Aircraft Engineering and Aerospace Technology

PyPAD: a multidisciplinary framework for preliminary airframe design

Lorenzo Travaglini Sergio Ricci Giampiero Bindolino

PyPAD: a multidisciplinary framework for preliminary airframe design

Lorenzo Travaglini, Sergio Ricci and Giampiero Bindolino
Department of Aerospace Science and Technology, Politecnico di Milano, Milan, Italy

Abstract

Purpose – The purpose of this paper is to describe the development of an integrated framework suitable for preliminary airframe design, called PyPAD (Python module for Preliminary Aircraft Design), providing the capability to define models to compute loads and to perform the structural sizing.

Design/methodology/approach – The modules developed until now allow for the definition of multi-fidelity aero-structural models starting from a Common Parametric Aircraft Configuration Schema (CPACS) input file and to compute static loads (trim) and flutter margin with minimum user effort. PyPAD take advantages of Abaqus-CAE, and the main functions are developed in Python, to take advantages of the simplicity in terms of software development and maintenance, but the core routines are developed in Fortran, taking advantages of parallel programming to get the best performances.

Findings – A complete test case, starting from the CPACS input and ending with the definition of structural, aerodynamic and aero-elastic models, with the computation of different design loads, is reported. An example will show that the framework developed is able to handle different problematics of the preliminary projects using quite complex global models.

Practical implications – All the tools developed in the framework, and the ones currently under development, could be a valid help during the preliminary design of a new aircraft, speeding up the iterative process and improving the design solution.

Originality/value – PyPAD is the first framework developed around Abaqus-CAE for the preliminary aircraft design and is one of the few tools looking at the different problematics involved in a preliminary airframe design: design, loads and aero-elasticity, sizing and multi-disciplinary optimization.

Keywords Aero-elasticity, Loads, Preliminary design

Paper type Research paper

Introduction

The present work describes the development and application of PyPAD (*P*ython module for *P*reliminary *A*ircraft *D*esign), a framework suitable for preliminary airframe design.

The most important considerations that motivate this work are:

- New projects request high-performing design, and the multi-disciplinary nature of the problem must be taken in consideration as soon as possible during the project.
- High-fidelity tools like computational structural dynamics (CSD) and computational fluid dynamics (CFD) are extremely powerful, but not suitable for the early phases of the project, when thousands of analysis must be performed.
- While several *ad hoc* tool have been developed during the years for the conceptual design of a new aircraft using simple models, frameworks suitable for the early phases of the project, able to handle complex models, seem to be lacking.

These key points can be translated in requests that this work tries to meet:

- To define a framework suitable for the early phases of the project of a new aircraft, using tools able to catch the multi-disciplinary nature of the problem by means of complex models.
- These tools should represent the best compromise between the level of fidelity of the results and the time needed to analyze the large number of conditions taken into consideration during the early phases of the project.

To improve the analysis and the fidelity of the results as soon as possible in the project, CFD and CSD are also becoming widely used in the conceptual design. This could look like a simple solution, but the difficulties during the preliminary design are not only linked to the complexity of the models, but also to the number of analysis to consider and to the number of configurations and different designs to analyze. CFD and CSD are useful tools, but they need complex mesh and models to be used and the definitions of such models is time-consuming. For the conceptual design, several frameworks exist and offer not only a tool for analysis but also a complete framework for the definition of models, optimizations and post-processing, like NeoCASS (Cavagna *et al.*, 2011), and the CEASIOM packages

The current issue and full text archive of this journal is available on Emerald Insight at: **www.emeraldinsight.com/1748-8842.htm**

Aircraft Engineering and Aerospace Technology: An International Journal
88/5 (2016) 649–664

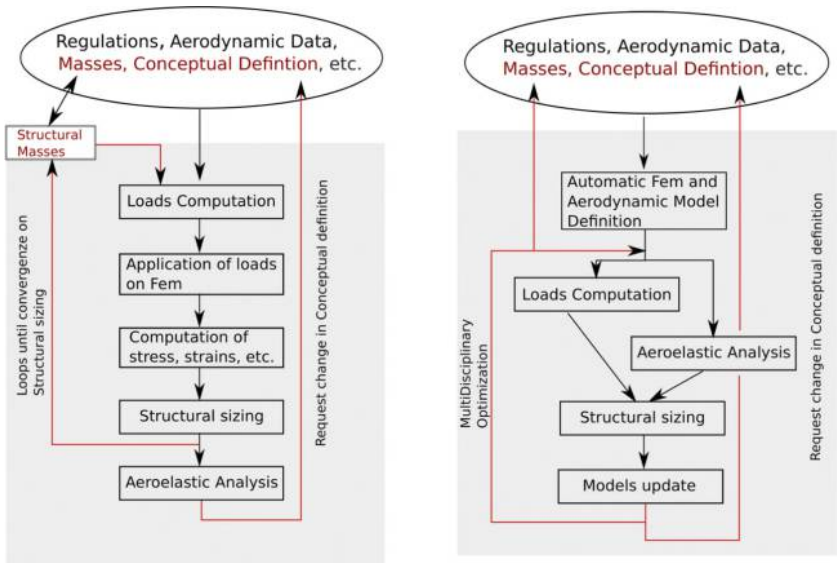
[DOI 10.1108/AEAT-02-2015-0061]

Received 28 February 2015
Revised 13 February 2016
Accepted 12 May 2016

(CFS Engineering, 2010) developed during the SimSAC European project or SUAVE (Aerospace Design Lab) developed by the Aerospace Design Lab at Stanford University (2013). Until few years ago, there was a lack of this kind of framework for the preliminary design, that was able to handle a higher level of complexity and model fidelity. In the recent past, several research laboratories are working to fill this gap between conceptual and preliminary design:

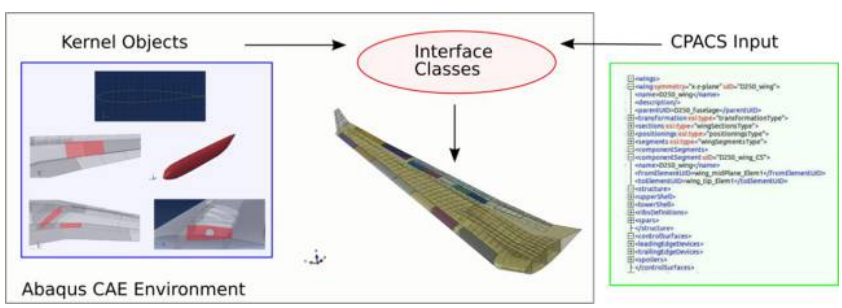
- the Integrated Aircraft Design laboratory of the DLR (Dieter Kohlgrueber) Air Transportation Systems Institution has developed several tools for the multidisciplinary design of new aircrafts. Besides the development of the Common Parametric Aircraft Configuration Schema (CPACS) file format (DLR, 2013), (Nagel *et al.*, 2012), almost a standard in the European projects, they developed several tools for the geometry definition and manipulation [TiGL TIVA Geometry Library (DLR, TIGL)], for the definition of detailed finite element (FE) structural models [ELWIS (Dorbath *et al.*, 2011)], for the multidisciplinary optimization process [RCE Remote Component Environment (DLR, RCE, 2011)] and for the sizing of wings under multidisciplinary constraints (Dorbath *et al.*, 2012);
- The multi-disciplinary design and analysis optimization (MDAO) research laboratory (Martins, 2010) of the University of Michigan has developed a full framework for the multidisciplinary optimization of an aircraft (Liem *et al.*, 2015; Chen, 2015) that is able to optimize the external shape of an aircraft to minimize the drag or the fuel consumption or the structural mass. In their work, they use an adjoint method for the optimization, thanks to the possibility to directly access their core solvers (CFD and finite element method (FEM)). Even if their approach is extremely multidisciplinary, while the aerodynamic aspect of the problem is extremely high-fidelity, the structural part of the problem seems to not be the focus of their studies.
- One of the first attempt to define a multidisciplinary environment for the aircraft design has been started at Airbus-Military, by the software LAGRANGE (Daoud *et al.*, 2012; Schuhmacher *et al.*, 2012), dedicated to the multidisciplinary design of airframes. In the recent past, they also started to develop a model generator (Deinert *et al.*, 2013) using a CPACS parametrization and the OpenCASCADE library.

Figure 1 Preliminary design



Notes: Typical logic flow (left), proposed approach (right)

Figure 2 PyGFEM: main view



Aircraft Engineering and Aerospace Technology: An International Journal
Volume 88 · Number 5 · 2016 · 649–664

All these works highlight the need to introduce high-fidelity tools and multidisciplinary process in the design of a new aircraft as soon as possible. The present work describes the development and an application of *PyPAD*, a framework dedicated to preliminary airframe design, fully developed using Python and taking advantage of Abaqus-CAE environment. The framework has a special emphasis on the design, loads and aero-elasticity and structural sizing. All other aspects are not neglected, but considered as inputs, constraints or output requests. Clearly, even considering a partial aspect of the project, the problem is still strongly multidisciplinary. The traditional approach iterates manually across models definition, loads' computation, aero-elastic analysis and structural sizing. This could lead to a slow convergence to the best solution or, even worst, the solution found could not be the optimal one. Nowadays, thanks to computing power and engineering knowledge, the preliminary design process can be approached in a fully automatic way, taking advantage of the multidisciplinary design optimization methods. Figure 1 depicts two logical flows, summarizing the traditional approach to preliminary design problem and the one here proposed. The main disadvantages of the first one are:

- All the steps are performed manually and by different departments inside a typical small, medium-size aircraft company. Engineers deal with repetitive works and the processes are error prone.
- A change in the conceptual design (because of aero-elastic behaviors or other problems) is very expensive. It is therefore hard to investigate different conceptual designs and the solution found could be not the best one.
- All the steps are performed manually and by different departments inside a typical small- or medium-sized aircraft company.

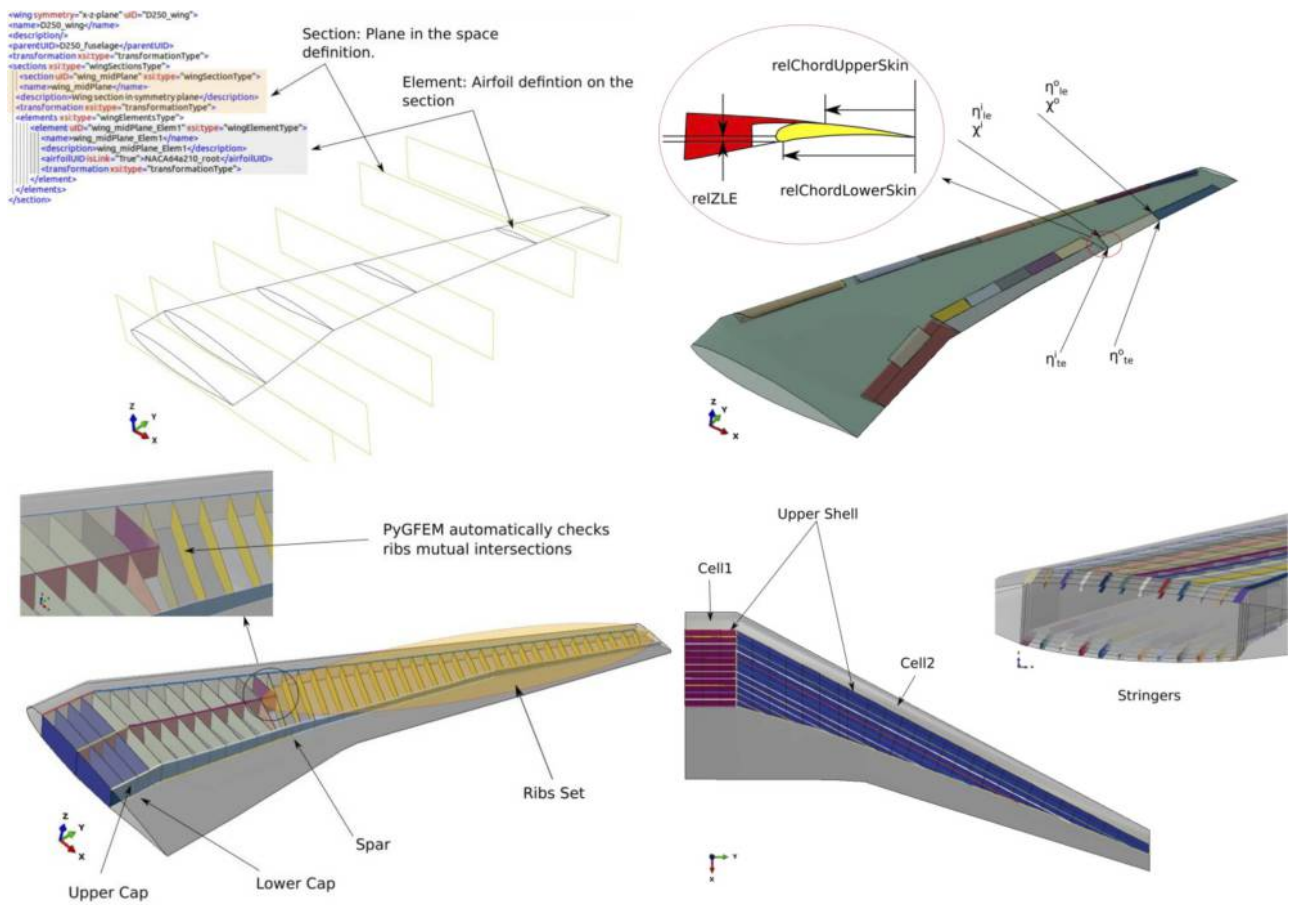
The proposed approach tries to solve all the highlighted problems:

- All the steps are performed using multidisciplinary optimization algorithms.
- Both structural and aerodynamic models (for loads computation and aero-elastic analysis) are defined in a fully automatic way. Therefore, different conceptual designs can be investigated.
- Aero-elastic behavior is considered in the sizing process.

To achieve these results, a multidisciplinary framework called *PyPAD* that is composed of different modules, is under development. The three main modules included in the full framework are:

1 *PyGFEM* (a *Py*thon module for the *G*eneration of *F*inite *E*lement *M*odels): It is an object-oriented tool developed under Abaqus-CAE environment that is able to define

Figure 3 Wings' definition



Notes: Sections and elements (top left); movables surfaces (top right); spars and ribs (bottom left); skins and stringers (bottom right)

Aircraft Engineering and Aerospace Technology: An International Journal
Volume 88 · Number 5 · 2016 · 649–664

both structural and aerodynamic models starting from CPACS file input.

2 *PyAERO* (a *Py*thon module for the *AERO*elastic analysis): It is a tool able to perform all the analyses needed to compute loads and aero-elastic responses.

3 *PySIZE* (a *Py*thon module for the structural sizing): It will be a module dedicated to the sizing of the different components.

PyPAD runs embedded in the Abaqus-CAE environment, and so it takes advantage of the large library of low-level routines for FEM manipulation that is accessible via Python calls.

One of the key factor in defining a procedure that is able to quickly generate aircraft models for aero-structural analysis at different levels of fidelity is represented by the strategy adopted for aircraft description and parameterization. In this work, the parameterization called CPACS (DLR, 2013) proposed by DLR is adopted. CPACS is a data definition schema for the air transport system based on a single XML file.

This data definition schema is becoming a standard because of its extendibility and completeness. Extensive data can be stored in a CPACS file; the most important ones for this work are the geometry definitions, both external lofts and structural configurations, and the related properties definition. In the following, a quick overview of two of the main modules of *PyPAD* are reported with some specific examples.

Aero-structural models' generation: *PyGFEM* module

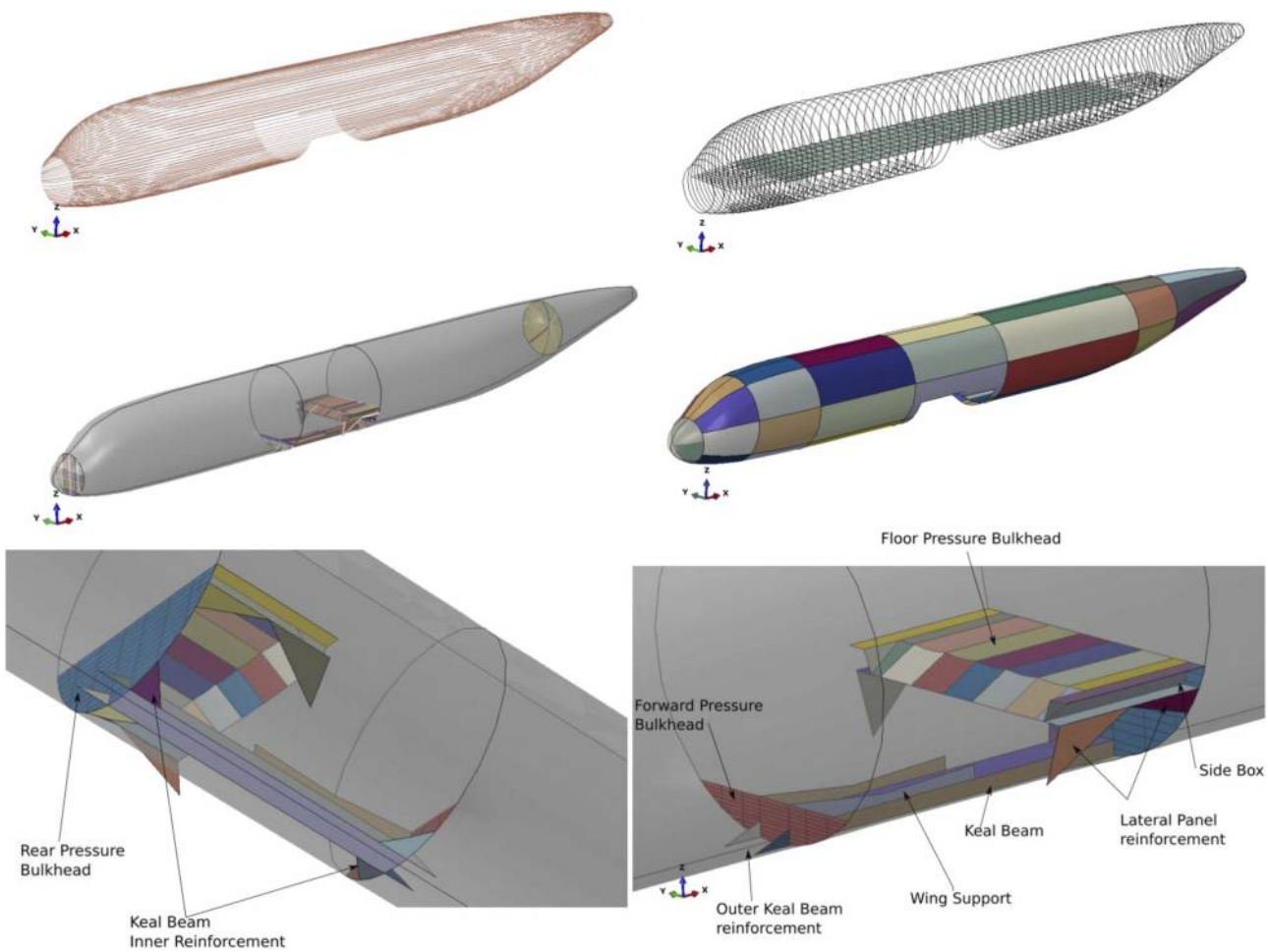
PyGFEM is the module dedicated to the automatic generation of the full finite element model of the complete aircraft. The capability to quickly generate a large spectrum of aerostructural models is a must because of the need to find the best aircraft optimization by means of different iterations and multi-disciplinary optimization runs. Figure 2 depicts the main idea driving the development of *PyGFEM*, where the main core can be divided in two branches:

1 The core classes providing the definition of basic objects, like edges, airfoils, planes, cuts, lofts, materials, properties, etc.

2 The interface classes providing the coupling between CPACS input and core objects.

In such a way, the core objects have been developed without constraints by the inputs, providing generality and reusability. The task to represent the final objects starting from the CPACS file is carried out by the interface classes; the only ones depending on the input file. First of all, *PyGFEM* reads from CPACS file all

Figure 4 Fuselages' definition



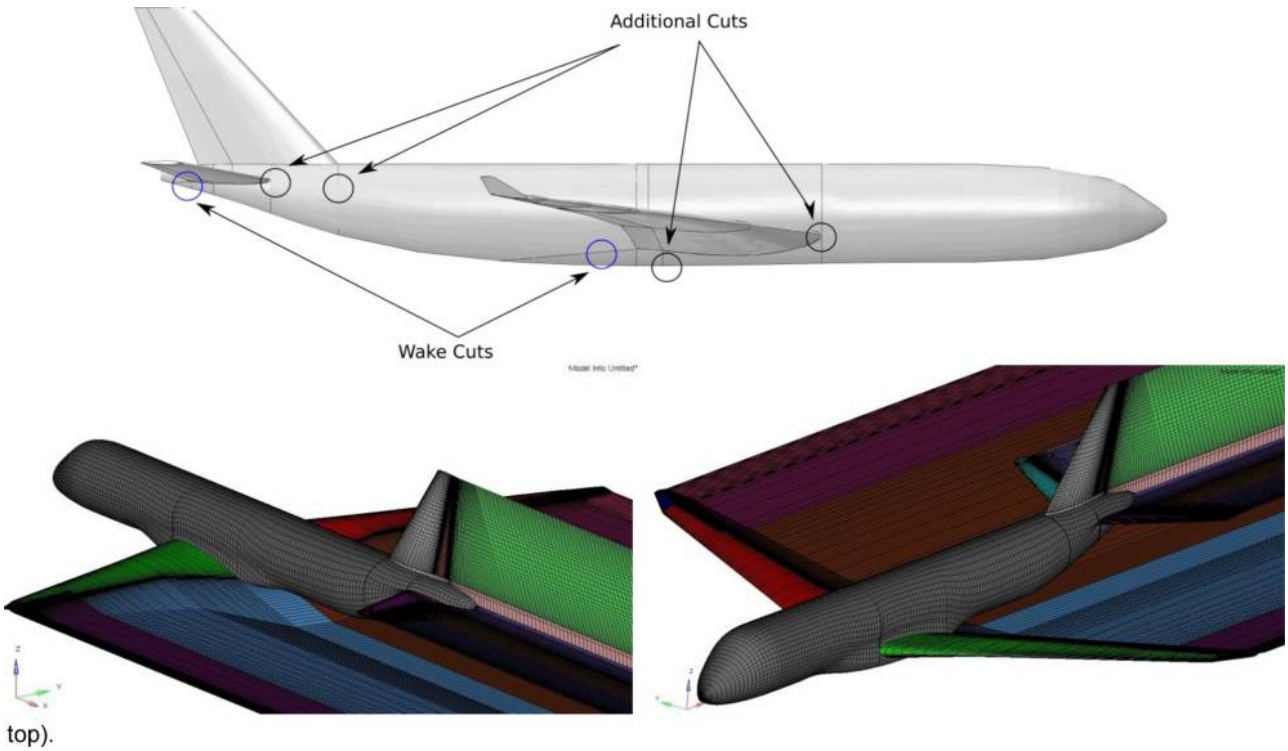
the information concerning airfoils (both for wings and fuselages), materials, beam profiles and properties definitions, common data needed for the definitions of all the others objects. Once all these data are stored in the Abaqus-CAE database, *PyGFEM* can define wings and fuselages.

Wing

The loft definition (for both wings and fuselages) is described by sections and elements. The section defines a plane in the space, and each section can hold several elements, and each of them defines an airfoil on its plane. To drive the loft, the leading and trailing edges are defined and used as path for the loft extrusion (Figure 3 top left). Once the external loft is defined, the movable surfaces (leading edges and trailing edges devices and spoilers) can be defined (Figure 3 top right).

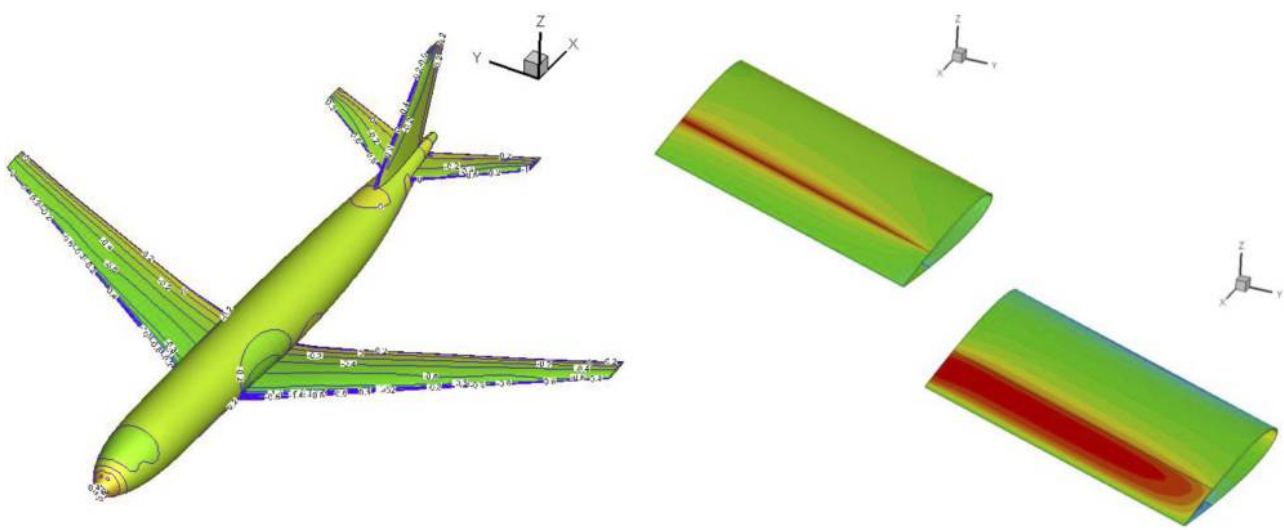
The idea is to take advantage of the same classes used to describe the wing lofts. The user defines inboard and outboard positions (relative to the "parent" wing), while for the

Figure 5 Aerodynamic mesh



top).

Figure 6 SUnPaM results



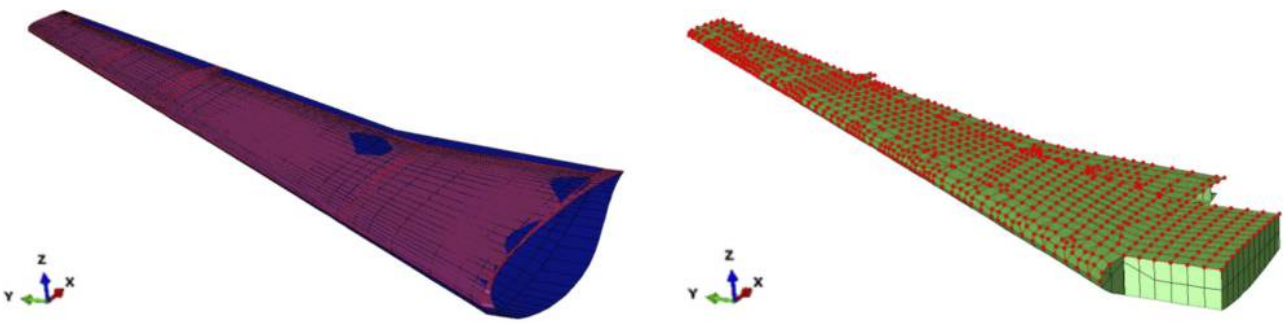
Notes: Steady solution (left); unsteady solution (imaginary and real part, right)

definition of the 2D geometry, he can choose airfoils (among the ones defined) or can define them starting from the wing geometry. In the last case, *PyGFEM* automatically computes the correct airfoils and defines the movable surfaces using the same approach used for the wing. In the end, it modifies the parent loft by removing the faces belonging to the new component. Also, the leading edge devices are defined using this idea, but in this case, a new surface is added to the parent geometry, to redefine the structural leading edge. It is important to underline that the movable surfaces are objects derived by the same class of the wing one; therefore, it is possible to define other devices referring to a movable surface (like double-slot flaps). Once all the movable surfaces are defined, *PyGFEM* starts assembling the structural model of the several components (wings, movables surfaces and fuselages). For the wings, the principal structural component are spars, ribs, skins and stringers.

PyGFEM can define spars that are arbitrarily oriented in the space, with arbitrary property partition. On the upper and lower edge of the spar, if specified, two stringers, representing the lower and the upper caps, are added. Ribs are collected in a set, and if specified, each rib, in the ribs set, can have a different property.

As for spars also, the ribs can be defined arbitrarily in space. The ribs can start from each characteristic geometry, like leading edge and spars, can end on a trailing edge and spars, and if specified, it can end when *PyGFEM* finds an intersection with another rib. Moreover, it is possible to force the rib to be perpendicular to a reference geometry (leading edge, trailing edge, a spar and the local y axis) (Figure 3, bottom left). Knowing the position of the spars and ribs, it is possible to define skins and stringers. CPACS provides a nested data structure for the description of the skins. The basic object of these data is the simple skin that can be defined using spars and ribs, with leading and trailing edges as the border lines, or by the definition of bounding parametric points on the loft. Inside this object, the user can define (using relative parameters) other skins with different properties. Besides, in each skin, the user can define a stringers set specifying the number of stringers or the pitch between stringers or each relative position. Moreover, the stringers can

Figure 7 Aero-elastic coupling sets



Note: Aerodynamic (left) and structural (right)

Figure 8 Flutter: Agard 445.6 wing test

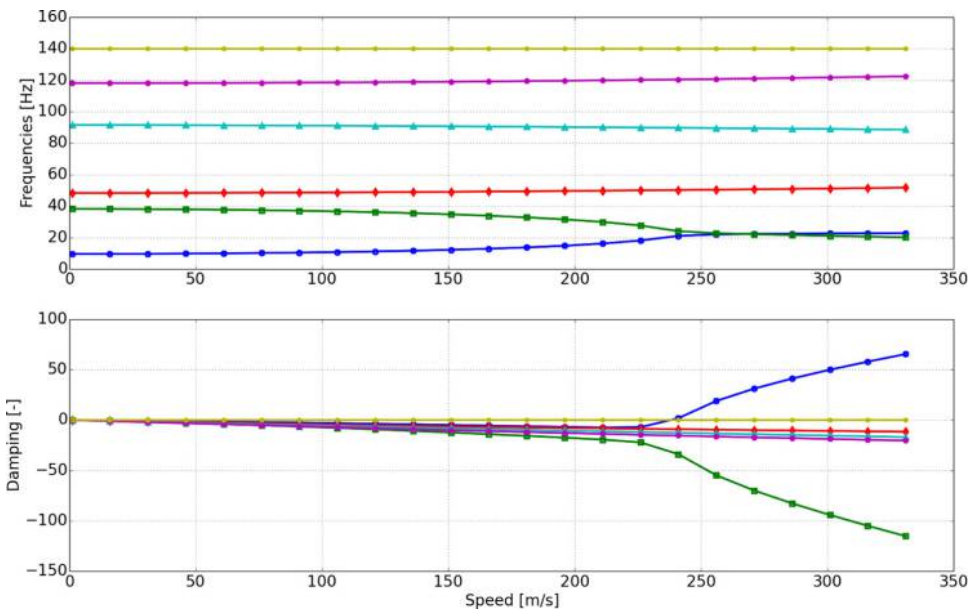
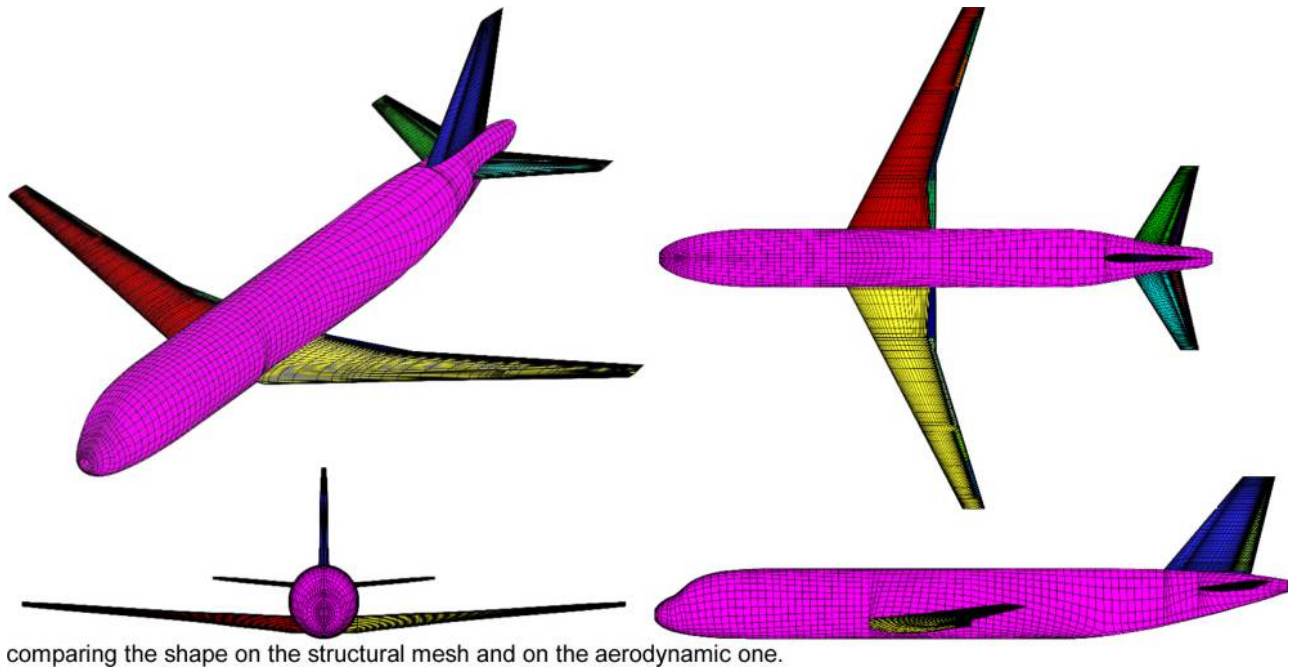


Table I Mass and geometry properties

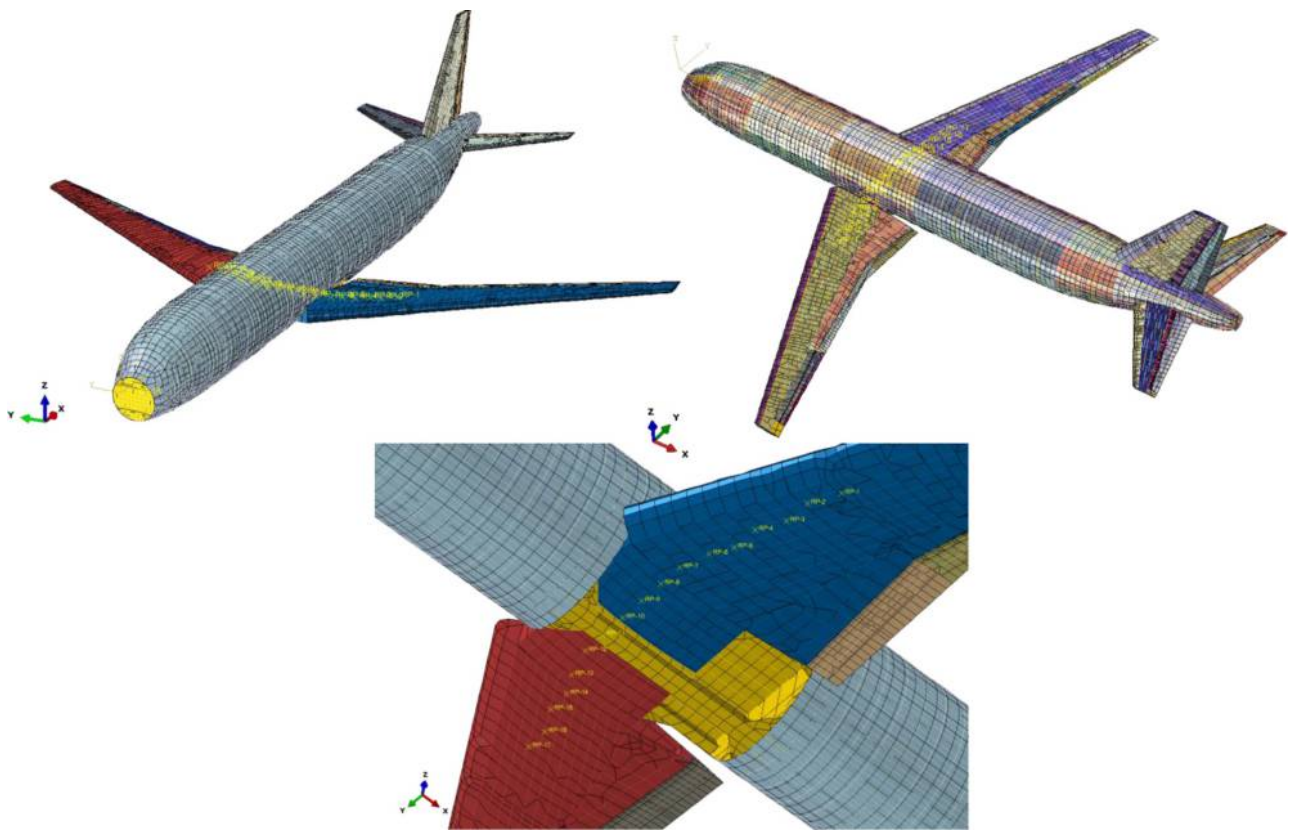
M [kg]	I_{xx} [Kgm^2]	I_{yy} [Kgm^2]	I_{zz} [Kgm^2]	I_{xz} [Kgm^2]	S [m^2]	c [m]	b [m]
48475	961948.5	1895684	2768967	−133950.7	122.4	4.2	16

Figure 9 Structural model



comparing the shape on the structural mesh and on the aerodynamic one.

Figure 10 Aerodynamic model



be rotated with respect to the direction defined by the leading edge (Figure 3 bottom right).

Fuselage

Regarding the structural definition of the fuselage, a different approach was used. While for the wing, all the structural objects were defined on the same part, sharing edges and points, for the fuselages, different parts are defined and then linked together by means of rigid links. Using this approach, the time needed by the tool to define the model is substantially reduced. This because the whole execution time is driven by the stringers' (and frames) definition, particularly by cuts on

Figure 11 First three elastic modes, structural and aerodynamic mesh

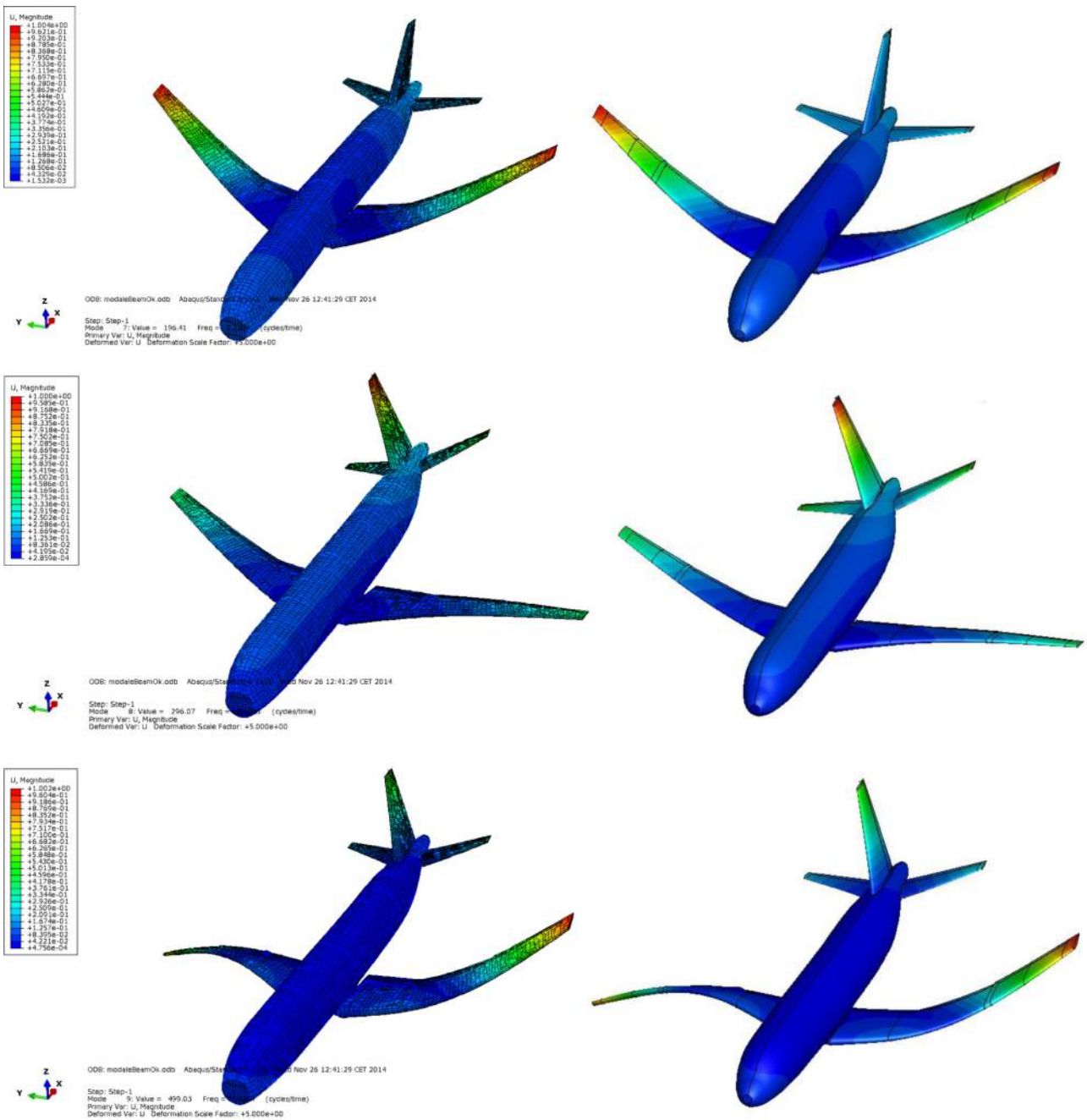


Table II Stability and control derivatives, rigid model

Mach	$C_{l\alpha}$	$C_{M\alpha}$	$C_{l\delta_e}$	$C_{M\delta_e}$	$C_{y\beta}$	$C_{L\beta}$	$C_{N\beta}$	$C_{y\delta_r}$	$C_{L\delta_r}$	$C_{N\delta_r}$	$C_{L\delta_{a2}}$
0.3	5.65	−1.876	0.455	−1.895	−0.66	0.255	−0.259	−0.304	0.082	−0.336	0.167
0.5	5.896	−1.973	0.47	−1.976	−0.68	0.26	−0.274	−0.315	0.0849	−0.349	0.172

Table III Stability and control derivatives, elastic model

Mach	$c_{l\alpha}$	$c_{M\alpha}$	$c_{l\delta_e}$	$c_{M\delta_e}$	$c_{y\beta}$	$c_{L\beta}$	$c_{N\beta}$	$c_{y\delta_r}$	$c_{L\delta_r}$	$c_{N\delta_r}$	$c_{L\delta_{a2}}$
0.3	5.67	−1.868	0.455	−1.895	−0.662	0.254	−0.26	−0.305	0.082	−0.337	0.167
0.5	4.74	−1.484	0.504	−1.636	−0.649	0.244	−0.243	−0.29	0.0956	−0.327	0.0999

the surfaces. The number of stringers on the wing could be very high, but it is possible to perform the cut of all the stringers at the same time, saving a lot of computational time. The same approach is not feasible on the fuselage because of the curvature of the geometry and the mutual cuts between stringers and frames.

At the moment, *PyGFEM* is able to define a complete FE model of a transport aircraft fuselage, and the classes are easily extendible to others type of fuselages. At the moment, the limitation is on the input file. Figure 4 shows all the different parts and details of the central fuselage. The floor structure is a modeled coupling beam and plate elements. Cross beams, strut beams and floor beams are described by beam elements, while the floor is described by plate elements (Figure 4; top left). *PyGFEM* automatically defines the central fuselage, the part where the wing is linked to the fuselage (Figure 4 bottom) using several objects:

- The keel beam and its reinforcements defined to restore the bending stiffness is decreased by the cut for the wing box.
- The forward and rear pressure bulkheads are defined to carry on the pressure load of the cabin.
- The side boxes are defined to restore the lateral bending stiffness.
- The lateral panel reinforcements are defined to stiffen the main landing gear bay.
- The floor pressure bulkhead is defined to carry the pressure load of the cabin.

Properties and sets

The database defined in the Abaqus-CAE environment can be managed and modified by the user using PyGFEM. The objects defined are able to handle all the properties of the model; moreover, all the objects hold information about their geometry and mesh using several definition sets. This will be very helpful during an optimization study to run analyses on local objects using *ad hoc* solutions.

Table IV Level flight trim solutions

Mach = 0.3	α [deg]	δ_e [deg]	Mach = 0.5	α [deg]	δ_e [deg]
Rigid	4.95	−5.694	Rigid	0.535	−1.20
Elastic	4.92	−5.64	Elastic	1.245	−2.016

Table V Pull-up trim solutions

Mach = 0.3	α [deg]	δ_e [deg]	Mach = 0.5	α [deg]	δ_e [deg]
Rigid	15.09	−16.13	Rigid	4.03	−4.84
Elastic	15.02	−16	Elastic	5.73	−6.30

Aerodynamic models

PyGFEM can define a structured aerodynamic mesh to be used as input for *SUnPaM* (*S*ubsonic *Un*steady *Pa*nel *M*ethod), the aerodynamic solver available inside *PyPAD*, based on the Morino's method. Figure 5 (bottom) shows two views of the result obtained on a test case. Starting from the different parts, the program merges all the components finding each intersection, then starts to check all the different couples of intersecting parts, adding construction cuts to simplify the geometry or cuts to define the projection of wings wake on the other part involved in the intersection (Figure 5 top).

Loads and aero-elasticity: *PyAERO* module

PyAERO is a module developed to compute all the aero-elastic responses, such as trim, flutter, dynamic analysis, to get all the loads and all the responses needed for structural sizing. *PyAERO* uses the structural model defined by *PyGFEM* and the aerodynamic database computed using *SUnPaM*. All the routines are written in Python, but to get better computational time performances, some core functions are written in FORTRAN, to take advantages of numerical libraries like Lapack, Optimized Blas and FFTW and to exploit the power of parallel computing using OpenMP.

SUnPaM

SUnPaM (*S*ubsonic *Un*steady *Pa*nel *M*ethod) is a panel aerodynamic solver, based on the Morino method (Morino *et al.*, 1975; Morino and Kuo, 1974) and developed in the Department of Aerospace Science and Technology of Politecnico di Milano. The tool solves linearized subsonic compressible flow both in steady and unsteady conditions. The tool can exploit the power of modern multi-core CPU, both using external libraries for linear algebra [OpenBlas, Plasma (The University of Tennessee, 2008)] and OpenMP during the definition of the matrices. Figure 6 shows some test cases for the steady solution (full aircraft Figure 6 left) and on a single wing with oscillating flap (Figure 6 right).

Aero-elastic models

PyAERO couples the structural and the aerodynamic models coming from *PyGFEM* by Radial Basis Functions

Table VI Sideslip trim solutions

Mach = 0.3	α	δ_e	δ_{a2}	δ_r	$\ddot{y}$	Mach = 0.5	α	δ_e	δ_{a2}	δ_r	$\ddot{y}$
Rigid	4.95	−5.70	−11.56	−7.59	−1.17	Rigid	0.535	−1.20	−11.31	−7.74	−3.30
Elastic	4.95	−5.64	−11.517	−7.60	−1.17	Elastic	1.245	−2.013	−17.65	−7.059	−3.21

interpolations. Regarding the structural model, at the moment, *PyAERO* works using as input an eigenvalue solution computed by Abaqus (Figure 7).

It imports all the grids' definition (used to couple the structural model with the aerodynamic one), the modal displacements, all the modal information and the nodal mass and stiffness matrices. The definition of the aero-elastic coupling between structural and aerodynamic models becomes very simple, taking advantage of the automatic definition of both the models by *PyGFEM*. The structural model is divided in several parts, each described by several sets, and the same partition is reflected also on the aerodynamic mesh. In this way, to define an interface element between structural and aerodynamic mesh, the user must specify only the part name (of the structural model), the body name (of the aerodynamic model) and the two sets of the structural model describing the upper and lower surface. Using this information, the solver is able to compute the interfaces matrices to get displacements and velocities on an aerodynamic mesh and aerodynamic forces on the structural one.

Aero-elastic solvers

As introduced, *PyAERO* provides several aero-elastic solvers to be able to perform all the analyses requested by the preliminary sizing of an aircraft.

Trim solution

PyAERO solves the trim problem using a classical approach; therefore, assuming a steady solution on the deformable motion and considering a steady approximation of the aerodynamic. The trim problem can be written as:

Figure 12 Trim deformations and Von-Mises stress distributions level flight, pull-up and sideslip for Mach 0.3 (left) and Mach 0.5 (right)

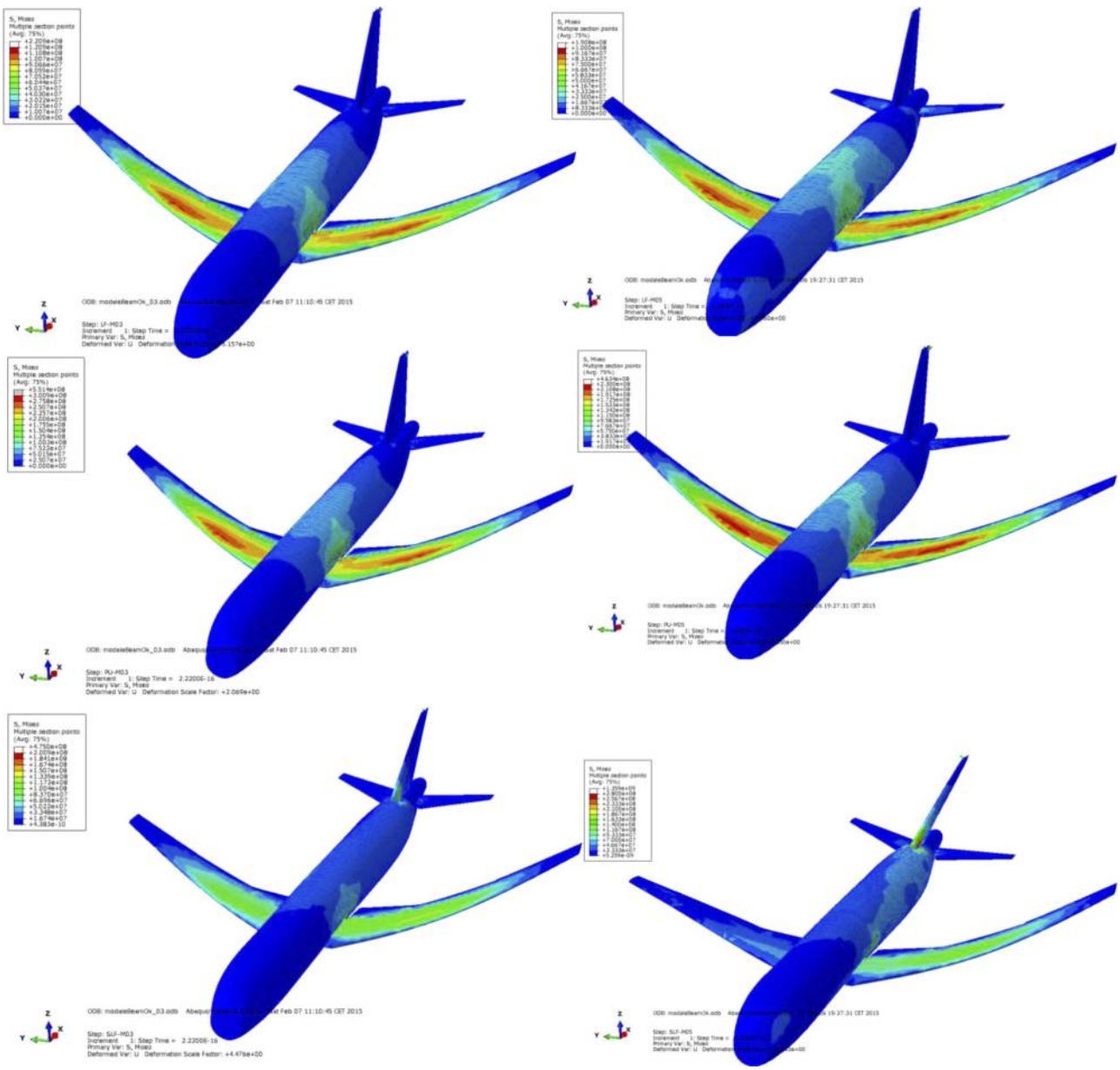
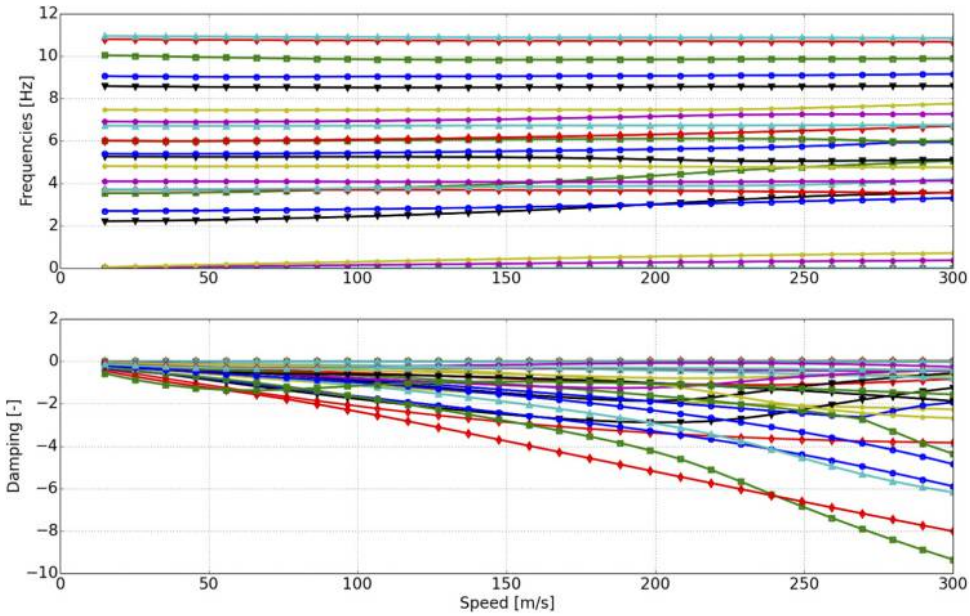


Figure 13 Flutter diagram at Mach 0.5



$$M\ddot{u} + Ku = F_0 + q_\infty Q_{su} u + q_\infty Q_{sx} u_x$$

where M and K are the mass and the stiffness matrices, respectively, u is the vector of nodal displacements, $\boldsymbol{F}_0$ is the vector of initial nodal forces, q_∞ is the dynamic pressure, Q_{su} is the steady approximation of aerodynamic forces due to nodal displacements and Q_{sx} is the steady approximation of aerodynamic forces due to mechanical flight degree of freedom u_x (like α, β, p, q, r, etc.). Given the complexity of the structural model, the nodal displacements, $\boldsymbol{u}$, are described using a modal bases:

$$u = Uq$$

Where $\boldsymbol{U}$ is the modal base and $\boldsymbol{q}$ is the modal amplitude. The modal base is defined considering also the six rigid degrees of freedom, described in physical coordinates. Replacing the approximation of u in the trim equation:

$$MU\ddot{q} + KUq = F_0 + q_\infty Q_{su} Uq + q_\infty Q_{sx} u_x$$

To improve computation performances, *PyAERO* directly computes (using *SUnPAM*) the aerodynamic matrix $\widetilde{Q}_{su} = Q_{su}U$.

Thanks to the orthogonality properties of eigenvector, scaling the previous equation by U^T, the problem becomes:

$$\begin{bmatrix} M^r & 0 \\ 0 & m^l \end{bmatrix} \begin{bmatrix} \ddot{q}_r \\ \ddot{q}_l \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & k^l \end{bmatrix} \begin{bmatrix} q_r \\ q_l \end{bmatrix} = \begin{bmatrix} F_0^r \\ F_0^l \end{bmatrix} + q_\infty \begin{bmatrix} \check{Q}_{su}^{rr} & \check{Q}_{su}^{rl} \\ \check{Q}_{su}^{lr} & \check{Q}_{su}^{ll} \end{bmatrix} \begin{bmatrix} q_r \\ q_l \end{bmatrix} + q_\infty \begin{bmatrix} \check{Q}_{sx}^{r} \\ \check{Q}_{sx}^{l} \end{bmatrix} u_x$$

Where M^r is the global barycentric mass matrix, m^l is the diagonal modal mass matrix associated to flexible modes and k^l is the diagonal stiffness matrix. Under the hypothesis of steady solution of the deformable part, the problem has $2nr + nl + nx$ degrees of freedom with $nr + nl$ equations, where nl, nr and nx are the number of modes, the number of rigid degrees of freedom (six) and the number of flight mechanics degrees of freedom, respectively. The six unknowns, q_r, can be

Figure 14 Gust condition: center of gravity (CG) responses

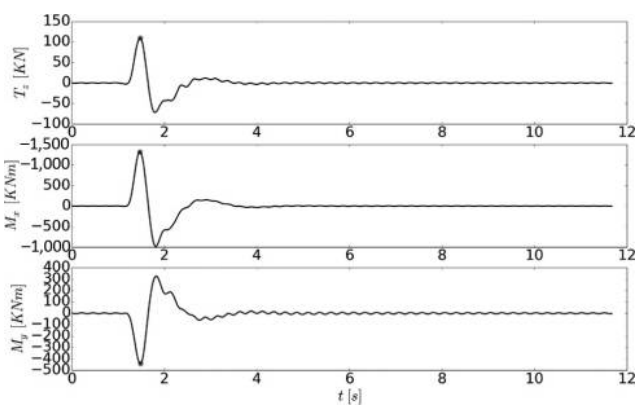
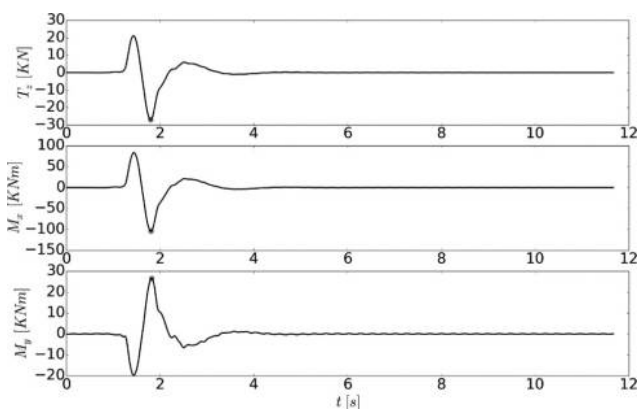


Figure 15 Gust condition: wing root responses



Aircraft Engineering and Aerospace Technology: An International Journal
Volume 88 · Number 5 · 2016 · 649–664

neglected considering that the gravity center position and the rotation around the x axis (speed direction) do not give aerodynamic loads, while the pitch and yaw rotation are already described by α and β angle of the flight mechanics set, respectively. Therefore, the number of unknowns is $nr + nl + nx$ with $nr + nl$ equations. Therefore, to have a close problem, the user must fix nx unknowns.

Once the trim problem is solved, using $\widetilde{Q}_{su}$ and Q_{sx}, it is possible to compute the nodal aerodynamic loads. Moreover, knowing the rigid accelerations and the nodal mass matrix, it is also possible to compute the inertia nodal loads. In this way, for every trim condition, the user can export the corresponding nodal loads to compute the corresponding stress solution.

Figure 16 Gust conditions: horizontal tail responses

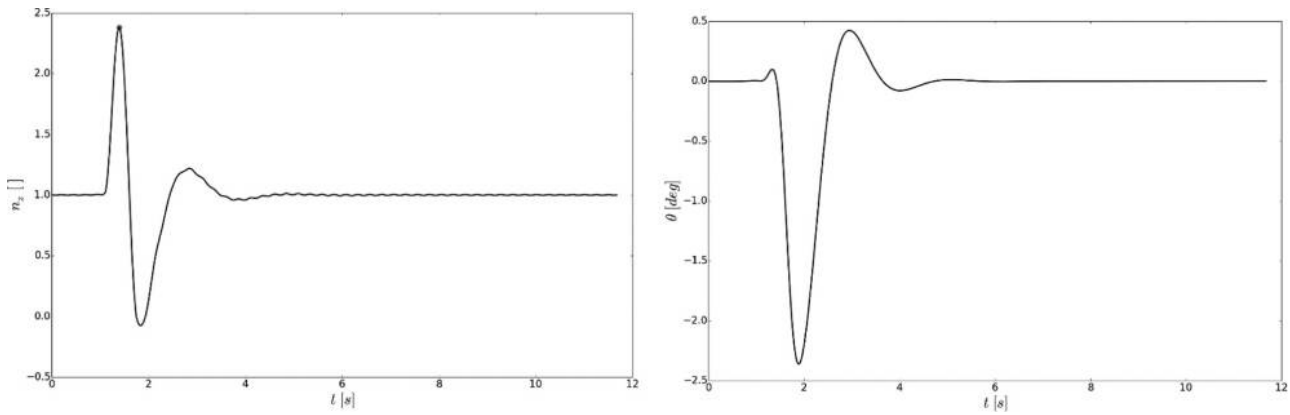


Figure 17 Gust conditions: fuselage responses

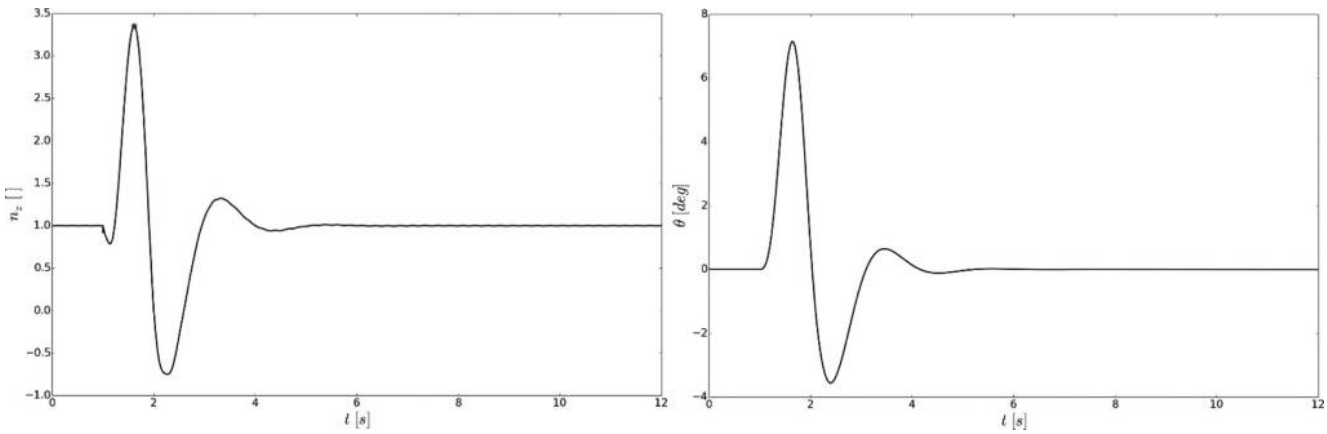
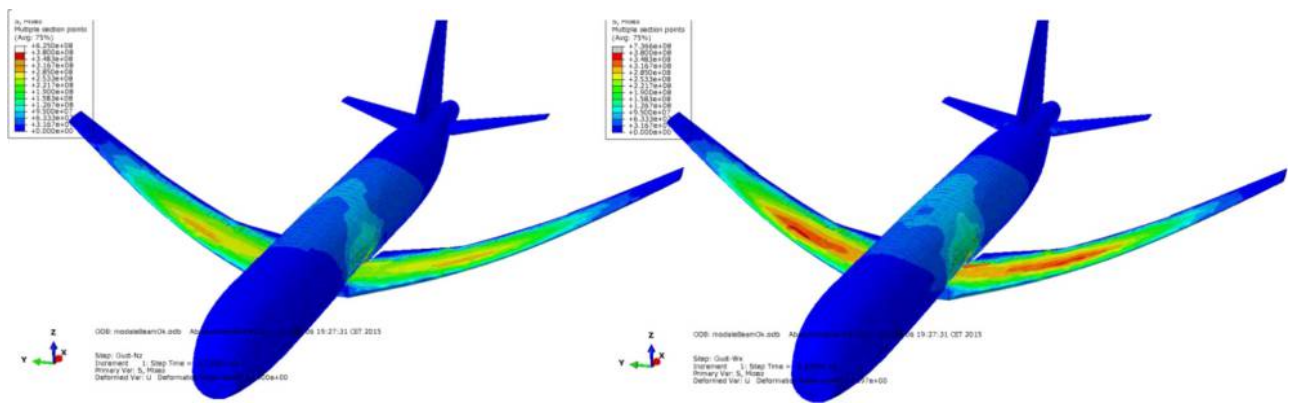


Figure 18 Gust condition: deformation and Von-Mises stress for maximum load factor (left) and maximum wing bending moment (right)



Aircraft Engineering and Aerospace Technology: An International Journal
Volume 88 · Number 5 · 2016 · 649–664

Flutter

PyAERO includes a dedicated well-know (MSC.NASTRAN; Harder *et al.*, 1979) p-k futter solver to get the best compromise between solution robustness and time efficiency. The flutter problem can be written as:

$$(s^2M + sC + K - q_\infty Q_{qq}(Mach, p))q = 0$$

Where M, C and K are the modal mass, damping and stiffness matrices, respectively, and p is the complex-reduced frequency:

$$p = \frac{sc}{2V_\infty} = g + k$$

With c being the reference cord and V_∞ the flight speed.

The aerodynamic matrix is computed at several pure harmonic reduced frequencies. The p-k method considers the Q_{qq} matrix as the sum of the real part Q^R_{qq} and the imaginary part Q^I_{qq}. Considering the imaginary part, it is possible to write:

$$q_\infty Q^I_{qq} = \frac{\rho}{2}V_\infty^2 Q^I_{qq}\frac{k}{k} = \frac{\rho}{4}cV_\infty\frac{Q^I_{qq}}{k}\omega \simeq \frac{\rho}{4}cV_\infty\frac{Q^I_{qq}}{k}s$$

Using this approximation, the flutter equation becomes:

$$\left(s^2M + s\left(C + \frac{\rho}{4}cV_\infty\frac{Q^I_{qq}(Mack, k)}{k}\right) + K - q_\infty Q^R_{qq}(Mach, k)\right)q = 0$$

The problem now is defined by pure real matrices and it can be solved by a normal eigenvalue algorithm. The solution must iterate the different eigenvalues to get the correct Q_{qq} approximation at the correct reduced frequencies k. Figure 8 shows a flutter diagram on the AGARD 445.6 wing test (Yates *et al.*, 1963; Yates, 1987) performed for validation purpose.

Dynamic response

The dynamic modal problem in the frequency domain can be described by:

$$(\omega^2M + j\omega C + K - q_\infty Q_{qq}(Mach, k))q = F(\omega)$$

Where $F(\omega)$ can be both deterministic and stochastic vector of forces.

At the moment, in *PyAERO*, the user can simply define deterministic gusts, prescribed control surfaces motions and nodal forces. Once the reduced modal problem is solved, the software is able to compute nodal responses on the structural model in terms of both displacements and forces. The outputs can be displacements, velocities, accelerations, total force on specific set of nodes, total force on a part and nodal forces. Moreover, the user, to save memory, can choose to get the wanted output only at a fixed time step. The dynamic response can be solved also in a time domain, computing a state-space aerodynamic model starting from the frequencies database of $Q_{qq}(Mach, k)$.

PyAERO takes advantages of the modern techniques developed in our department in the past years (Ripepi and Mategazza, 2013; Ripepi, 2014) to compute the aerodynamic state-space representation. The aerodynamic model in time domain can be described by:

$$\dot{x}_A(t) = 2\frac{V_\infty}{c}A^A x_A(t) + 2\frac{V_\infty}{c}B^A q(t)$$

$$f_A(t) = q_\infty\left(C^A x_A(t) + D^A_0 q(t) + \frac{c}{2V_\infty}D^A_1\dot{q}(t) + \left(\frac{c}{2V_\infty}\right)^2 D^A_2\ddot{q}(t)\right)$$

Where $x_A(t)$ is the aerodynamic state; A^A, B^A, C^A, D^A_0, D^A_1 & D^A_2 are the matrices of the aerodynamic state space model; and $f_A(t)$ is the vector of the aerodynamic force acting on the structural modal coordinates.

In the time domain, the dynamic response is described by the equation:

$$M\ddot{q}(t) + C\dot{q}(t) + Kq(t) = f_A(t) + f_G(t)$$

Where $f_G(t)$ is the vector of gust loads. Coupling the aerodynamic system and the structural one, the aero-elastic model becomes:

$$\begin{bmatrix}\dot{x}_A\\ \dot{q}\\ q\end{bmatrix} = \begin{bmatrix}2V_\infty/cA^A & 2V_\infty/cB^A & 0\\ 0 & I & 0\\ q_\infty\hat{M}^{-1}C^A & \hat{M}^{-1}\hat{K} & \hat{M}^{-1}\hat{C}\end{bmatrix} + \begin{bmatrix}0\\ 0\\ \hat{M}^{-1}\end{bmatrix} f_G$$

Defining:

$$\hat{M} = M - q_\infty\left(\frac{c}{2V_\infty}\right)^2 D^A_2$$

$$\hat{C} = C - q_\infty\frac{c}{2V_\infty}D^A_1$$

$$\hat{K} = K - q_\infty D^A_0$$

Regarding the gust input f_G, at the moment, *PyAERO* uses a hybrid approach that combines the Fourier transform and the time-domain approaches, presented by Karpel *et al.* (2005). The purpose of this approach is to obtain a time-domain state-space model without performing rational approximation to the spiral gust columns. The Fourier transform of the gust profile is computed, then the modal loads $f_G(\omega)$ are computed in the frequency domain and the vector $f_G(t)$ is obtained by means of inverse Fourier transform.

Application

A test case involving all the different capabilities of the framework just described is reported. The test case is based on a CPACS model of a transport aircraft, courtesy Dieter Kohlgrueber (DLR), that is used for the development of the models and analysis reported in by Scherer *et al.* (2013). Table I summarizes the principal data of the model.

Structural and aerodynamic models

The structural model is defined by 21 parts, 29,945 nodes and 40,969 elements. Figure 9 depicts some views of the model and the details of the wing fuselage interface. Figure 10 shows several views of the aerodynamic mesh used both for steady and unsteady analysis. The total number of panels is approximately 20,000. The aero-elastic model is defined using the first 25 normal modes of the aircraft. Figure 11 shows the

Aircraft Engineering and Aerospace Technology: An International Journal
Volume 88 · Number 5 · 2016 · 649–664

first three deformable normal modes of the model, comparing the shape on the structural mesh and on the aerodynamic one.

Trim analysis

Six different trim analysis (all at sea level) are reported:

- 2 level flight conditions at Mach 0.3 and Mach 0.5;
- 2 pull-up conditions (nz = 2:5) at Mach 0.3 and Mach 0.5; and
- 2 sideslip conditions with b = 10° at Mach 0.3 and Mach 0.5.

For all the conditions, the nodal loads are computed and exported in Abaqus format. These loads are then used to perform a static analysis in Abaqus to get the stress solution. Tables II and III report the principal stability and control derivatives for the rigid and elastic models, Table IV reports the trim solutions (rigid and elastic model) for the level flight conditions, Table V reports the trim solutions for the pull-up conditions and Table VI reports the solutions for the sideslip analysis [angles in degree and accelerations in (m/s^2)]. Figure 12 shows the displacement solution and the Von-Mises stress distribution for the different load cases.

Figure 19 Elevator maneuver: CG responses

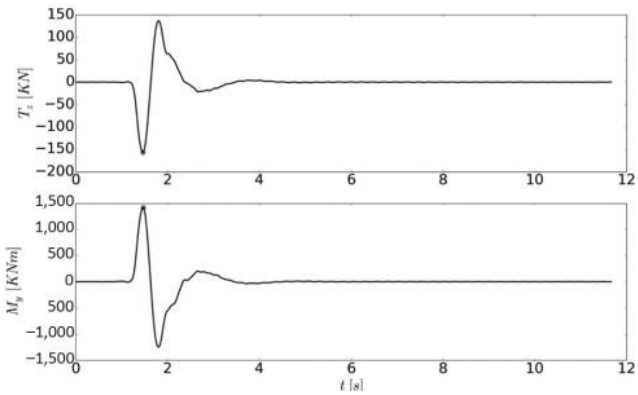


Figure 20 Elevator maneuver: wing root responses

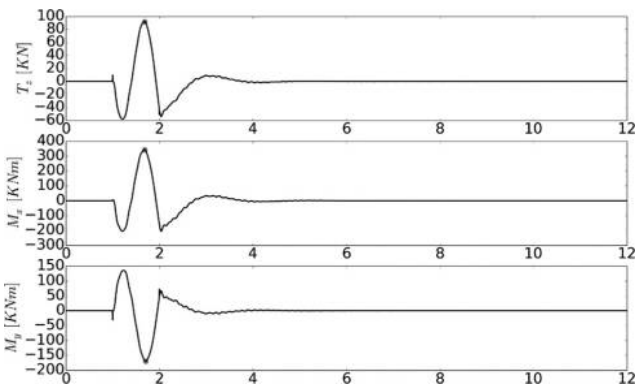
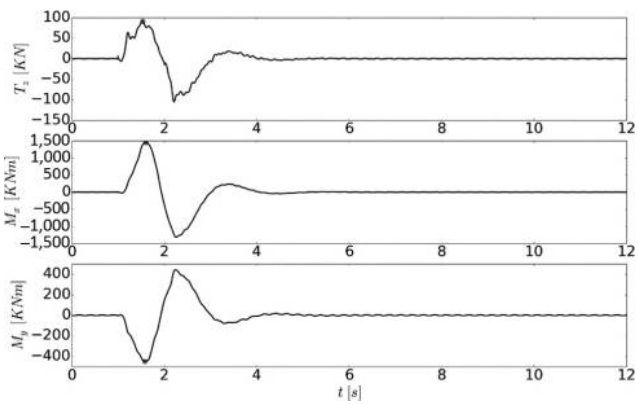


Figure 21 Elevator maneuver: horizontal tail responses



Dynamic analysis

Before running the dynamic responses, the flutter analysis is computed at Mach 0.5. The analysis is performed with fixed movable surfaces. Figure 13 shows the flutter diagram. Once checked that the model has no instabilities, several dynamic analyses are performed, and two of them are reported here:

1 Discrete gust condition, $v_g = 15(1 - \cos(3\pi t))[m/s]$, Mach = 0.5.
2 Elevator rotation, $\delta_e = 15\sin(2\pi t)[deg]$, Mach = 0.5.

The outputs requested are:

- the wing root internal loads;
- the horizontal tail root internal loads; and
- the internal loads of the fuselage section after the wing.

All the internal loads are computed through the summation of force techniques.

For each solution, two time steps are selected; for the gust analysis, the first one corresponds to the time when the load factor reaches the maximum value and the second one when the bending moment of the wing is maximum. For the elevator maneuver, the first time step corresponds to the time when the bending moment on the horizontal tail is minimum and the second one when the bending moment of the wing is maximum. In these time steps, the nodal loads, both the aerodynamic and inertial ones (computed using the nodal mass matrix) of the whole aircraft, are exported and an Abaqus steady solution is executed to get the stress solutions.

Figures 14-17 show the time history of the outputs for the gust analysis and Figure 18 shows the deformation and the Von-Mises stress obtained at the two selected time steps.

Regarding the elevator maneuver, Figures 19-22 report the time history of the outputs and Figure 23 shows the deformation and the Von-Mises stress obtained at the two selected time steps.

Figure 22 Elevator maneuver: fuselage responses

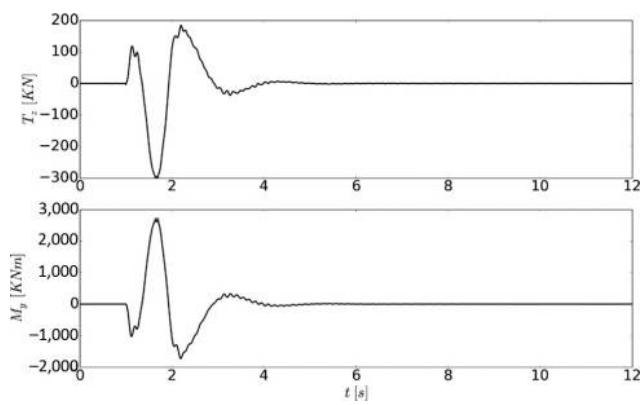
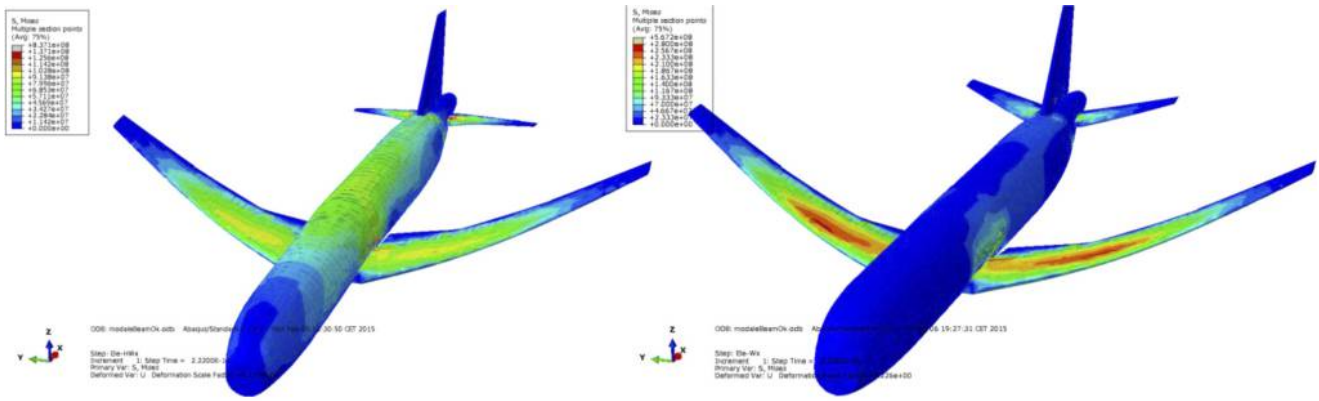


Figure 23 Elevator maneuver



Note: Deformation and Von-Mises stress for minimum bending moment on the horizontal tail (left) and maximum wing bending moment (right)

Conclusion

This work describes the definition of a new framework designed to be able to perform all the different analyses needed during the preliminary design, managing considerably big databases, both for structural and for aerodynamic models. The tool is able to communicate with Abaqus very easily to take advantage of one of the most powerful computer aided design (CAD)/CAE software. As shown by the test case, the tool at the moment is able to define a complete aircraft models and to perform several aero-elastic simulations to get responses and loads useful for the aircraft design. Several new features are planned for both mid-term and long-term time. Regarding the mid-term:

- At the moment, the coupling of the aero-elastic state space model with the multi-body solver MBDyn (Masarati, 2014) to be able to compute landing conditions and therefore landing loads on the whole aircraft is under development.
- The second feature that will be added is a smart tool that is able to define all the load cases and analyses requested by the regulations (EASA-CS23 and EASA-CS25) starting from few inputs, like the aircraft class category and the flight envelope.

Regarding the long-term development:

- A tool able to identify the sizing load cases, among all the computed ones, for the different structures will be developed.
- Knowing the design load cases, the sizing process can be defined through an optimization problem. Several optimization tools are under evaluation to identify the best solution and approach.

References

Aerospace Design Lab at Stanford University (2013), *SUAVE An Aerospace Vehicle Environment for Designing Future Aircraft*, available at: http://suave.stanford.edu/.

Cavagna, L., Ricci, S. and Travaglini, L. (2011), "NeoCASS: an integrated tool for structural sizing, aeroelastic analysis and MDO at conceptual design level", *Progress in Aerospace Sciences*, Vol. 47 No. 8, pp. 621-635.

CFS Engineering (2010), *CEASIOM Conceptual Aircraft Design*, available at: www.ceasiom.com/.

Chen, S., Lyu, Z., Kenway, G.K. and Martins, J. (2015), "Aerodynamic shape optimization of the common research model wing-body-tail configuration", *Journal of Aircraft*, Vol. 53 No. 1, pp. 276-293.

Daoud, F., Petersson, O., Deinert, S. and Bronny, P. (2012), "Multidisciplinary airframe design process: Incorporation of steady and unsteady aeroelastic loads", *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSM, Indianapolis, Indiana*.

Deinert, S., Petersson, O., Daoud, F. and Baier, H. (2013), "Aircraft loft optimization with respect to aeroelastic lift and induced drag loads", *10th World Congress on Structural and Multidisciplinary Optimization, Orlando, Florida*.

DLR (2013), "CPACS – a common language for aircraft design", available at: http://code.google.com/p/cpacs/

Dorbath, F., Nagel, B. and Gollnick, V. (2011), "A knowledges based approach for automated modelling of extended wing structures in preliminary aircraft design", 60th German Aerospace Congress September 2011, Bremen Germany, *Manching, Bavaria*.

Dorbath, F., Nagel, B. and Gollnick, V. (2012), "Implementation of a tool chain for extended physics-based wing mass estimation in early design stages", *71th Annual Conference of Allied Weight Engineers*.

Harder, R.L., Rodden, P. and Bellinger, E.D. (1979), Aeroelastic addition to nastran, CR 3094, NASA, NASA contractor report, 3094, Washington.

Karpel, M., Moulin, B. and Chen, P.C. (2005), "Dynamic response of aeroservoelastic systems to gust excitation", *Journal of Aircraft*, Vol. 42 No. 5.

Liem, P.R., Kenway, G.K.W. and Martins, J.R.R.A. (2015), "Multimission aircraft fuel burn minimization via multipoint aerostructural optimization", *AIAA Journal*, Vol. 53 No. 1, pp. 104-122.

Masarati, P. (2014), "Mbdyn - free multibody dynamics simulation software", available at: www.mbdyn.org

Morino, L., Chen, L.T. and Suciu, E.O. (1975), "Steady and oscillatory subsonic and supersonic aerodynamics around complex configurations", *AIIA Journal*, Vol. 13 No. 3.

Morino, L. and Kuo, C.C. (1974), "Subsonic potential aerodynamics for complex configurations: a general theory", *AIIA Journal*, Vol. 12 No. 2.

MSC.NASTRAN (1994), *MacNeal-Schwendler Corporation*, Aeroelastic analysis user's guide.

Nagel, B., Bohnke, D., Gollnick, V., Schmollgruber, P., Rizzi, A., La Rocca, G. and Alonso, J.J. (2012), "Communication in aircraft design: can we establish a common language?", *28th International Congress of the Aeronautical Sciences*, *Brisbane*.

Osterheld, C.M., Heinze, W. and Horst, P. (2000), "Influence of aeroelastic effects on preliminary aircraft design", *ICAS*, *Harrogate*.

Ripepi, M. (2014), "Model Order Reduction for Computational Aeroelasticity", PhD thesis, Politecnico di Milano, Milano.

Ripepi, M. and Mategazza, P. (2013), "Improved matrix fraction approximation of aerodynamic transfer matrices", *AIAA Journal*, Vol. 51 No. 5.

Scherer, J., Kohlgruber, D., Dorbath, F. and Sorour, M. (2013), "A finite element based tool chain for structural sizing of transport aircraft in preliminary aircraft design", *CEAS 2013*, *Delft, Netherlands*.

Schuhmacher, G., Daoud, F., Petersson, O. and Wagner, M. (2012), "Multidisciplinary airframe design optimization", *28th International Congress of the Aeronautical Sciences*, *Brisbane*.

The University of Tennessee (2008), "Plasma homepage", available at: http://icl.cs.utk.edu/plasma/

Yates, E.C.J. (1987), Agard standard aeroelastic confgurations for dynamic response i – wing 445.6. NASA, TM 100492, NAS 1.15:100492, August 01, 1987.

Yates, E.C.J., Land, N.S. and Foughner, J.T. (1963), "Measured and calculated subsonic and transonic flutter characteristics of a 45deg swepteback wing planform in air and in freon-12 in the langley transonic dynamics tunnel", NASA, TN D-1616.

Further reading

Cavallaro, R. (2008), "code for surface modeling and grid generation coupled to a panel method for aerodynamic configuration design", *Master's thesis*, Universitá degli Studi di Pisa, Pisa.

DLR (2011), RCE: Remote Component Environment, available at: http://rcenvironment.de/

DLR (2011), The TiGL Geometry Library, available at: https://github.com/DLR-SC/tigl

Hurlimann, F. (2010), "Mass estimation of transport aircraft wingbox structures with a CAD/CAE-based multidisciplinary process", PhD thesis, ETH Zurich, Zurich.

Klimmek, T. (2013), "Development of a structural model of the crm configuration for aeroelastic and loads analysis", *IFASD 2013, International Forum on Aeroelasticity and Structural Dynamics*, *Bristol*.

Ledermann, C., Ermanni, P. and Kelm, R. (2006), "Dynamic cad objects for structural optimization in preliminary aircraft design", *Aerospace Science and Technology*, Vol. 10 No. 7, pp. 601-610.

Ledermann, C., Hanske, C., Wenzel, J., Ermanni, P. and Kelm, R. (2005), "Associative parametric CAE methods in the aircraft pre-design", *Aerospace Science and Technology*, Vol. 9 No. 7, pp. 641-651.

Maierl, R., Petersson, O. and Daoud, F. (2013), "Automated creation of aeroelastic optimization models from a parameterized geometry", *IFASD 2013, International Forum on Aeroelasticity and Structural Dynamics*, *Bristol*.

Martins, J.R.R.A. (2010), Multidisciplinary Design Optimization Laboratory, available at: http://mdolab.engin.umich.edu/

Moerland, E., Zill, T., Nagel, B., Spangenberg, H., Schumann, H. and Zamov, P. (2012), "Application of a distributed MDAO framework to the design of a short- to medium-range aircraft", Deutscher Luft-und Raumfahrtkongress.

Parrinello, A. and Mantegazza, P. (2010), "Independent two-fields solution for full-potential unsteady transonic flows", *AIAA Journal*, Vol. 48 No. 7, pp. 1391-1402.

Parrinello, A. and Mantegazza, P. (2012), "Improvements and extensions to a full-potential formulation based on independent fields", *AIAA Journal*, Vol. 50 No. 3, pp. 571-580.

Ricci, S. (2013), "Neocass suite homepage", available at: www.neocass.org/

Rocca, G.L. (2011), "Knowledge based engineering techniques to support aircraft design and optimization", PhD thesis, Technische Universiteit Delft, Delft.

Tarkian, M. and Tessier, F.J.Z. (2007), "Aircraft parametric 3d modelling and panel code analysis for conceptual design", Master's thesis, Division of Machine Design, Linkoping University Institute of Technology, Linkoping.

Vescovini, R. (2011), "Analytical formulation for buckling and post-buckling analysis and optimization of composite stiffened panels", PhD thesis, Politecnico di Milano, Milano.

Werner-Westphal, C., Heinze, W. and Horst, P. (2008), "Multidisciplinary integrated preliminary design applied to unconventional aircraft configurations", *Journal of Aircraft*, Vol. 45 No. 2.

Corresponding author

Lorenzo Travaglini can be contacted at: lorenzo.travaglini@polimi.it