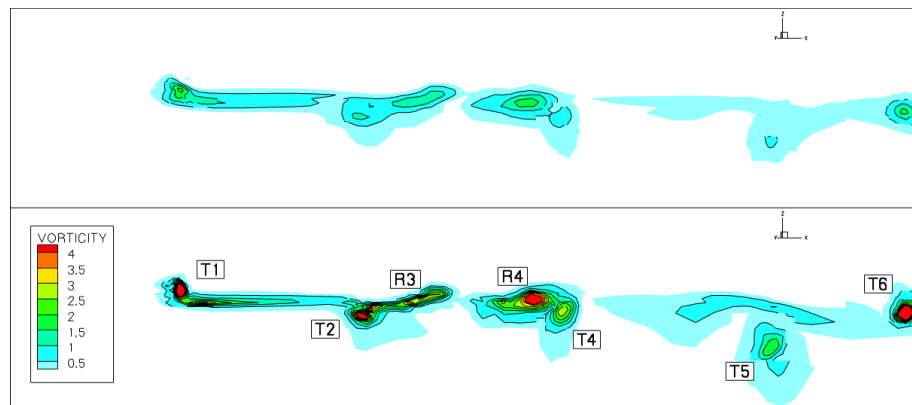


Fig. 24 Comparison of vortex resolution in the slice defined in Figure 23 with second order simulation (top) and third order simulation (bottom)

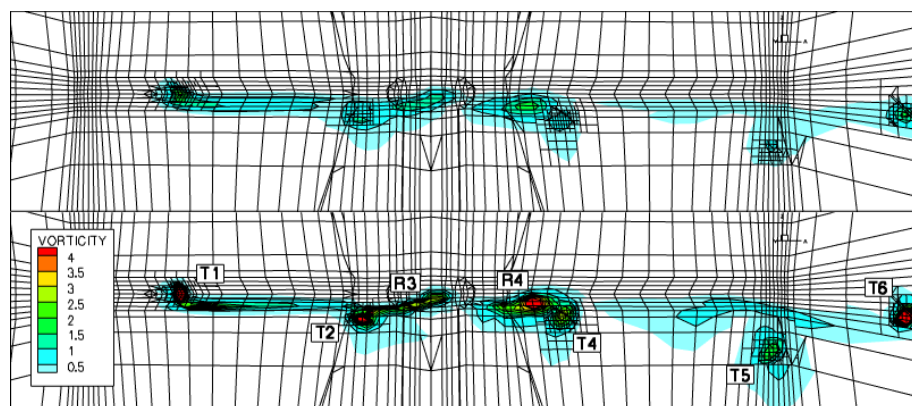


Fig. 25 Comparison of vortex resolution in the slice defined in Figure 23 with second order simulation (top) and third order simulation (bottom)



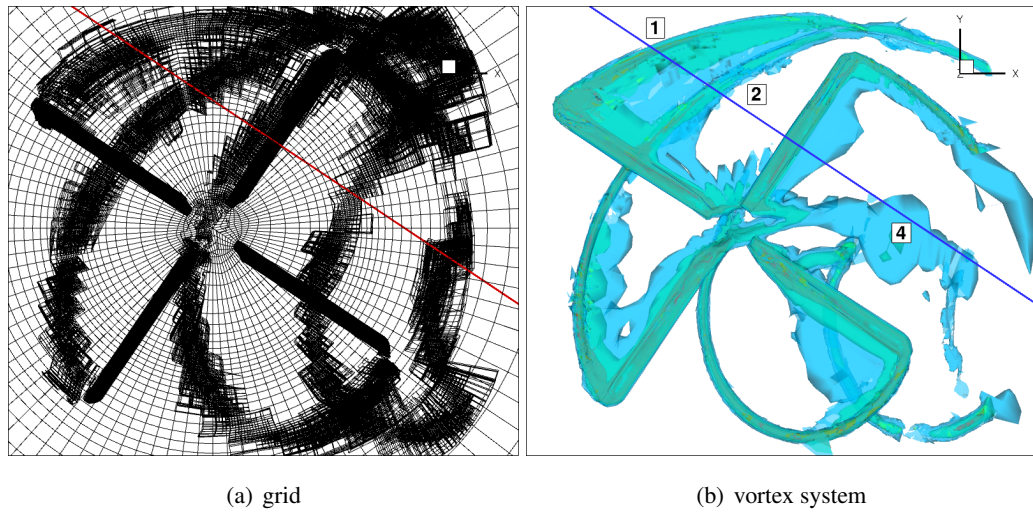


Fig. 26 Definition of the cross-sectional slice, shown in red, at  $\psi = 56^\circ$

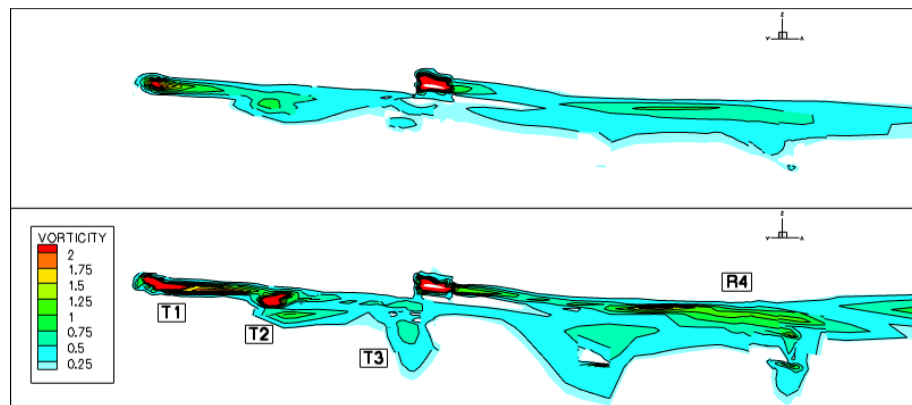


Fig. 27 Comparison of vortex resolution in the slice defined in Figure 26 with second order simulation (top) and third order simulation (bottom)

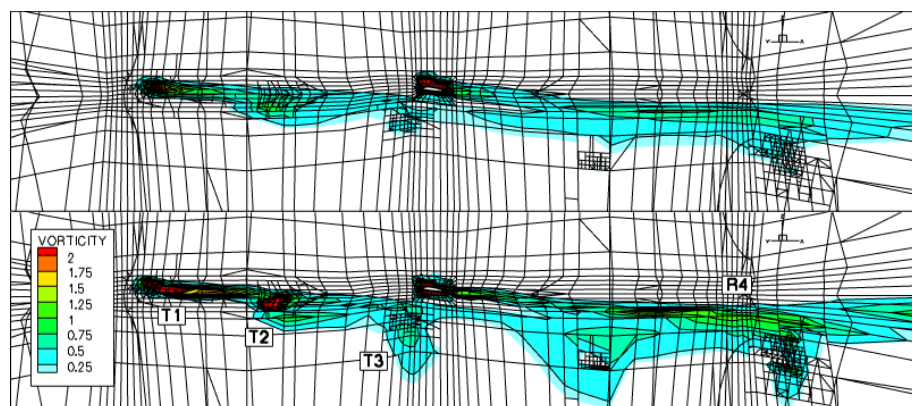


Fig. 28 Comparison of vortex resolution in the slice defined in Figure 26 with second order simulation (top) and third order simulation (bottom)



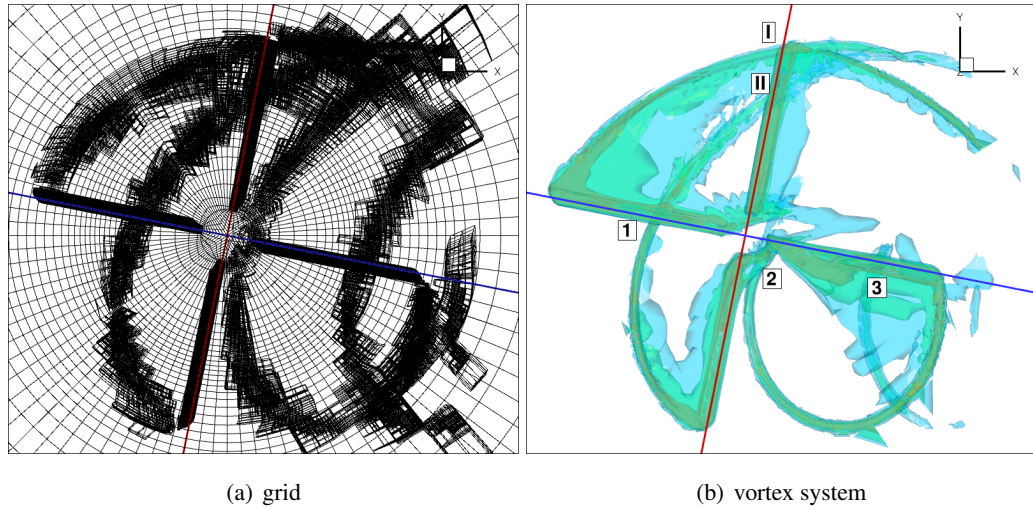


Fig. 29 Definition of the cross-sectional slices, shown in red and blue, at  $\psi = 80^\circ$

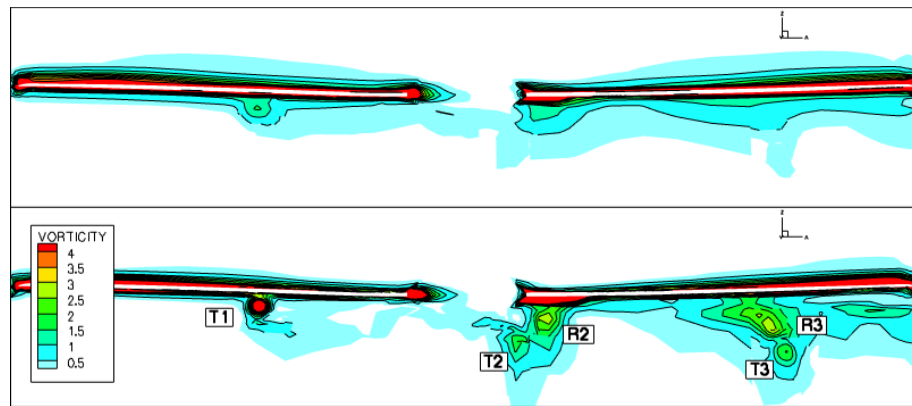


Fig. 30 Comparison of vortex resolution in the slice defined by the blue line in Figure 29 with second order simulation (top) and third order simulation (bottom)

## 7 Conclusions

These notes present a comprehensive and consistent approach to the simulation of rotor wakes using first-principles CFD. The multi-time multi-grid technique allows efficient computation of all physics involved in rotorcraft aerodynamics, such as trim, aero-elasticity, and the vortex wake. A feature-based pre-adaptation strategy has been described which aims at resolving the vortex wake. The techniques have been applied to the simulation of a rotor in forward flight. Given the limitations in computational capability, the necessary resolution cannot be obtained. Nonetheless, the vortex persistence is significantly improved by the local grid refinement.

For those who want to pursue this method, the following challenges are open:

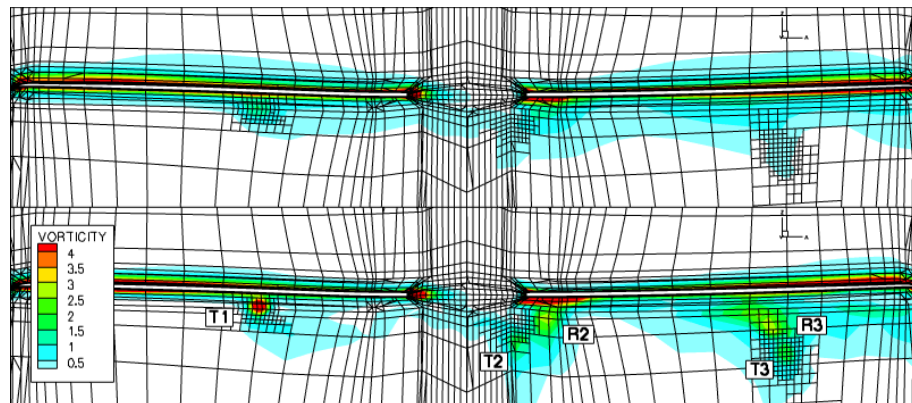


Fig. 31 Comparison of vortex resolution in the slice defined by the blue line in Figure 29 with second order simulation (top) and third order simulation (bottom)

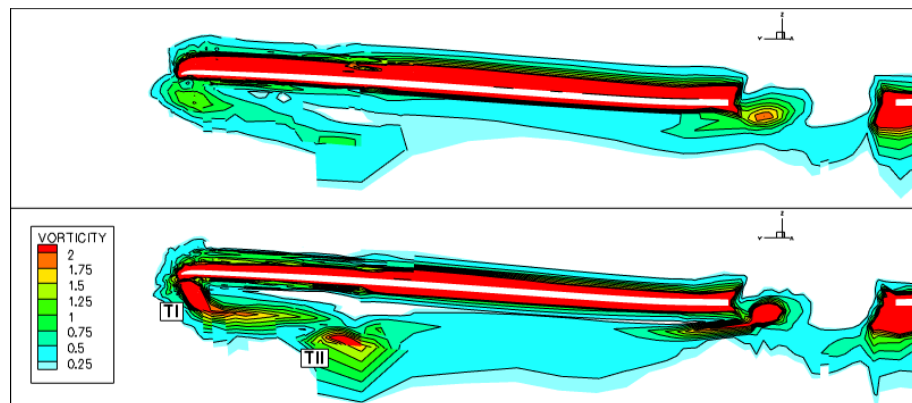


Fig. 32 Comparison of vortex resolution in the slice defined by the red line in Figure 29 with second order simulation (top) and third order simulation (bottom)

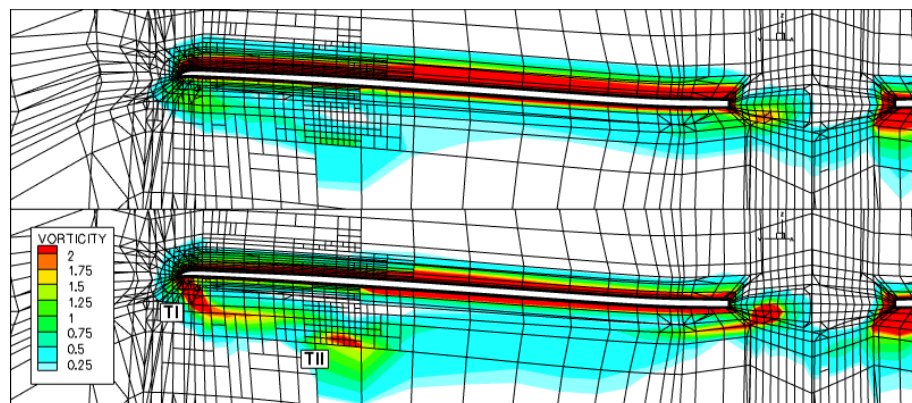


Fig. 33 Comparison of vortex resolution in the slice defined by the red line in Figure 29 with second order simulation (top) and third order simulation (bottom)

- Perform a simulation of a rotor in forward flight with adequate resolution;
- Investigate the possibilities of four-dimensional grid generation around a rotor-fuselage configuration;
- Develop a theory of space-time goal-oriented adaptation for rotor wakes.

### Acknowledgements

The work described in this paper is partially carried out within the European project ADIGMA on the development of innovative solution algorithms for aerodynamic simulations, and partially by NLR's programmatic research.

The contents of this report is based on the following previous publications (Ref. 2, 22, 23, 24, 25, 26)

### References

1. A. R. M. Altmikus, S. Wagner, G. Servera, and P. Beaumier. A comparison: weak versus strong coupling for trimmed aeroelastic rotor simulations. In *Proceedings of the 29th European Rotorcraft Forum, September, 2003*, 2003.
2. O. J. Boelens, H. van der Ven, B. Oskam, and A. A. Hassan. Boundary conforming discontinuous Galerkin finite element approach for rotorcraft simulations. *J. of Aircraft*, 39 (5):776–785, sep-oct 2002.
3. Carlo L. Bottasso and Mark S. Shephard. A parallel adaptive finite element Euler flow solver for rotary wing aerodynamics. *AIAA Journal*, 35 (6):937–944, 1997.
4. B. Buchtula and S. Wagner. Rotory wing aeroelasticity in forward flight with refined wake modeling. In *Proceedings of the 24th European Rotorcraft Forum, Marseille, September 1998*, 1998.
5. F. X. Caradonna. Development and challenges in rotorcraft aerodynamics. *AIAA paper*, 2000-0109, 2000.
6. A. Datta, M. Nixon, and I. Chopra. Review of rotor loads prediction with the emergence of rotorcraft CFD. *J. of the American Helicopter Society*, 52 (4):287–317, 2007.
7. M. Dietz, E. Kramer, and S. Wagner. Tip vortex conservation using on a main rotor in slow descent flight using vortex-adapted chimera grids. *AIAA*, 2006-3478, 2006.
8. K. Duraisamy and J. D. Baeder. High resolution wake capturing methodology for hovering rotors. *J. of the American Helicopter Society*, 52 (2):110–122, 2006.
9. J. Jeong and F. Hussain. On the identification of a vortex. *J. Fluid Mech.*, 285:69–94, 1995.

10. Wayne Johnson. *Helicopter Theory*. Princeton University Press, Princeton, New Jersey, 1980.
11. C. M. Klaij, M. H. van Raalte, H. van der Ven, and J. J. W. van der Vegt.  $h$ -Multigrid for space-time discontinuous Galerkin discretizations of the compressible Navier-Stokes equations. *J. Comput. Phys.*, 227:1024–1045, 2007.
12. Joon W. Lim and Roger C. Strawn. Prediction of HART II rotor BVI loading and wake system using CFD/CSD loose coupling. *AIAA paper*, 2007-1281, 2007.
13. N. D. Melson, M. D. Sanetrik, and H. L. Atkins. Time-accurate Navier-Stokes calculations with multigrid acceleration. In *Proc. 6th Copper Mountain Confer. on Multigrid Methods, NASA Conference Publication 3224*, pages 423–437, 1993.
14. A. Ochi, T. Aoyama, S. Saito, E. Shima, and E. Yamakawa. BVI noise predictions by moving overlapped grid method. In *Proceedings of the AHS 55th Forum, Montreal, Canada, May 25-27, 1999*, pages 1400–1413, 1999.
15. S. Piperno, C. Farhat, and B. Larroutrou. Partitioned procedures for the transient solution of coupled aeroelastic problems, Part I: Model problem, theory and two-dimensional application. *Comput. Meth. Appl. Mech. Engrg.*, 24:79–112, 1995.
16. H. Pomin and S. Wagner. Aeroelastic analysis of helicopter rotor blades on deformable Chimera grids. *J. of Aircraft*, 41(3):577–584, 2004.
17. L. N. Sankar, S. Y. Ruo, and J. B. Malone. Application of surface transpiration in computational aerodynamics. *AIAA paper*, 86-0511, 1986.
18. K.-J. Schultz, W. Splettstoesser, B. Junker, W. Wagner, E. Schoell, G. Arnaud, E. Mercker, K. Pengel, and D. Fertis. A parametric wind tunnel test on rotorcraft aerodynamics and aeroacoustics (helishape) — test documentation and representative results. Technical Report DLR-IB-129-96/25, DLR, 1996.
19. S. P. Spekrijse and J. W. Boerstoel. Multiblock grid generation. In *27th Computational Fluid Dynamics Course, Von Karman Institute for fluid dynamics, March 25-29, 1996*.
20. J. J. W. van der Vegt and H. van der Ven. Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows. Part I. General formulation. *J. Comput. Phys.*, 182:546–585, 2002.
21. J.J.W. van der Vegt. Discrete fourier analysis of multigrid algorithms. In *36th CFD/ADIGMA course on hp-adaptive and hp-multigrid methods*, 2009.
22. H. van der Ven. An adaptive multitime multigrid algorithm for time-periodic flows. *J. Comput. Phys.*, 227:5286–5303, 2008.
23. H. van der Ven and O. J. Boelens. Towards affordable CFD simulations of rotor in forward flight – a feasibility study with future application to vibrational analysis. In *proceedings of the 59th American Helicopter Society Forum, Phoenix, Arizona, USA, May 6-8, 2003*, 2003.

24. H. van der Ven and O. J. Boelens. A framework for aeroelastic simulations of trimmed rotor systems in forward flight. In *proceedings of the 30th European Rotorcraft Forum, Marseille, September 14-16, 2004*, 2004.
25. H. van der Ven and O. J. Boelens. High-order simulation of a rotor in forward flight using a four-dimensional adaptive flow solver. In *proceedings of the 34th European Rotorcraft Forum, Liverpool, September, 2008*, 2008.
26. H. van der Ven, O. J. Boelens, and B. Oskam. Multitime multigrid convergence acceleration for periodic problems with future applications to rotor simulations. In (*Ref. 30*), pages 355–363, 2002.
27. H. van der Ven and J. J. W. van der Vegt. Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows. Part II. Efficient flux quadrature. *Comput. Meth. Appl. Mech. Engrg.*, 191:4747–4780, 2002.
28. S. Wagner, A. R. M. Altmikus, and H. Pomin. Coupled numerical simulation of aerodynamics and rotor dynamics of a helicopter in forward flight. In *Proceedings of the Fifth World Congress on Computational Mechanics, Vienna, Austria, July 7th - 12th, 2002*, <http://wccm.tuwien.ac.at/>, 2002.
29. B. E. Wake and D. Choi. Investigation of high-order upwind differencing for vortex convection. *AIAA Journal*, 34, (2):332–337, 1996.
30. P. Wilders, A. Ecer, J. Periaux, N. Satofuka, and P. Fox, editors. *International Parallel CFD 2001 Conference*. North-Holland, Elsevier, May 21-23 2002.