NUMERICAL METHODS FOR CONTROL SURFACES AERODYNAMICS WITH FLEXIBILITY EFFECTS

Florian Blanc\(^1\), François-Xavier Roux\(^2\), Jean-Christophe Jouhaud\(^3\), Jean-François Boussuge\(^4\)

\(^1\)Ph.D. student, Aerodynamics Department, Airbus, and CERFACS (European Center for Research and Advanced Training in Scientific Computation)
Airbus France, 316 route de Bayonne, 31060 Toulouse, France
florian.blanc@airbus.com

\(^2\)Senior Researcher, ONERA/DTIM/HPC, Chatillon, France

\(^3\)Senior Researcher, CERFACS/AAM, Toulouse, France

\(^4\)CERFACS/AAM Team Project Leader

Keywords. chimera, aeroelasticity, time spectral method, flexibility effects

Abstract. A tool for computing control surfaces aerodynamics with flexibility effects is presented. This tool is based on the combination of three advanced numerical techniques: Chimera technique for modeling control surfaces, a dynamic aeroelasticity solver to handle flexibility effects and Time Spectral Method to reduce the computation times. All these techniques had to be adapted to work properly together. A Chimera grid assembly process was developed to treat control surfaces meshes in an aeroelasticity solver and it is shown how it was adapted to produce grid assemblies compatible with the Time Spectral Method. It is also presented how the aeroelasticity solver was modified to use the Time Spectral Method. To finish, an application of this tool to the computation of aircraft aileron aerodynamics with flexibility effects is presented.

1 INTRODUCTION

Improving the accuracy of tools used for aerodynamics computations is a key point in the design of even more efficient aircrafts. The Reynolds Averaged Navier Stokes (RANS) computational fluid dynamics (CFD) solvers that are nowadays widely used in the aeronautics industry are so accurate that taking into account multi-physics effect, such as aircraft flexibility, has become a necessity for going on increasing the accuracy of aerodynamics computations.

Another way to increase the accuracy of aerodynamics computations is to work on the capability of the flow solvers to take complex geometrical features into account. This requires to adapt the meshing techniques.

Control surfaces aerodynamics is at the meeting of these two trends. On one hand, control surfaces are complex geometrical features that require an adapted meshing strategy. On the other hand, aircraft flexibility plays an important role in their efficiency. In this paper, a solution is presented for computing aircraft control surfaces aerodynamics with flexibility effects. This solution is based on three numerical techniques that have been adapted and combined.

The ground element of this solution is elsA \([1]\) flow simulation environment. This compressible unsteady RANS (URANS) flow solver includes aeroelasticity functionalities \([2]\).
Figure 1: Overview of the key components of the solution presented for the computation of control surfaces aerodynamics with flexibility effects.

These features are used as a starting point to take flexibility effects into account. Two main problems had to be solved to make this tool fully operational for control surfaces aerodynamics computations with flexibility effects.

As elsA solver uses structured multi-blocks meshes, that are difficult to generate, complex geometrical details like control surfaces cannot be easily taken into account. To overcome this issue, the Chimera technique [3,4] (also called overset grid technique) has been used. This technique allows mesh blocks to overlap each other. This eases the mesh generation process by releasing the block matching constraint.

Independently from the meshing difficulties, a work had to be done on the numerical methods used for the aeroelastic computations. The algorithms implemented in elsA aeroelasticity module, initially developed for flutter and limit cycle oscillations computations, lead to high computation times on control surfaces simulations. To reduce these computation times a new time discretization algorithm, based on the Time Spectral Method (TSM) has been used [5].

The outline of the paper follows the main structure of the solution proposed for doing control surfaces aerodynamics simulations with flexibility effects (as summarized in Fig. 1): first the algorithm developed to simplify the use of Chimera technique is briefly presented. The second part presents how Chimera meshes are used in elsA aeroelastic solver. The third part shows how TSM was used to speed up aeroelastic computation and the fourth part explains how the Chimera grid assembly process is adapted to be used with the TSM. Lastly, an application to an aileron aerodynamics computation with flexibility effects is presented.

2 CHIMERA TECHNIQUE

2.1 Chimera technique description

The Chimera technique is based on two features:

- **Spatial interpolations:** when several mesh blocks overlap each other there is no coincident interfaces between these blocks. Thus, it is not possible to exchange directly fluxes between these blocks. Chimera technique proposes to exchange these
informations using spatial interpolations. On the boundaries where some flow information have to be obtained from other blocks, the values of flow variables are directly interpolated from the flow variables computed in other blocks.

- **Blanking:** within Chimera technique, the computation in some cells can be disabled. These are called *blanked* cells. When a region is blanked, its borders define new boundaries of the computational domain on which boundary conditions must be set. Chimera technique proposes to set these boundary conditions using interpolations.

These two features are illustrated on a simple two dimensional test case in Fig. 2.

![Figure 2: Example of Chimera grid: a spoiler on an airfoil. The blanked cells are hidden and the interpolated cells are in gray.](image)

To perform a computation on a Chimera mesh, all information describing the location of blanked cells and the interpolations have to be prepared. This pre-processing step is called *grid assembly*.

The simplest technique for doing this is based on the use of objects, called *masks*, defined by the user. Masks are geometric entities, like parallelepipeds or spheres, used to define regions of the computational domain in which the cells are blanked [6–8]. The Chimera mesh presented in Fig. 2 was assembled using a mask defined by the spoiler skin.

This technique suffers from an important issue. If the masks are not correctly defined, it may happen that no interpolation cells can be found for some of the cells which must be interpolated. These cells, called *orphans*, can make the computation impossible to perform. In this case, the user must modify the masks accordingly and restart the Chimera pre-processing.

This problem is even more difficult when Chimera technique is used for doing unsteady aeroelastic simulations. In these cases the Chimera grid assembly has to be re-computed during the simulation, while the bodies are moving. Then, it is possible that a mask correctly set at the beginning of the computation becomes wrong during the computation, causing a failure after a long computation time.

To tackle this issue, an automated grid assembly tool, which ensures that no orphan cells are produced, has been used.
2.2 Automated grid assembly

Automated grid assembly processes are numerous throughout the literature [9–14]. All of them are efficient but they are quite complex because they are designed to be able to handle all types of Chimera grid assembly.

Here, a new simple grid assembly process has been developed. This process was added to the Chimera functionalities already available in elsA [15]. It was specifically designed to process control surfaces meshes. Two types of Chimera grids can be distinguished:

- the background grid that meshes the aircraft on which the control surfaces are added,
- the patch grid that meshes the control surfaces.

For this type of meshes, the new grid assembly process automatically assembles the grid by blanking as much cells of the background grid as possible. The main idea used by this algorithm is to check, before selecting the blanked cells, that all the cells that must be interpolated, can effectively be interpolated. This ensures that there is no orphan cells in the grid assembly. Moreover the user workload for preparing the grid assembly is reduced: the user just has to identify which blocks are in the patch grid.

Fig. 3 presents a Chimera mesh assembled with the automated grid assembly algorithm. Note that, in this grid assembly, the overlap between the patch and the background grid is smallest than the overlap obtained with masks defined on the spoiler skin (to compare to Fig. 2). This reduced overlap is said to improve the computation convergence [9]. Moreover it simplifies the post-processing of the flow computed on this mesh (see following section).

3 USING CHIMERA MESHES IN AN AEROELASTIC SOLVER

The aeroelastic solver requires to compute the aerodynamic forces acting on the structure. This computation is straightforward if a coincident mesh is used for the CFD computa-
tion. In this case, the aerodynamic forces can be computed by summing directly all the elementary forces computed in each cell of the aerodynamic mesh.

When a Chimera mesh is used, some regions of the computational domain are computed by several meshes overlapping each other. In these regions, summing all the elementary forces available leads to an over-estimation of global aerodynamic forces.

To solve this problem, two main solutions are currently available:

- **The zipper grids:** the cells which are overlapping each other are removed. The resulting gaps are replaced by triangle cells automatically generated (see sketch Fig. 4). This results in a mesh with no overlaps, on which the aerodynamic forces can be computed like on a coincident meshes. Main software implementing the zipper grid strategy are FOMOCO [16] and MIXSUR/POLYMISUR [17].

- **Surface weighting:** The principle of this technique is to compute the forces using, as usual, a sum on each cell. In the regions of overlap, a coefficient that modifies the contribution of each cell is used. Thanks to this weighting, the global computed aerodynamic forces are right. The computation of these coefficients is based on a polygon clipping algorithm applied to the overlapping cells. USURP software implements this technique [18, 19].

![Zipper grids method](image)

Figure 4: Zipper grids method: first the overlapping cells are removed and secondly the gap are filled with triangle cells.

In the solver presented here, the surface weighting method was used because:

1. this strategy is easy to integrate in an existing aeroelastic solver designed for coincident aerodynamic meshes: the weighting coefficients just have to be added in the sum that computes the aerodynamic forces,
2. the generation of the zipper grids may suffers from robustness problems when highly stretched triangle cells have to be generated [18].

The surface weighting method has been used through USURP software [18, 19], developed at Penn State University and available under a collaborative, open source license. Most of the work for coupling USURP to elsA aeroelastic solver was done by ONERA/DADS team [20].

In the following, a validation of the use of USURP for computing aerodynamic forces in elsA flow solver is briefly presented. This validation is based on two computations:
1. a computation of an aircraft wing body configuration in a coincident mesh,
2. a computation on the same mesh in which Chimera boxes, containing only fluid, 
   were added on aircraft skin (see Fig. 5).

As the Chimera boxes contain only fluid, they do not modify the computed flow, thus 
forces computed in both cases should give the same results.

Convergence curves for computed lift and drag are presented in Fig. 6. First the over 
estimation of the aerodynamic forces computed on the Chimera mesh without correction 
can be clearly seen. With the use of the weighting coefficients processed by USURP, the 
aerodynamic forces are correctly computed.

![Figure 6: Convergence history of the computed lift and drag on the coincident mesh and on the Chimera mesh, with and without USURP correction.](image)

### 4 ACCELERATING AERODYNAMICS OF CONTROL SURFACE COMPUTATIONS WITH FLEXIBILITY EFFECTS USING TSM

As explained in the introduction, flexibility effects in control surfaces aerodynamics are 
taken into account using elsA aeroelasticity module [2]. Based on this tool, the computa-
tion process consists in running the simulation on an aircraft with a structural model on 
which a control surface is periodically moved. The computation of aerodynamics coeffi-
cients that characterizes the control surfaces aerodynamics requires to run the simulation 
until a periodic solution is found.
In practice, it appears that this technique leads to computation times too high for an industrial use. This is mainly due to the very long, aperiodic, transient state which has to be simulated before reaching a periodic state.

In order to accelerate the computation of control surfaces aerodynamics with flexibility effects, TSM has been applied to the aeroelastic solver. TSM has been developed in the field of CFD to accelerate the computation of periodic flows by reducing the time spent on computing the aperiodic transient [5,21–25].

In the following the TSM used in CFD is briefly presented. Consider URANS equations semi-discretized in space:

\[
\frac{\partial w}{\partial t} = R(w)
\]  

(1)

where \(w\) is the vector of conservative flow variables and \(R\) is the residuals operator. This equation can be written for a set of \(N_t\) time steps equally spaced in the period of the solution computed: \(W = [w_1, \ldots, w_{N_t}]^T\).

\[
\frac{\partial W}{\partial t} = R(W)
\]

(2)

Considering that the solution searched is periodic, \(W\) can be approximated by its Inverse Discrete Fourier Transform:

\[
\forall k \in [1, \ldots, N_t] \ w_k = \sum_{n=-\frac{N_t}{2}}^{\frac{N_t}{2}-1} \hat{w}_n e^{jkn2\pi \delta_t}
\]

(3)

Using this expression of \(W\) has two advantages:

1. it enforces the periodicity of the computed solution,
2. it ease the time derivation of \(W\).

Using equation (3) the time derivative of \(W\) can be written:

\[
\forall k \in [1, \ldots, N_t] \ \frac{\partial w_k}{\partial t} = \sum_{n=-\frac{N_t}{2}}^{\frac{N_t}{2}-1} 2\pi jk \hat{w}_n e^{jkn2\pi \delta_t}
\]

(4)

In this expression \(\hat{w}_n\) can be expressed as a function of \(W\) using the Discrete Fourier Transform of \(W\):

\[
\forall n \in [1, \ldots, \frac{N_t-1}{2}] \ \hat{w}_n = \frac{1}{N_t} \sum_{k=0}^{N_t-1} w_k e^{-j2\pi nk/N_t}
\]

(5)

This leads to the following formula for the time derivative:

\[
\forall k \in [1, \ldots, N_t] \ \frac{\partial w_k}{\partial t} = \sum_{m=1-\frac{N_t}{2}}^{\frac{N_t}{2}-1} d_m w_{k+m} = D_k W
\]

(6)

with:

\[
d_m = \begin{cases} 
\frac{\pi (-1)^{m+1}}{\sin(\pi m/N_t)} & \text{if } m \neq 0 \\
0 & \text{if } m = 0
\end{cases}
\]

(7)
The time discretization of equation (1) using TSM is then written:

$$D.W = R(W)$$

(8)

with \( D = [D_1, \ldots, D_{N_t}] \). Equation (8) is a system of steady RANS equations, each equation corresponding to a time step. These equations are coupled by the left hand side \( D.W \) that computes the time derivative of \( W \) using \( W \) values at each time steps.

The TSM procedure was applied to aeroelasticity equations. Consider the following equations of an aeroelastic system (semi-discretized in space):

$$\frac{dW}{dt} = R(W, \frac{dx^m}{dt}, x^m)$$

(9)

$$M \frac{d^2x^s}{dt^2} + C \frac{dx^s}{dt} + Kx^s = F(W)$$

(10)

$$x^m = \text{MeshDef}(x^s)$$

(11)

In Equation (9) \( x_m \) are the coordinates of the mesh used for flow computations. As an Arbitrary Lagrangian-Eulerian (ALE) [26] formulation of URANS equations is used, residuals depend on the mesh deformation speed, \( \frac{dx^m}{dt} \). Equation (10) are the structure equations. \( x_s \) is the vector of structure coordinates and \( M, C \) and \( K \) are respectively the inertia, damping and stiffness matrices of the structure. \( F \) is the operator that computes the aerodynamic forces acting on the structure. Lastly equation (11) represents the operator that deforms the fluid mesh according to current structural coordinates.

The aeroelasticity techniques available in elsA for solving these equations use a time marching scheme to discretize the time derivatives. With this scheme the computation of the periodic state of the system requires firstly to compute its transient.

Here the TSM time discretization operator was used to compute directly a periodic solution. This significantly reduces the time spent on computing transients.

Using the TSM time derivation operator leads to the following equations (with \( X^s = [x^s_1, \ldots, x^s_{N_t}]^T, X^m = [x^m_1, \ldots, x^m_{N_t}]^T \)):

$$\begin{align*}
(D.W) &= R(W, D.X^m, X^m) \\
MD^2\cdot X^s + CD.X^s + KX^s &= F(W) \\
\forall k \in [1, \ldots, N_t] \quad X^m_k &= \text{MeshDef}(X^s_k)
\end{align*}$$

(12)

(13)

(14)

These equations are a system of static aeroelasticity equations (one per time step) coupled by the terms corresponding to the TSM-discretization of the time derivatives. They are solved using a procedure based on the fixed point algorithm (FPA) used for solving static aeroelasticity problems [27]. This procedure reads:

1. Solve equation (12). To solve this equation an iterative procedure based on a pseudo time is used.
2. From the solution of (12) compute the aerodynamic forces acting on the structure at each time step.
3. Solve structure equations (13) to compute structure displacements at each time step.
4. If necessary, under-relax the computed displacements to ensure the stability of the whole procedure.
5. For each time step, deform the fluid mesh according to structure displacements.

TSM for aeroelasticity was implemented using the TSM features already available in elsA for turbo-machinery computations [28].
5 CHIMERA GRID ASSEMBLY FOR TSM

Motion of bodies requires to modify the Chimera grid assembly between two time steps. In particular, the set of blanked cells may change from one time step to the other. This raises a specific issue when TSM is used.

Equation (12) can be rewritten to explicitly show the computation done in each cell $l$:

$$\forall (k, l) \in [1, \ldots, N_t] \times [1, \ldots, N_{\text{cells}}]: \sum_{p=1}^{N_t} D_{kp} w_p^l = R(w_k)^l$$

(15)

The sum in equation (15) shows that the computation in cell $l$ uses the values of $w$ in this cells at all time steps. Thus, if at any time this value is not available because the cell is blanked, the computation is impossible.

Two solutions can be proposed to this problem:

1. **Modification of TSM equations:** the TSM time discretization process that results in equation (15) could be modified to be able to compute a value of a given time step without necessarily using the value at all other time steps. In this case, if some time steps are not used for the computation, the sampling will not be regular. As a result, the discrete Fourier transforms that have been used to establish the TSM time discretization, will be impossible to use. More sophisticated mathematical tools, such as wavelet analysis, that are able to deal with non-uniform samplings, would have to be used here.

2. **Constant blanking:** Chimera grid assembly can be done to ensure that the set of blanked cells is the same at all time steps. This requires to improve the grid assembly algorithm to satisfy this constraint. The main issue with this technique is that it cannot be applied to all cases. For example, in the case of an helicopter rotor mesh made of Chimera blade meshes moving in a background grid, no constant blanking can be found. However, in the cases considered here, control surfaces motion are relatively small (at least when compared to rotor blades motions) and these are moving around a mean position. Thus it can be guessed that a constant blanking can be found.

Considering that the first solution requires a deep modification of TSM equations, the second solution was chosen. A technique is presented to obtain automatically, when it is possible, an identical blanking for all time steps.

Globally, a two step procedure is used:

1. First a usual Chimera grid assembly is performed separately for each time step. Any Chimera grid assembly algorithm can be used here.
2. Secondly an algorithm is applied to modify the grid assembly of each time step to obtain a constant blanking.

To do so, the following iterative procedure is applied:

(a) The blanking of all time step are “summed”: if a cell is blanked at any time step then the algorithm blanks it at all other time steps.
(b) For some time steps, the previous operation will add new blanked cells. Around these new blanked cells, new interpolated cells are added to ensure that a fringe of interpolated cell remains available around all the blanked cell, at all time steps.
(c) For each new interpolated cells, interpolation cells are searched.
(d) It may happen that no donor cell can be found for some cells at some time steps. To solve this issue, the set of blanked cell is modified so that the cell that cannot be interpolated does not need to be interpolated anymore. In this case the algorithm goes back to step (2) and it iterates until donor cells can be found for each interpolated cells.

The result given by this algorithm is presented on a simple two dimensional test case consisting in an airfoil on which an aileron is added using Chimera technique (see Fig. 7). The Chimera grid was assembled using the automated grid assembly tool presented in this paper. The blanking at each time step obtained with this mask before the modification for TSM compatibility is shown Fig. 8 (left). The result of the algorithm for TSM compatibility applied on this case is shown in Fig. 8 (right). After the use of this algorithm the blanking obtained is the same for each time step.

![Figure 7: 2D Chimera airfoil with aileron. Top, the airfoil mesh. Bottom the aileron mesh.](image)

6 EXAMPLE OF AILERON AERODYNAMIC SIMULATION WITH FLEXIBILITY EFFECTS

The presented tool was applied to the computation of the unsteady flow over a moving aileron on a flexible aircraft. For the aerodynamic part of the computation, the URANS equations with Spalart-Allmaras [29] turbulence model have been solved on a mesh containing 16 millions of nodes. Chimera technique has been used for meshing the aileron (see Fig. 9 the Chimera mesh strategy used). For the structural part, a modal representation of the structure has been used. Only the 10 first Eigen modes of the full structural model were used. The ratio between the highest and the lowest Eigen-Frequencies kept was around 27 (see the two first Eigen-modes Fig. 10). Amplitude of the forced aileron oscillations was six degrees. The frequency of the oscillation was 94 percents of the first Eigen-Mode frequency.

The Chimera mesh was assembled using the automated algorithm described in this paper (see Fig. 11). This grid assembly was adapted for compatibility with TSM. TSM computations have been done using five time steps equally spaced in the period of the aileron
motion. This means that the computation of the dynamic aeroelasticity problem solved five coupled static aeroelasticity problems. 13 iterations of the static aeroelasticity FPA were necessary to complete the computation. CFD equations were not fully solved at each iteration of the FPA. Only 100 iterations of the CFD solver were performed at each FPA iteration. The CFD computation at a given FPA iteration was initialised with the result of the CFD computation of the previous iteration. With this strategy the overall CFD computation and the global aeroelasticity problem converged together. The complete computation, including Chimera grid-assembly, took around 20 hours on 64 POWERPC POWER5 processors.

Figure 12 plots the fluid density residuals curves of all CFD computations carried out by the FPA. This curve is an indicator of the convergence of the algorithm.

Figure 13 presents the computed position-velocity diagram of the structure coordinates along its first and second Eigen-modes. This shows how the aircraft is deformed by the unsteady aileron motions. The cycles computed at each iteration of the FPA are also plotted on this picture. A convergence in these computed cycles can also be observed.
Figure 10: View of the two first Eigen-Modes used for the structural model. (displacements magnified 10 times)

Figure 11: Grid setup achieved, left a cut of the background mesh and of the patch mesh on the right. (Blanked cell are hidden).

7 CONCLUSION

A solution for efficiently doing aircraft control surfaces aerodynamics simulations with flexibility effects has been presented. This solution is based on an original combination of some advanced numerical features:

- Chimera technique to simplify mesh generation for control surfaces, with an automated tool for recomputing the grid assembly during the aeroelastic simulation,
- an aeroelasticity solver for taking aircraft flexibility into account,
- TSM, to reduce computation times down to values compatible with an industrial use of the simulation tool.

The combination of these features raised some new issues to which solutions have been proposed, like for example, the use of Chimera technique in a TSM based solver.

This combination was made possible only because the features used here were all implemented in the elsA flow simulation environment. From this point of view, this work enlightens the significance for research teams to work in common simulation environment to be able to easily combine their developments to build complex solutions that meet industry needs.
Figure 12: Concatenated fluid density residuals of all CFD computations done at each iteration of the static aeroelasticity FPA.

Figure 13: Computed Position-Velocity diagrams for the two first structure Eigen-modes. Plain: result of the last coupling iteration, dotted: result at each iteration of the TSM-aeroelasticity procedure.

8 ACKNOWLEDGMENTS

This work is part of a thesis research funded by Airbus and done in strong collaboration with CERFACS. The author wishes to thank all the people of Airbus Aerodynamic Methods and Tools team in Toulouse. elsA developers from CERFACS and ONERA, and especially Bjoern Landmann, Guillaume Dufour, Frederic Sicot, are gratefully acknowledged for their help on the Chimera technique and on the TSM.

9 REFERENCES


