International Forum on Aeroelasticity and Structural Dynamics
IFASD 2024
17-21 June 2024, The Hague, The Netherlands

TOWARDS MODULAR AEROELASTIC SIMULATIONS: RECENT DEVELOPMENTS AND APPLICATIONS AT ONERA

A. Placzek[1], A. Riols-Fonclare[1], A. Dugeai[1], C. Liauzun[1], C. Blondeau[1], P.E. Des Boscs[1], C. Mollet[1]

[1] DAAA, ONERA, Institut Polytechnique de Paris, 92320 Châtillon, France

Keywords: computational aeroelasticity, fluid-structure coupling, solver, coupling library

Abstract: This paper details current work carried out at ONERA for the development of a modular framework dedicated to the resolution of aeroelastic problems. Historically, an aeroelastic module was implemented inside ONERA's CFD code elsA, that includes all the necessary components to perform aeroelastic analyses (mesh deformation methods, transfer of loads and displacements, specific simulations drivers). Aeroelastic simulations capabilities within this elsA's legacy aeroelastic module are however restricted mainly to the coupling with linear elastic structures and multiblock structured aerodynamic grids, as elsA was initially developed for structured meshes. To upgrade the aeroelastic coupling capabilities, some developments have been performed to externalize the different components outside elsA's kernel. This modular approach offers greater flexibility in terms of coupling and also makes it possible to work not only with elsA but also with other CFD codes, while reusing the same components for coupling. Some recent applications performed in this context will be presented in this paper and some perspectives for the development of advanced modular solvers are finally presented.

1 INTRODUCTION

The numerical simulation of aeroelastic phenomena involves the resolution of a coupled problem to describe the interaction between the fluid and the structure responses. This coupled problem can be dealt with in several ways, considering either a monolithic approach where a single solver is used to model both the fluid and the structure, or a partitioned approach where dedicated solvers are implemented for each physics. The last option is often preferred when complex physical phenomena have to be modeled in each physics but the coupling between the different solvers has then to be addressed to enable a proper exchange of data between the solvers.

This partitioned approach has been implemented by ONERA's Aeroelasticity Modelling and numerical Simulation research unit several decades ago in elsA's CFD software [1] [2] [3] to predict the aeroelastic behavior of aeronautical structures in various application areas such as military and civil aircraft, turbomachinery, propellers or helicopters.

The standpoint was that for transonic compressible flows, complex aerodynamic phenomena required a modelling by (U)RANS equations, but a basic modelling of the structure by a simple linear elastic model was considered to be sufficient to predict the structural behavior. This basic structural model has then been implemented historically in elsA's kernel in a dedicated aeroelastic

module. Although the structural solver is implemented in elsA, it is nevertheless independent of the CFD solver and the aeroelastic problem is treated with a partitioned approach. Therefore, elsA's aeroelastic module contains also all the necessary components to perform aeroelastic analyses, including some components for mesh deformation, transfer of loads and displacements between non-matching fluid and structural meshes, and specific simulations drivers to deal with several types of aeroelastic simulations [2] [3].

Aeroelastic simulations capabilities with elsA's legacy aeroelastic module are therefore restricted mainly to the coupling with linear elastic structures, either represented by a condensed Finite Element model in the static case or a modal representation in the dynamic case. Besides, the aeroelastic module was also implemented in such a way that only structured aerodynamic grids can be considered since elsA was originally developed in this context.

In today's climate context, future generations of aircraft and aeronautical engines must reduce their environmental footprint by cutting down their fuel consumption and CO2 emission. To meet these requirements, new configurations are emerging, involving lighter and more slender structures, that are consequently more flexible, and therefore more prone to aeroelastic phenomena involving non-linear structural responses. Besides, the assessment of overall aircraft or engines performance increasingly needs to take account of technological and integration effects. This means that complex geometries have to be dealt with, for which a discretisation with structured mesh topologies becomes very difficult or even impossible to deal with.

An evolution of elsA's legacy aeroelastic module was therefore required to tackle these difficulties concerning the modelling of non-linear structural effects and the capabilities to deal with unstructured grids. An additional constraint is also to ensure the compatibility of these evolutions for the new generations of CFD solvers which are developed at ONERA, like CODA [4] and SoNICS [5] (successor of elsA).

The coupling between the CFD codes of interest and a more sophisticated structural solver could be addressed with existing coupling libraries such as the following, to name a few:

- preCICE [6] developed by the German universities of Stuttgart and Munich, and dedicated to partitioned multi-physics simulations, including, but not restricted to fluid-structure interaction and conjugate heat transfer simulations

- ParaSiF_CF [7] developed by University of Manchester and dedicated to the resolution of massively parallel partitioned fluid-structure interaction problems, based on the MUI interface

- CWIPI [8] developed by ONERA and dedicated to the resolution of multi-physics problems in parallel distributed environments.

From a more general perspective, multi-physics platforms are also being developed to address very general coupling problems and improve interoperability in an HPC context among software libraries developed by independent teams. This is for example the objective of the xSDK package [9], but such a platform is beyond the scope of the coupling level considered in this paper.

Furthermore, the use of such libraries does not allow to reuse directly some existing components of elsA's legacy module and the strategy which has been initiated consists first in the externalization of some components from elsA's kernel. Work has then been conducted to build a new aeroelastic library outside elsA's kernel to reproduce in a first step the same simulations as those available with the legacy module. Then some components have been improved and new capabilities are progressively developed. The compatibility with CWIPI is also anticipated to address HPC requirements in massively parallel environments.

In the next section, additional details about the new library with respect to the legacy one are provided. Then examples of common aeroelastic simulations (static/dynamic coupling, forced motion) are presented in the context of the new modular framework. Then advanced aeroelastic coupling simulations are illustrated in section 4 for specific applications involving for example a non-linear modelling of the structural behavior, or coupling with flight dynamics. The implementation of specific modular solvers for time spectral analysis and adjoint problems is described in section 5 and finally some conclusions and perspectives are given.

2 MODULAR FRAMEWORK FOR AEROELASTIC COUPLING

In this section we first recall some details about the legacy aeroelastic module implemented in elsA and we then describe the work conducted to externalize some components of this module, before being able to rebuild a fully modular external aeroelastic library.

2.1 elsA's legacy aeroelastic module

elsA's project started in 1997 within ONERA's aerodynamic department and has since been developed by a large number of contributors from several departments inside ONERA, but also by industrial or academic partners, such as AIRBUS, SAFRAN, CERFACS, ECL/LMFA and CENAERO. It is currently a ONERA-SAFRAN property.

Within the CFD software elsA, the optional "Ael" module has been developed since the early stage of elsA's project to enable the simulation of aeroelastic phenomena in a unified framework. Multiple aeroelastic simulations drivers give access to non-linear and linearized harmonic forced motion computations, static coupling and consistent dynamic coupling simulations in the time-domain, all simulations being compatible with the flow solver features [2] [3].

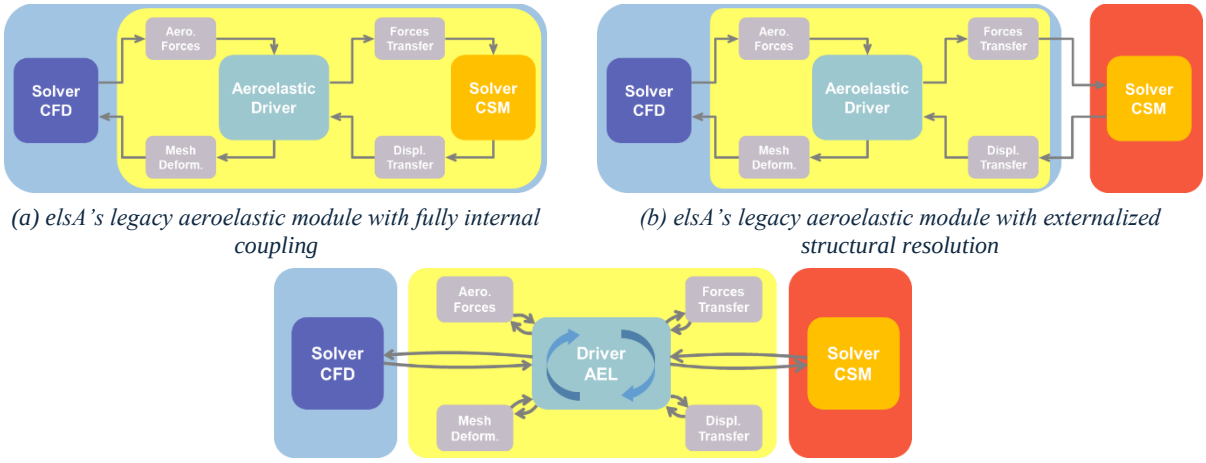
In addition to the specific aeroelastic simulation drivers, the historical "Ael" module depicted by the yellow box in Figure 1 (a) includes three submodules: a basic submodule for the resolution of the structural linear response (orange box), a submodule to deal with the volumic fluid mesh deformation (gray box), and a submodule for data transfer of load and displacement (gray boxes) between the fluid and structure solvers. It should be noted that all developments within this module have been carried out in the context of structured mesh topology and the extension to unstructured grid is not straightforward.

The structural module deals only with linear elastic structures, whose behavior is represented by a condensed Finite Element model in the static case or a modal basis in the dynamic case. The mesh deformation module includes several methods to deal with the challenging issue of preserving a proper deformed mesh quality. To this end, a first technique relies on the assumption that the fluid domain can be assimilated to a linear elastic continuous medium. A classical Finite Element problem is then solved, assuming that the elastic properties of the artificial structure are linked to the volume of the cell mesh, and that boundary conditions correspond to the displacement of the

aerodynamic grid at the aeroelastic interfaces. A second technique is based on a combination of the Inverse Distance Weighting (IDW) method and the TransFinite Interpolation (TFI) which is possible in the case of multiblock structured configurations. Transfers of loads and displacement fields through the fluid structure interfaces are handled with specific interpolation methods or smoothing techniques, involving the nearest neighbour or virtual work principle.

Simulations with a prescribed motion are typically performed to obtain the aerodynamic response to the excitation of different modal shapes. In this case, there is no feedback from the aerodynamics on the structure and the structural solver is not called since the motion imposed. The aeroelastic problem is solved a posteriori in a weakly coupled way, using for example the p-k method, Karpel's minimum state smoothing method or energy considerations for flutter evaluation.

The aeroelastic problem can also be solved with a stronger coupling for static or dynamic analyses. In this case, the interaction between the fluid and the structure requires an iterative process to balance the fluid loads and the structural displacements until an equilibrium is reached. In the static case, exchanges between the fluid and the structure are performed at preselected iterations of the CFD simulation after partial convergence of the fluid state, with a relaxation step to mitigate the discontinuities when the coupling occurs. In the dynamic case, the structural response is solved using a Newmark method, while the fluid is time-integrated with a Gear or DTS method. A fixed-point strategy is implemented in an additional external loop to iterate multiple times on the same physical time step to eventually reach equilibrium.



(a) elsA's legacy aeroelastic module with fully internal coupling

(b) elsA's legacy aeroelastic module with externalized structural resolution

(c) fully externalized resolution with modular components

Figure 1 : Different levels of modularity for the aeroelastic simulations

2.2 Externalization of aeroelastic components

Starting from the original implementation of elsA's legacy aeroelastic module illustrated Figure 1 (a), a first step towards the externalization had already been performed in the past by enabling the external resolution of the structural response. This intermediate step is presented in Figure 1 (b) where the structural module (red box) lives now outside the CFD software (blue box). This externalization gives access to aeroelastic coupling procedures with more sophisticated structural models including non-linearities which are taken into account either with a reduced-order model

or with a dedicated Finite Element structural solver to benefit from a fully representative solution. This capability has already been exploited to couple elsA with NASTRAN for the prediction of UHBR and CROR blade hot shapes, where geometric non-linearities occur due to large displacements, see [10]. In this case, the simulation is still driven by the CFD code, and elsA's aeroelastic module is used for the mesh deformation and the transfer of loads and displacements.

This first step of externalization offers new capabilities to deal with nonlinear structures but it is still limited to aeroelastic simulations with structured aerodynamic meshes and remains closely tied to elsA, as it is still mostly based on the legacy aeroelastic module.

An additional work of externalization has then been carried to extract the other components of the aeroelastic module outside the CFD software kernel. This has been performed for several mesh deformation techniques and transfer methods. The interaction between all these externalized components has then to be organized and a new library has to be redeveloped to recover at least similar aeroelastic simulation capabilities as those of the legacy aeroelastic module. This implementation is illustrated in Figure 1 (c), where the CFD code in the blue box could be elsA or another code. Some details about the requirements and implementation of this new library are presented in the next section.

2.3 Modular aeroelastic library

The coupling will now be addressed externally by the modular aeroelastic library from a Python environment, whereas it was originally driven directly by elsA. Consequently, for this modular approach to work efficiently, the solvers must provide functionalities for:

- read/write memory accesses to several objects that need to be exchanged for the coupling
- interrupting/restarting the iterative resolution process at the desired iteration
- advancing the resolution of a step (increment, time step) or a substep of the dual pseudo-time loop

The physical solvers (CFD, CSM) are classically written in C++ and Fortran and the functionalities listed above can be accessible through the use of some code generation tools like SWIG (Simple Wrapper Interface Generator) and in-memory data sharing using some dedicated buffer protocol.

The CFD code must provide memory accesses at every coupling step to aerodynamic loads and to the aerodynamic grid coordinates and velocities. Indeed, the aeroelastic library has to transfer loads to the structure, and the aerodynamic grid is deformed externally by some module of the library, before being sent back to the CFD solver, while ensuring that the update of the grid metrics is properly triggered. Similar accesses are also required on the structural solver side which receives aerodynamic loads and returns structural displacements that are transferred on the fluid grid, finally used as input for the mesh deformation module. The memory access to these quantities is made possible in elsA and CODA through dedicated buffer protocols enabling a coherent exchange of pointers between the C++ and Python representation of the objects containing those quantities.

The mechanism for interrupting/restarting iterations in the solvers' resolution process, as well as the functionality for advancing a resolution step are enabled through swigged methods. In this way, the iterative loop of the fluid and structural solvers can be exposed outside the code kernels, at the Python level where the user can define his proper coupling process.

Note that on the structural side, these accesses to the displacements and forces and the functionalities for advancing the solver are straightforward when a basic structural model is implemented directly inside the aeroelastic library. If a dedicated structural solver is needed, these functionalities are not necessarily available, especially when dealing with commercial codes but several solutions exist, such as the OpenFSI interface offered through MSC NASTRAN development environment.

```python
# Definition of fluid, structural and coupling
parameters
FluidModel     = dict(...)
StructureModel = dict(...)
CouplingModel  = dict(...)

Config = dict(FluidModel=FluidModel,
           StructureModel = StructureModel,
           CouplingModel = CouplingModel,
           ...)

# Prepare setup and data
AelLib = AelLibApi(Conf)
setup = AelLib.prepareSetup()
AelLib.prepareInputData()

# Initialize driver and compute
driver = AelLib.initDriver()
driver.precompute()
driver.compute()
```

(a) basic "black-box" coupling

```python
# Definition of fluid, structural models + coupling tree and
mesh deformation
fluid_model  = AelLib.newFluidModel(...)
struct_model = AelLib.newStructureModel(...)

coupling_tree =
AelLib.newCouplingTree(fluid_model,struct_model,AelInterfaces)
mesh_deformer = AelLib.newMeshDeformer(fluid_model,...)

CFDSolver = AelLib.newSolver(...)
CSMSolver = AelLib.newSolver(...)

# Prepare CFD <-> CSM transfers
coupling_tree.createTransfer(...)
CSMSolver.prepare(...)
CFDSolver.prepare(...)

# Iterative loop
for iteration in range(0,N):
    # CFD computation
    CFDSolver.computeStep()
    ForcesCFD = CFDSolver.extractForcesAndMetrics()

    # Process force transfer
    coupling_tree.updateAeroForces(ForcesCFD)
    coupling_tree.transfer(['Force'])

    # CSM computation
    ForcesCSM, ids = coupling_tree.getStructForces()
    CSMSolver.setForces(ForcesCSM, ids=ids)
    CSMSolver.computeStep()
    stateCSM = StaticSolver.extractState()

    # Process displacement transfer
    coupling_tree.updateStructDispAndCoord(stateCSM)
    coupling_tree.transfer(['Displacement'])

    # Mesh Deformation
    WallDispl = coupling_tree.getAeroFields(...)
    mesh_deformer.prescribe(WallDispl)
    Disp = mesh_deformer.compute()

    # Update aerodynamic grid in CFD solver
    CFDSolver.updateGrid(Disp)
```

(b) advanced coupling for specific user defined drivers

Table 1: Code snippet of basic use of the modular library (a) or with advanced user mode exposing the different objects and the iterative loop accessible at the Python level (b)

The code snippets presented in Table 1 illustrates how the modular library can be used for the coupling with a more or less advanced level of user intervention. The code snippet (a) corresponds to a basic use for classical aeroelastic simulations where the user calls some predefined drivers in a black box mode. This is similar to the coupling process involved with elsA's legacy aeroelastic module since the simulation run with elsA relied on predefined drivers. In the second mode (b), an advanced user has the possibility to manipulate directly at the Python level the different objects for the CFD and CSM solvers, and for the transfers and mesh deformation. This offers a great

flexibility for the user to define its own coupling, and to introduce additional operations inside the iterative loop (for example trim, flight dynamics, gust,…). These adaptations were previously only possible with elsA's legacy aeroelastic module after a thorough modification of the C++ classes and methods of the CFD solver kernel. A further advantage is that the simulation drivers can be reused for different CFD solvers, avoiding the need for a tedious reimplementation of each simulation driver in each CFD code kernel.

The advanced mode presented in Table 1 (b) highlights the existence of a coupling tree. This object serves as a unique structure to store all the required information to facilitate interoperability between CFD and CSM solvers, allowing for the transfers of displacements and forces across the two different grids. It supports parallel processing, enhancing the efficiency of coupled simulations, and can be easily saved into CGNS format.

In summary, the modular aeroelastic library implements the following components:

- aeroelastic simulation drivers for static/dynamic coupling, or forced motion simulations, including capabilities for the user to design its own driver
- mesh deformation algorithms for structured and unstructured aerodynamic grids, running in a parallel MPI context:
    - Transfinite Interpolation (TFI) method for structured grids
    - Inverse Distance Weighting (IDW) available for structured or unstructured grids; can be combined with TFI for structured grids
    - Structural analogy method: only available for structured grids and running in sequential mode; parallel mode possible when combined with IDW method
    - Quantum for structured or unstructured grids, based on IDW and a decomposition of the displacement field into a rotation and a translation terms through a quaternion algebra
    - Rigid motion to apply solid transformations simulating the trajectory and grid velocity of any kind
- transfer techniques for structured or unstructured grids
    - Radial Basis Functions (RBF) to transfer displacements from the structural grid to the fluid
    - Nearest Neighbors method (KNN) to transfer aerodynamic loads to the structure
    - Virtual Work method to transfer aerodynamic loads to the structure, while ensuring the conservation of virtual work between the aerodynamic and structural grids
- basic structural solver for linear elastic structures, including modal dynamics model
- low-levels objects to operate on fields, meshes, boundary conditions and specific objects for the coupling (aeroelastic interfaces, coupling tree)
- interfaces to communicate with CFD solvers (elsA, CODA)
- coprocessing tools for specific aeroelastic quantities (generalized aerodynamic forces, harmonic analyses,…)

The library implements a converter that transforms the input data into low-levels objects. In particular, the converter interprets CGNS data structures that are used by multiple CFD codes, but other input format like FSDM arrays used by CODA are also under consideration.

2.4 Advanced modular coupling: user defined solvers

An even more advanced use of the modular approach is possible, depending on the level of externalization for the resolution of the coupled non-linear flow problem. As an example, we consider the steady flow problem $R_a(W) = 0$, involving the fluid residual R_a depending on the conservative flow variables W. This problem is typically solved with a pseudo-unsteady method:

$$V\frac{dW}{d\tau} + R_a(W) = 0$$

Considering a classical backward Euler implicit time discretization with $\Delta W = W^{n+1} - W^n$, the residual is evaluated at the next time instant:

$$V\frac{\Delta W}{\Delta\tau} + R_a(W^{n+1}) = 0$$

The first order Taylor approximation of the residual involves the Jacobian $J = \frac{\partial R_a}{\partial W}$ of the residual such that $R_a(W^{n+1}) = R_a(W^n) + J\Delta W$ and the steady problem finally writes:

$$\left[\frac{V}{\Delta\tau}I + J\right]\Delta W = -R_a(W^n)$$

The resolution is usually performed by the CFD code, but if external accesses for the evaluation of the residual and the Jacobian matrix (or the Jacobian-vector product) are available, it then becomes possible to perform the resolution outside the CFD code.

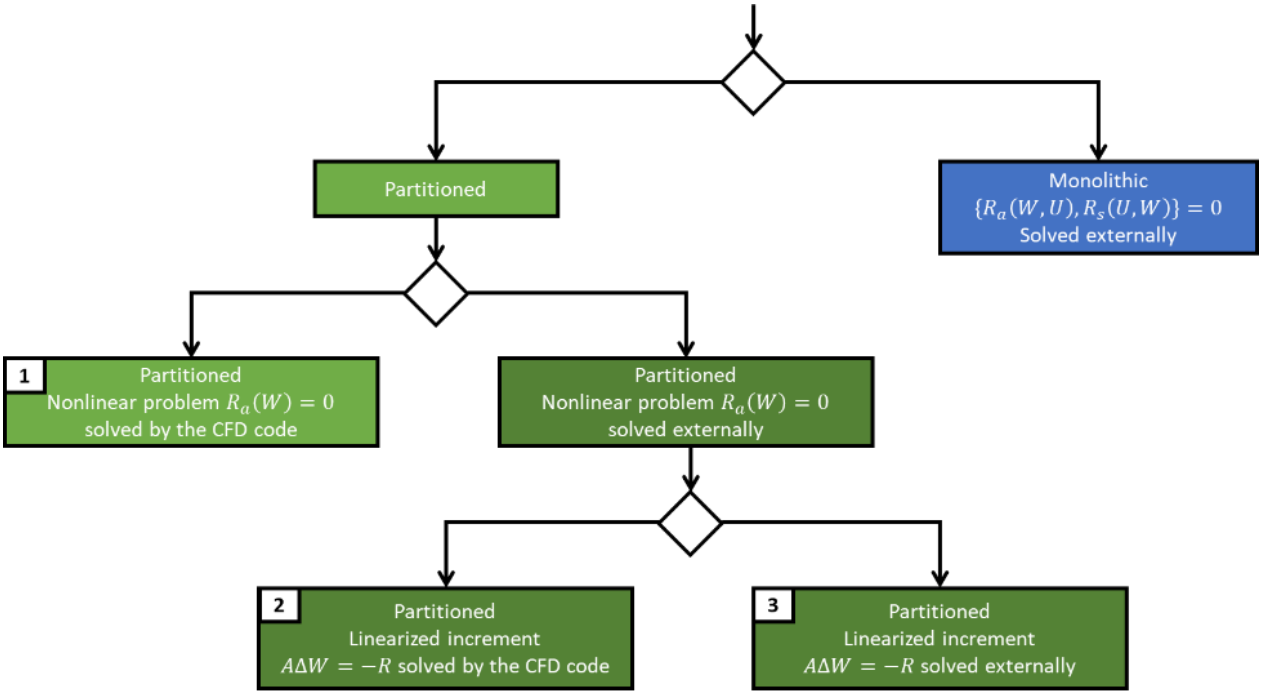


Figure 2: Options of externalization for the resolution of the fluid problem

Figure 2 summarizes different ways to solve the aeroelastic problem, with a focus on the resolution of the fluid problem. Although the monolithic approach mentioned in the blue box on the right is not covered in this paper, it should be noted that building such a solver becomes possible with a modular implementation, provided that the accesses to the residuals and Jacobians are available.

With the partitioned approach, several levels of externalization for the resolution of the fluid problem are also possible. Indeed, the nonlinear fluid problem can either be solved directly by the CFD code (option 1), or the iterative loop can be exposed in the Python level where advanced users can operate at the level of the linearized step of the resolution. In this case, the resolution for the linearized increment ΔW can still be delegated to the CFD solver (option 2), or the user can rely on its own solver or external libraries (option 3).

Option 1	**Option 2**	**Option 3**
For n=1,…,N: Residual evaluation: $-R_a(W^n)$ Jacobian evaluation: $J = \frac{\partial R_a}{\partial w}$ Get Cell Volume: V Compute Pseudo Time step: $\Delta\tau$ Compute increment: $\left[\frac{V}{\Delta\tau}I + J\right]\Delta W = -R_a(W^n)$ Update solution: $W^n \leftarrow \Delta W + W^n$	For n=1,…,N: Residual evaluation: $-R_a(W^n)$ Jacobian evaluation: $J = \frac{\partial R_a}{\partial w}$ Get Cell Volume: V Compute Pseudo Time step: $\Delta\tau$ Compute increment: $\left[\frac{V}{\Delta\tau}I + J\right]\Delta W = -R_a(W^n)$ Update solution: $W^n \leftarrow \Delta W + W^n$	For n=1,…,N: Residual evaluation: $-R_a(W^n)$ Jacobian evaluation: $J = \frac{\partial R_a}{\partial w}$ Get Cell Volume: V Compute Pseudo Time step: $\Delta\tau$ Compute increment: $\left[\frac{V}{\Delta\tau}I + J\right]\Delta W = -R_a(W^n)$ Update solution: $W^n \leftarrow \Delta W + W^n$

Table 2: Different options of resolution for the fluid problem depending on the level of externalization. Lines highlighted in blue correspond to the operations performed by the CFD code.

With the first option the resolution of the fluid problem is performed by the CFD code and the main advantage is to benefit from the different solvers available and from optimized HPC implementations, with minimal exchanges between the CFD solver and the Python layer.

In the second option, we still benefit from the linear solver implementations of the CFD code but the convergence loop can be driven externally and the left- and right-hand size terms of the linear problem can be modified in the Python layer to define new types of solvers (e.g. for the Time Spectral Method) provided that the modified terms can be updated in the CFD code.

In the third option, all steps of the linear system resolution are executed externally. This approach offers a great flexibility for prototyping new solvers, and allows the use of external libraries like PETSc [11]. This last option can be used to investigate some strategies to improve the resolution of coupled adjoint systems or to design coupled monolithic solvers for example [12] [13] [14].

3 STANDARD AEROELASTIC SIMULATIONS

3.1 Static aeroelastic coupling

The partitioned static coupling procedure involves the resolution of the steady fluid equations (Euler or RANS, solved with a pseudo-time step method) $R_a(W, X) = 0$ where the grid coordinates $X(U)$ depend on the structural displacements U. The structural equations $R_s(U, F_a) = 0$ are derived from a Finite Element discretization for linear elastic structures, such that $R_s(U, F_a) = KU - F_a$ with K the structural stiffness matrix and $F_a(W)$ the aerodynamic forces depending on the fluid conservative variables. The coupled aeroelastic problem is solved in a partitioned way, with a fixed-point method including a relaxation of the structural displacements. In practice, the structural stiffness matrix is often condensed to deal with a matrix of smaller dimension.

Static coupling computations have thus been carried out to validate the modular process in terms of external coupling loop with mesh update (to obtain the fluid grid coordinates $X(U)$ from the

structural displacements U at the fluid structure interface) and force transfer process (to evaluate the forces on the structural nodes $F_a(W)$ from the fluid conservative variables W) with respect to legacy aeroelastic simulations performed with elsA's aeroelastic module.

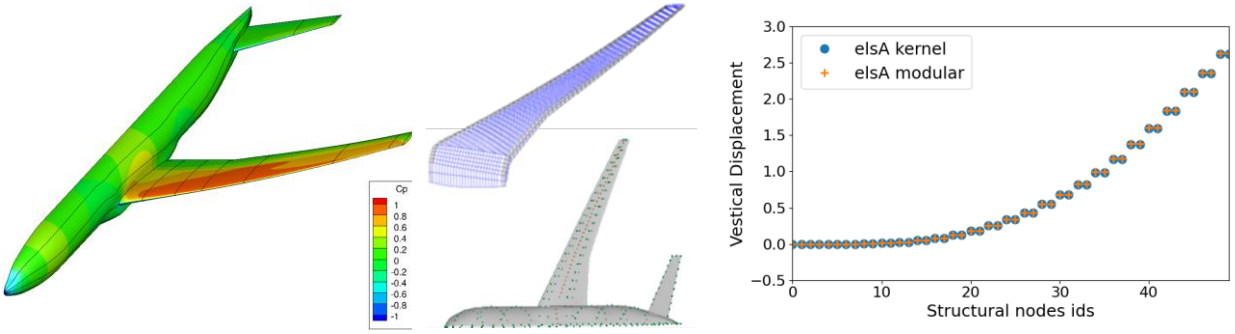


Figure 3: CRM test case for static coupling. Pressure distribution (left), structural model with force (red) and displacement (green) transfer nodes (middle), comparison of vertical displacements (right).

The Common Research Model (CRM) configuration is first considered. The fluid domain is discretized by a multi-block structured mesh including ~10^6 cells. A static coupling computation performed with elsA's legacy implementation is compared to the modular approach for a transonic point at Mach 0.85 and AoA = 1.32°. The pressure distribution for this operating point is shown in Figure 3 (left). For the aeroelastic coupling, the Finite Element model is condensed into a reduced flexibility matrix. Fifteen fixed point iterations between the fluid and the structural solvers are performed to reach the equilibrium. The comparison of the vertical displacement on some of the structural nodes is presented in Figure 3 (right), resulting from the simulations performed either with elsA's legacy aeroelastic module or with the modular implementation. The agreement between both computations is excellent, although the mesh deformation (TFI method) and transfer methods (KNN and RBF) are not exactly implemented in the same way.

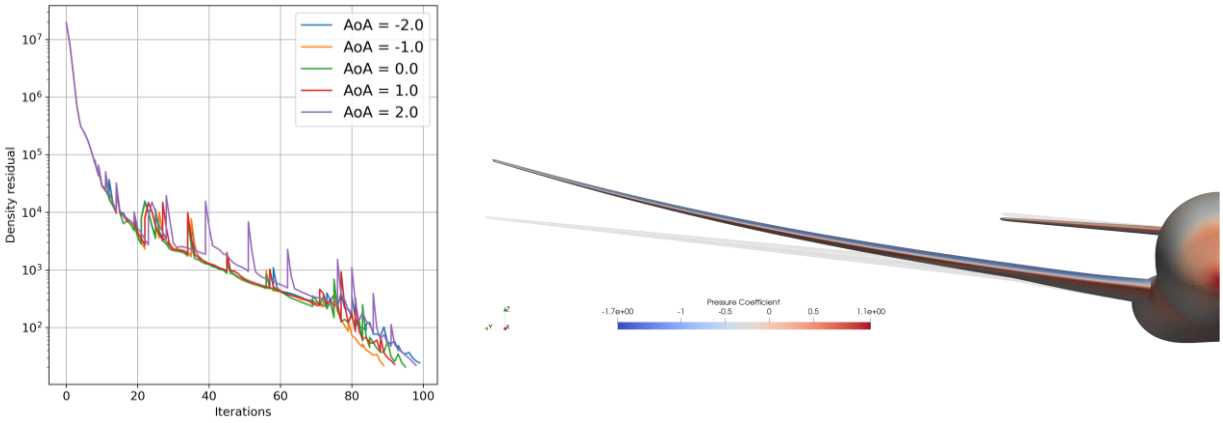


Figure 4 : Static coupling with CODA on the DLR F25 configuration. Convergence residual and deformed shape for α=2°

The modular static coupling capability has also been assessed with CODA. The DLR F25 wing-body-tailplane configuration in cruise condition is considered and the structure is modeled by a condensed flexibility matrix. The RANS SA-neg simulations are run on an unstructured mesh with 4.4M cells in a modular way, calling CODA's method for external mesh update and using mesh deformation tools (IDW method) and transfer techniques (nearest neighbor for loads, RBF for

displacements) from the modular library. The resolution of the fluid problem with CODA involves a 3 stages approach with improving reconstruction scheme and Jacobian accuracy at each step. The coupling with the structural resolution is triggered when the residual is decreased by a certain order of magnitude. The convergence of the solution is presented in Figure 4 (left) for several angles of attack α and a visualization of the deformed wing shape for α=2° is shown in Figure 4 (right). Further work is ongoing to compare results to elsA's legacy simulations.

These first examples validate the modular approach with elsA and CODA for updating the metrics from a mesh deformation process run outside the CFD code and for handling static coupling resolution with a simple structural model.

3.2 Forced motion

Validation of the modular approach is then performed in the unsteady case for forced motion simulations. In this case, the coordinates of the deformed mesh, but also the grid velocity of the deformed aerodynamic mesh has to be provided from the modular library to the CFD code to feed the contribution of the convective flux term arising from the Arbitrary Lagrangian Eulerian formulation. The fluid equations (Euler or URANS, solved with a Dual Time Step or a Gear method) write formally $\frac{d(VW)}{dt} + R_a(W, X) = 0$. The update of the grid coordinates $X(U)$ modifies the computation of the cell volumes V and the evaluation of the residual because of the metrics changes and the additional contribution from the grid velocity $\dot{X}$ in the convective flux term. Forced motion simulations are usually performed to get the aerodynamic forces associated to structural modes of vibration. The structural displacement prescribed at the fluid structure interface is thus obtained from selected modes shapes, weighted by a generalized coordinate: $U(t) = \Phi q(t)$.

The test case is a state-of-the-art fan blade representative of a modern UHBR engine, presented in Figure 5 (left). A typical second bending mode shape ϕ^{2F} illustrated in Figure 5 (right) is considered to prescribe the deformation of the fluid structure interface with a harmonic forcing, such that the physical displacement applied is: $u = \phi^{2F} q^{\star} \cos(\omega t)$. Note that the mode shape is complex due to cyclic symmetry condition but only the real part is considered for these preliminary validations. Besides, simulations are run only for a single sector of the whole fan.

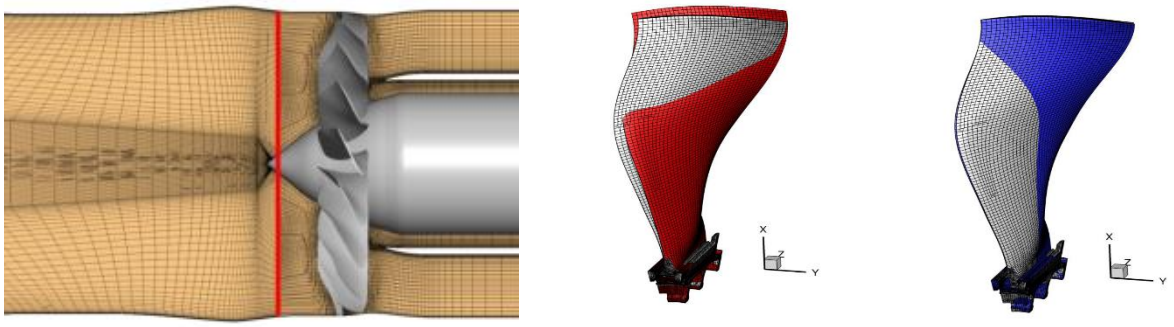


Figure 5: UHBR fan blade test case and real/imaginary mode shapes of interest (artificially deformed shape for visualization)

Simulations are first performed for a structured mesh to compare with elsA's legacy solution. Since a modal deformation is prescribed, the deformed mesh is precomputed and then interpolated at each time step with respect to the current value of the harmonic generalized coordinate. The time history of the reference computation run with elsA's legacy aeroelastic module is presented in

Figure 6 (left) with the blue line for the integrated Generalized Aerodynamic Force (GAF) over several cycles of vibration. The same computation is then carried out with the modular implementation, using exactly the same preprocessed deformed mesh for the modal shape of interest. The orange line on the left of Figure 6 compares perfectly to elsA's legacy solution and validates the modular implementation for forced unsteady motions.

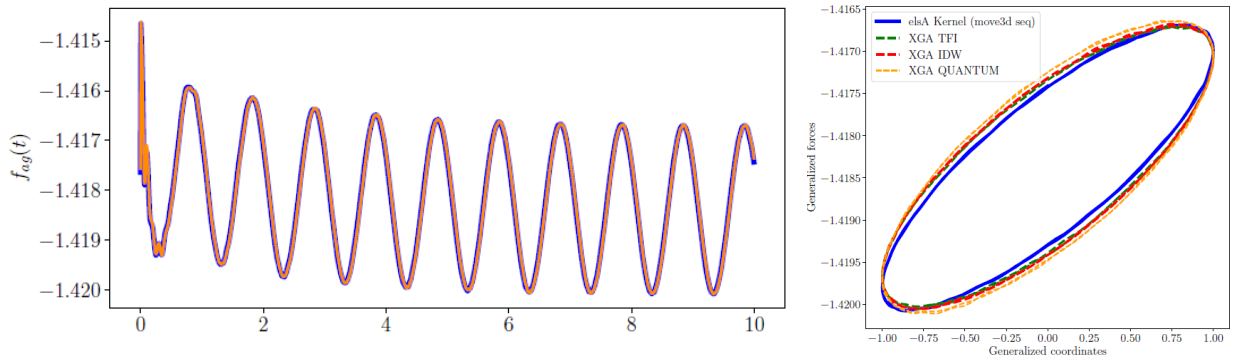


Figure 6: Comparison of elsA's legacy and modular solution for forced motion using the same deformed mesh (left) or alternative modular mesh deformation algorithms (right).

With the modular implementation, other mesh deformation methods are available. The phase portrait (Lissajous) of the GAF is compared on Figure 6 (right) on the last cycle of vibration: a very good agreement is obtained when using also pure TFI, IDW or the quaternion approach (Quantum). The modular approach offers a large variety of mesh deformation methods, and larger amplitudes of vibration can be prescribed than with the legacy implementation.

Another significant advantage is the possibility to perform aeroelastic simulations for unstructured grids. In the present case, the structured grid has been "destructured" for comparison purpose: the same mesh with hexahedral cells is used, but the mesh is treated as an unstructured one. A very good agreement is also obtained in this case (see Figure 7). The IDW mesh deformation method is used in this case since the structural analogy approach is not available yet in the modular implementation for unstructured meshes.

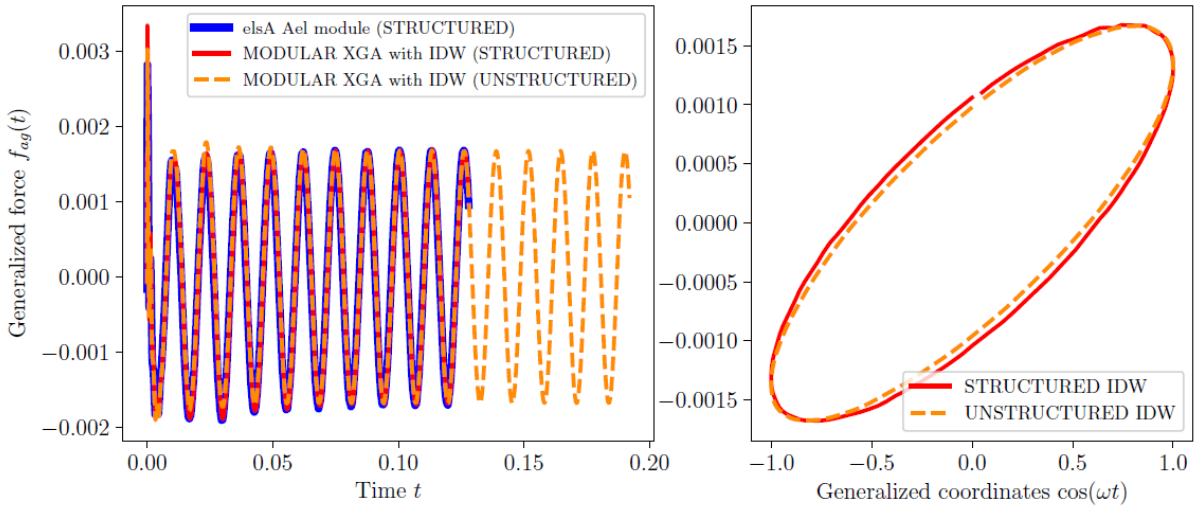


Figure 7: Comparison of forced motion aeroelastic computation with structured or unstructured (destructured) mesh

3.3 Dynamic coupling

We consider now unsteady aeroelastic coupling, involving the same unsteady fluid equations (Euler or URANS) as in the forced motion case: $\frac{d(VW)}{dt} + R_a(W,X) = 0$. On the structural side, the displacements are no longer prescribed but result from the resolution of the dynamic equations of motion. For a linear elastic structure, the equations write: $M\ddot{U} + C\dot{U} + KU = F_a(W)$. The fluid equations are time integrated with a Dual Time Step or Gear method, whereas the structural equation is time integrated with a Newmark method. As in the static coupling case, a fixed-point strategy is also needed to balance aerodynamic forces and the structural displacements since the aerodynamic forces are not implicited in the Newmark scheme.

In elsA's legacy module the fixed-point approach is implemented in an additional coupling loop iterating several times for each physical time step. The modular implementation of the coupling is slightly different: exchanges between the fluid forces and the structural displacements are now performed during the subiterations of the time integration method (either Dual Time Step or Gear), thus avoiding iterating multiple times on the same time step. Both approaches are illustrated in Figure 8, assuming that the structure is linear.

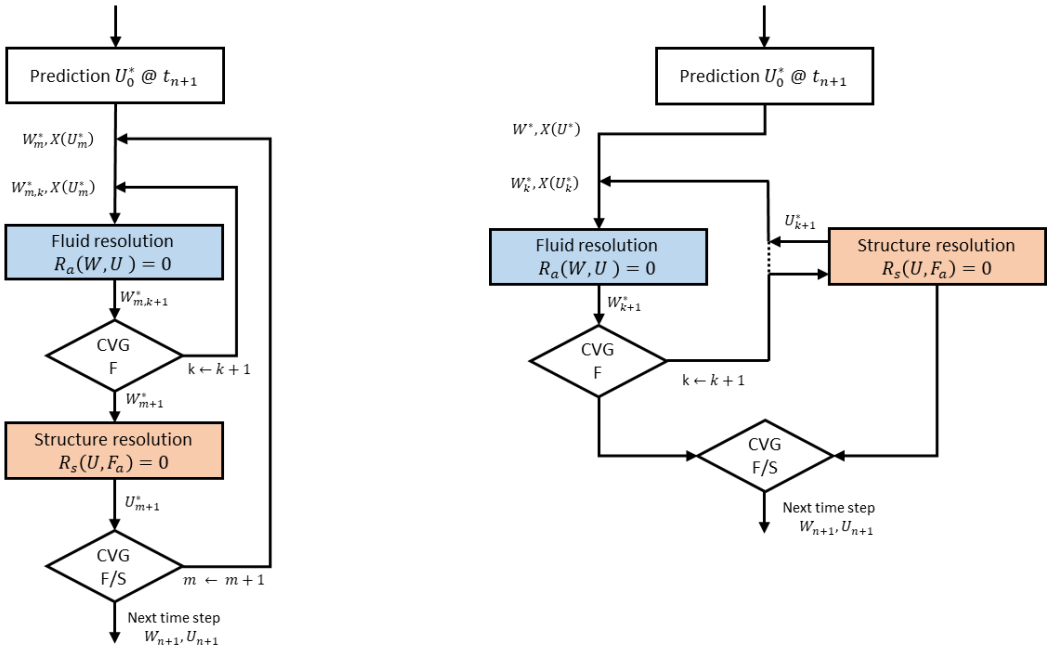


Figure 8: Flowcharts of the dynamic coupling implementation. On the left, the original implementation with an additional loop to reach equilibrium between the fluid and the structure; on the right the new implementation where the structural displacements are updated during the iterative loop of the fluid resolution

Note that in practice the linear dynamic behavior of the structure is usually modeled by a small set of eigenmodes, such that the displacements are approximated as $U = \Phi q$ with $\Phi = [\varphi_1, \dots, \varphi_r]$. The dynamic equation of motion is then reduced by projection to a very limited number of decoupled equations: $\mu\ddot{q} + \beta\dot{q} + \gamma q - f_{ag}(W) = 0$, where μ, β, γ are the generalized mass, damping and stiffness matrices respectively and $f_{ag} = \Phi^T f_a$ is the vector of the generalized aerodynamic forces. The reduced set of equations, still solved by a Newmark algorithm, provides the generalized coordinates q, from which the physical displacements are finally evaluated.

For validation purpose, we consider here the standard AGARD 445.6 configuration with a structural model modified to have either stable or unstable behavior. As illustrated in Figure 9(left), the flow is transonic (Ma=0.96, AoA=1°) and the dynamic behavior of the structure is modeled by the first 6 eigenmodes whose deformed shape transferred on the fluid grid of the aeroelastic interface is shown in Figure 9(right).

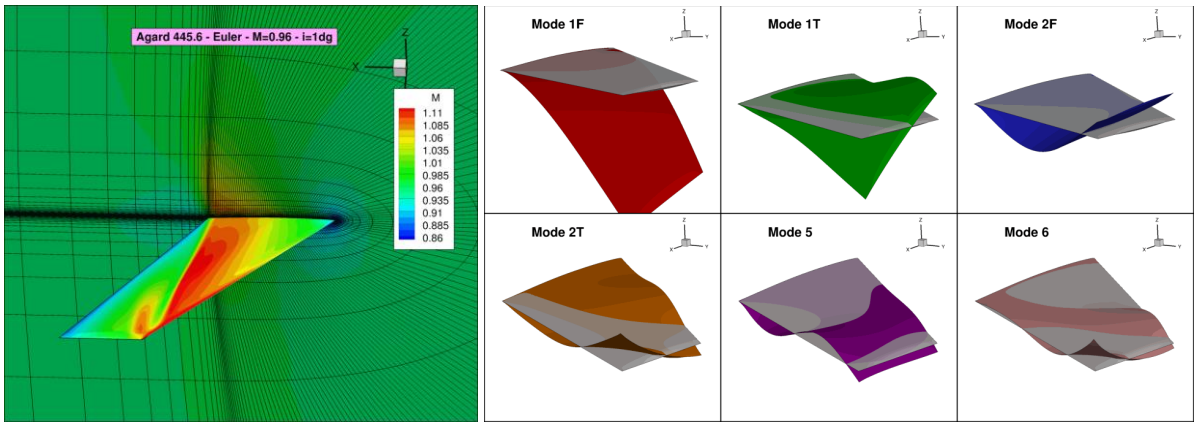


Figure 9: AGARD configuration – Mach distribution (left) and first mode shapes (right)

A coupled aeroelastic simulation is performed assuming an inviscid flow (Euler) and a linear elastic structure. The flow conditions are selected here such that the system has an unstable behavior (flutter). The simulation is started from a steady equilibrium state where a perturbation is introduced. The amplitudes of the generalized coordinates are therefore increasing with time (positive damping). This behavior is shown in Figure 10 for the first three generalized coordinates (plain lines for elsA's legacy reference solution), and is compared to the modular solution (dashed lines). The agreement between the original reference solution and the modular one is very good, although the coupling process has been implemented differently as explained before.

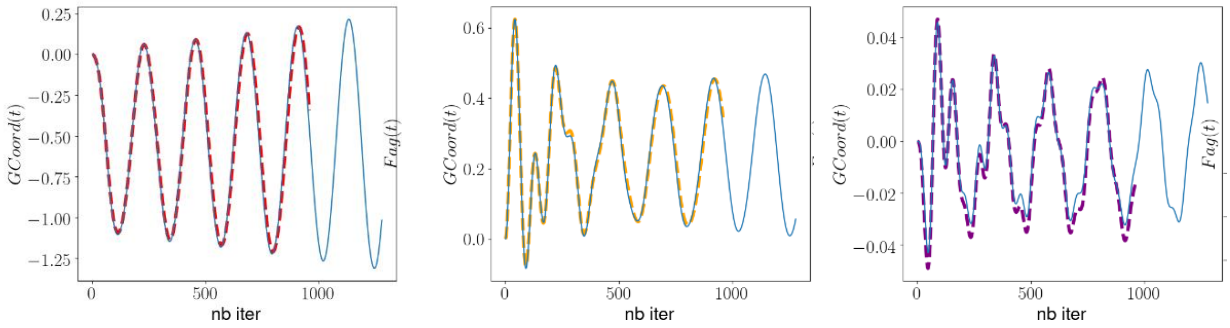


Figure 10: Comparison of first 3 generalized coordinates time response – unstable case

In this section, several test cases have been considered to validate the modular implementation of the aeroelastic coupling with respect to elsA's legacy aeroelastic module, for the most common aeroelastic computations (static coupling, forced motion, dynamic coupling). Additional capabilities have been highlighted (new mesh deformation approaches, dealing with unstructured meshes, coupling with CODA,…) and it is worth noting that no computational overhead is observed since in memory data exchanges are involved for the coupling.

4 ADVANCED AEROELASTIC COUPLING FOR SPECIFIC APPLICATIONS

Additional capabilities are now presented in this section, whose modular implementation is much easier than if developments were conducted in the CFD solver kernel.

4.1 Coupling with non-linear structural models

High aspect ratio wings, and large diameter fans or propellers that are currently investigated to increase efficiency have an increasingly nonlinear behavior due to large displacements. Also, nonlinear effects due to friction (e.g. for blade-disk assemblies) or material properties may influence significantly the aeroelastic response of aeronautical structures. A nonlinear modelling of the structure is therefore required in the resolution of the coupled aeroelastic problem.

When structural nonlinearities like large displacements are considered, the dynamic equation of motion contains an additional nonlinear term: $M\ddot{U} + C\dot{U} + KU + f_{nl}(U) = F_a(W)$. In the static case, the equation reduces to $Ku_s + f_{nl}(u_s) = F_{a,s}(W)$ with u_s the nonlinear solution resulting from a steady aerodynamic force. The dynamic equation can be reformulated, after splitting the total displacement into a static and a fluctuating contribution: $U(t) = u_s + u(t)$. Assuming the same breakdown of the forces $F_a(t) = F_{a,s} + f_a(t)$, the dynamic equation of motion are rewritten for the unsteady contribution of the displacement only: $M\ddot{u} + C\dot{u} + \big(K + K_{nl}(u_s)\big)u + g_{nl}(u) = f_a(W)$, with the tangent stiffness matrix $K_{nl}(u_s) = \partial f_{nl}(U)/\partial u|_{u_s}$ and the purely nonlinear contribution $g_{nl}(u)$. The treatment of the nonlinear term requires an iterative Newton process that is combined in the dynamic case with the time integration scheme, still performed with the Newmark method.

Dealing with such nonlinear structural models within the CFD solver kernel is not fully relevant, as different methods and solvers than those used for the fluid are required. Besides, the partitioned coupling is well suited to rely on an external structural solver. The Finite Element analysis code NASTRAN is thus considered here to benefit from advanced and efficient nonlinear modelling capabilities of the structure.

A static coupling procedure is implemented, where the structural resolution is performed with NASTRAN using the SOL400 nonlinear solution enabling large displacements modelling, and with elsA as fluid solver. The coupling is performed with a basic approach, where structural displacements and forces provided by each solver are written in files. The fluid solver is stopped by the aeroelastic driver of the modular library at each coupling iteration to wait for new structural displacements. However, NASTRAN cannot be stopped to wait for aerodynamic loads during the resolution, unless the OpenFSI interface is used. Such a coupling with OpenFSI has already been performed in the past at ONERA, using elsA's legacy aeroelastic module enabling a partial externalization (only for the structural resolution, the mesh deformation and transfer being performed within elsA), see [10] for example. This type of coupling is currently being reimplemented to work with the modular library.

In the meantime, basic file-based coupling is possible with the modular library and an example of static coupling involving a nonlinear resolution of the structure and the fluid is shown in Figure 11. The configuration is a very high aspect ratio wing derived for research purpose from Airbus' XRF1 geometry [15]. The very large deflection of the wing tip is illustrated in Figure 11 (left) for several flight conditions. The convergence of the vertical and longitudinal wing tip displacements and lift are plotted in Figure 11 (right). The solution obtained with a nonlinear structural modelling (dark colored lines) is compared to the solutions obtained with a classical linear modelling (light

colored lines). A significant difference is observed for the longitudinal displacements that are not well captured by a linear model since the geometrical nonlinearity is not modelled. The vertical wing tip displacements are nonetheless rather close and the lift integrated over the whole wing surface is very similar in both cases.

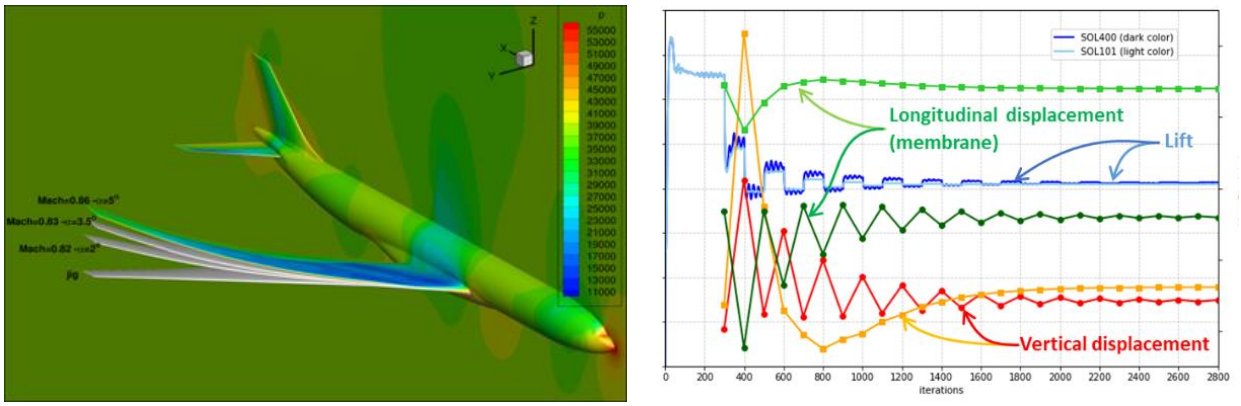
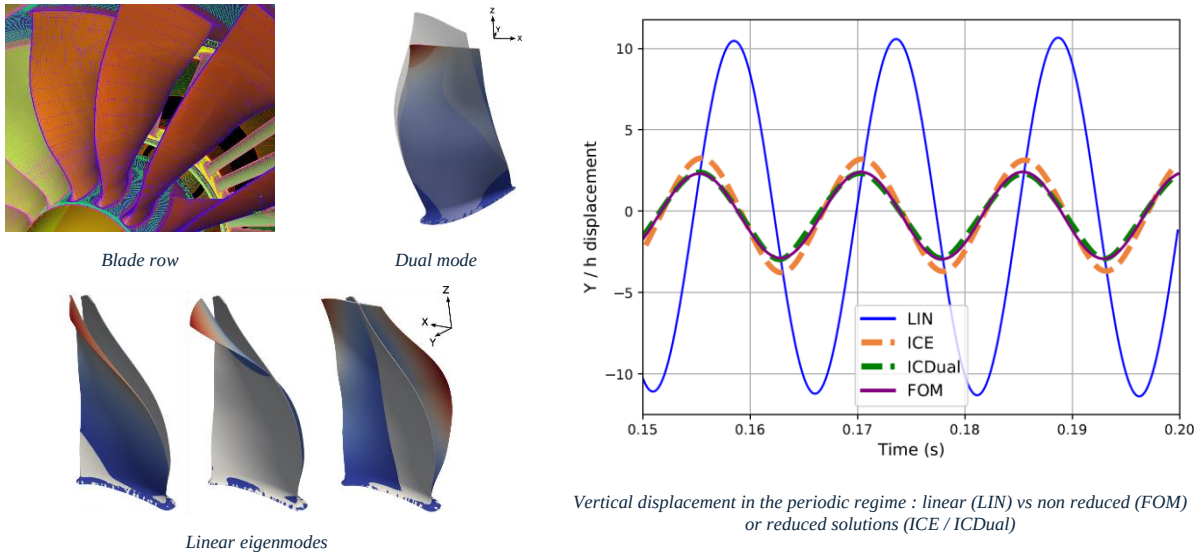


Figure 11: Nonlinear CFD/CSM static coupling on Airbus' XRF1 configuration. Wing tip deflection for several flight conditions (left) and convergence of vertical/longitudinal displacements and lift (right).

The structural non-linearities can also be approximated by a reduced order model. This is particularly useful for unsteady aeroelastic computations to avoid the coupling with an external Finite Element solver like in the example presented above. Large displacements can be taken into account using a well-chosen projection basis that contain information on the nonlinear coupling of the structural response, such that $U \approx Vq$ with $V = [\Phi, D]$ containing linear eigenmodes Φ and additional modes D (modal derivatives, or dual modes like in the present case). The nonlinear dynamic equations of motion are then projected on this small dimension basis, leading to the reduced set of nonlinear equations: $\mu\ddot{q} + \beta\dot{q} + \big(\gamma + \gamma_{nl}(u_s)\big)q + \tilde{g}_{nl}(u) = f_{a,g}(W)$. The k-th coefficient of the purely nonlinear force vector is then approximated by a third order polynomial as: $\tilde{g}_{k,nl} \approx \sum_{i,j} \rho_{ij}^k q_i q_j + \sum_{i,j,m} \theta_{ij}^k q_i q_j q_m$. The coefficients of the polynomials are identified from a set of precomputed nonlinear static solutions, following the Implicit Condensation method. An explicit expression of the reduced order model with respect to the generalized coordinates q is thus obtained, that can be efficiently coupled to the fluid solver without relying on an external Finite Element solver.

The nonlinear reduced order model detailed above has been coupled to the fluid solver elsA on the simplified test case of a flexible beam placed in the wake of a cylinder generating an unsteady forcing [16], and it has also been used to evaluate the dynamic response of a UHBR fan blade subject to an external forcing induced by an inlet distortion [17]. The fluid mesh of the fan blade is depicted in Figure 12 (top left), along with the first linear eigenmodes and the dual mode selected for the reduced basis (middle and bottom left). The unsteady aerodynamic forces induced by the inlet distortion have been determined in a preprocessing step, and have then been applied to the structure as an external load. The nonlinear dynamic response of the structure is finally assessed in a decoupled way using the reduced order model representation. The comparison with the solution using a linear representation of the structure or with the full (non-reduced) Finite Element model reveals a good agreement that provides confidence for the use of such models in aeroelastic computations, see Figure 12 right.



Vertical displacement in the periodic regime : linear (LIN) vs non reduced (FOM) or reduced solutions (ICE / ICDual)

Figure 12: Aeroelastic solution using a reduced representation of the structure with non-linearities (large displacements)

4.2 Flexible trim capability

Thanks to the accesses granted by the modular framework to the different steps of resolution of the aeroelastic problem, the determination of trimmed aircraft conditions can be addressed, while considering the flexibility of the aircraft. The trimming problem comes down to finding the angle of attack α and the HTP deflection δ to reach a target lift $C_L^\star$, while ensuring zero pitching moment C_M and considering the wing and fuselage flexibility of the aircraft.

The residual formulation of the problem writes $R_{trim}(T) = (C_L(T) - C_L^\star, C_M(T))^T$ with the unknown vector $T = (\alpha, \delta)^T$. An iterative loop for the resolution of the nonlinear trim problem is implemented in the modular environment, within which the coupled static aeroelastic problem is solved to take into account the flexibility of the aircraft. The resolution of the nonlinear problem involves the Jacobian matrix

$$J_{trim} = \begin{bmatrix} \frac{\partial C_L}{\partial \alpha} & \frac{\partial C_L}{\partial \delta} \\ \frac{\partial C_M}{\partial \alpha} & \frac{\partial C_M}{\partial \delta} \end{bmatrix}$$

Two approaches are considered for the resolution trim problem: the first one solves the nonlinear static problem $R_{trim}(T) = 0$, whereas the second one considers a dynamic formulation. For the first solution, the modular implementation offers flexibility to assess different improvements such as the use of Broyden's method for the evaluation of the Jacobian, or Aitken's relaxation for the resolution of the coupled problem.

The dynamic approach is derived from the formulation of a second order mechanical system: $\mu_{trim}\,\ddot{T} + \beta_{trim}\,\dot{T} = R_{trim}(T)$ such that the steady solution corresponds to the static flexible trim problem [18]. Only the diagonal terms are kept for simplification and two separate dynamic equations are solved for α and δ, involving the evaluation of the coefficients $\frac{\partial C_L}{\partial \alpha}$ and $\frac{\partial C_M}{\partial \delta}$, that are approximated with simplifying assumptions. The fictitious mass and damping of the dynamic equations are carefully determined to ensure a significantly damped behavior (aperiodic solution).

An example of this modular trim procedure is illustrated in Figure 13, using the pseudo dynamic approach: the trimmed solution including the rotation of the HTP and the modification of the angle of attack is illustrated on the bottom left box, to be compared to the initial undeformed jig shape on the top left box. The deformed wing shape involves mainly the first bending mode, but other modes also contribution to a lesser extent. New positions of the rotated HTP are obtained during the resolution of the trimmed problem with a mesh deformation strategy, and a smoothing procedure to have a continuous shape at the HTP/fuselage intersection.

The convergence of the angle of attack and HTP deflection on one side and of the global force on the other are shown in Figure 13, middle and right respectively, when the resolution is performed with the dynamic approach. The convergence has an aperiodic behavior, as expected by the choice of the fictitious mass and stiffness matrices. This modular approach also makes it easy to control the behavior of multiple control surfaces and offers therefore a great potential to control the behavior of multifunctional trailing edges in an aeroelastic computation.

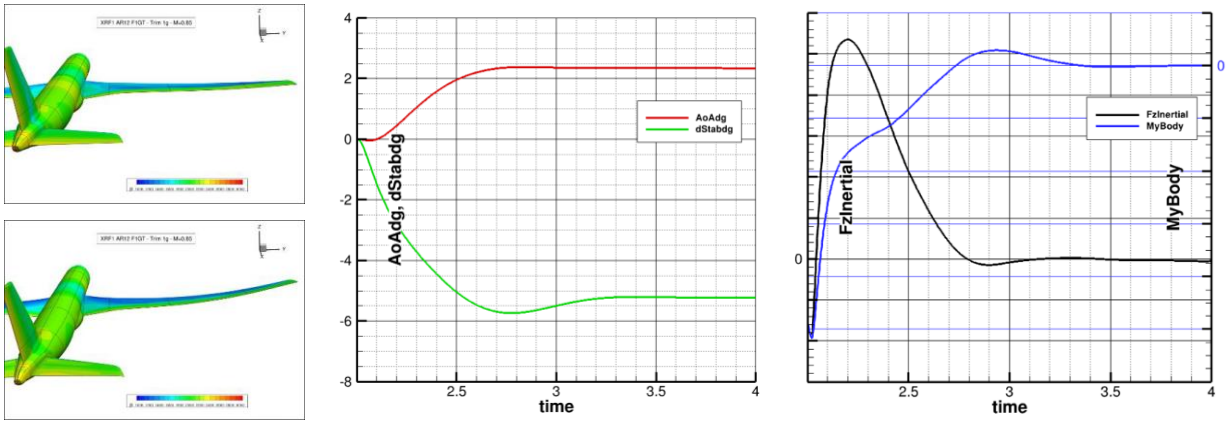


Figure 13: Initial and trimmed aircraft solution with a deflected HTP (left). Convergence of α, δ (middle) and global force and moment (right)

4.3 Coupling with flight dynamics and control

The design of very high aspect ratio wings leads to an increase of the aerodynamic loads and a different structural dynamics behavior, affecting the dynamic response to a maneuver or a gust, but also the flutter behavior. Indeed, these highly flexible structures exhibit very low frequency elastic eigenmodes, that get very close to the rigid body modes. An interaction between elastic and rigid mode thus becomes possible, and the aircraft flight dynamics response has to be considered to investigate the aeroelastic behavior of such structures.

This kind of interactions can be investigated with low/mid-fidelity aerodynamics models [19], but we rather focus here on the use of high fidelity CFD. For that purpose, further capabilities need to be implemented in the modular framework to investigate the dynamic behavior of such structures: coupling with a flight dynamics module, ability to actuate control surfaces, trim capabilities. Eventually, the objective is to address gust load alleviation problems (see the dedicated paper [18] for more details and results) and additional capabilities for gust modelling and control laws application to actuate the surface control are required. The modular library is then expanded, as shown in Figure 14 below.

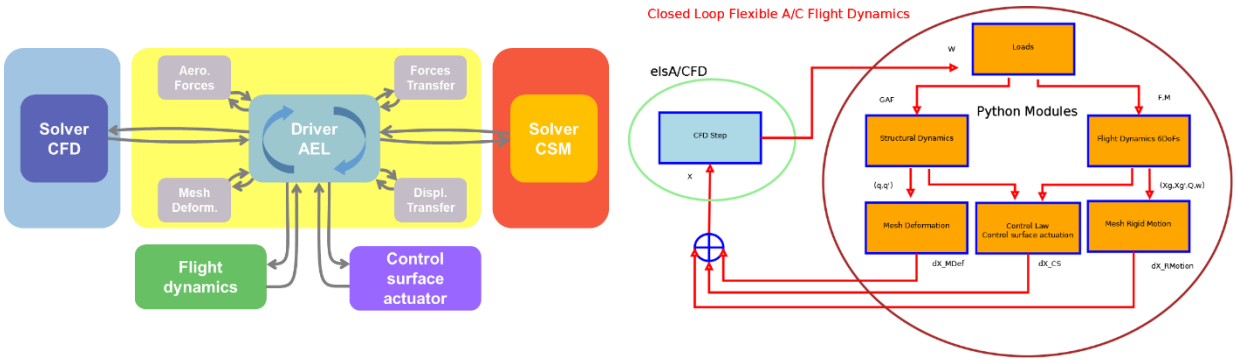


Figure 14: Modular aeroelastic coupling involving additional flight dynamics and surface control actuation modules

The aircraft motions and deformations are defined in the inertial frame of reference, and are prescribed at each time step by updating the fluid grid coordinates and velocities to cope with the ALE formulation. The mesh motion is determined from the aircraft trajectory and attitude that are computed with the flight dynamics module. In this module, Newton equations of motion are solved in the inertial ground frame of reference for the translations and rotations.

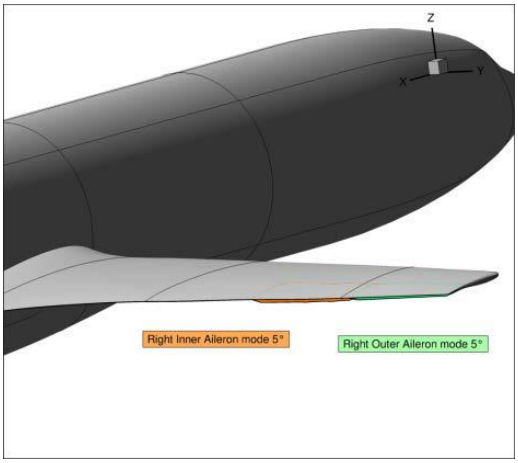
The translation equations provide the acceleration of the center of gravity $\ddot{X}_G$ with respect to the external forces applied on the aircraft (aerodynamic forces and gravity): $m\ddot{X}_G = F_a + F_g$. The rotation equations provide the rotation rate $\omega_b = (p, q, r)$, depending on the moment of the external forces and the inertial matrix J: $\dot{\omega}_b = J^{-1}(M_b - DJ\omega_b)$. Finally, the angular position is determined from the rotation rate vector using a quaternion $Q = (e_0, e_1, e_2, e_3)$ defined by a rotation axis and an angle, such that $\dot{Q} = \frac{1}{2}L^T(Q)\omega_b$, to avoid singularities encountered with Euler angles and to provide a more robust numerical resolution.

The second order equation in time for the translation equation is rewritten as a first order system, and the flight dynamics state variables are finally $W = (\dot{X}_G, X_G, \omega_b, Q)$, such that a first order equation in time has to be solved: $\frac{dW}{dt} = F(W)$. More details about the implementation of the flight dynamics module results are presented in the dedicated paper [18].

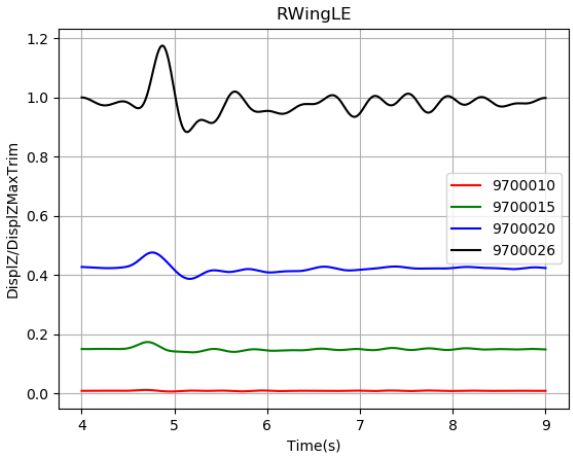
The rigid body behavior of the aircraft is thus determined with the flight dynamics module on one side, and on the other side the flexible behavior is characterized by a set of structural eigenmodes. These modes are computed for the free-free configuration. The projection of the structural dynamics equation on these modes leads to a separation of the rigid and flexible contributions that is compatible with the determination of the rigid body contribution by the flight dynamics module. The coupling between the rigid body and the flexible dynamics occurs through the external forces that depends on both rigid body and flexible dynamics states.

These capabilities combining the resolution of the rigid body behavior with the flight dynamics module and the flexible behavior with a modal structural solver has been demonstrated on Airbus' XRF1 research configuration. First, control surface actuations were prescribed on aileron, rudder or HTP to analyze the aircraft dynamics response of the free-free aircraft. A database of typical harmonic or step excitation has then been produced to train a control law aiming at reducing maximum bending moment. This control law is finally integrated in the modular framework to drive the surface control actuation during the CFD simulation. Figure 15 (a) illustrates the capability for aileron deflection that is required for the gust load alleviation. The rigid body response of the structure is illustrated by the plots Figure 15 (c) and (d) comparing the behavior

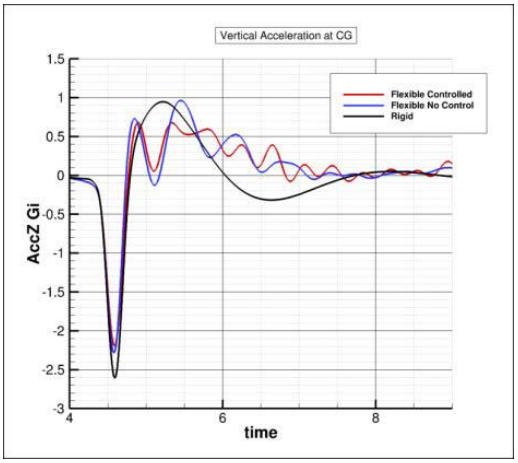
when a rigid structure is considered (black line), or with a flexible structure that is controlled (red line) or not (blue line). The peaks of response are slightly reduced in the controlled closed-loop case, using only the external aileron as control surface. Figure 15 (b) illustrates the flexible response of the wing in terms of structural displacements for several nodes located on the wing in the closed-loop case. The black line corresponds in this case to a node placed at the wing tip, where maximal displacements are observed. See [18] for more results.



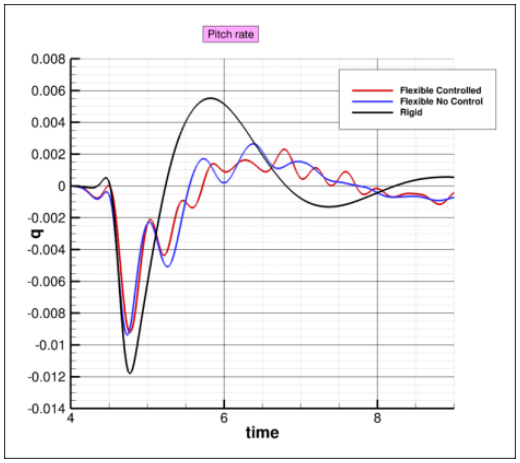
(a) Aileron activation



(b) Time evolution of wing structural displacements



(c) Time evolution of accelerations



(d) Time evolution of angular position

Figure 15:Gust response of the XRF1 aircraft configuration considering a rigid structure, of a flexible structure, with or without actuation of several control surface for gust load alleviation

5 SPECIFIC MODULAR SOLVERS FOR AEROELASTICITY

5.1 Time Spectral Solver for unsteady solutions

The characterization of dynamic aeroelastic phenomena is often based on the evaluation of the periodic response of the system, for example to determine the generalized aerodynamic forces induced by a prescribed modal motion, or to evaluate the response to a periodic gust or distortion. In this case, simulating the transient regime is not of primary interest and numerical methods to compute directly the periodic solution should be preferred.

Starting from the semi-discrete unsteady fluid equations

$$\frac{\partial VW}{\partial t} + R(W) = 0$$

the Time Spectral Method (TSM) considers a Fourier approximation with N harmonics of the conservative variables $VW \approx \sum_{k=-N}^{N} \widehat{VW}_k e^{ik\omega t}$ and the residual $R \approx \sum_{k=-N}^{N} \hat{R}_k e^{ik\omega t}$ for a given fundamental frequency ω. For deformable grids, the change of cell volumes is also accounted for with the Arbitrary Lagrangian Eulerian formulation and additional flux terms are included in the residual. Replacing the Fourier approximation in the unsteady fluid equation and making use of orthogonality properties leads to a set of $M = 2N + 1$ equations for each harmonic. The problem is preferably reformulated in the time domain using the inverse Fourier transform, to make use of the existing implementation in CFD codes for the residual evaluation. This leads to M steady equations, that are formally written for equally spaced time instants $t_n = 2\pi n/\omega M$ in the period: $D_t(V_n W_n) + R_n = 0$ with V_n, W_n and R_n referring to the evaluation of the variables at the time instant t_n. The operator $D_t(V_n W_n) = \sum_{j=0}^{M-1} d_{nj} V_j W_j$ may be viewed as a spectral discretization of the time derivative operator, and couples the M steady equations together. A pseudo-time step formulation is then introduced, such that:

$$V_n \frac{\partial W_n}{\partial \tau_n} + D_t(V_n W_n) + R_n = 0$$

The pseudo-time derivative is approximated by a first-order scheme involving the solution increment $\Delta W_n = W_n^{q+1} + W_n^q$ between iterations $q + 1$ and q for the time instant n, and the pseudo-time increment $\Delta\tau_n$ based on a CFL number. Then the backward Euler scheme is applied to the residual and the time spectral source term leading to the fully implicit TSM formulation for each time instant:

$$\left(\frac{V_n}{\Delta\tau_n} + J_n + D_t(V_n\,\cdot)\right)\Delta W_n = -R_n^q - D_t\left(V_n W_n^q\right) = -R_{\mathrm{TSM}}\left(W_n^q\right)$$

The full system involving all time instants $W = [W_1, \ldots, W_{M-1}]^T$ may be formally written as $A\Delta W = -R_{\mathrm{TSM}}(W^q)$. The block structured matrix A has diagonal terms $\frac{V_i}{\Delta\tau_i} + J_i$ and off-diagonal terms $V_i d_{ji}\, I$ derived from the application of the time derivative operator $D_t(\cdot)$. The dimension of the square matrix is very large (M times the number of fluid degrees of freedom) and the size increases with the number of harmonics.

To solve the TSM problem efficiently, the resolution must be parallelized in both space and time, and efficient numerical methods have to be considered to deal with the matrix ill-conditioning as the number of harmonics increases. For that purpose, an implementation of the TSM solver is carried out outside the CFD code kernel, to easily prototype different resolution strategies. This requires specific accesses from the CFD code: residual vector and Jacobian matrix (eventually at different orders of approximation) or the matrix-vector product for each instant.

Such accesses are available with CFD codes like elsA and CODA that provide only the necessary numerical evaluation of the quantities of interest, while the problem is assembled outside through a Python interface that is code independent. A preliminary implementation of this solver was presented in [20] with a block Jacobi formulation, where only the diagonal terms of the matrix A were retained; the coupling between the time instants then occurs only through the TSM source

term on the right-hand side. However, the convergence becomes difficult as the number of time instants increase and the fully implicit formulation is thus addressed with preconditioned GMRES algorithms. Advanced resolution strategies are being considered, making possibly use of the external linear algebra library PETSc [11]. Finally, recent work has been carried out to deal with deformable grids for aeroelastic solutions where the mesh metric depends on the time instant. The residuals vectors R_n and the Jacobian matrix J_n must therefore account for the additional flux due to the mesh deformation velocity and for additional terms in the boundary conditions at the fluid structure interface. More details about the current implementation are presented in [21].

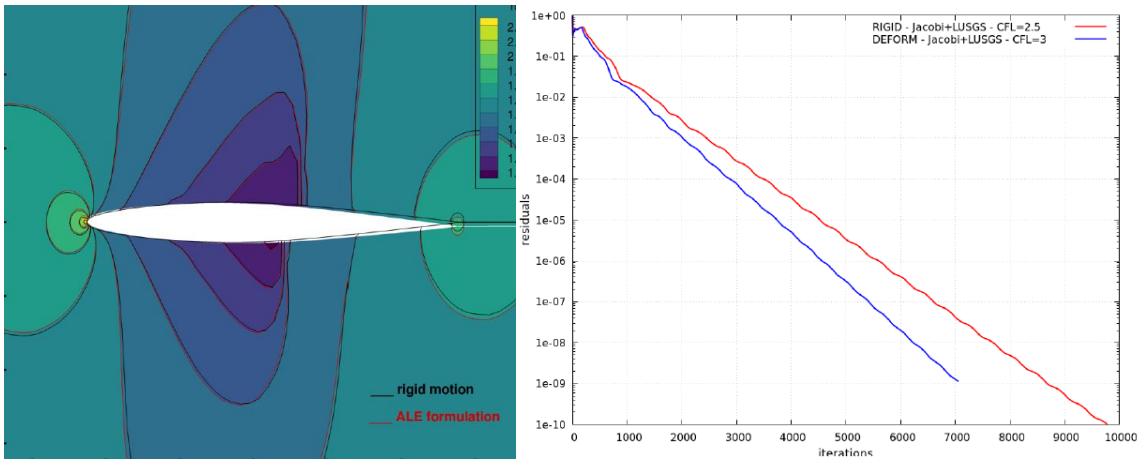


Figure 16: Comparison of rigid and deformable TSM formulations for the pitching airfoil: isolines of density for the 2nd TSM time instant (left) and residual convergence (right)

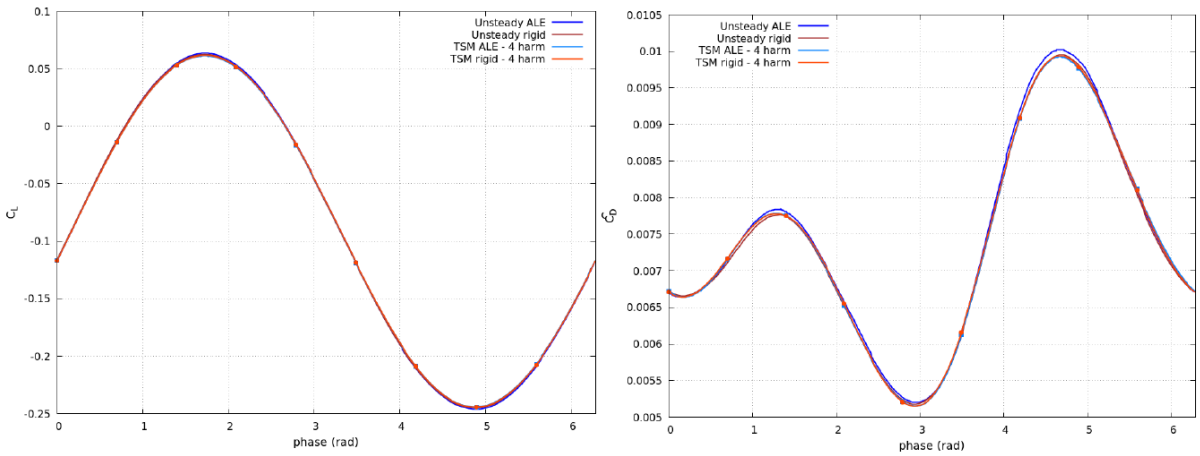


Figure 17: Comparison of lift (left) and drag (right) solutions resulting from reference unsteady computations and from TSM computations, both with a rigid or deformable formulation for the mesh rotation.

Some results are presented here for the classical NACA64A010 test case airfoil under a prescribed harmonic pitching motion in a transonic inviscid flow (Ma=0.796). Two formulations are considered for this simple test case, either by prescribing a rigid rotation to the whole mesh with moving boundaries on the far field, or by deforming the mesh while keeping the far field boundaries fixed. Figure 16 (left) compares at the second time instant both TSM solutions computed with a single harmonic (3 time instants), showing a good agreement; note that isolines are not matching since the mesh is rotated only with the mesh deformation approach. With a single harmonic, the lift coefficient is correctly reproduced by the TSM but the drag is not accurate. The

number of harmonics is then increased to 4 (i.e. 9 time instants). The convergence to the solution is illustrated in Figure 16 (right) for both TSM formulations (rigid and deformable mesh), and the reconstructed unsteady lift and drag coefficients are compared in Figure 17 to the reference unsteady solutions, showing a very good agreement even for the drag coefficient. Additional results are presented in [21], including the response to periodic gusts that are prescribed externally from the CFD code, as an additional grid velocity. Ongoing work is also carried out to improve further the solver robustness and efficiency.

5.2 Aerostructural adjoint systems

In order to further improve performances, the design of aeronautical structures is increasingly based on multidisciplinary optimization techniques. Fluid-structure interaction effects then becomes essential in the optimization process, especially when high-fidelity aerodynamic and structural models are considered in the optimization process. The objective of such a process is to minimize a quantity of interest $\mathcal{J}$ (such as the drag for example) that depends on a design parameter p, such as geometric parameters defining profile sections for example. The objective function $\mathcal{J}(W, U, p)$ derives from the solution of an aeroelastic problem, and depends therefore on the conservative fluid state W and the structural displacements U. Strictly speaking, the structural grid $X_s(p)$ and aerodynamic grid $X_a(p)$ depend directly on the design parameters and in turn, the fluid and structural states inherit from this dependency. For ease of reading, the grid dependency is not detailed in the following and the fluid and structural residual equations are formally written as:

$$R_a(W, U, p) = 0$$
$$R_s(W, U, p) = 0$$

During the optimization process, the gradient of the objective function is needed to guide the optimizer towards appropriate descent directions. The derivative of the objective function writes:

$$\frac{d\mathcal{J}}{dp} = \frac{\partial \mathcal{J}}{\partial p} + \frac{\partial \mathcal{J}}{\partial W}\frac{dW}{dp} + \frac{\partial \mathcal{J}}{\partial U}\frac{dU}{dp}$$

Note that the grid dependency with respect to the parameters is partially included in the term $\partial\mathcal{J}/\partial p$. The expression above also involves the sensitivities of the aerodynamic dW/dp and structural dU/dp states with respect to the design parameter. These quantities can be derived from the residual equations, leading to a direct approach for their evaluation [13] [22]:

$$\left.\begin{aligned} \frac{dR_a}{dp} &= \frac{\partial R_a}{\partial p} + J_{aa}\frac{dW}{dp} + J_{as}\frac{dU}{dp} = 0 \\ \frac{dR_s}{dp} &= \frac{\partial R_s}{\partial p} + J_{sa}\frac{dW}{dp} + J_{ss}\frac{dU}{dp} = 0 \end{aligned}\right\} \quad \Rightarrow \quad \begin{bmatrix} J_{aa} & J_{as} \\ J_{sa} & J_{ss} \end{bmatrix} \begin{bmatrix} \frac{dW}{dp} \\ \frac{dU}{dp} \end{bmatrix} = -\begin{bmatrix} \frac{\partial R_a}{\partial p} \\ \frac{\partial R_s}{\partial p} \end{bmatrix}$$

The Jacobians $J_{aa} = \partial R_a/\partial W$ and $J_{ss} = \partial R_s/\partial U$ are provided by the fluid and structural solvers, whereas the coupled terms $J_{as} = \partial R_a/\partial U$ and $J_{sa} = \partial R_s/\partial W$ arise from the forces and displacements transfer operators and from the mesh deformation operator that need to be differentiable. Here again the grid dependency is not detailed but is included in the derivatives of the residuals $\partial R_a/\partial p$ and $\partial R_s/\partial p$. With this direct approach, the linear system above has to be solved for each design parameter. This process is costly when high fidelity CFD simulations are performed, and an adjoint approach is thus preferred. The direct problem is formally rewritten $[\boldsymbol{J}]\left[d_p(W, U)\right] = -\left[\partial_p(R_a, R_s)\right]$ and the derivative of the objective function is reformulated as:

$$\frac{d\mathcal{J}}{dp} = \frac{\partial \mathcal{J}}{\partial p} + \left[\frac{\partial \mathcal{J}}{\partial W}, \frac{\partial \mathcal{J}}{\partial U}\right]\left[d_p(W,U)\right] = \frac{\partial \mathcal{J}}{\partial p} - \underbrace{\left[\frac{\partial \mathcal{J}}{\partial W}, \frac{\partial \mathcal{J}}{\partial U}\right][\boldsymbol{J}]^{-1}}_{\Lambda^T}\left[\partial_p(R_a, R_s)\right]$$

The adjoint vector $\Lambda = [\Lambda_a, \Lambda_s]$ introduced in the equation above is solution of the adjoint problem $[\boldsymbol{J}]^T\Lambda = -\left[\frac{\partial \mathcal{J}}{\partial W}, \frac{\partial \mathcal{J}}{\partial U}\right]^T$. The formulation becomes independent of the number of design parameters since only the derivative of each objective function appear in the right-hand size. The adjoint aerostructure problem finally writes:

$$\begin{bmatrix} J_{aa} & J_{as} \\ J_{sa} & J_{ss} \end{bmatrix}^T \begin{bmatrix} \Lambda_a \\ \Lambda_s \end{bmatrix} = -\begin{bmatrix} \frac{\partial \mathcal{J}}{\partial W} \\ \frac{\partial \mathcal{J}}{\partial U} \end{bmatrix}$$

The coupled aerostructure adjoint problem is usually solved with a partitioned procedure, alternating between the resolution of the fluid adjoint equation and the structural one. The system resulting from the linearized RANS equations of the fluid block is often very stiff, and the coupling with the structural term may be even more challenging for the numerical methods used for the resolution. Work has therefore been conducted recently to (i) improve the robustness and efficiency the adjoint fluid solvers using advance Krylov subspace methods [12], and (ii) to accelerate the convergence of the partitioned aerostructural adjoint solver using Krylov subspace recycling strategies [13].

Different solvers and resolution strategies have been prototyped outside the CFD code kernel, and compared to the legacy implementation in elsA. This has been made possible by the external accesses to the fluid Jacobian, together with the capability to assemble the contributions from the force/displacements transfer operators as well as the mesh deformation operator. The different solvers implemented include on one side the Generalized Minimal RESidual with Deflated Restarting (GMRES-DR) and its flexible variant allowing variable preconditioning. On the other side the family of Generalized Conjugate Residual with inner Orthogonalization algorithm with Deflated Restarting (GCRO-DR) and its flexible counterpart have also been investigated to recycle information from previous coupling with the structure at each restart.

Numerical experiments have been conducted with the ONERA M6 wing configuration at Ma=0.84 and an angle of attack of 3.06° (Figure 18) corresponding to a Reynolds number of 11.7×10^6. The flow is transonic and is modelled by RANS equations with a Spalart-Allmaras turbulence model. The fluid domain is discretized with a multiblock structured mesh of 3.8 million of cells.

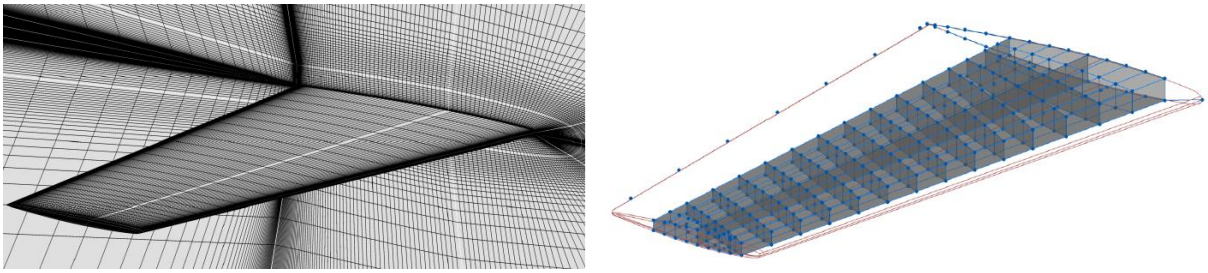


Figure 18: M6 wing aeroelastic model: 42 block-structured RANS CFD mesh and FEM internal layout

The performance of the different solvers implemented have been thoroughly investigated in [13] and some results are briefly reported here. The coupled adjoint system is first solved with the

GMRES-DR algorithm, with various preconditioners based either on the first order approximation of the Jacobian or the exact Jacobian (Figure 19, left). The red line presents the convergence of the legacy implementation using a preconditioner based on a combination of a Restrictive Additive Schwarz (RAS) domain decomposition method coupled with Lower-Upper Symmetric Gauss Seidel (LU-SGS) relaxation. The number of matrix-vector products is reduced with a BILU preconditioner applied to the exact first-order Jacobian, and even more substantial gains can be achieved with the flexible GMRES-DR (not presented here).

Further improvements are also obtained with the GCRO-DR solver, since information are recycled in this case at each coupling with the structure. The acceleration of convergence is illustrated in Figure 19, right, for different recycling strategies. The black line corresponds to the convergence of the standard GMRES-DR, clearly exhibiting some plateaus at each restart of fluid-structure coupling. Recycling information from the second cycle onwards considerably improves convergence, and the number of matrix-vector products is significantly reduced with respect to the initial solution without recycling.

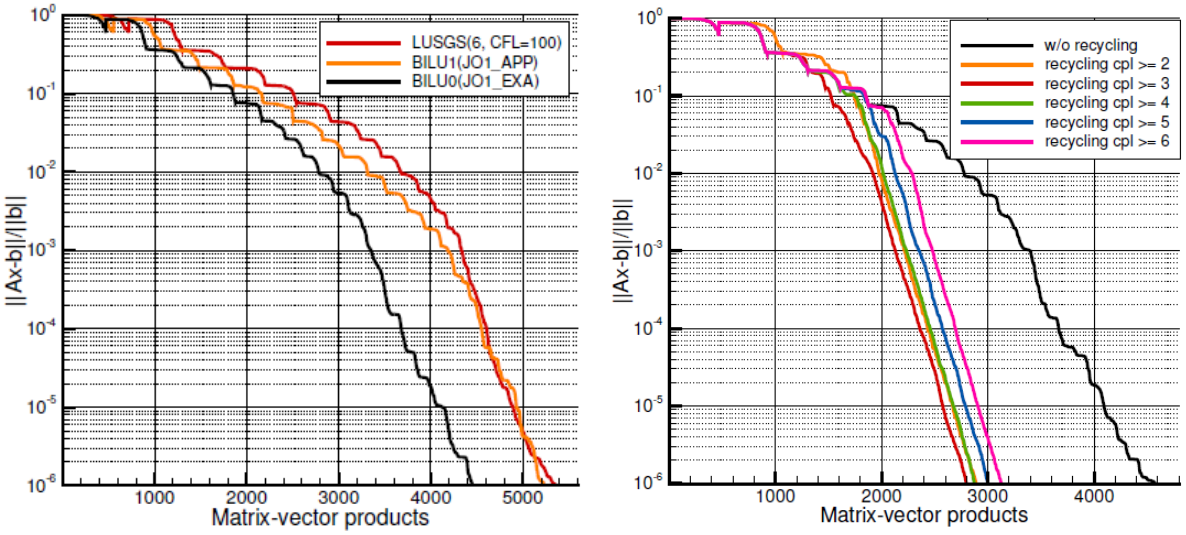


Figure 19:Convergence of the coupled adjoint relative residual norm with a GMRES-DR solver and various preconditioners (left) or a GCRO-DR solver with different recycling strategies (right).

Note, that the developments of these solvers carried out in a modular way have also led to preliminary investigations to solve the coupled adjoint problem with a monolithic formulation ([14], chapter 5). Besides, the TSM solver and its adjoint version will benefit from the progress made in the various solvers investigated here.

6 CONCLUSIONS

In this paper, recent work carried out to build a modular coupling environment for aeroelastic computations has been detailed. The evolutionary context of the CFD codes in which ONERA is involved has been recalled and justifies the need to externalize the different components historically embedded inside elsA's fluid solver kernel. A new versatile library is currently being developed around the externalized components, the aim of which being first to recover the original standard aeroelastic simulation capabilities. Results have been presented here to demonstrate the confidence gained with this modular implementation on different test cases of different levels of complexity, covering static coupling, forced harmonic motion and dynamic coupling simulations

in the time domain. New capabilities are also enabled by the modular implementation for these standard aeroelastic computations, like the compatibility with unstructured meshes required by next generation codes (CODA, SoNICS). The modular framework also offers a great potential to easily set up user defined aeroelastic simulations, powered by enriched physical modeling capabilities (nonlinear structure), or additional couplings (trim, flight dynamics, control) for example. Beyond the definition of new simulation drivers and additional coupling capabilities, the modular approach also provides in-depth interactions with the CFD code through specific accesses to Jacobians (or matrix-vector products) and residual evaluations, making it possible to build dedicated solvers (e.g. TSM or adjoint solvers) and prototype new strategies for efficient resolution of aeroelastic problems. Further work is underway to improve the library and, in particular, to extend its general use to next generation codes like CODA and SoNICS to benefit from state-of-the-art CFD codes.

REFERENCES

[1] L. Cambier, S. Heib and S. Plot, "The ONERA elsA CFD software: input from research and feedback from industry," *Mechanics & Industry,* vol. 14, no. 3, p. 159–174, 2013.

[2] P. Girodroux-Lavigne, "Progress in steady/unsteady fluid-structure coupling with Navier-Stokes equations," in *International Forum on Aeroelasticity and Structural Dynamics*, Munich, 2005.

[3] A. Dugeai, A. Madec and A.-S. Sens, "Numerical unsteady aerodynamics for turbomachinery aeroelasticity," in *9th International Symposium of Unsteady Aerodynamics, Aeroacoustics and Aeroelasticity of Turbomachines*, Lyon, 2000.

[4] S. Görtz, T. Leicht, V. Couaillier, M. Méheut, P. Larrieu and S. Champagneux, "CODA: A European Perspective for a Next-Generation CFD, Analysis and Design Platform," in *NATO AVT-366 Workshop on Use of Computational Fluid Dynamics for Design and Analysis: Bridging the Gap Between Industry and Developers*, 2022.

[5] D. Gueyffier, S. Plot and M. Soismier, "SoNICS: a New Generation CFD Software for Satisfying Industrial Users Needs," in *NATO AVT-366 Research Workshop on Use of Computational Fluid Dynamics for Design and Analysis: Bridging the Gap Between Industry and Developers*, 2022.

[6] G. Chourdakis, K. Davis, B. Rodenberg, M. Schulte, F. Simonis, B. Uekermann, G. Abrams, H.-J. Bungartz, L. Cheung Yau, I. Desai, K. Eder, R. Hertrich, F. Lindner, A. Rusch, D. Sashko, D. Schneider, A. Totounferoush, D. Volland, P. Vollmer and O. Ziya Koseomur, "preCICE v2: A sustainable and user-friendly coupling library," *Open Research Europe,* vol. 2, no. 51, 2022.

[7] W. Liu, A. Skillen and C. Moulinec, "ParaSiF_CF: A Partitioned Fluid-Structure Interaction Framework for Exascale," Zenodo, 2022.

[8] F. Duchaine, S. Jauré, D. Poitou, E. Quémerais, G. Staffelbach, T. Morel and L. Gicquel, "Analysis of high performance conjugate heat transfer with the OpenPALM coupler," *Computational Science & Discovery,* vol. 8, no. 1, 2015.

[9] R. Bartlett, I. Demeshko, T. Gamblin, G. Hammond, M. A. Heroux, J. Johnson, A. Klinvex, X. Li, L. C. McInnes, J. D. Moulton, D. Osei-Kuffuor, J. Sarich, B. Smith, J. Willenbring and U. M. & Yang, "xSDK Foundations: Toward an Extreme-scale Scientific Software Development Kit," *Supercomputing Frontiers and Innovations,* vol. 4, no. 1, pp. 69-82, 2017.

[10] A. Dugeai, Y. Mauffrey, A. Placzek and S. Verley, "Overview of the Aeroelastic Capabilities of the elsA Solver within the Context of Aeronautical Engines," *Aerospace Lab Journal,* vol. 14, 2018.

[11] S. Balay, S. Abhyankar, M. Adams and al., "PETSc Web page," 2023. [Online]. Available: https://petsc.org/.

[12] M. Jadoui, C. Blondeau and F.-X. Roux, "Recycling Krylov Subspaces for Efficient Partitioned Solution of Aerostructural Coupled Adjoint Systems," in *AIAA Aviation Forum*, 2022.

[13] C. Blondeau and M. Jadoui, "Recycling Krylov Subspaces for Efficient Partitioned Solution of Aerostructural Adjoint Systems," *Journal of Computational Physics, To appear,* 2024.

[14] M. Jadoui, "Solveurs de Krylov robustes pour la résolution partitionnée et monolithique du système adjoint couplé aéro-structure," PhD Thesis, https://theses.hal.science/tel-04336964, 2023.

[15] J. Pattinson and M. Herring, "High Fidelity Simulation of the Active Winglet," in *International Forum on Aeroelasticity and Structural Dynamics*, 2013.

[16] T. Flament, J. Deü, A. Placzek, M. Balmaseda and D. Tran, "Reduced-order model of geometrically nonlinear flexible structures for fluid–structure interaction applications," *International Journal of Non-Linear Mechanics,* vol. 158, p. 104587, 2024.

[17] T. Flament, J.-F. Deü, A. Placzek, M. Balmaseda and D.-M. Tran, "Reduced Order Model of Nonlinear Structures for Turbomachinery Aeroelasticity," *Journal of Engineering for Gas Turbines and Power,* vol. 146, no. 3, p. 031005, 2024.

[18] A. Dugeai and P. Vuillemin, "Highly Flexible Aircraft Flight Dynamics Simulation Using CFD," in *International Forum on Aeroelasticity and Structural Dynamics,* The Hague, 2024.

[19] R. Palacios and C. Cesnik, Dynamics of Flexible Aircraft, Cambridge University Press, 2023.

[20] C. Blondeau and C. Liauzun, "A modular implementation of the spectral method for aeroelastic analysis and optimization on structured meshes," in *International Forum on Aeroelasticity and Structural Dynamics*, Savannah, 2019.

[21] C. Liauzun and C. Blondeau, "A modular TSM solver for aeroelastic analysis and optimization," in *International Forum on Aeroelasticity and Structural Dynamics,* The Hague, 2024.

[22] T. Achard, C. Blondeau and R. Ohayon, "High-Fidelity Aerostructural Gradient Computation Techniques with Application to a Realistic Wing Sizing," *AIAA Journal*, vol. 56, no. 11, 2018.

COPYRIGHT STATEMENT

ACKNOLEDGEMENTS

The authors gratefully acknowledge the financial support provided by French Civil Aviation Authority (DGAC) under the France 2030 investment plan in several projects (SONICE, MAJESTIC) and European Union’s Horizon 2020 research and innovation program funded projects (Clean Sky2 ADEC, Clean Aviation UPWING, H2020 NEXTAIR). The results, opinions, conclusions, etc., presented in this work are those of the authors only. The authors also acknowledge SAFRAN and Airbus for the geometries shared in these projects.