UNIVERSITÀ DEGLI STUDI DI GENOVA

SCUOLA POLITECNICA

DIME

Dipartimento di Ingegneria Meccanica, Energetica, Gestionale e dei Trasporti

MASTER OF SCIENCE THESIS
IN
MECHANICAL ENGINEERING

Optimization of an Aeroelastic System for Energy Harvesting

Supervisors:

Dr. Luca Magri - University of Cambridge

Chiar.^{mo} Prof. Ing. Alessandro Bottaro

Candidate:
Kevin Wittkowski

March 2021

Acknowledgements

I miei ringraziamenti vanno ai miei relatori, il Prof. Bottaro e il Dr. Magri. Il primo per le molte opportunità che mi ha offerto, per aver creduto in me e soprattutto per avermi dedicato tempo quando ero in difficoltà e avevo bisogno di capire meglio le materie trattate o le idee che avevo in mente. Il secondo per avermi dato sostegno, anche durante il periodo pandemico, per aver alleggerito l'inevitabile peso del lavoro da casa e per avermi consigliato e corretto col giudizio di un esperto e col tatto di un collega, più che di un supervisore.

In secondo luogo, i miei ringraziamenti vanno ai miei genitori, che mi hanno sopportato e aiutato nei momenti più duri di questi 5 anni e che hanno gioito con me dei miei successi. Ringrazio anche i miei zii, Odo e Gloria, per avermi sempre sostenuto economicamente e per avermi sempre dato il loro punto di vista, rispettando il mio. Parimenti, ringrazio la mia fidanzata per avermi sopportato in questo anno difficile.

Un ringraziamento particolare va ai miei colleghi di corso, che sono stati la mia famiglia per scelta. Ciri, Marco, Bar, Gotz: grazie di cuore per questi anni. Spero che non ci perderemo mai di vista.

Ottimizzazione di un Sistema Aeroelastico per Energy Harvesting

Sommario

Sistemi aeroelastici per la produzione di energia sono in fase di sviluppo per applicazioni micro-elettroniche e terrestri per via della loro abilità di sfruttare venti che sarebbero poco o per nulla produttivi per turbine convenzionali. Questi dispositivi producono potenze inferiori rispetto alle aero-turbine convenzionali ma possono lavorare con venti a bassa velocità e inoltre flussi irregolari o turbolenti o una disposizione a plotoni possono avere effetti benefici su questi dispositivi (contrariamente alle turbine convenzionali). In aeronautica, dispositivi simili ad aerogeneratori (detti RAT, ram-air turbine) sono impiegati per fornire potenza meccanica/elettrica ai sistemi di bordo in caso di emergenza o per alimentare dispositivi esterni (come i pod per guerra elettronica) o per diffondere trattamenti liquidi su vaste colture. In questa tesi viene discussa l'applicazione di dispositivi aeroelastici per produzione di energia in campo aeronautico e viene presentata l'ottimizzazione di un modello aeroelastico di un profilo NACA 64-A010 a due gradi di libertà (traslazione e rotazione). L'obiettivo del processo è trovare un set di parametri meccanici che ottimizzi l'output utile del dispositivo (ossia la massima potenza scambiata nel senso di traslazione oppure forza elettromotrice fornita). Il processo viene eseguito utilizzando i software MatLab e SU2 7.0.8. In particolare l'algoritmo di Nelder-Mead, implementato in MatLab, viene usato per lanciare iterativamente simulazioni in SU2 in regime transonico, a condizioni di aria libera corrispondenti a un volo a 10000m. Al termine del processo, le condizioni ottimali trovate vengono perturbate al fine di esaminare la stabilità delle soluzioni trovate.

Optimization of an Aeroelastic System for Energy Harvesting

Abstract

Aeroelastic systems for energy harvesting are being developed for terrestrial and micro-electronics applications for their ability to exploit winds that are not suitable for conventional wind turbines. These devices have a power production that is lower than conventional wind turbines but can work with low-speed winds, benefiting from turbulence and irregular flow patterns and may benefit from an arrangement in platoons. In aeronautics, devices similar to wind turbines (the ram-air turbines, RATs) are used to provide mechanical/electrical power to systems in case of emergency or to power external devices (such as electronic-warfare pods) or pumps to diffuse liquid agents on crops. In this thesis, the application of aeroelastic energy-harvesting devices to the aeronautical field is discussed. An optimization process is run on a pitch-plunge NACA 64-A010 airfoil model to find a set of mechanical parameters that are optimal for energy harvesting in the transonic regime (in terms of electromotive force or power extraction). The optimization process is performed using MatLab and SU2 7.0.8 CFD software: the Nelder-Mead algorithm implemented in MatLab is used to iteratively run simulations in transonic regime, at free-stream conditions corresponding to a flight at 10000m. The optimal condition is examined and a robust-design study is performed to examine the stability of the solutions.

Nomenclature

Notations

α	pitch angle [°]
AoA	angle of attack [°]
b, c	airfoil semichord and chord [m]
h	airfoil plunge [m]
$r_\alpha = \sqrt{\frac{I_\alpha}{mb^2}}$	radius of gyration [-]
x_α	distance of the center of gravity behind the elastic line [% of b]
$V^* = \frac{U_\infty}{b\omega_\alpha\sqrt{\mu}}$	flutter speed index [-]
$\mu = \frac{m}{\pi\rho_\infty b^2}$	airfoil mass ratio [-] (m=mass per unit span)
m	mass per unit span [kg/m]
I_α	section moment of inertia about the elastic axis [kg m^3]
$S_\alpha = mbx_\alpha$	static unbalance, positive for CG aft of mid-chord [kg m]
k_h, k_α	bending [N/m] and torsional [N/rad] spring stiffnesses
$\omega_h = \sqrt{k_h/m}$, $\omega_\alpha = \sqrt{k_\alpha/I_\alpha}$	uncoupled natural frequencies in plunge and pitch respectively [rad/s]
$f_{red} = \omega b/U_\infty$	reduced frequency [-]
$C_l = L'/(0.5\rho_\infty U_\infty^2 c)$	section lift coefficient [-]
$C_d = D'/(0.5\rho_\infty U_\infty^2 c)$	section drag coefficient [-]
$C_m = M'_{EA}/(0.5\rho_\infty U_\infty^2 c^2)$	section moment coefficient, referred to the elastic axis, positive nose-up [-]

$C_p = (p - p_\infty)/(0.5\rho_\infty U_\infty^2)$	pressure coefficient [-]
$Re_L = \frac{UL}{\nu}$	Reynolds number, referred to the Reynolds characteristic length "L" [-]
$Ma = \frac{U}{\sqrt{kRT}}$	Mach number [-]
$\vec{u} = (u, v, w)^T$	velocity vector [m/s], $U = \|\vec{u}\|$
p	static pressure [Pa]
T	static temperature [K]
ρ	density [kg/m^3]
ν	kinematic viscosity [m$^2/s$]
E	specific total energy [m$^2/s^2$]
H	specific total entalpy [m$^2/s^2$]
$k = c_p/c_v$	gas specific heats ratio [-]
R, R^*	gas-specific constant [J/K kg] and gas universal constant [J/K mol]
t	time [s]
$\tau = \omega_\alpha t$	non-dimensional structural time [-]
$\vec{n}$	normal vector [-]
k	turbulent kinetic energy [J/kg]
ω	turbulent specific dissipation rate [s^{-1}]

Superscripts and Subscripts

∞	evaluated in the free-stream
α	about the elastic axis/in the direction of pitching
$\dot{}$	time derivative
$\vec{}$	vector

Contents

1 Introduction

1.1 Aeroelasticity

Aeroelasticity is the study of the mutual interaction between inertial, elastic and aerodynamic forces acting in a solid exposed to a fluid flow [1]. The action of the wind on aerodynamic structures may be considered in different ways:

- strictly aerodynamic phenomena: the structural response has a negligible influence on the flow field;
- aero-elastic phenomena: the influence of the structural response is not negligible. A further distinction between dynamic and static fluid-structure interaction (FSI) can be made:
  - static: e.g. torsional divergence [2];
  - dynamic: e.g. flutter, buffeting [2], propeller-whirl flutter [3].

Sometimes aeroelastic effects generate small deformations, but there are cases in which the deformations are large. In the latter, the fluid-structure interactions need to be computationally modelled.
To set a historical example, the *Supermarine Spitfire Mk V* faced severe control-reversal problems due to the limited stiffness of its wing. Compared to the German *Messerschmitt BF-109* and *Focke-Wulf FW-190*, the *Spitfire* had a lower roll rate in dogfight combat. To improve the manoeuvrability, the designers of the *Spitfire* increased the aileron size. The larger ailerons led to an increase in the torsional moments on the wing when the manoeuvres were executed at high speed and, thus, the designers had to clip the wingtips to control this aero-servo-elastic problem (*control reversal*, in this case), decreasing the wing efficiency [4].

Fig. 1.1: Left: Spitfire Mk.V C (elliptical wing). Right: Spitfire LF Mark V B (clipped wingtips) in flight over North Africa. ©Imperial War Museum [5]

From this historical example, it is clear that aeroelastic design of airplanes is a key aspect of the structural and aerodynamic design optimization. Indeed, dynamic aeroelastic interaction should be limited, or avoided, in aeronautics [6].
In other fields, aeroelastic interactions may sometimes be useful. This is the case of energy harvesting and Formula 1 racing cars, for example. The former aims at producing electrical power from a flapping rigid wing (or a similar, but deformable, device) exposed to the wind [7], while the latter uses the non-isotropic behaviour of the composite materials to optimize the shape of the main spoilers depending on the instantaneous flux pattern [8] without the need for controls or actuators (which would be, in most of cases, prohibited by the FIA regulations [9]).

1.2 Flutter

1.2.1 General description

Among fluid-structure interactions, flutter is central for energy-harvesting. This aeroelastic interaction occurs when a streamlined structure exchanges energy with the fluid flow. The structure is deformed as a consequence of the interaction, hence the flow field is changed as a consequence of this deformation, producing a variation of the forces acting on the structure. If the fluid and structural forces reinforce each other, the system may experience large deformations, leading to the collapse of the structure or to limit-cycle oscillations (LCOs). A classic example of flutter interaction is the *Tacoma Bridge Disaster*, while in aeronautics this problem was discovered in 1916, when a *Handley Page O/400* bomber collapsed because of the coupling of its fuselage torsion modes with the independently-actuated elevators on the tail [10]. Flutter was described first with linear theories, which are useful in the low-Mach regime for 2D studies, but fail to capture the non-linearities of the transonic regime and LCOs.

1.2.2 Wagner and Theodorsen theories

Before the birth of Computational Fluid Dynamics (CFD), aerodynamics was mainly studied experimentally, or with inviscid theories using conformal mapping. Flutter modelling has to take into account several features:

- the wake released by an airfoil oscillates as the airfoil pitches up and down;
- the boundary layer can detach as a consequence of the oscillations;
- the detaching vortices change the velocity profile on the airfoil, hence the airfoil undergoes unsteady motion, so added-mass-terms must be taken into account;
- flow compressibility may be responsible for changes in the aeroelastic behaviour;

To model the physics on an airfoil, some conditions on the model can be imposed: impermeability (i.e. the airfoil is a streamline), Kutta-condition at the trailing edge (i.e. the tangential flow components at the trailing edge must be equal) and Kelvin's constant-circulation theorem (i.e. circulation enclosed in a material line is constant). The theory developed by H. Wagner in 1925 [11] to describe the starting vortex of a

flat plate is the first attempt to describe the unsteady dynamics of an impulsively-pitching airfoil. This theory was applied to the simple pitch-plunge airfoil aeroelastic problem (Fig. 2.1), whose structural governing equations are described in 1.1.

$$\begin{bmatrix} m & S_\alpha \\ S_\alpha & I_\alpha \end{bmatrix} \begin{Bmatrix} \ddot{h} \\ \ddot{\alpha} \end{Bmatrix} + \begin{bmatrix} k_h & 0 \\ 0 & k_\alpha \end{bmatrix} \begin{Bmatrix} h \\ \alpha \end{Bmatrix} = \begin{bmatrix} -l \\ m_{EA} \end{bmatrix} \tag{1.1}$$

where l and m_{EA} are the section lift and moment coefficients about the elastic axis, S_α is the static unbalance, I_α is the airfoil moment of inertia about the elastic axis, k_h and k_α are the spring stiffnesses in the plunge and pitch direction and h and α are the plunge and pitch coordinates. The aerodynamic problem is solved in the complex plane using conformal mapping, then the system of equations for this case can be rearranged to the real-valued ODE

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \pi\rho U c \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} + 2\pi\rho U^3 \boldsymbol{w} = \boldsymbol{f} \tag{1.2}$$

where $\boldsymbol{q}$ contains the states of the system, i.e. $\boldsymbol{q} = (h, \alpha)^T$ (plunge and pitch, respectively), $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{K}$ are 2x2 matrices (mass, damping and stiffness matrices, respectively), $\boldsymbol{f}$ is a 2x1 vector of forces, and $\boldsymbol{w}$ is a 4x1 vector. These elements are obtained by modelling the flux as inviscid, irrotational and using a conformal transformation from a plane containing a circle to a plane containing a flat plate. The vector $\boldsymbol{w}$ contains the aerodynamic states, which are integral functions

$$w_i(t) = \int_0^t e^{-\epsilon_1 U(t-t_0)/b} f(t_0) dt_0 \tag{1.3}$$

After some algebra, using the approximate form of the Wagner function (i.e. the function that mathematically introduces the history of the detaching vortices in this inviscid model and that is "stored" in the aerodynamic states), one gets

$$\begin{bmatrix} \ddot{h} \\ \ddot{\alpha} \\ \dot{h} \\ \dot{\alpha} \\ \dot{w}_1 \\ \dot{w}_2 \\ \dot{w}_3 \\ \dot{w}_4 \end{bmatrix} = \begin{bmatrix} -\boldsymbol{M^{-1}C} & -\boldsymbol{M^{-1}K} & -\boldsymbol{M^{-1}W} \\ \boldsymbol{I} & \boldsymbol{W} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W_0} & \end{bmatrix} \begin{bmatrix} \dot{h} \\ \dot{\alpha} \\ h \\ \alpha \\ w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \tag{1.4}$$

This is solved by Laplace transforming and solving the characteristic polynomial. For a detailed description of the method, refer to [10]. Wagner method produces solutions in the time domain, but it does not respect the impermeability boundary condition. This approach is valid only for very simple systems.

The natural improvement of Wagner's theory is the the theory by Theodorsen, that respects the impermeability constraint [10]. In this theory the assumptions are:

- the airfoil is a flat plate (with a flap possibly, so asymmetric airfoils are modelled in this theory);
- the wake is flat and the vorticity travels at the free stream airspeed;
- the flow never detaches (small amplitudes).

Furthermore, this introduces the added-mass terms and leading to an integro-differential equation from which one can compute the lift and moment on the airfoil. Specializing the equations of motion of the flat plate for a sinusoidal case, one obtains an algebraic system whose solutions can be found for

- free sinusoidal oscillations;
- forced sinusoidal oscillations;
- the airfoil is flying at the critical flutter condition.

The latter can be used to compute the flutter critical conditions by solving the disperions relation for eq.1.5 for $U = U_F$ and $\omega = \omega_F$

$$\begin{vmatrix} k_h - \omega^2 m + \pi\rho U c C(k) j\omega + & \omega^2 S + \rho\pi b^2 U j\omega + \rho\pi b^2 (x_f - c/2)\omega^2 + \\ -\omega^2 \rho\pi b^2 & +\pi U c C(k)(U + (0.75c - x_f) j\omega) \\ \\ \omega^2 S - \pi\rho U e c^2 C(k) j\omega + & k_\alpha - \omega^2 I_\alpha + (0.75c - x_f)\rho\pi b^2 U j\omega + \\ +(x_f - c/2)\rho\pi b^2 \omega^2 & -\pi\rho U e c^2 C(k)(U + (0.75c - x_f) j\omega) + \\ & -(x_f - c/2)^2 \rho\pi b^2 \omega^2 - 0.125\rho\pi b^4 \omega^2 \end{vmatrix} = 0 \quad (1.5)$$

Details of the method are explained in [10].
Both Wagner's and Theodorsen's theories are not able to describe the effects of compressibility and the effects of all geometrical parameters of an airfoil and viscosity. For this reason, numerical methods were developed to handle more general problems using the continuum hypothesis. For the purpose of this thesis, the inviscid theories suggest that:

- as viscosity is the feature that contains naturally the history of the flux (to set an example: assume that viscosity is negligible and the flow is incompressible and irrotational, Laplace equations allow for solving first the velocity field and then the pressure field, thus the pressure field is immediately updated after a change of the velocity field and does not depend on the previous time-step—viscosity introduces the dependence on the previous time-step), modelling the problem as inviscid is reasonable only when compressibility effects are dominant (i.e. when the Mach number is sufficiently large);
- if the problem is modelled as viscous, the mesh must be refined in the wake to describe the vortices released from the trailing edge.

1.2.3 The flutter transonic dip

The transonic regime presents severe conditions for the flutter-free requirement of various wings. This situation is critical for sweptback wings because they experience the so-called "transonic dip" (fig. 1.2). This is a sharp drop of the flutter boundary [12] [13].

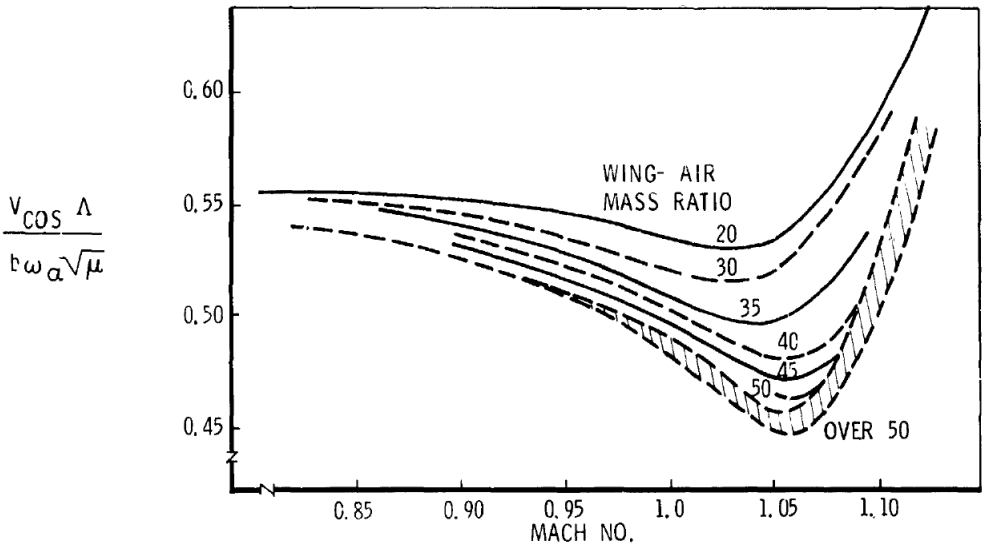


Fig. 1.2: Flutter transonic dip of a sweptback cantilevered wing as a function of the mass/air-ratio. $\Delta = 45°$ is the sweep angle [14]. This figure is taken from the original article [15]

Since the transonic regime is important for both civil and military aeronautics, and it is non-linear, a correct prediction of flutter characteristics of aircraft in the transonic regime is fundamental for stability studies. At the same time, it is costly to investigate flutter experimentally and to produce a high-fidelity simulation of this regime on a complete aircraft model. Since in the 1960s-70s supersonic aircraft were already being built, but computers had a limited power, there was a widespread use of reduced-order models to predict flutter numerically. Following this idea, in some references 3D wings were modelled using a 2D model of the wing section, taken at some distance from the root (usually chosen by experience, even if some references suggest that taking it at 3/4 span from the root is a good choice [6]). The 2D model is a simple pitch-plunge 2 DOFs airfoil, whose structural characteristics are chosen in order to achieve a similar behaviour with respect to the 3D wing. This is the case of the 2 DOFs NACA 64-A010 airfoil, which was used as a reduced order model for transonic flutter predictions by Isogai [13] [16]. In their article, Isogai pointed out that

- the first bending mode of a sweptback wing can absorb energy from the air flux and the streamwise sections of the wing pivot around an elastic axis placed upstream of the leading edge;
- the work per cycle done by the aerodynamic pitching moment about this axis

becomes positive if there is a lag between the airfoil motion and the pitching moment signal;

- this lag is caused by the vortex shedding and by compressibility, the latter being the most influential in the transonic regime.

Isogai examined the articles by Farmer and Hanson [12], and Mykytow [15], on 3D wings. To the best of the author of this thesis knowledge, it is not clear exactly how Isogai defined the equivalence between the 3D case and the reduced order 2D model. Indeed, their original paper [13] reports:

"The behavior of the streamwise sections [of the 3D wing] having such vibrational characteristics can be well simulated by an ordinary binary (bending-torsion) system of a two-dimensional airfoil [...] by carefully choosing the values of the structural parameters. Placing $a = -2.0$ *and* $\omega_h/\omega_\alpha = 1.0$*, we obtain the natural modes [...] that closely simulate the above-mentioned characteristic behaviors of the streamwise sections of a sweptback wing."*

The inertial parameters chosen by Isogai were also taken from the article by Theodorsen and Garrick [17]. Secondly, in Isogai's article it is not clear to which sweptback wing they exactly refer. [1].

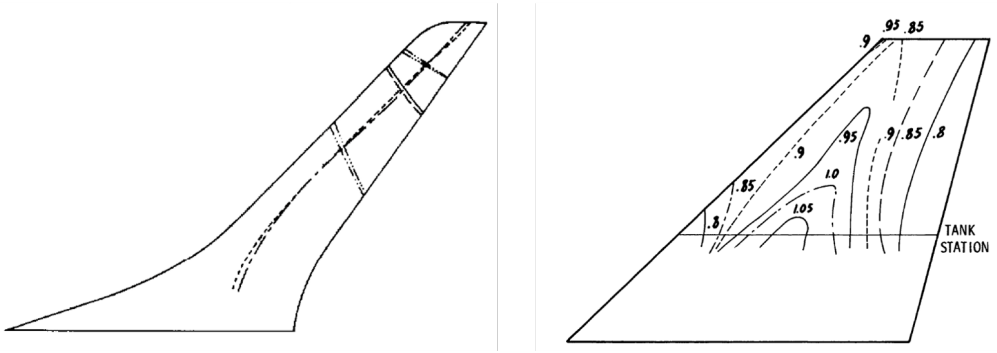


Fig. 1.3: Left: supercritical and conventional model wing for flutter tests on a NASA F-8 "Crusader" jet. This wing is tested in ref. [12]. Right: model of a 45° *back-swept wing. This model is tested in ref. [15] and its flutter boundary is shown in fig. 1.2. Sizes are not in scale. These images are taken from the original references.*

In this thesis, the results on the NACA 64-A010 pitch-plunge model will be taken for granted and only qualitative conclusions on a 3D configuration will be made.

1.3 Energy harvesting from aeroelastic flutter (EHAF)

Aeroelastic interactions may be a major source of danger for aircraft wings and aerodynamic surfaces in general, but it may also become useful in energy-harvesting applications. Aeroelastic energy harvesting devices have been studied for terrestrial and maritime applications. Most of these studies are related to small self-powered

[1] It is not clear, in particular, if the wing to which Isogai refers is the one examined in Farmer and Hanson's article [12] or one of the cases described in Mykytow's technical report [15]

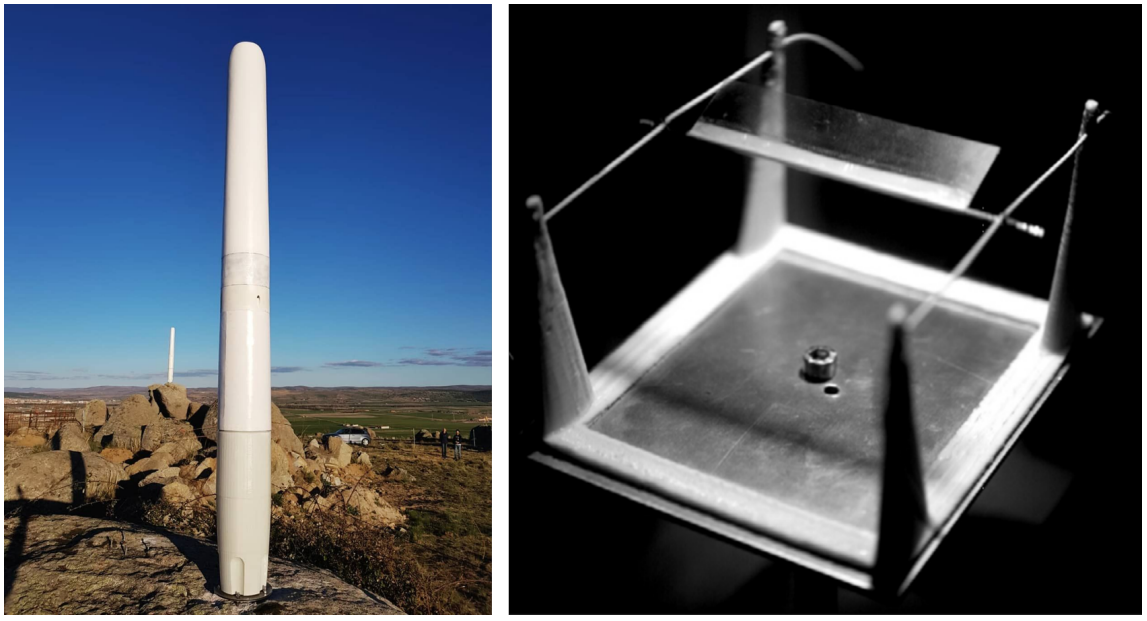


Fig. 1.4: Left: Prototype of "Tacoma" bladeless wind turbine by Vortex Bladeless [19] The tower has a dimension of about 3m. Right: Flutter Energy Harvester for Autonomous Powering (FLEHAP) device developed by Oliveri, Boccalero, Mazzino and Boragno. This image is taken from [21]. The wing has a span of 70mm.

transducers or small microelectronic applications [7] that can use either flutter, galloping, wake galloping or vortex-induced vibrations to harvest energy from oscillations. The harvester is usually placed in a fixed position and its mechanical parameters are studied to make the system experience amplifying oscillations first and then limit-cycle oscillations (LCOs). Energy is usually extracted from the plunging oscillations [7]. In a few studies harvesters were placed in the wings of aircraft scale models and the deformable-wing motion was used to charge small batteries or power a small camera [7] [18]. Large-scale applications of aeroelastic energy harvesters are being developed by Vortex Bladeless SL [19], which received a funding by the European Community in 2016 [20] and has already produced several prototypes.

Since the way energy is converted from mechanical to electric does not depend on the type of aeroelastic interaction, an overview of all methods is presented. There are several means of conversion:

- using piezoelectric devices. Piezoelectricity is the attitude of some materials (quarzum, $BaTiO_3$, among others) to convert the strain into electric charge and vice versa. This family of harvesters is the most common in literature and it is used mainly for the exploitation of low-speed winds. Harvesters of this kind were proposed by Bryant and Garcia [22] and Beltramo et al. [23], among others;

- using Faraday-Neumann-Lenz law (electromagnetic coupling). This law states that *the electromotive force induced by a magnetic field in a closed line is opposed to the variation in time of the magnetic flux on the surface enclosed by that line.* This is the case of devices that use the coupling with an alternator to convert mechanical energy into electric (as in Bladeless Vortex prototypes

[19]) or devices that have a magnetic equipment on the moving parts and a circuit on the basement (like the harvester proposed by Park [24], that was also equipped with a funnel to increase the airspeed near the device);

- using more complex electric equipment. This section collects electrostatic devices (such as electret harvesters like the one by Perez et al. [25]) and devices that act as electric amplifiers (such as the harvester proposed by Olivieri et al. [21]).

In transonic flow the available power can be orders of magnitude larger than in microelectronic devices. This work leaves out of consideration the way electric power is produced in the EHAF device, but it is devoted to the optimization of the maximum power that can be extracted from a similar device. For this reason, some of the methods discussed previously are more suitable than others. This is the case of electromagnetic coupling, that can be scaled easily from small to large devices with relative simplicity.

1.4 SU2 CFD software

SU2 ("Stanford-University Unstructured") is an open source computer-aided-engineering (CAE) analysis and design software developed to solve multi-physics and optimization tasks [26]. The code is available on GitHub. PointWise and Paraview softwares will be used for meshing and post-processing, respectively. A great part of post-processing activity is made also using MatLab 2017R and 2020a, which is able to handle the .csv SU2 output files. [2]

1.5 Computational resources

The studies were conducted using the following resources

- a workstation: CPU 16-cores INTEL Xeon 2.10GHz, 32 GB RAM, 64-bit architecture, Ubuntu OS;
- a desktop computer: CPU 8-core INTEL i-7 10700K 3.8GHz-5GHz, 32 GB RAM, 64-bit architecture, MS Win10 OS;
- a laptop: CPU quad-core INTEL i-7 7700HQ 2.8GHz, 16 GB RAM, 64-bit architecture, MS Win10 OS.

the laptop was mainly used for meshing, programming and post-processing activities.

[2] PointWise is suitable because it can handle the .su2 mesh format directly and thus it is not necessary to pass by .cgns mesh format conversions to obtain a mesh readable in SU2. This software is particularly efficient in creating structured meshes (more suitable than unstructured meshes in simple 2D cases because they are more computationally-efficient).

2 Transonic NACA 64-A010 airfoil flutter analysis

Before starting with the airfoil analysis, it is worth spending some words on how SU2 initializes the free-stream conditions. The computations presented in this section are "dimensional" and the definition of the free-stream conditions follows this workflow:

1. Only Ma_∞ (free-stream Mach number), Re (Reynolds number based on the chord c), V^* (flutter speed index), ω_α (pitch natural frequency), μ (freestream dynamic viscosity) and the chord are necessary to define the free-stream condition;

2. the free-stream speed, U_∞, is computed from V^* and, using the expression of Ma for an ideal gas, it is used to compute the free-stream temperature, T_∞;

3. the Sutherland law is used to compute free-stream dynamic viscosity, μ_∞, from T_∞;

4. Re, U_∞ and μ_∞ are used to find ρ_∞ and, using the ideal gas model, the free-stream pressure p_∞.

In the case of inviscid calculations the Reynolds number is not meaningful, thus it is necessary to define a free-stream pressure p_∞ value in the configuration file. The free-stream pressure was set in all inviscid cases to 101325 Pa. These settings are valid only for the aeroelastic case definition, which is described in the file *CSolver.cpp* in the SU2 7.0.8 source code.
An example of SU2 working configuration file is provided in Appendix. The SU2 code for 2D aeroelastic calculations is based on Alonso and Jameson's work [27] and, following the original reference, the calculation can also account for a mechanical damping in the pitching and plunging direction. The analysis described in the next chapters are performed with zero damping.

2.1 Problem definition

A model used in literature to describe the aeroelastic performances of airfoils is that composed of an airfoil that can move in the pitching and plunging directions, constrained by springs and dampers. The airfoil moves because of the interaction with the fluid flow and/or non-equilibrium initial conditions. This setup, where the considered airfoil is a NACA 64-A010 (symmetric) was examined by Isogai [13], who investigated flutter transonic dip in a back-swept wing. This case became one of the benchmark cases in 2D aeroelasticity. In this section, the setup is shown in fig. 2.1

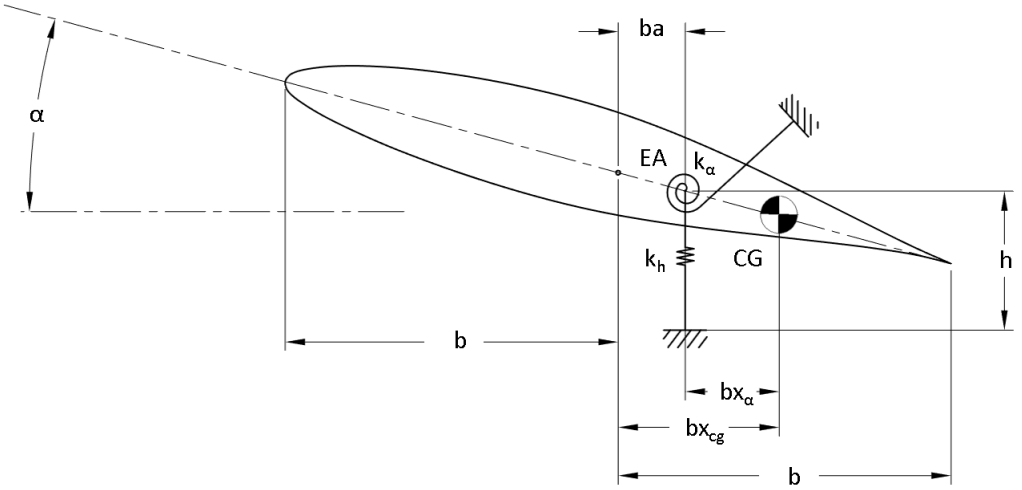


Fig. 2.1: Aeroelastic model with 2 degrees of freedom, corresponding to airfoil pitch and plunge. b is the semi-chord; EA is the elastic axis; CG is the center of gravity; k_h and k_α are the spring stiffnesses in the plunge and pitch direction respectively; a is the non-dimensional distance between the center of the airfoil and the elastic axis; x_α is the non-dimensional distance between the elastic axis and the center of gravity and x_{cg} is the non-dimensional distance between the center of the airfoil and the center of gravity; h and α are the plunge (positive is spring is compressed) and pitch (positive nose-up) coordinates.

2.1.1 Structural governing equations

The governing equations of the structural system are

$$\begin{bmatrix} m & S_\alpha \\ S_\alpha & I_\alpha \end{bmatrix} \begin{Bmatrix} \ddot{h} \\ \ddot{\alpha} \end{Bmatrix} + \begin{bmatrix} k_h & 0 \\ 0 & k_\alpha \end{bmatrix} \begin{Bmatrix} h \\ \alpha \end{Bmatrix} = \begin{bmatrix} -L' \\ M'_{EA} \end{bmatrix} \tag{2.1}$$

which can be re-written in the non-dimensional form as

$$\begin{bmatrix} 1 & x_\alpha \\ x_\alpha & r_\alpha^2 \end{bmatrix} \begin{Bmatrix} \ddot{h}/b \\ \ddot{\alpha} \end{Bmatrix} + \begin{bmatrix} \left(\frac{\omega_h}{\omega_\alpha}\right)^2 & 0 \\ 0 & r_\alpha^2 \end{bmatrix} \begin{Bmatrix} h/b \\ \alpha \end{Bmatrix} = \frac{V^{*2}}{\pi} \begin{Bmatrix} -C_l \\ 2C_m \end{Bmatrix} \tag{2.2}$$

$\tau = \omega_\alpha t$ is the non-dimensional time.
The mechanical properties are similar to those in Isogai [13]:

$a = -2.0 \qquad x_{CG} = -0.2 \qquad x_\alpha = 1.8 \qquad r_\alpha^2 = 3.48 \qquad \mu = 60 \qquad \omega_h = \omega_\alpha = 100$

In this thesis, the damping was not considered for two reasons. First, SU2 aeroelastic package is built without the possibility to prescribing mechanical damping. Second, damping has a limited importance in our case: the method that will be used to convert the mechanical power into electrical power in the device is not pre-defined, thus it is possible only to optimize the *maximum* available power and not the net output power. For any condition, maximum power is always obtained when there is no damping, thus it is not worth implementing the damping in the C++ SU2 solver scripts.

2.2 Inviscid calculations

2.2.1 Inviscid flow governing equations

The governing equations for an inviscid 2D flow are the Euler equations [27]. Let A be a control volume with boundary ∂A which moves with cartesian velocity components u_A, v_A. The equations of the fluid flow can be written in the integral form as

$$\frac{d}{dt}\iint_A \boldsymbol{w}dxdy + \oint_{\partial A}(\boldsymbol{f}dy - \boldsymbol{g}dx) = \boldsymbol{0} \tag{2.3}$$

where

$$\boldsymbol{w} = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{Bmatrix} \qquad \boldsymbol{f} = \begin{Bmatrix} \rho(u-u_A) \\ \rho u(u-u_A)+p \\ \rho v(u-u_A) \\ \rho E(u-u_A)+pu \end{Bmatrix} \qquad \boldsymbol{g} = \begin{Bmatrix} \rho(v-v_A) \\ \rho u(v-v_A) \\ \rho v(v-v_A)+p \\ \rho E(v-v_A)+pv \end{Bmatrix}$$

and the equation of state for an ideal gas is

$$p = (k-1)\rho\left[E - \frac{1}{2}(u^2+v^2)\right] \tag{2.4}$$

Since the flow is non-viscous, the non-penetration boundary condition

$$\vec{u}\cdot\vec{n} = 0 \tag{2.5}$$

has to be imposed on the airfoil surface ("Euler wall" in the configuration file), while at the farfield boundary, static pressure and Mach number are prescribed. The use of the farfield boundary condition requires the boundary to be very distant from the airfoil, with an increase in the cell count. A good agreement (also taking into account that the mesh will be deformed during calculation) is to set the radius of the farfield boundary to 200 times the chord length (or more). It was verified that, with these values, solutions are not sensitive to the farfield radius.
In SU2 [28] *EULER* solver, the discretization in space is done using a finite volume method (FVM). The convective and viscous fluxes are evaluated at the midpoint of an edge. The discretization in time is implemented using a dual-time stepping scheme (2nd order).

2.2.2 Setup

Alonso and Jameson [27], Marti and Liu [29], among others, described numerically the flutter mechanism of a NACA 64-A010 airfoil in transonic regime using inviscid

calculations (and without mechanical damping), finding that the computations were in good agreement with experimental observations [13] with a resolution of 36 time-steps per period of oscillation based on the pitching natural frequency. Using this resolution, Alonso found that at $Ma = 0.9$, (where the second flutter mode becomes significant) only the first mode was correctly computed by the simulation, while the second mode was not computed properly. The aforementioned calculations were performed in two steps:

1. the airfoil is forced to pitch for 3 periods about the quarter of the chord with an amplitude $\Delta\alpha = 1.0°$ and a frequency equal to the characteristic frequency of the first flutter mode;

2. when the airfoil passes through the $\alpha = 0°$ position, the imaginary pin at the quarter-chord is removed and the airfoil is left free to move, following the aeroelastic configuration described in the previous section.

This configuration could be simulated in SU2 by generating a set of restart files from the pitching simulation (the exact number depends on the time discretization) and by initialising the aeroelastic simulation using these files. However, a faster and simpler way of obtaining aeroelastic solutions is to force the airfoil directly in the aeroelastic simulation using a wind-gust. This can be done in SU2 configuration file, details about the procedure will be given in the following paragraphs.
Taking into account the results by Alonso [27], the discretization in time is done with 36 time-steps per period, based in the pitching natural frequency. Finer discretizations in time do not cause significant differences in the results. Thus, the time-step is 0.001745 s. The number of internal iterations is set to 100, which is sufficient to achieve a good convergence of forces and displacements at each time-step. The flow time discretization is solved with Euler-Implicit. The elastic equations are solved every 3 internal iterations of the fluid solver. The maximum number of external iterations is chosen as:

- for validation purposes, since the available validation data cover a range of structural time from 0 to 50, the total number of time-steps is set to 360;

- for flutter boundary description, the number of time-steps is set to 720, to achieve a better description of the transient (and of the saturated condition, if possible, when the aeroelastic response is undamped).

A sinusoidal wind gust of amplitude $\Delta\alpha = 1°$ is used to force the initial motion of the airfoil for a single period, then the flow angle of attack is set to 0° (i.e the flow is horizontal in the cartesian reference and the airfoil is left free to move).
The gradients are computed using Green-Gauss and the convective numerical method is the Jameson-Schmidt-Turkel (JST) with a 2nd order spatial integration. The grid deformation algorithm is $Flexible - MGRES$, with linear preconditioner $LU - SGS$ and the deformation is handled using the Inverse Volume method. A brief description of the aforementioned methods is provided in Appendix A.

2.2.3 Mesh

Four different meshes were used for the inviscid calculations:

- "mesh 0" is an hybrid mesh of 9313 cells, with quadrilaterals near the airfoil and triangles in the farfield, see fig.2.2a. This mesh is available at [30]. In this mesh the farfield is placed 200 times the cord from the airfoil;

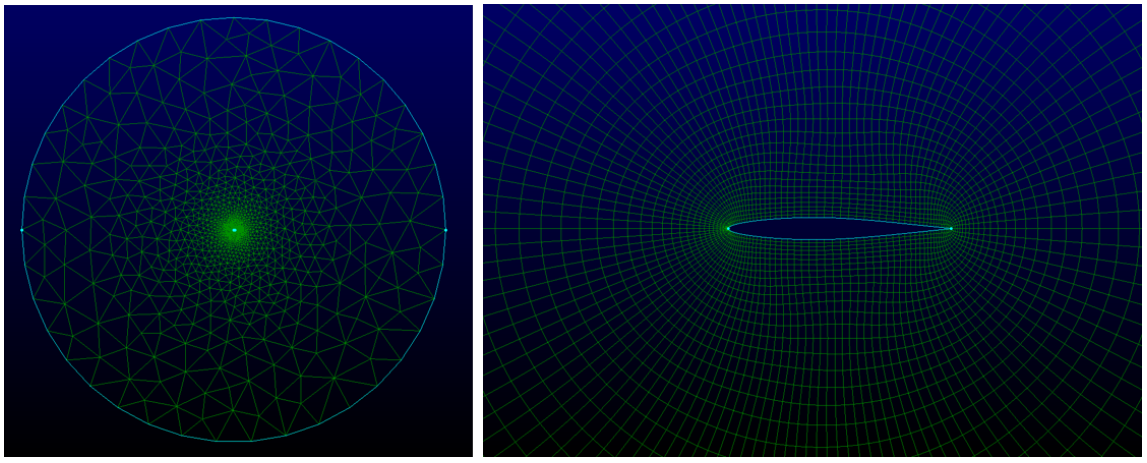
- "mesh 1" is a structured mesh with 15300 cells, with farfield placed at 210 times the chord from the airfoil, see fig.2.2b;

- "mesh 2" is a structured mesh made originally for viscous calculations and thus refined on the wall: this mesh is used in this case only to see the effect of refinement near the wall, that for an inviscid simulation should result either in no change or in a slower convergence rate. This mesh has 38760 cells and the farfield is placed at $\sim$270 times the chord from the airfoil, see fig.2.2c.

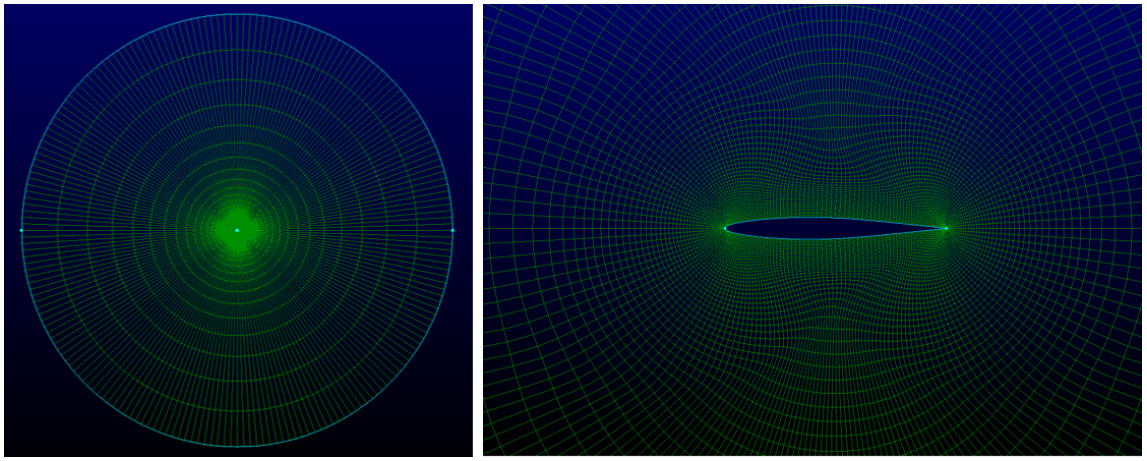
All meshes are O-type meshes, which are documented [27] [31] and are easier to build in PointWise than C-type meshes (PointWise extrusion layers converge to circles at large distances from the central body both for algebraic and hyperbolic methods). In mesh 1 the first cell height is 5mm and the expansion ratio is gradually increased from 1.06 to 1.2 as the distance from the body is increased (in this way the cells near the body are more square-like). In mesh 2 the first cell height is of order of 10^{-7}m and the expansion ratio is kept ≤ 1.1 to resolve the boundary layer in compressible viscous calculations, with an inevitable increase in the computational cost.

2.2.4 Validation and mesh sensitivity study

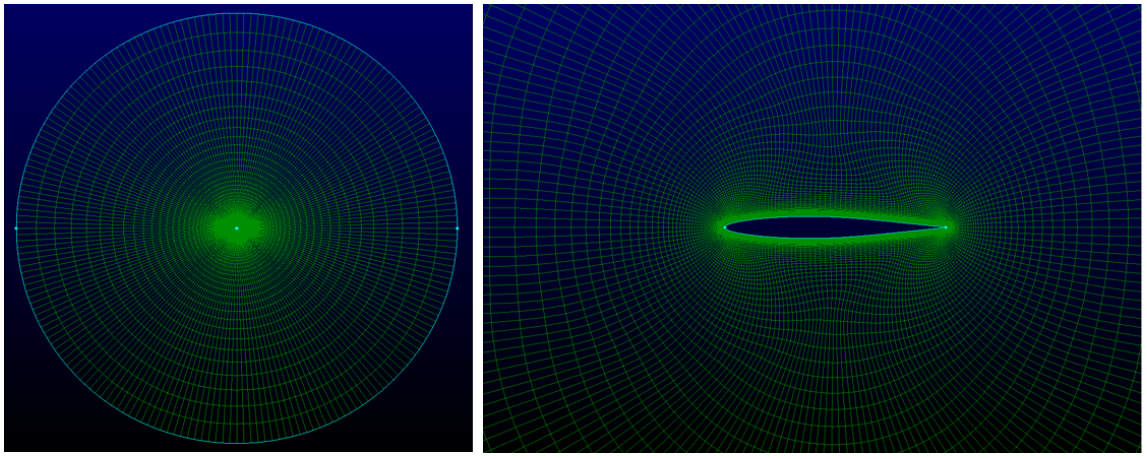
In literature a wide variety of inviscid studies on aeroelastic NACA 64-A010 are available since this case is used as a benchmark for flutter prediction [32]. The first studies were conducted numerically with vortex-doublet method by Isogai [13]. Alonso and Jameson [27] were the first to conduct a 2D numerical inviscid study solving Euler equations using the finite-volume method. Most of the studies report first a static validation, then a forced pitching case and, hence, the aeroelastic case. This work will follow a similar line, but the pitching airfoil case will not be presented, as in our case the airfoil is forced using a wind gust.
The static validation case is obtained resolving the steady inviscid flow equations on the airfoil, which is set at an AoA of 0°. In this case no structural equation is solved. The results of the static validation are presented in fig. 2.18. Among others, Hall et al. [31] was chosen for comparison. Agreement between Hall data and the present calculations is good and, in particular, the position of the shock-wave is predicted correctly with all meshes.



(a)



(b)



(c)

Fig. 2.2: mesh 0 (2.2a), 1 (2.2b) and 2 (2.2c): general view and closeup on the airfoil.

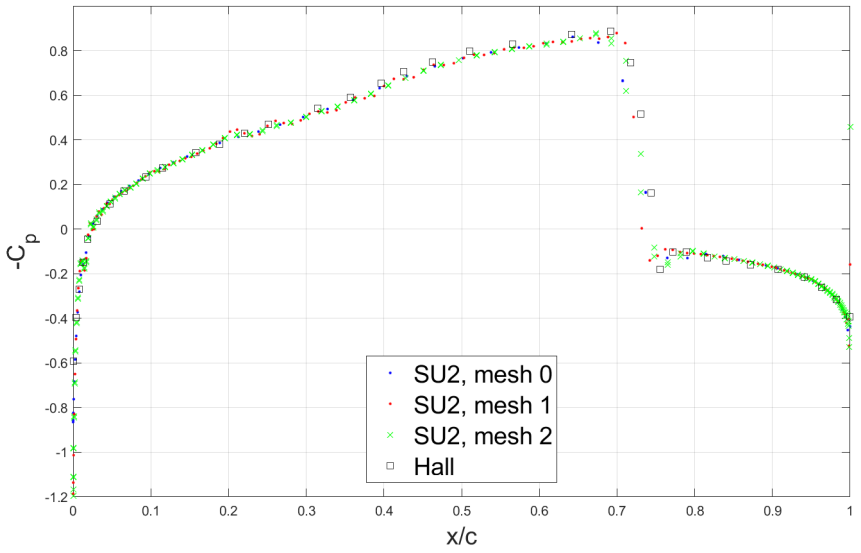


Fig. 2.3: Pressure coefficient vs non-dimensional position along chord for a static NACA 64-A010 airfoil in inviscid steady flow at $Ma = 0.85$, $\alpha = 0°$. The airfoil is symmetric, thus the plotted data represents both the upper and the lower side of the airfoil.

The dynamic validation is based both on the work by Alonso and Jameson [27] and that by Kassem et al. [33] (who reported lift coefficient time-traces), while for the flutter boundaries Hall [31] and Li and Ekici [34] were used as a reference since they are the highest and lowest flutter boundaries found in literature for the Mach range $0.825 \div 0.875$ in the inviscid case.

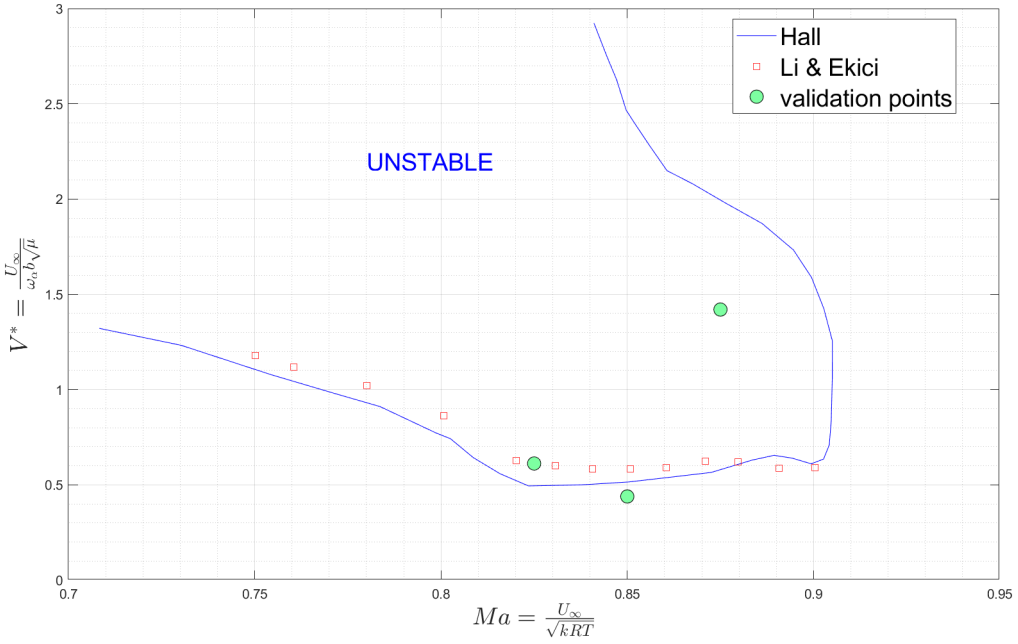


Fig. 2.4: Mach number vs flutter speed index boundary for different references [34] , [31]. In green the position of the validation points for the inviscid case are reported.

The aeroelastic calculations were compared with [27] [33] by means of C_l, pitch and plunge time-traces. From fig. 2.5, the SU2 flutter simulations predicted correctly the aeroelastic response and, in particular, in the $Ma = 0.825$ case SU2 predicted a neutral behaviour as in [27], while [33] response seems more damped. Note that the saturated condition is not presented in [33]. In the C_l case, the unstable behaviour is predicted correctly but the C_L seems under-predicted in the transient zone. The opposite is true for the pitch-plunge comparison in fig. 2.6 with [27].

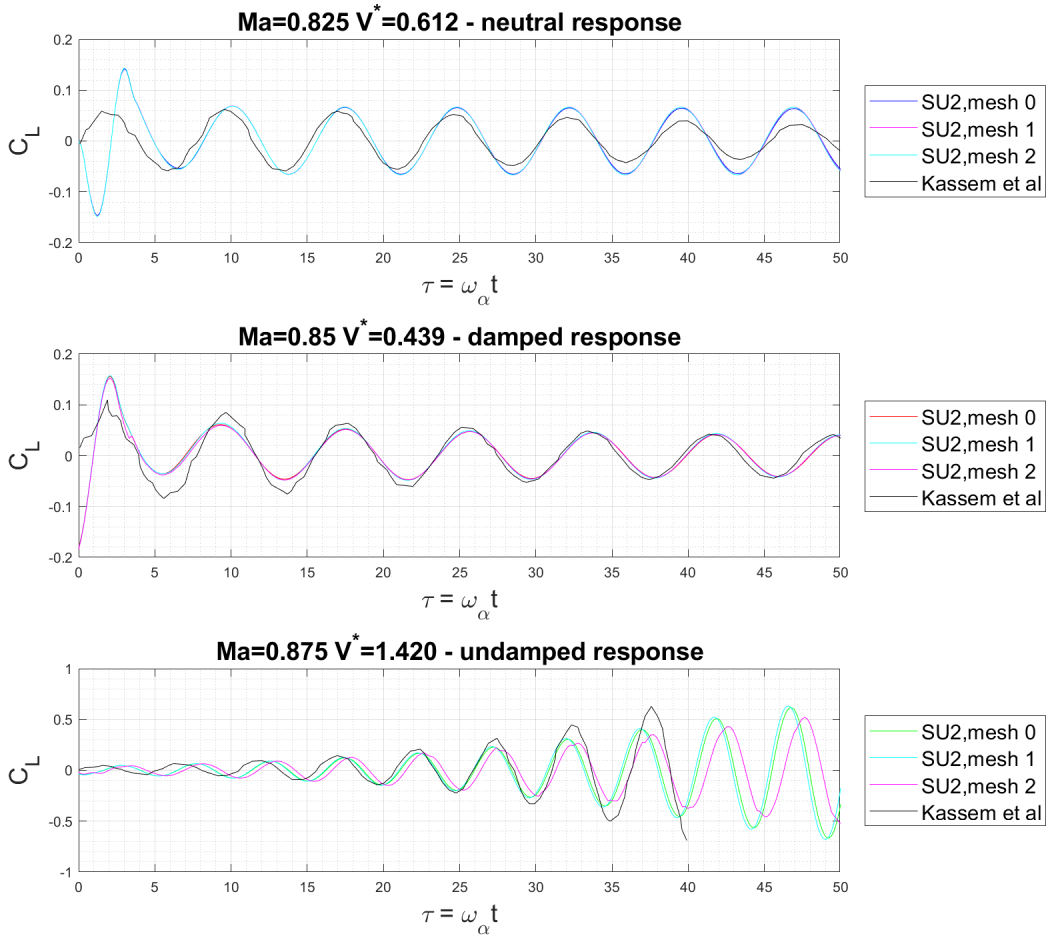


Fig. 2.5: Lift coefficient time-traces for 3 different meshes and for 3 different validation points in SU2 compared with the inviscid simulations by Kassem, Liu and Banerjee [33]. Due to different initial conditions (in [33] they do not use a wind gust), the time-traces were shifted to start in phase with the reference.

Since the larger the deformations, the larger the forces involved, it concluded that the two references predict a different unstable transient and that the one computed using SU2 lies in the range of the two references. *Mesh 2* yields results that overlapped the results given by the other meshes in most of cases. This does not happen in the unstable cases, where the unstable behaviour is captured, but the time-trace is slightly different. This might be due to the slower convergence rate experienced

using this mesh.
Overall, the inviscid study made in SU2 is thus validated.

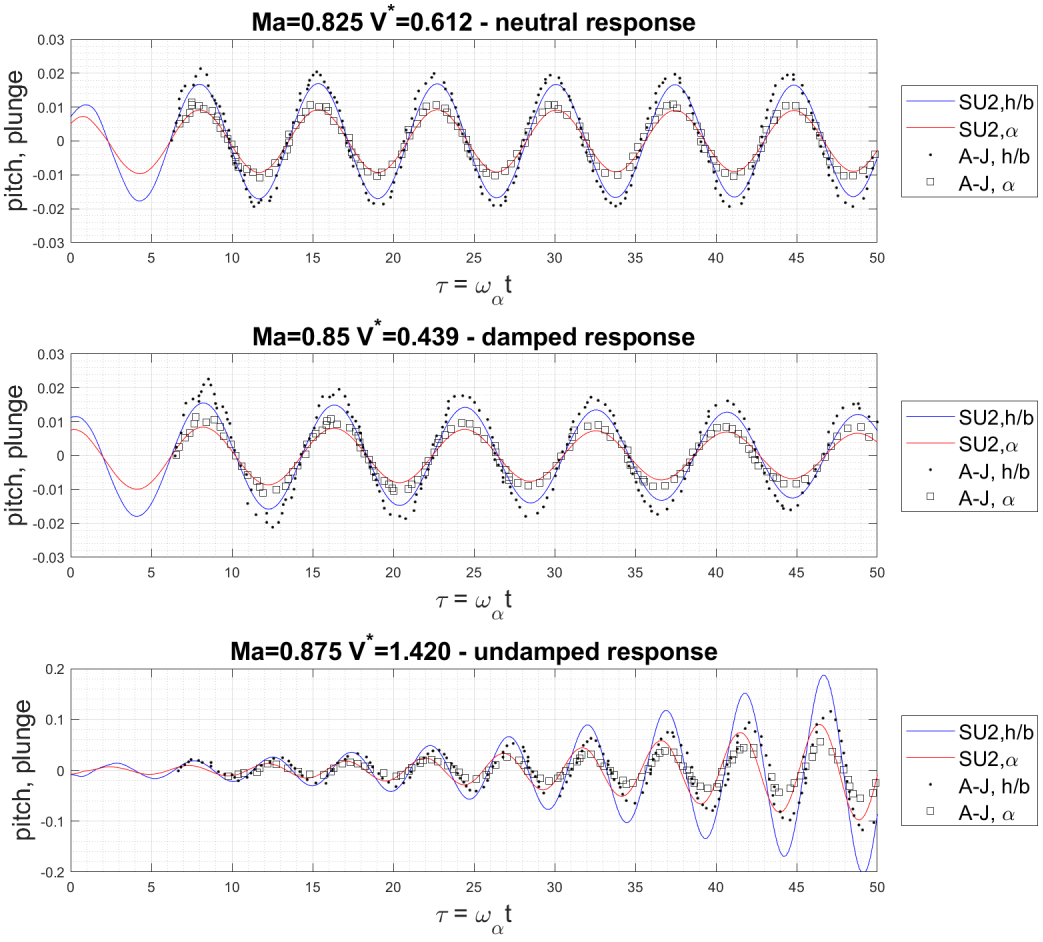


Fig. 2.6: non-dimensional plunge and pitch time-traces for 3 different validation points in SU2 compared with the inviscid simulations by Alonso and Jameson [27]. Due to different initial conditions, the time-traces were shifted to start in phase with the given reference. SU2 time-traces were obtained using mesh 0 and the results of mesh 1 are not shown as they overlap with those of mesh 0.

The time-traces for the validation study were obtained from their original references using the MatLab function "grabit" [35] manually, thus a very minor source of error on the effective point positions is due to this manual acquisition.

In conclusion, a mesh sensitivity study was performed in order to investigate how the solution is influenced by the coarseness of the mesh and thus to study if it is necessary to use a finer grid for the study present in the next section or if *mesh 0* is suitable for the case with a negligible error. The mean flow fields are obtained by setting as SU2 output files *restart-flow* .csv and importing those files in MatLab,

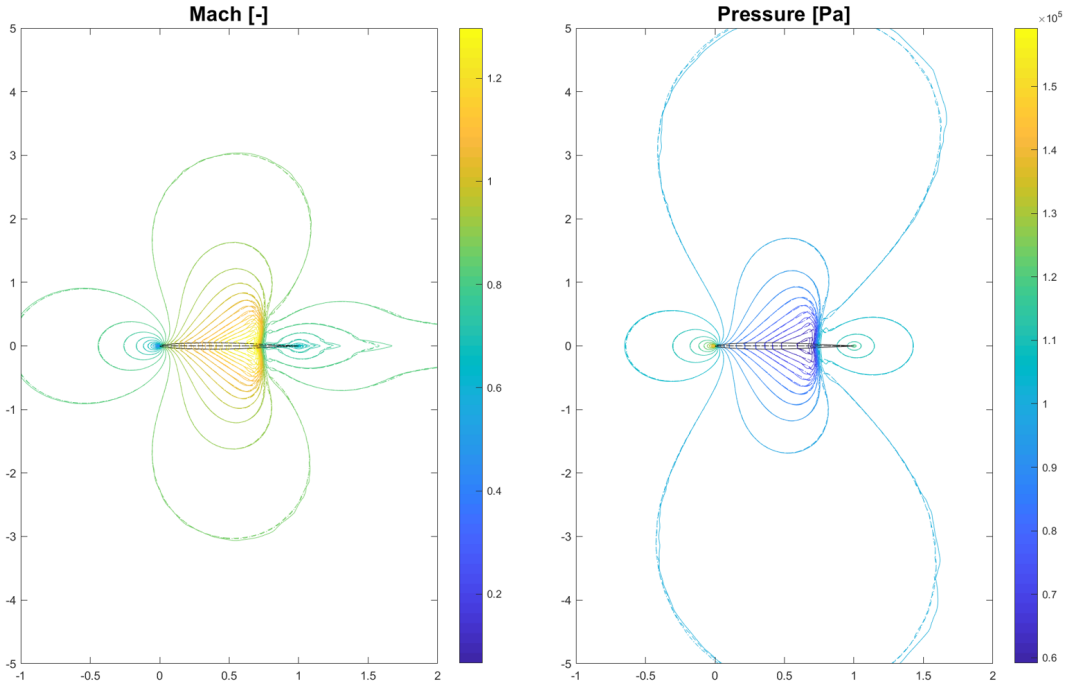


Fig. 2.7: Mean pressure [Pa] and Mach [-] contours around the airfoil mean position in time for the case $Ma = 0.85$, $V^* = 0.439$. The time traces were cut for $\tau = 17.45$ to remove the initial transient due to the wind gust. Mesh 0,1,2 are represented with continuous, dash-dash and dash-dot lines, respectively.

then using the point identifiers (each mesh node has a corresponding number in SU2 flow fields files) to order data by point identifier and clustering the data into a 3D matrix for pressure and another for Mach. The "thickness" of these two matrices correspond to the time variable. At this point, one averages in the time direction, obtaining a matrix that represents the mean field of pressure or Mach. The flow field is then resampled to obtain a regular grid and use it to plot the contours in MatLab. For computational reasons, the re-sampling method "natural" was used in MatLab [36]. A similar procedure is followed for the position of the airfoil, using the *surface-flow* .csv files. In that case no resampling was necessary. Fig. 2.7 shows mean Mach and pressure fields around the mean position of the airfoil (note that field contours inside the airfoil boundary are not physical) for the three meshes 0, 1, 2. The contours show a good agreement in most of the points and minor differences can be seen in the pressure farfield, which is not very important for force computation on the airfoil, and in the wake field, in the Mach contour.

Fig 2.8 shows contours of pressure and Mach number RMS: these charts are obtained similarly to the one used for the mean fields, the only difference being that in this case the field variables RMS is plotted in the mean positions of the mesh nodes and the airfoil plot is the same in fig 2.7. For this reason, the airfoil plot is used only to provide an idea as to where the computed field has no physical meaning and is only

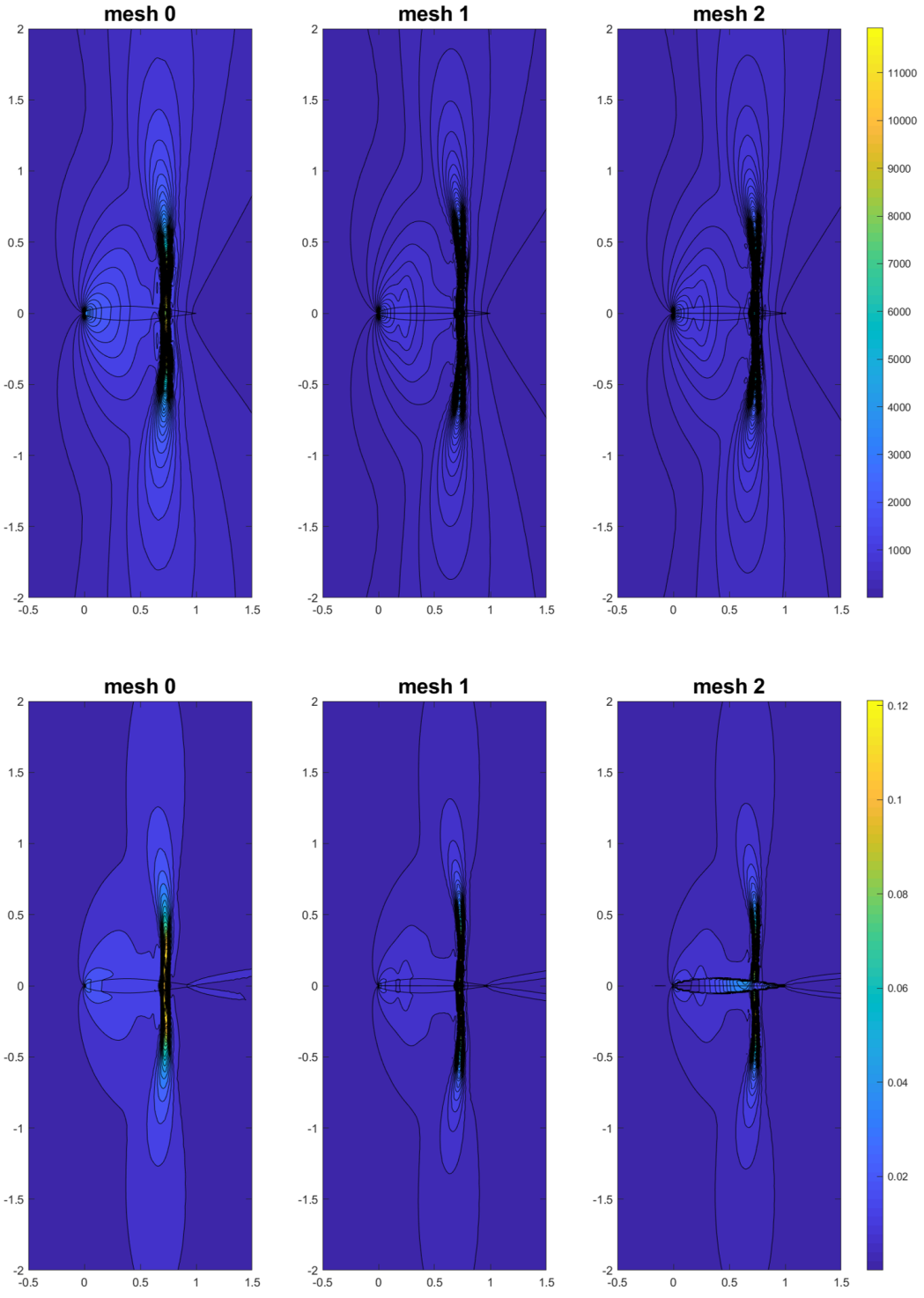


Fig. 2.8: Iso-contours of pressure RMS (upper) and Mach RMS (lower) around the airfoil mean position in time for the case $Ma = 0.85$, $V^* = 0.439$, for the three meshes. The time traces were cut for $\tau = 17.45$ to remove the initial transient due to the wind gust. Data is in [Pa] and non-dimensional, respectively.

due to the interpolation procedure.
The RMS fields show an overall good agreement but some differences are present. In particular, in *mesh 0*, the zone where the highest RMS values are present (that corresponds to the shock-wave) is thicker than in the other two meshes. This is due to the mesh coarseness. Then there are some differences near the leading edge, but they are too close to the airfoil mean position to assess whether they are physical or numerical.
Given the results in this section, it is reasonable to use *mesh 0* for the reconstruction of the flutter boundary, presented in the next section.

2.2.5 A time accurate study of the flutter transonic dip

To study the flutter boundary at least 3 different methods in time domain are available in literature [37]: the *Moving-Block Method* (MBM), the *Least-Square Curve Fitting Method* (LSCFM) and the *Autoregressive Moving-Average Method* (AMAM). The simplest method to apply in this case is the LSCFM, which is summarized as follows:

1. let $f(t)$ be the transient time-trace of the airfoil plunge or pitch (or another related physical property), the LSCFM solves a least-square problem to fit an exponential function to $f(t)$ and estimate the exponential of the time-decay function;

2. the chosen model is

$$f(t) = a_0 + \sum_{i=1}^{N} e^{-\zeta_i \omega_i t}[a_i cos(\omega_i t) + b_i sin(\omega_i t)] \tag{2.6}$$

   where the unknowns are a_i, b_i, ω_i and these values minimize the Euclidean norm of the difference between the two sides of the previous equation. It is a non-linear least-square (LS) problem;

3. one way to solve the problem is to provide initial guess values for ω_i and ζ_i and then solve the problem as a linear LS for the remaining unknowns and iterate until convergence.

This method can be long to run, but in our case only one mode is present (this is clear from the fast-Fourier-transform of the time-traces, see fig. 2.14), thus $i = 1$, then the time-traces clearly show a sinusoidal behaviour, thus it is possible to set either $a_i = 0$ or $b_i = 0$. At this point, since we have no interest in guessing the frequency of the signal, we can extract the peaks from the signal's absolute value (usign MatLab $findpeaks$ function) and re-formulate the problem. The LS problem is now to fit an exponential function of unknown time-factor a and amplitude factor c on the sequence of the peaks of the absolute value of the original signal (y_i)

$$\begin{aligned} y_0 &= ce^{-at_0} \\ y_1 &= ce^{-at_1} \\ &\dots \\ y_N &= ce^{-at_N} \end{aligned}$$

where N is the number of points in the peaks time-series. Normalizing the signal and translating it so that it always starts from $t = 0$ and unitary value, c can be discarded from the equation and, doing some algebra, the problem becomes a linear LS:

$$\underbrace{\begin{bmatrix} t_0 \\ t_1 \\ ... \\ t_N \end{bmatrix}}_{\text{M}} \underbrace{[a]}_{\text{X}} = \underbrace{\begin{bmatrix} -ln(y_0) \\ -ln(y_1) \\ ... \\ -ln(y_N) \end{bmatrix}}_{\text{B}} \tag{2.7}$$

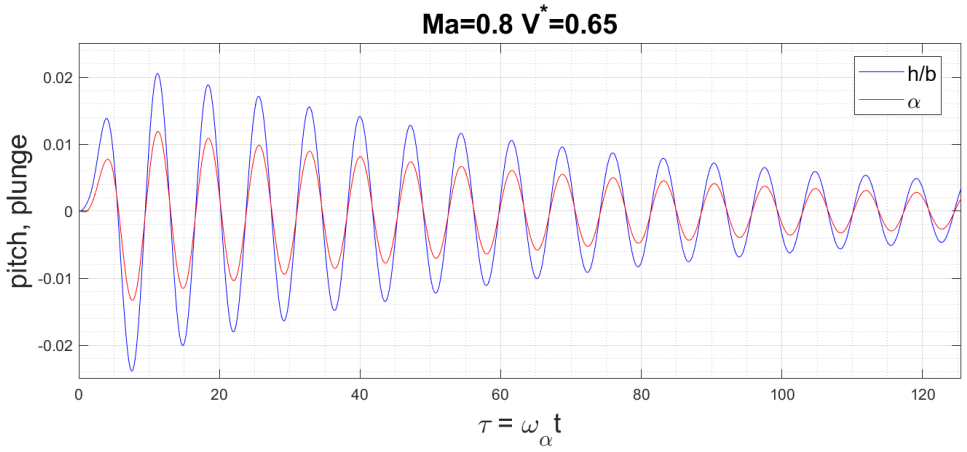
The solution of this LS problem uses the Moon-Penrose pseudo-inverse:

$$a = X = (M^T M)^{-1} M^T B = \frac{\sum_i t_i b_i}{\sum_i t_i^2} \tag{2.8}$$

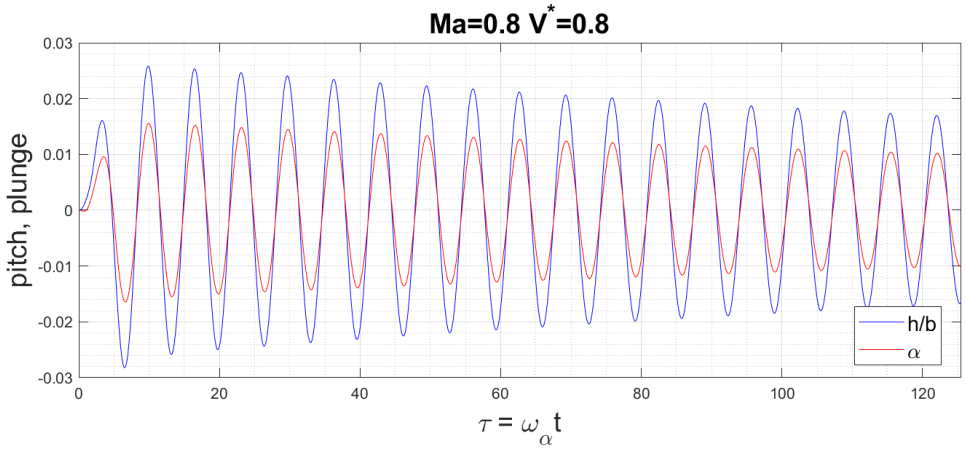
where b_i are the elements of B and the last passage is possible only because of the simple form of the equation in this case (and not in general).
Thus a is the best approximation of the exponential decay/growth of the time signal. To find exactly the flutter boundary, one should find where $a = 0$ (if $a > 0$ the signal decreases in amplitude with time, while for $a < 0$ it increases): to approximate the flutter boundary with a low computational cost, one could proceed in this way:

1. run at least 3 cases ($k = 1, 2, 3$) at the same Ma but at different V^*, to obtain at least one stable and one unstable response;
2. find a for each case (a_k);
3. plot $(V_k^*, a_k)_{|Ma}$ and find the zero of a_k by interpolation (in this case a *spline* interpolation was used);
4. verify that the point found in 3 shows a neutral behaviour (i.e. $a < \epsilon$, where ϵ is a user-defined tolerance value, close to zero) or use this verification point to start a new cycle

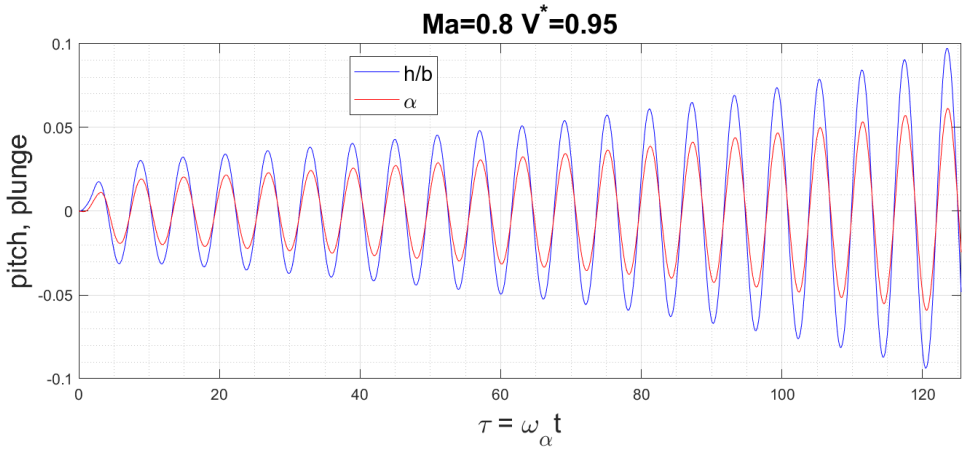
In the far-right side of the flutter boundary (see fig. 2.15) it is more convenient to run at constant V^* instead of at constant Ma, however, the aforementioned procedure can be easily converted for this case: the steps are exactly the same. (To be precise, it is necessary to specify that, to capture only the transient, all signals were cut from $\tau = 17.45$ on and time-traces of unstable responses were also cut superiorly if the signal saturates.) The results of this procedure show a good agreement with the the literature, as shown in fig. 2.15. Figures from 2.9 to 2.13 show the time-traces used to find the flutter boundary. Fig. 2.14 show the spectra obtained using the Fast Fourier Transform of some of the time-traces presented before. To compute the FFT in this case, since the timestep is constant, there is no need to resample the signal in time. Then, considering that cutting off the initial transient there are at least 10 periods in the time-trace, one can expect a good approximation in the definition of the dominant frequency of the signal. All signals show only one dominant reduced frequency and that the reduced frequencies grow as the flutter speed index decreases, as shown in fig.2.14. All signals show only the first mode of the system, as expected from literature.



(a)

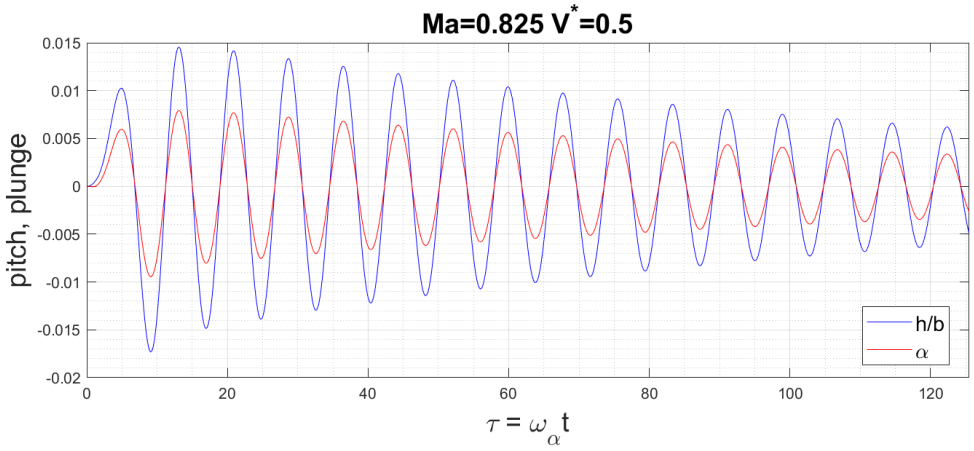


(b)

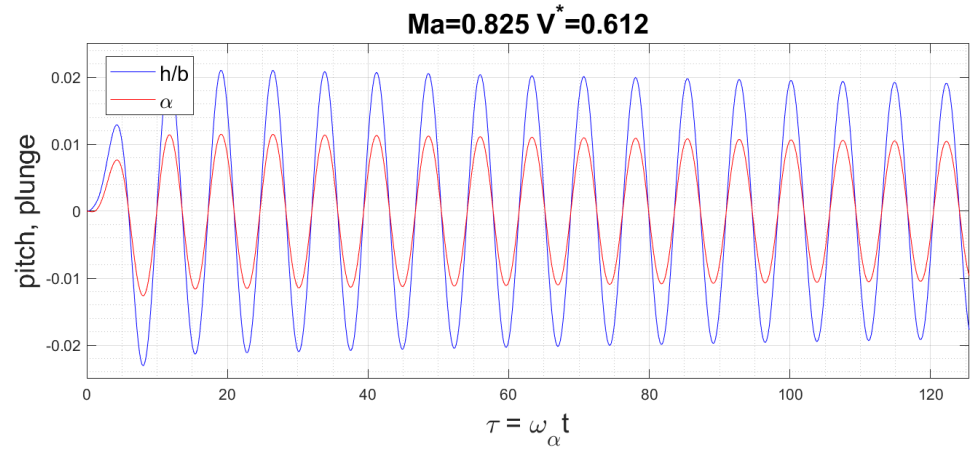


(c)

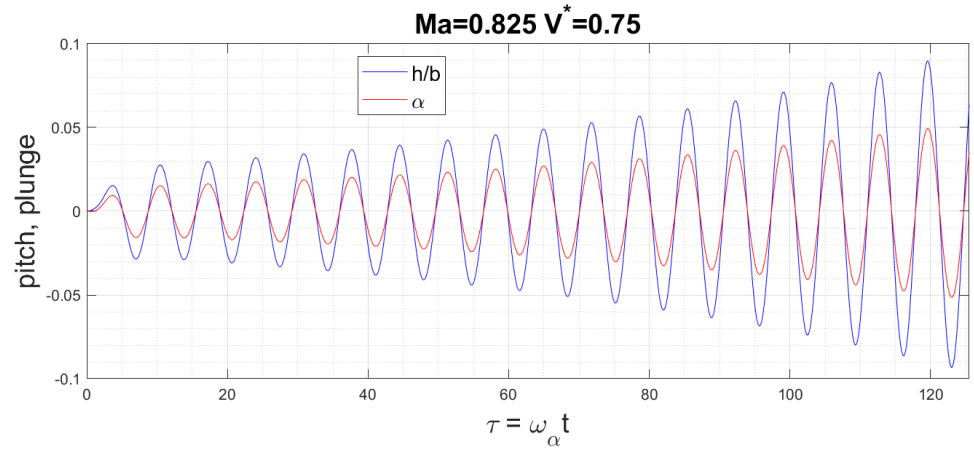
Fig. 2.9: Time-traces used to reconstruct the flutter boundary at $Ma = 0.8$



(a)

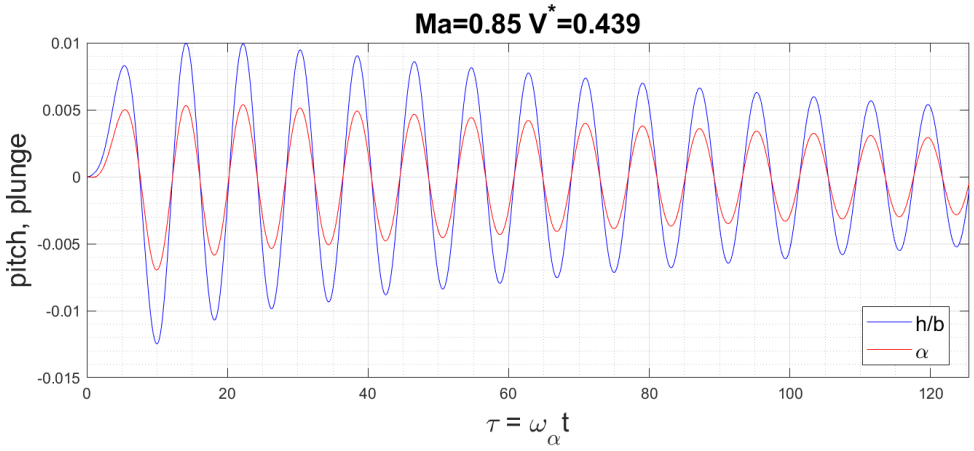


(b)

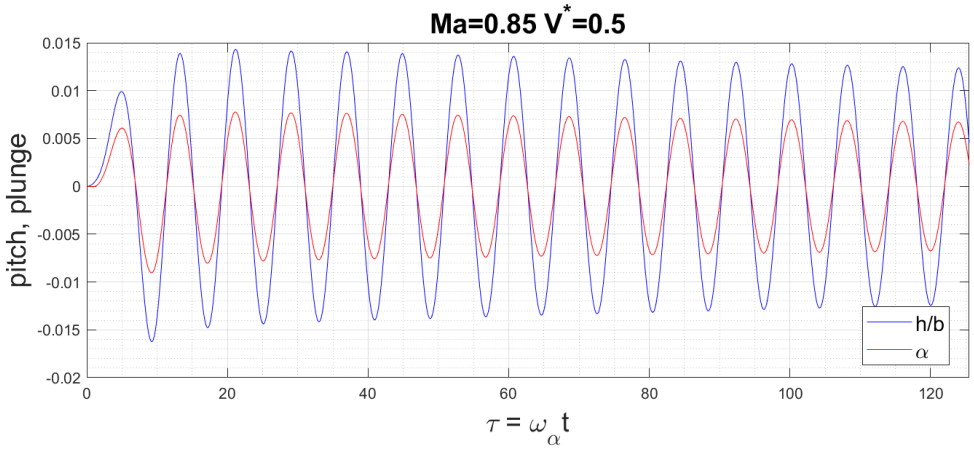


(c)

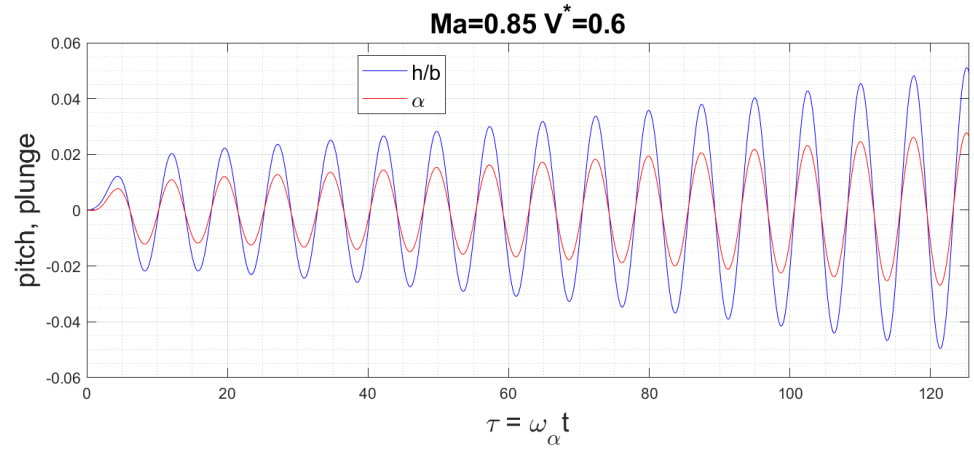
Fig. 2.10: Time-traces used to reconstruct the flutter boundary at $Ma = 0.825$



(a)

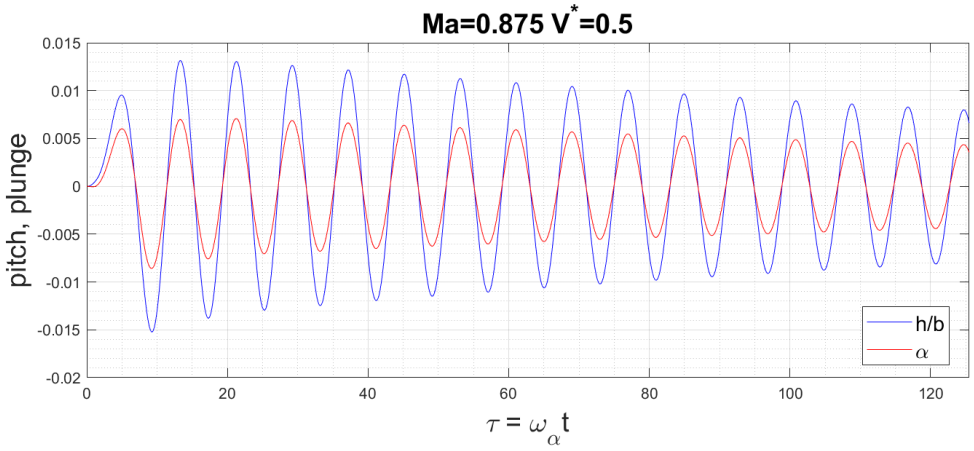


(b)

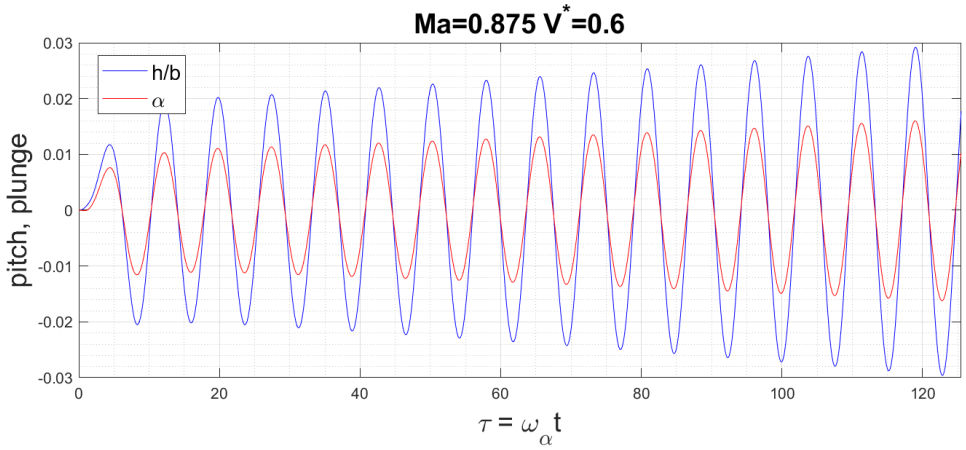


(c)

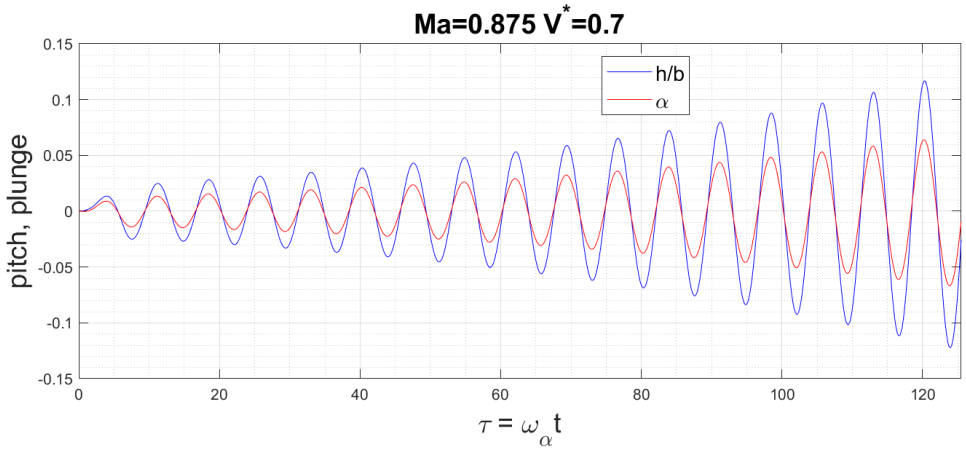
Fig. 2.11: Time-traces used to reconstruct the flutter boundary at $Ma = 0.85$



(a)

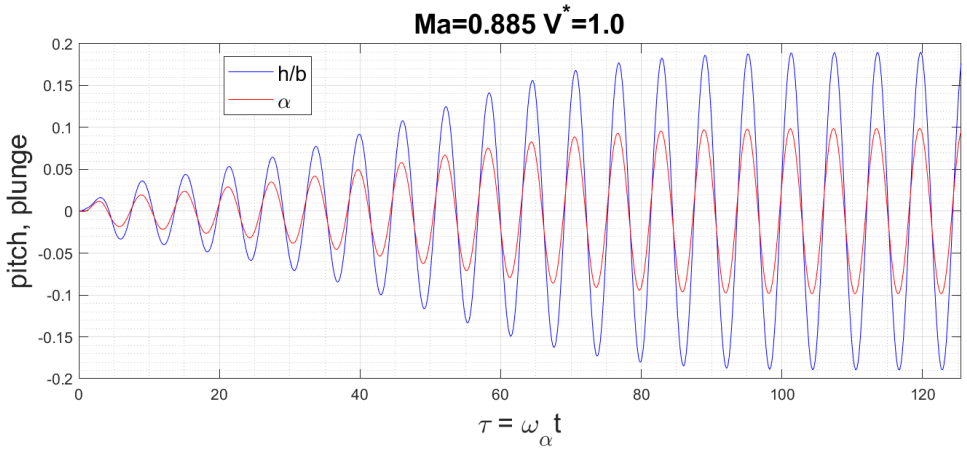


(b)

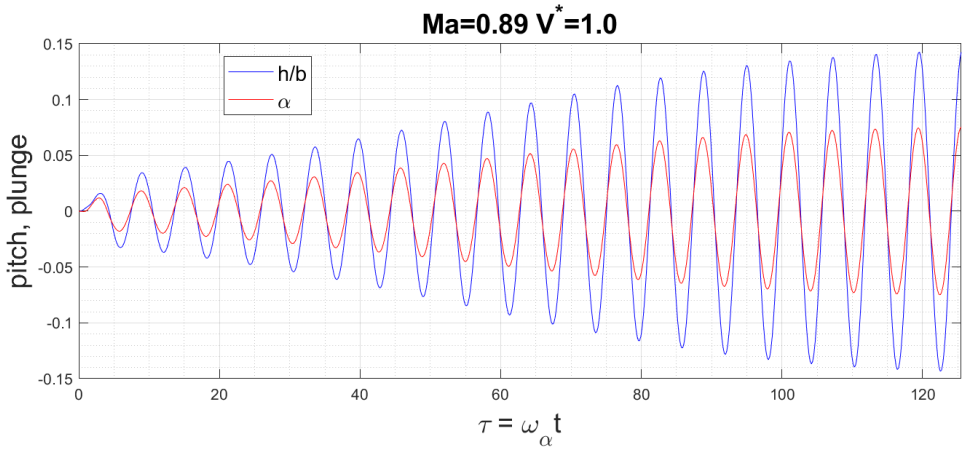


(c)

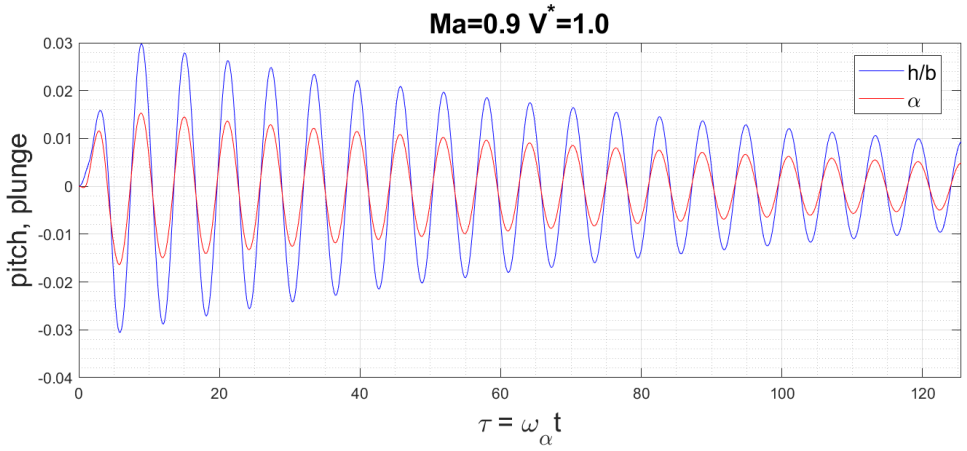
Fig. 2.12: Time-traces used to reconstruct the flutter boundary at $Ma = 0.875$



(a)



(b)



(c)

Fig. 2.13: Time-traces used to reconstruct the flutter boundary at $V^* = 1$

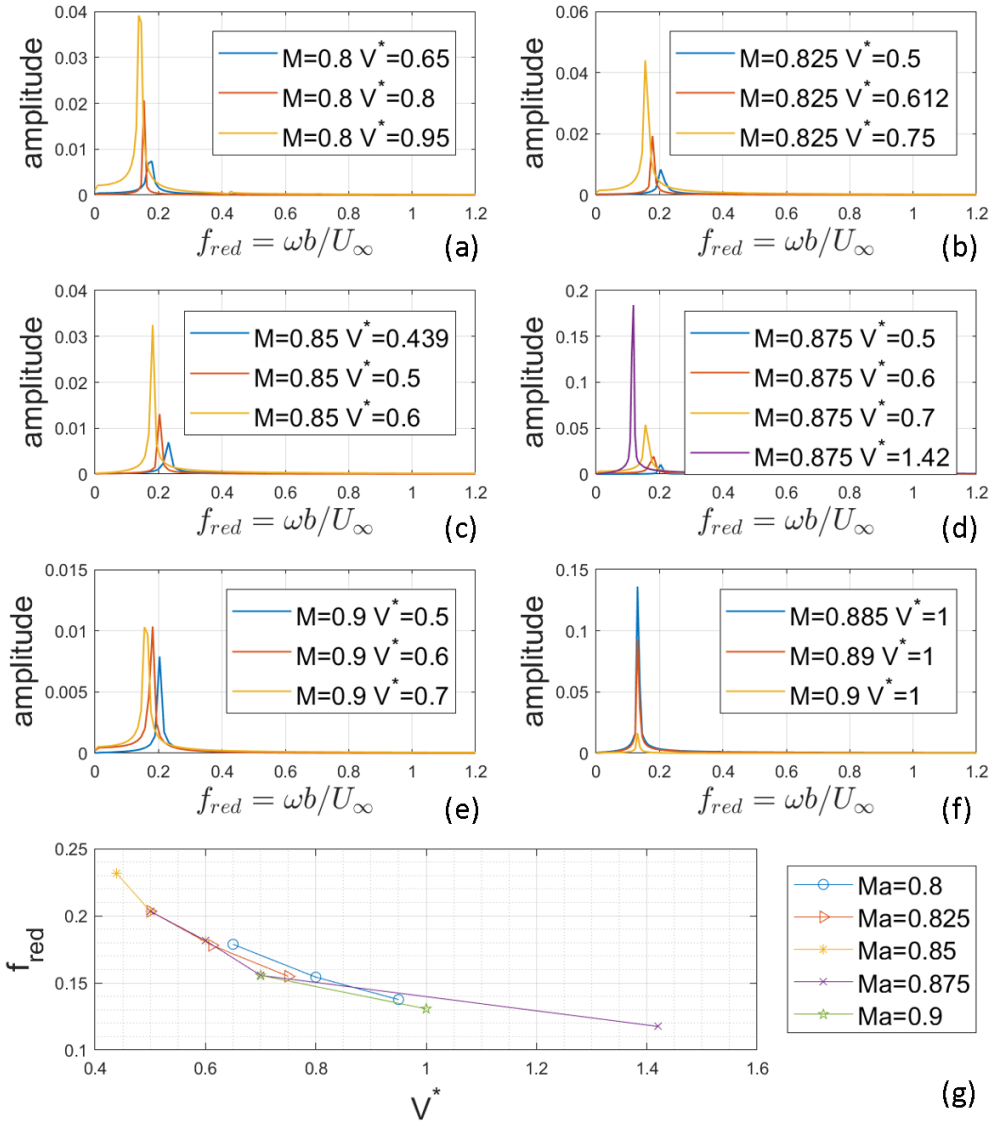


Fig. 2.14: (a-f) spectra of the plunge signal after the initial transient and (g) plot of the characteristic reduced frequency of the plunge signal vs flutter speed index for different Mach numbers. Fig. (g) shows the common trends of the first 6 figures: the reduced frequencies of the plunge signals are inverse function of the flutter speed index.

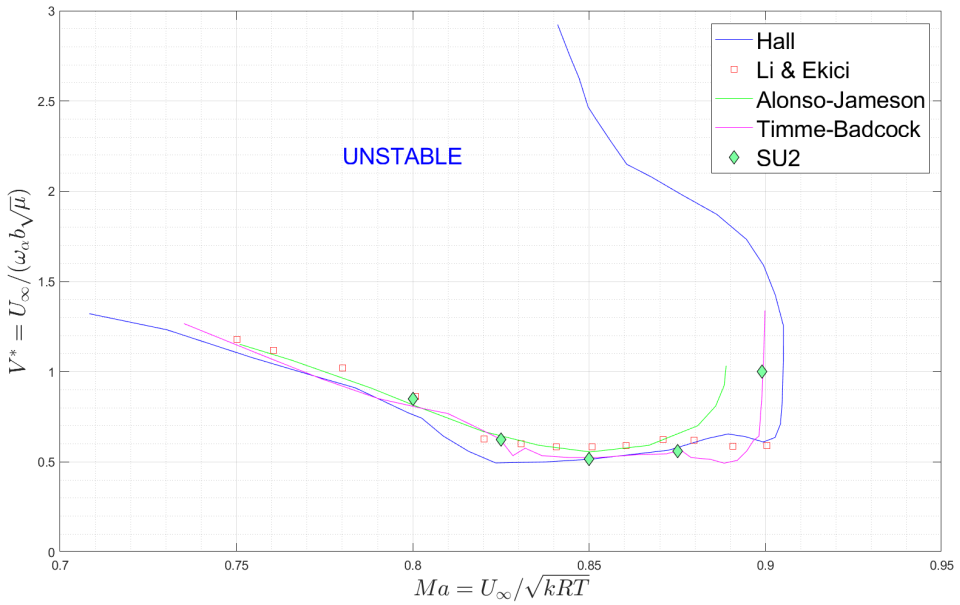


Fig. 2.15: Mach number vs flutter speed index boundary for different references [34], [31], [27], [38]. Green diamonds represent the present calculations.

2.3 Turbulent calculations

Viscosity and turbulence may play a significant role in the definition of the transonic flutter boundary since viscous fluxes are dissipative and thus viscosity damps the movement of the airfoil.

2.3.1 Turbulent flow governing equations

The flow governing equations are the unsteady Reynolds-averaged Navier-Stokes (URANS) equations. The solvable URANS equations obtained using Boussinesq hypothesis in index notation form are

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{2.9}$$

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{1}{\rho}\left(\frac{\partial p}{\partial x_i} + \frac{2}{3}\rho\frac{\partial k}{\partial x_i}\right) + \frac{\partial}{\partial x_j}\left[\frac{1}{\rho}(\mu + \mu_T)\frac{\partial u_i}{\partial x_j}\right] \tag{2.10}$$

where u_i is the generic component of the mean velocity vector.
Additional equations are needed to close the system: these equations are provided by the turbulence models, the viscosity model and the gas state equations. In this case, the Menter Shear-Stress Transport (SST) and the Spalart-Allmaras (SA) turbulence models are used, while the viscosity model is the Sutherland equation and the air is treated as an ideal gas (as in the inviscid calculations). A brief description of the aforementioned methods is available in Appendix A.
The equations in space are discretized using a finite volume method (FVM). The convective and viscous fluxes are evaluated at the midpoint of an edge. [28]

2.3.2 Setup

The setup is similar to the inviscid case, the main differences being:

- for viscous calculations it is necessary to define a Reynolds number. For comparison with cases present in literature, $Re = 1.256 \cdot 10^7$ is set for the dynamic cases and $Re = 1 \cdot 10^7$ for the static case. Viscosity is computed using Sutherland's law;

- since meshes for viscous calculations should be finer than the inviscid cases, the expected convergence rate is slower. For this reason, the number of internal iterations is increased to 2000 (in the inviscid case, 110-150 internal iterations were enough) and the timestep is slightly reduced: $\Delta t = 0.00163576s$;

- further details about the convective schemes are reported afterwards.

2.3.3 Mesh

Four different O-type meshes were used in the present case:

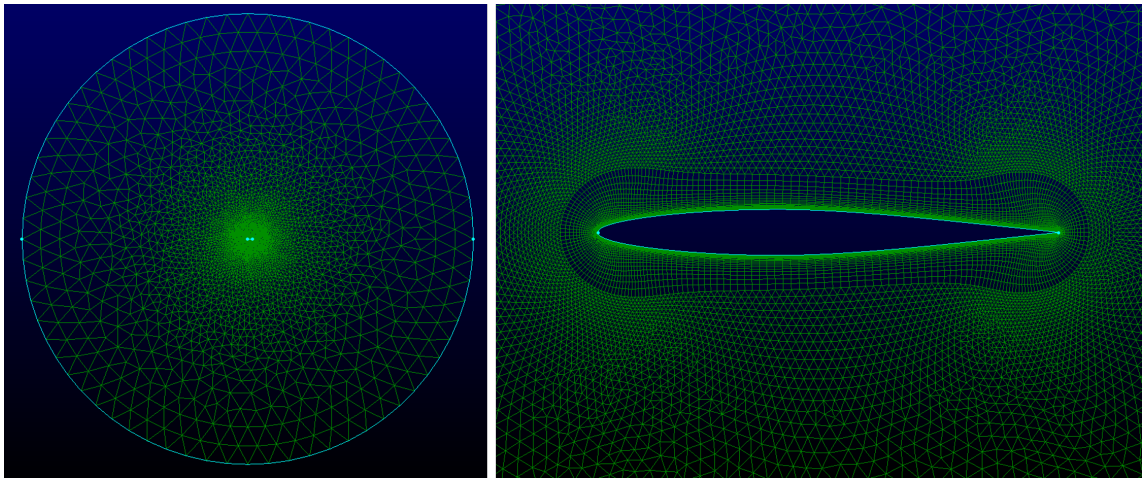
- *mesh A*: an unstructured mesh of 35404 elements with farfield placed at 50 times the chord from the airfoil and a first cell height of $2 \cdot 10^{-6}m$;

- *mesh B*: an unstructured mesh of 43632 cells with farfield placed at 70 times the chord from the airfoil and a first cell height of $1 \cdot 10^{-6}m$;

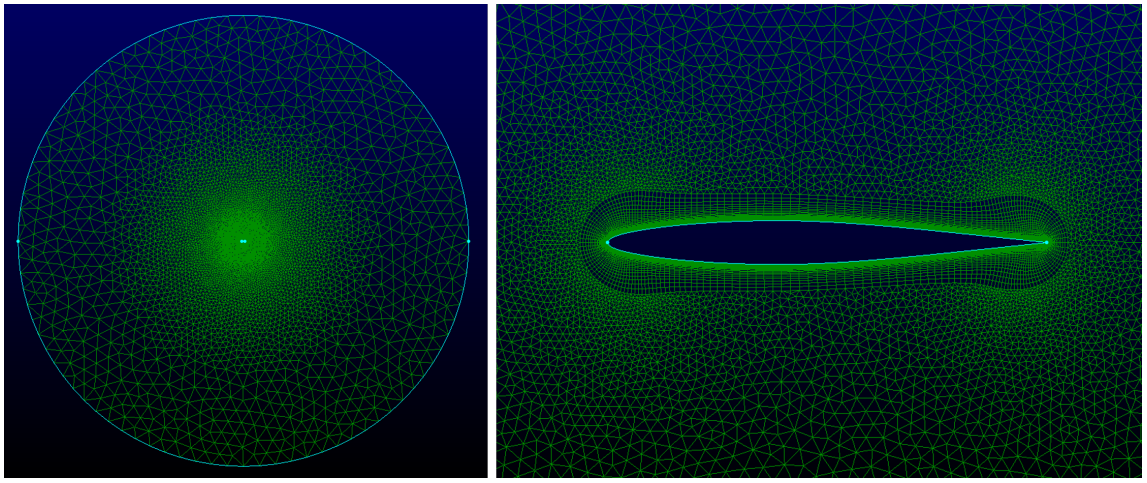
- *mesh C*: an unstructured mesh of 67359 cells with farfield placed at 100 times the chord from the airfoil and a first cell height of $1 \cdot 10^{-6}m$.

2.3.4 Validation and mesh sensitivity study

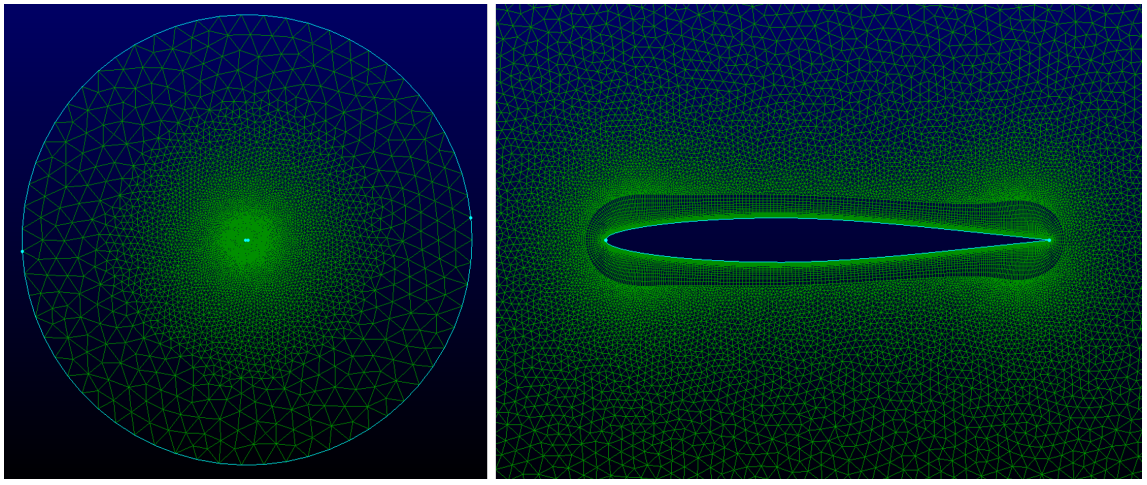
As in the inviscid case, the validation study is composed of a static and a dynamic part, the latter being performed using a wind-gust. The static validation is performed using a Spalart-Allmaras turbulence model and comparing the RANS results of SU2 with the work by Marti and Liu [29], who described flutter over a NACA 64-A010 airfoil, using an integral boundary layer code, both in the case of free transition and of fully developed turbulence using an e^N transition model.



(a)



(b)



(c)

Fig. 2.16: meshes A (2.16a) , B (2.16b), C (2.16c): general view and closeup on the airfoil

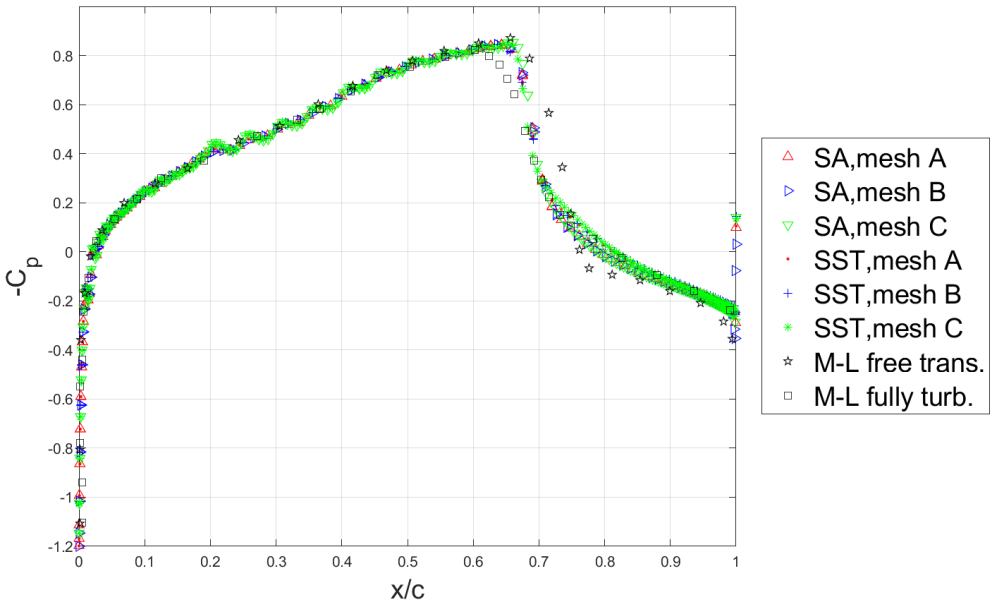


Fig. 2.17: Pressure coefficient vs non-dimensional position along chord for a static NACA 64-A010 airfoil in turbulent steady flow at $Ma = 0.85$, $Re = 10^7$, $\alpha = 0°$. The airfoil is symmetric, thus the plotted data represents both the upper and the lower side of the airfoil. Black curves represent data in literature [29]

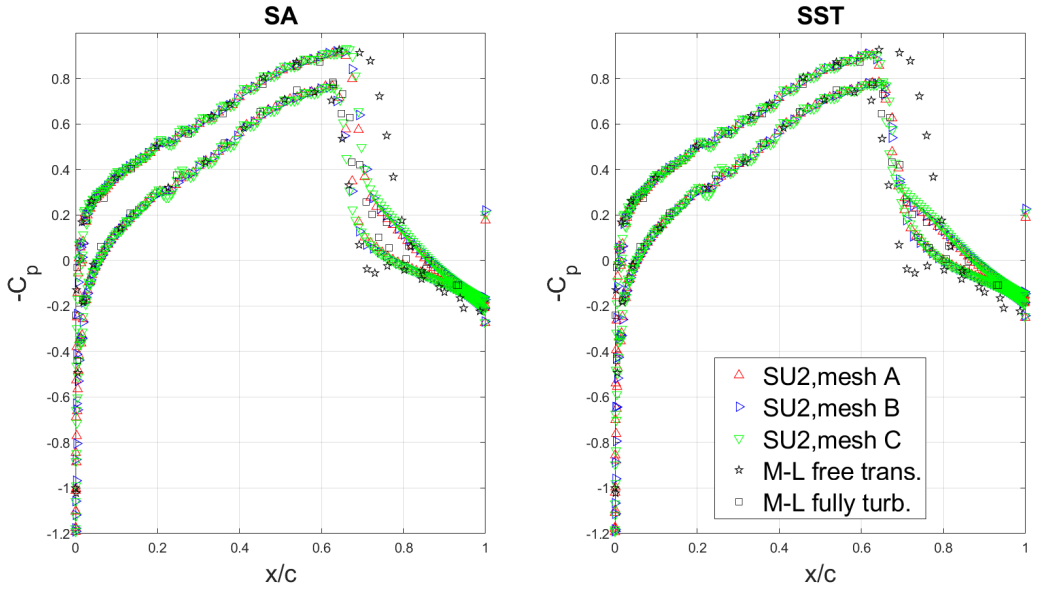


Fig. 2.18: Pressure coefficient vs non-dimensional position along chord for a static NACA 64-A010 airfoil in turbulent steady flow at $Ma = 0.85$, $Re = 10^7$, $\alpha = 1°$. The upper curves of each group represent the upper side of the airfoil and vice versa for the lower curves. The left figure compares the literature data [29] with the SU2 calculation made using the Spalart-Allmaras (SA) turbulence model, while the right figure is with the Menter SST $k - \omega$ (SST).

The agreement with the data by Marti and Liu is good as the main differences can

be found only near the shock-wave. Considering that a RANS calculation is being compared with an integral boundary layer code, some differences are natural, but the overall behaviour is well captured. In particular at $\alpha = 1°$ the SST calculations show a good agreement with the fully-turbulent plots in the reference, while the SA shows some minor differences in the shock-wave definition.
For the dynamic validation, there are some aspects that should be considered:

- preliminary calculations with fully turbulent models (SA, SST) show large overpredictions of the flutter boundary. This is a probable effect of the over-prediction of the viscous drag (a quite common problem in turbulence models for viscosity-driven cases) and numerical dissipation of the convective schemes;

- to overcome these problems, a transition model was added to the SA (transition models for SST are not implemented in SU2), slope limiters were disabled and several other convective schemes were used to search for less-dissipative solutions. The transition model is the BC [39], which proved to be more stable than the other models implemented in SU2. Turbulent parameters were set to near-zero values to mimic a quiet free-stream condition, so that the transitional model was properly conditioned;

- after exploring possible convective schemes and their low-dissipative options in SU2 with different meshes, the best choice (i.e. the choice of the parameters that minimized the discrepancy with data in literature [29]) was found to be the ROE 2nd-order convective scheme;

- turbulent calculations show a sharp tendency to diverge soon after the flutter boundary is crossed (this problem is also described in [40]), for this reason in figures 2.23a and 2.23b there are incomplete signals;

- turbulent cases are more time-consuming than inviscid cases: using 4 cores, a turbulent case runs in 60 hours using mesh A, while the same case with inviscid settings, using mesh 1, runs in 4 hours with 4 cores.

The results of the mesh sensitivity study are presented in figs. 2.21, 2.19 and 2.22.

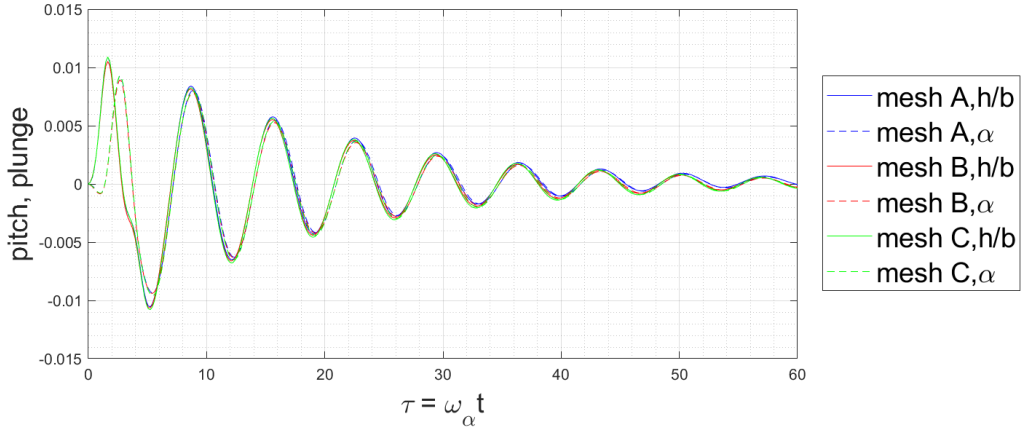


Fig. 2.19: Non-dimensional plunge and pitch time-signal for the three meshes

The mean and RMS contours are in good agreement for all considered meshes and minor differences can be found near the body. To understand this, one should consider that these contours are obtained from an unstructured set of data that is then resampled at constant spacing to plot the contours.

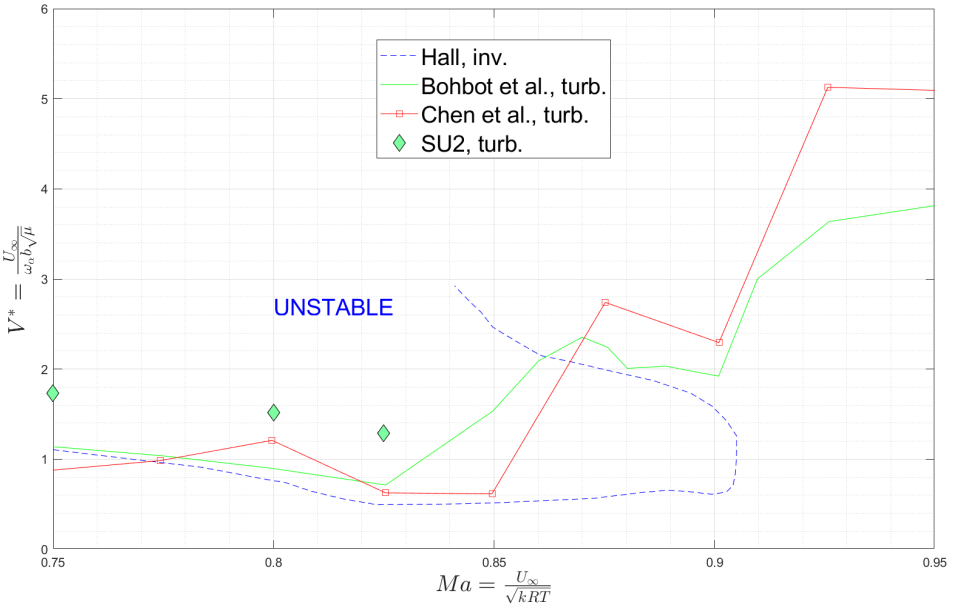


Fig. 2.20: Mach number vs flutter speed index boundary at $Re = 12,560,000$ for different references [41] , [42] for the turbulent cases and [31] for the inviscid case. In green is the position of the validation points for the turbulent case with SA model are reported.

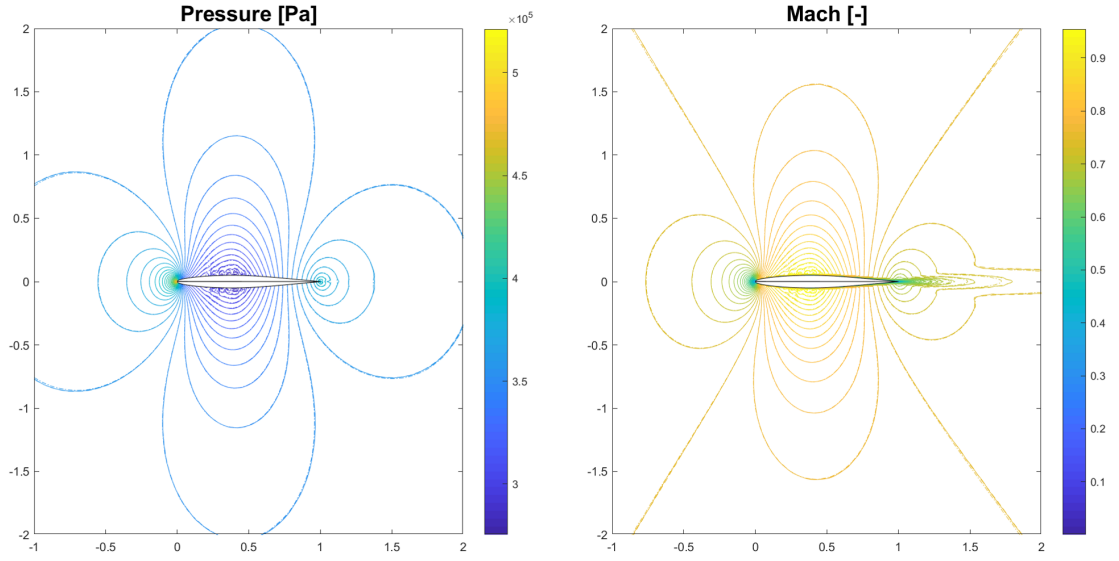


Fig. 2.21: Contours of the average pressure and Mach field on the average positions of the mesh points in time. Data is in $[Pa]$ and non-dimensional, respectively.

Since the smaller the spacing, the larger the matrix of the interpolated points, it is

not possible to resolve correctly the boundary layer region, where cells have dimensions of order $10^{-6}m$ in height. For this reason, the RMS field should be trusted out of the boundary layer, where the cell characteristic dimension is larger than the re-sampling size ($1mm$, in this case). The time-histories of pitch and plunge signal (fig. 2.19) show a nearly complete overlapping, confirming that the subsequent flutter boundary reconstruction can be achieved using the coarser mesh (A) without losing accuracy.

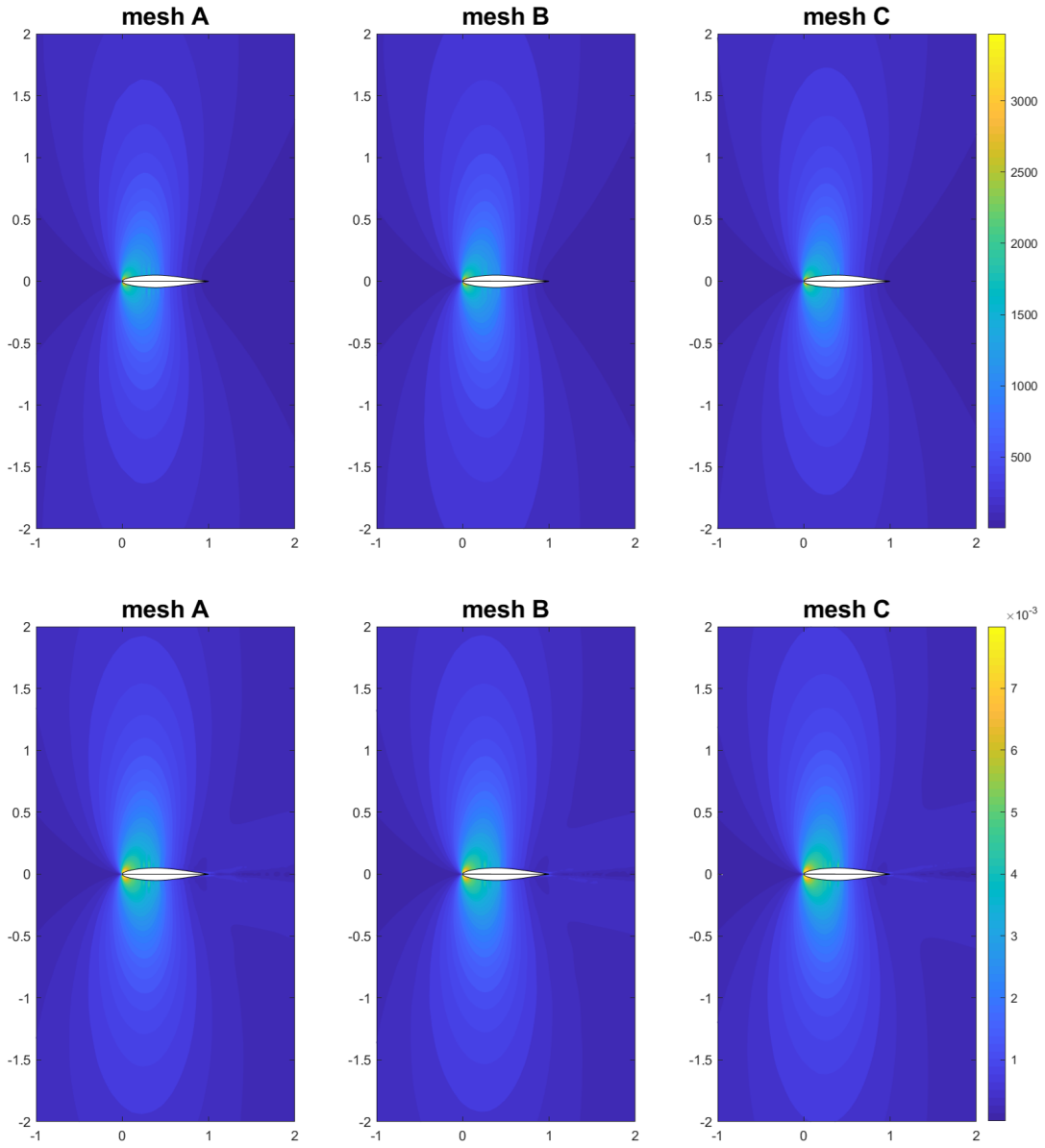
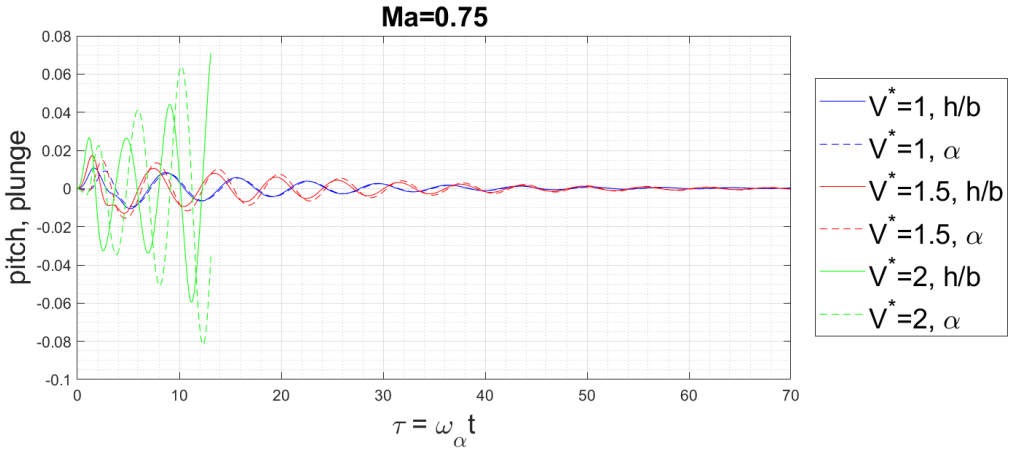
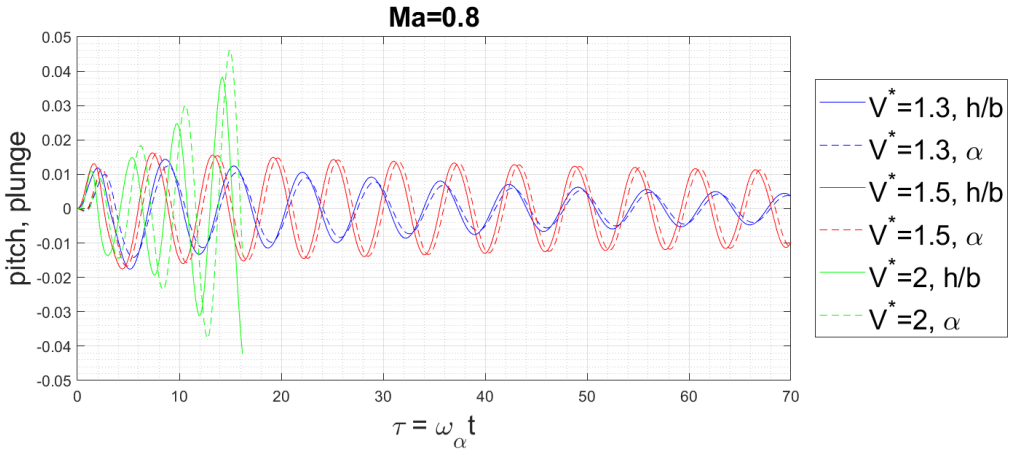


Fig. 2.22: Contours of the RMS of the pressure (upper) and Mach (lower) field on the average positions of the mesh points in time. Data is in [Pa] and non-dimensional, respectively.

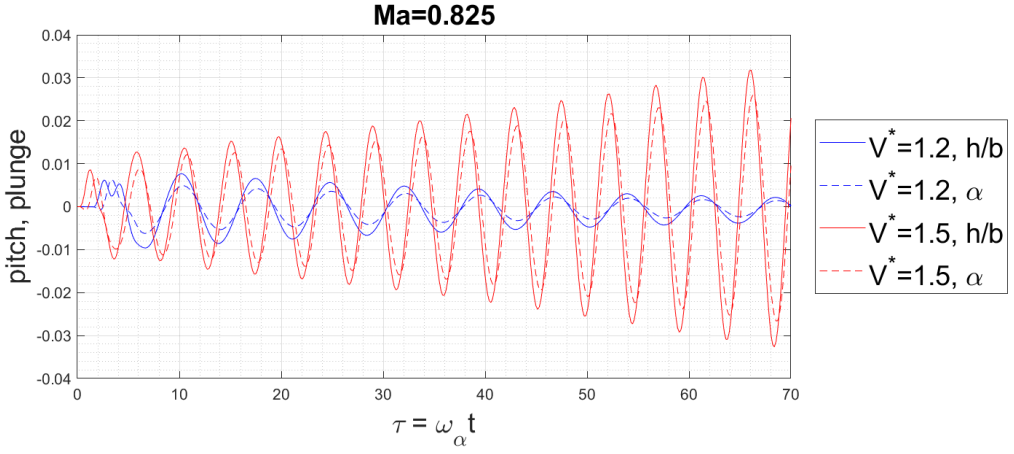
The results of the flutter boundary study in the turbulent case are presented in fig. 2.20.



(a)



(b)



(c)

Fig. 2.23: Time histories of the signal at $Ma = 0.75$ (2.23a), $Ma = 0.8$ (2.23b), $Ma = 0.825$ (2.23c) for different flutter speed indexes

By comparing the turbulent cases and the inviscid cases, the flutter boundary is overpredicted with respect to the inviscid calculation. it is even difficult to say that there is a transonic dip, which, on the contrary, is well recognisable in the inviscid case. Other authors, like Marti and Liu [29] found a flutter boundary similar to

the ones presented if fig. 2.20 (both for a fully turbulent case and for a transitional case), but with different values, suggesting that the turbulent cases presented in literature do show wider variations than the inviscid cases do. The SU2 calculations overpredict the flutter boundary with respect to the references for $Re = 12560000$. A reason that might explain this overprediction is the fact that a proper calibration of the turbulent parameters cannot be achieved directly in SU2, resulting in an increased dissipation. The time-traces used for the present case are shown in fig. 2.23.

2.3.5 Comparison of Turbulent and Inviscid results

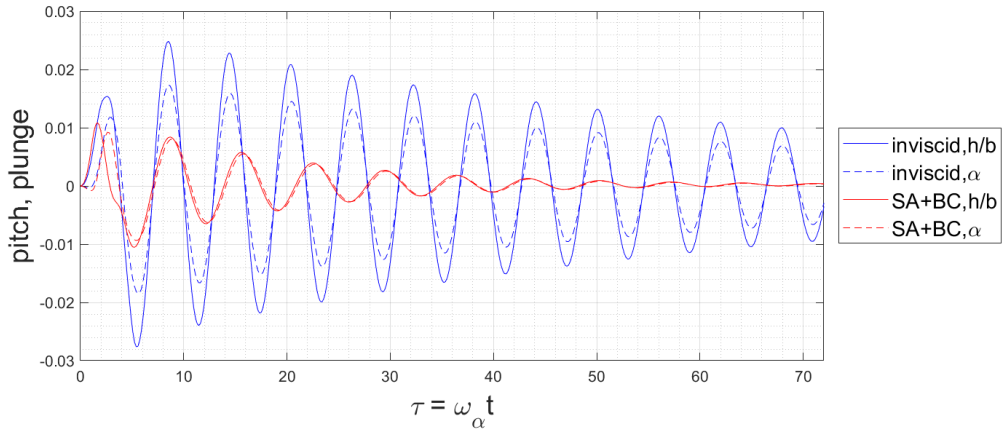


Fig. 2.24: comparison of turbulent and inviscid time histories at $Re = 12560000$, $Ma_{\text{inf}} = 0.75$ *and* $V^* = 1.0$

The main differences that can be found comparing viscous and inviscid results are:

- inviscid cases reach the flutter boundary faster than turbulent cases since viscosity is responsible for energy dissipation. This is clearly shown in fig 2.25, where the plunging signals related to the viscous calculations are smaller than those related to the inviscid case;

- turbulent Mach contours show a larger zone of low speed downstream of the trailing edge while inviscid cases are not able to capture this physical feature, as there is no viscous dissipation. Dissipation also plays a role in the determination of the characteristic frequencies of the system: the turbulent case shows a characteristic frequency that is slightly lower than that observed in the inviscid case;

- there is a phase-shift between the plunging and the pitching signal even in the stable zone for the turbulent cases, while this does not happen in the inviscid case. This may be ascribed to the damping introduced by the viscosity: pitching and plunging modes are damped in different ways as the area exposed in the plunging direction is different from that in the flow direction and the pitching angle is the main responsible for the generation of lift, which, in sequence, is responsible for plunge motion. Furthermore, the ratio between

the pitching and the plunging signal is higher in the turbulent case than in the inviscid one.

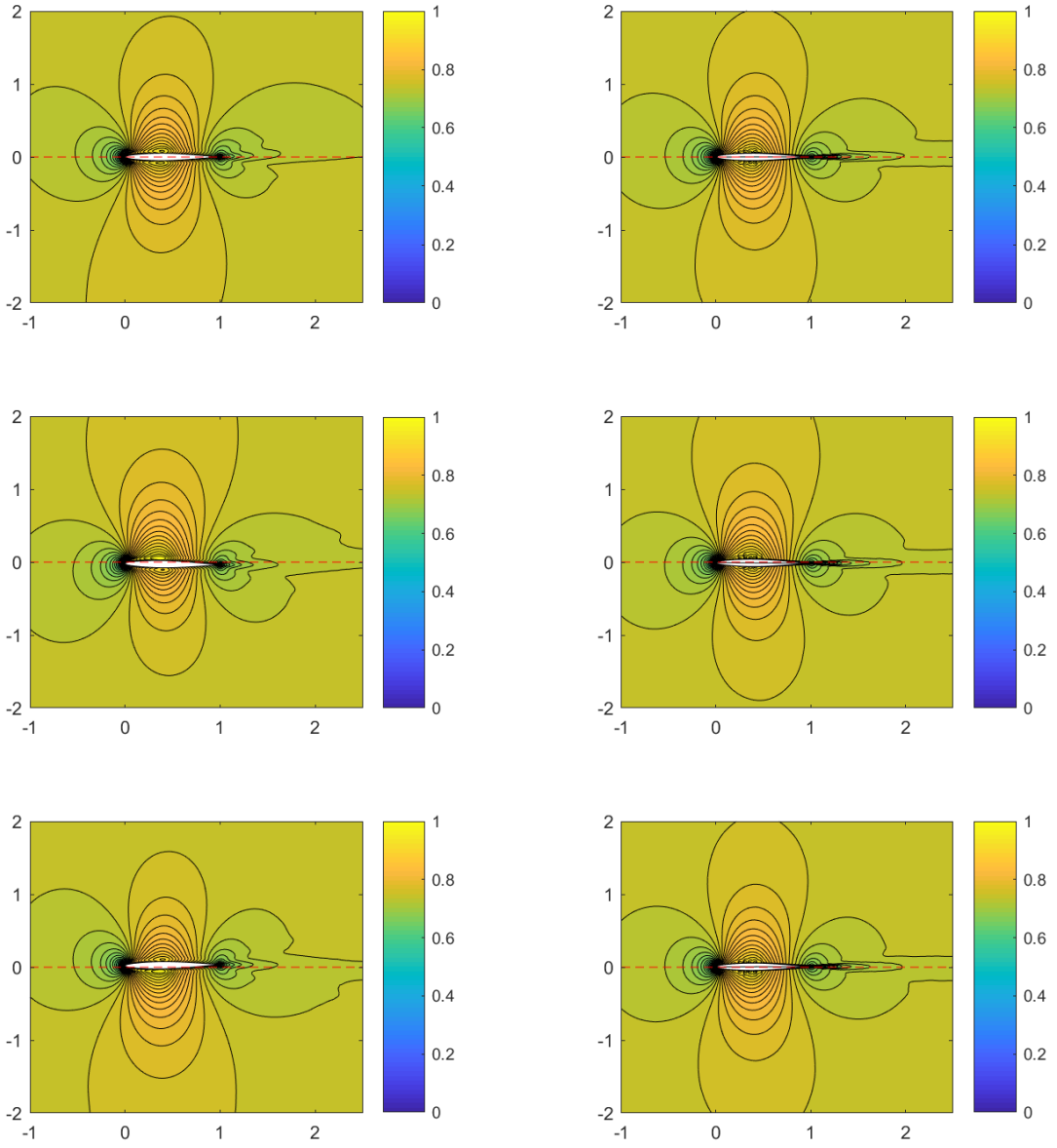


Fig. 2.25: Comparison of turbulent and inviscid instantaneous Mach fields at $Re = 12560000$, $Ma_\infty = 0.8$ *and* $V^* = 1.0$*: the pictures on the right represent the turbulent case and the inviscid case can be seen on the left. The first line represents a condition near* $h/b = 0$*, the second represents a maximum of* h/b *and the last a minimum. The red dashed line represents the starting condition (i.e. no plunge, no pitch).*

3 Optimization of a flutter energy harvester

Most works related to harvesters are aimed at proposing a wind energy-harvester, which could be more convenient than a classical wind turbine. Following this idea, it is natural to think about an Aeronautical application of flutter energy harvesters in place of Ram-Air Turbines (RATs) for supplying power to off-board equipment in military and agricultural aircraft, for example. Compared to conventional RATs, EHAF RATs may have the following advantages:

- smaller radar cross-section. The presence of a propeller may have a detrimental effect on radar detection [43]. New generation high-power RATs use a ducted fan instead of a propeller. This may reduce the radar cross-section but increases the weight of the device (a significant part of the device is occupied by the duct and the fan) [44];
- the power supplied by a RAT is a function of the air density and the airspeed and the efficiency of a propeller drops above a certain Mach number, typically. For a propeller, variable-pitch is used to optimize the performance. EHAF devices may benefit from operating in the transonic regime and show more degrees of freedom that could be tuned dynamically to achieve an optimal performance in a greater part of the flight envelope. The cut-in speed of a RAT is typically 80 knots [45], while flutter devices can work with lower winds;
- may be useful for low-power devices, which are currently supplied using on-board electric systems.

The present work aims at finding a set of values for the mechanical parameters of the problem to maximize the power that can be extracted from LCOs and eventually converted in electric energy.

Fig. 3.1: AN/ALQ-99 electronic-warfare pod mounted under a Boeing E/A-18G's wing. The RAT can be seen at the front of the device. Wikimedia Commons

Fig. 3.2: Emergency retractable RAT of a SAAB J-37 Viggen. Wikimedia Commons

3.1 Ram-Air Turbines

RATs are very common in aeronautics and may be used for three reasons:

1. for emergency power-supply in case of engine-fail: in this case the RAT is a simple propeller-driven generator, which is normally closed in an external compartment of the aircraft (fig. 3.2). In case of an engine-fail (and APU-fail, if present), to avoid the loss of the electric systems, the RAT propeller is extracted from the compartment and exposed to the airflow and electric power is generated from its rotation. Since this is an emergency system, it has to be reliable and there is no need for an optimization of the system in transonic or supersonic flight, since after an engine-fail an aircraft would glide in low-Mach flight;

2. for permanent power-supply in off-board equipment on military or agricultural aircraft. In this case, the RAT is a propeller-driven generator fixed (generally) at the front of the device. Examples are the AN/ALQ-99 radar jamming pods (fig. 3.1) used on the Boeing E/A-18G [46], or the high-power ducted ram-air turbine (Hi-RAT) developed by *ATGI* for reconnaissance and surveillance operations and its counterpart by *Raytheon* [44] for the New Generation Jammer (NGJ) project. In the military framework, RATs are also used as power supplies for the M61 Vulcan cannon [47] and for the *Blue Danube* British-made nuclear bomb radio-altimeter [48]. In agriculture, where small general-aviation aircrafts are used to diffuse liquid agents (pesticides or other treatment), RATs are used to move centrifugal pumps that pressurize the spray-diffusion systems

[49] [50]. This solution is optimal in agriculture because there is no direct coupling between the aircraft's engine and the centrifugal pump and thus no need for additional FAA certifications on the aircraft;

3. for permanent power-supply in on-board equipment. This case is the rarest and can be found on a few small airplanes. An historical example is the RAT mounted on the nose of the Messerschmitt Me-163 B "Komet". The Me-163 was a small "rocket-glider" combat plane of World War 2: during ascent and combat the aircraft was pushed by a rocket engine, then it approached and landed as a glider when fuel was exhausted. The peculiar power plant of this airplane could not power also the avionics, so a RAT was mounted for the purpose [51]. Similar use was made to power the actuator of the variable-pitch propeller on the airplanes pushed by the Argus As-410 piston engine [52].

3.2 Mechanical Optimization

The optimization problem is formulated as:

find the values of the mechanical parameters ω_h, ω_α, x_α and AoA that maximize a given cost function. The cost function is a measure of the maximum useful output that can be extracted from the airfoil's plunging motion.

The motion used to produce energy is only the plunging component as most of the works present in literature propose this method, at least for rigid-body foil structures. The way of defining the cost function to maximize is not unique, since the method to convert the plunging motion into electric energy is not known a priori. For this reason, two possible ways of defining the cost function are:

1. use the root-mean-square (RMS) value of the power exchanged in the plunging spring at LCO:

$$cost = -RMS\left(k_h h\frac{dh}{dt}\right) = RMS\left(h/b\frac{dh/b}{dt}\omega_h^2\mu\pi\rho_\infty b^2 b^2\right) \tag{3.1}$$

2. since energy would be generated using an induction system, considering that induction is described by Faraday's law (which states that *the electromotive force is proportional to the rate of change of the magnetic flux*), the maximum amplitude or the RMS of the derivative of the plunging motion is a suitable parameter to describe the "ability" of the system to make energy available.

To be precise, both methods have limitations because no mechanical damping is considered. However, it is reasonable to think that the highest power production can be found where the highest power exchange at the spring is found. As far as the non-dimensional distance between the center of gravity and the elastic axis is concerned, a variation of this parameter results in a variation of the moment of inertia and, thus, of the radius of gyration r_α. Since $r_{CG}^2 = 0.24$ [13] is obtained by mean of the Huygens-Steiner theorem: $r_\alpha^2 = r_{CG}^2 + x_\alpha^2$. The reference origin for moment calculation (rom) is obtained by $rom = x_{CG} - x_\alpha b$ to keep the CG in fixed position ($x/c = 0.4$). The freestream conditions are set for a flight at 10000m in standard ICAO atmosphere: $Ma_\infty = 0.85$, $p_\infty = 26436Pa$, $T_\infty = 223.15K$.

3.2.1 The Nelder-Mead algorithm

The optimization problem is solved using the Nelder-Mead downhill simplex algorithm [53].

Algorithm 1: Downhill Simplex Method

Let $f : \mathbb{R}^n \to \mathbb{R}$ be the function to minimize and let $\{x_0, x_1, ..., x_N\}$ be the current simplex and N the number of free parameters. Let k be the current iteration of the algorithm;

initialization: define a starting simplex: $\{x_i\}_{i=0}^{N}$;

$k \leftarrow 0$;

while *STOP CRITERION* ***and*** $k < k_{max}$ **do**

  sort the simplex in ascending order of $f(x_i)$;

  $h \leftarrow i \mid f(\boldsymbol{x_i}) = max(f(\boldsymbol{x}_i = 0, ..., N))$ *% the "worst" point;*

  $l \leftarrow i \mid f(\boldsymbol{x_i}) = min(f(\boldsymbol{x}_i = 0, ..., N))$ *% the "best" point;*

  $\bar{\boldsymbol{x}} \leftarrow mean(\{\boldsymbol{x_i}\}_{i=0,..N-1})$ *% centroid of the non-worst points ;*

  $\boldsymbol{x'} \leftarrow (1+\alpha)\bar{\boldsymbol{x}} - \alpha \boldsymbol{x_h}$;

  if $f(\boldsymbol{x'}) < f(\boldsymbol{x_l})$ **then**

    $\boldsymbol{x''} \leftarrow (1+\gamma)\boldsymbol{x'} - \gamma\bar{\boldsymbol{x}}$;

    if $f(\boldsymbol{x''}) < f(\boldsymbol{x_l})$ **then**

      $\boldsymbol{x_h} \leftarrow \boldsymbol{x''}$ *% expansion;*

    else

      $\boldsymbol{x_h} \leftarrow \boldsymbol{x'}$ *% reflection;*

  else if $f(\boldsymbol{x'}) > f(\boldsymbol{x_i}) \quad \forall i \neq h$ **then**

    if $f(\boldsymbol{x'}) \leq f(\boldsymbol{x_h})$ **then**

      $\boldsymbol{x_h} \leftarrow \boldsymbol{x'}$ *% reflection;*

    $\boldsymbol{x''} \leftarrow \beta\boldsymbol{x_h} + (1-\beta)\bar{\boldsymbol{x}}$;

    if $f(\boldsymbol{x''}) > f(\boldsymbol{x_h})$ **then**

      $\boldsymbol{x_i} \leftarrow \frac{\boldsymbol{x_i}+\boldsymbol{x_l}}{2}$ *% multiple contraction;*

    else

      $\boldsymbol{x_h} \leftarrow \boldsymbol{x''}$ *% contraction;*

  else

    $\boldsymbol{x_h} \leftarrow \boldsymbol{x'}$ *% reflection;*

  $k \leftarrow k + 1$;

Result: $\boldsymbol{x_l}$, *set of parameters that minimizes* f *(locally, in general)*

The implementation and standard coefficients for the algorithm are taken from [54]. The Simplex downhill does not need the gradient of the cost function f in the evaluation points $\boldsymbol{x_i}$ and the algorithm proceeds heuristically searching the steepest descent using a so-called *simplex* (i.e. a set of $N+1$ points -a polytope- in a N-parameters space). The algorithm is described in Alg. 1, where $\alpha = 1 > 0$, $0 < \beta = 0.5 < 1$, $\gamma = 2 > 1$ are the reflection, contraction and expansion coefficients, respectively, with their standard values. The $STOP\_CRITERION$ given in the original reference is

$$\sqrt{\frac{1}{N+1}\sum_{i=0}^{N}(f(\boldsymbol{x_i}) - \overline{f(\boldsymbol{x_i})})^2} \leq \epsilon \tag{3.2}$$

where $\bar{f}$ is the mean value of $f(\boldsymbol{x_i})$. This definition of the stop criterion links the size of the simplex with an approximation of the local curvature of f, thus it avoids running for long times to find a minimum in nearly-flat valleys of the function (if present).

3.2.2 Implementation in MatLab and SU2

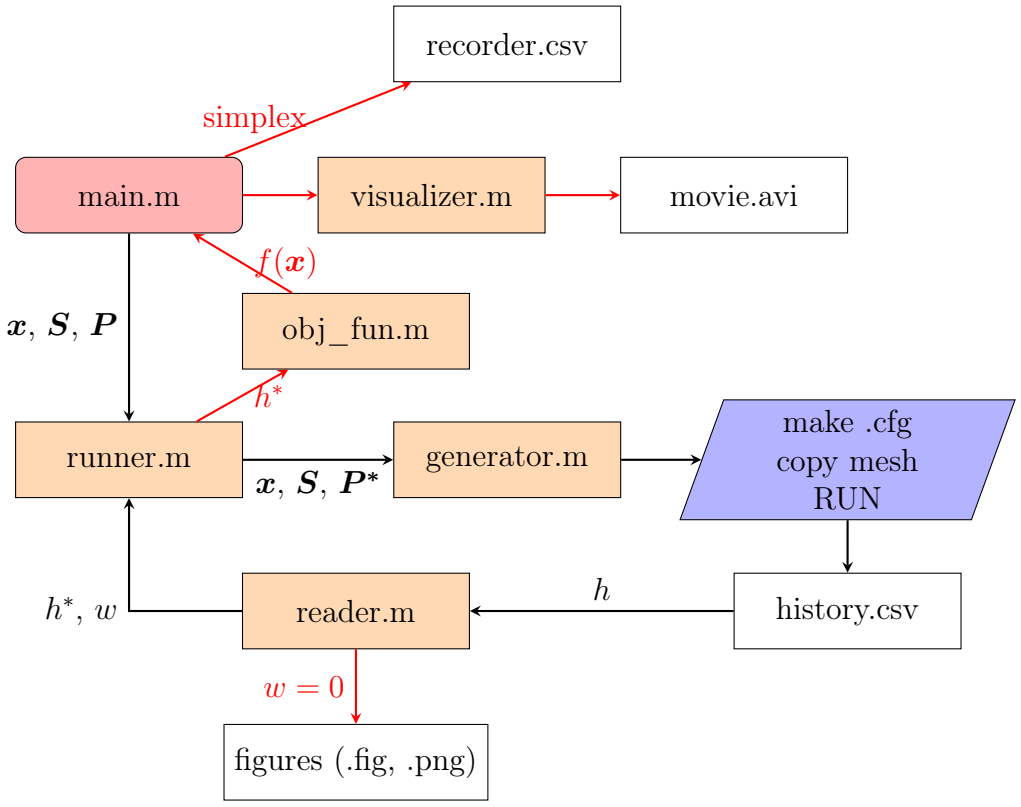


Fig. 3.3: Schematic of the implementation of the Downhill Simplex method in MatLab and SU2. Yellow blocks identify MatLab functions, the red block is the main MatLab script, blue blocks identify SU2 files and white blocks represent output files.

MatLab can be used to launch a command from the system command line both in Unix and MS systems. This feature can be useful in this case, since in the present work the main program will iteratively call some functions that will create

directories, create a SU2 configuration file and copy a SU2 mesh file there. In MatLab a function based on a modified version of the Downhill Simplex algorithm is already implemented (see *fminsearch* [55] for reference), but in the present case it is more suitable to implement the algorithm directly, as the action of reading signals and writing/launching the simulations are harder to implement (signals of the same kind may differ significantly for small variations of the parameters and the implementation must be robust enough to read correclty any kind of signal) than the optimization algorithm itself. Coding the algorithm in this case also offers more control on the outputs (for example, the restarting procedure is faster). The implementation is modular, assigning different tasks to different functions in a way such that the debugging procedure is fast and the different modules may be changed on-the-fly and re-used with minor (or no) modifications for other tasks (for example the perturbed cases for the robust-design study). The structure of the code is shown in fig. 3.2.2
The operations executed in the algorithm can be summarized as:

1. the parameters defining OS data $\boldsymbol{S}$ (e.g. the number of cores), fixed physical parameters (e.g. the Mach number) and simulation (e.g. maximum number of iterations) parameters $\boldsymbol{P}$ along with an initial simplex are defined in *main*;

2. the *main* calls the *runner* function and passes the data to it;

3. the *runner* calls *generator.m*, which creates a folder for the present calculation, writes a SU2 configuration file in that folder and copies the .su2 mesh file there. Then it calls the command prompt and launches the SU2 calculation in parallel mode and waits for the process to finish, after a predefined number of iterations. Then, it returns some parameters to the runner. These parameters are used to locate the folder of the current simulation (where the reader function will be working) and determine if it has diverged or not;

4. .vtk/vtu files, restart files and a time history of the simulation are created in the working folder by SU2: the *runner* calls the *reader* function, which reads the plunging signal h in *history.csv* file, computes its FFT and divides the signal in samples based on an estimate of its dominant frequency. Statistical moments are evaluated for each sample to determine where the signal becomes statistically stable. If this goal is reached, the reader returns the statistical stable part of the signal (*) to the runner and plots the time history and the FFT of the signal. There might be cases in which this does not happen:

    - in case the signal amplifies but it is too short to become statistically stable or that state is too short to correctly evaluate the cost function;
    - in case the signal either is neutral or damped or the signal does not tend to zero (for example because of a non-zero AoA) but converges to a finite value for long times (in this case the Fourier transform of the signal will show a peak near zero). In these sub-cases, the output signal is set to zero as it is not possible to exploit the system for energy harvesting;
    - the simulation diverged;

In these cases the *reader* issues a warning w and the *runner* restarts the simulation in case it is necessary to run longer or modifies the parameters P^* to avoid divergence;

5. the statistically stable signal is passed to the obj_fun function to evaluate the cost function $f(\boldsymbol{x})$ and its value is returned to the *main*;

6. the previous procedures are repeated for each component of the initial simplex and then for each new point in the Downhill Simplex algorithm in the *main*. Each simplex is stored in an external .csv file and in an internal 3D matrix in MatLab;

7. when stop criteria are met in the *main*, the *visualizer* function is invoked. This function represents the steps the algorithm has followed to find the given optimal solution, the first line of the last block in *recorder.csv*. A movie of the process is produced and saved.

The present code is tested on the Rosenbrock function to compare the results with [54]. This function is a classic testcase in optimization problems, which is described by a two-variable equation:

$$f(\boldsymbol{x}) = (1 - x_1)^2 + k(x_2 - x_1^2)^2 \tag{3.3}$$

the minimum of the function is in $(1, 1)$, where $f(1, 1) = 0$. In this case, $k = 10$ as in reference [54]. The *runner* function is substituted with a function that, given $\boldsymbol{x}$, returns $f(x)$ (the value of the Rosenbrock function), so only the *main* is tested in this case, while local tests on the other modules are carried out separately. The results of the application to the Rosenbrock function is provided in fig. 3.4.

With respect to the original version of the Nelder-Mead algorithm 1, the present implementation shows some modifications that are needed in order to make the code less sensitive to the choice of the initial simplex and offer the possibility to limit some parameters (the original algorithm is suitable only for unconstrained optimization):

- *constraint on ω:* the parameters ω_α and ω_h are constrained to be positive numbers. To do so, one can choose two ways:

    1. *way 1*: if one of these parameters is set by the algorithm to a negative value, return a cost function value that is higher than any other cost functions values in the current simplex. Since the cost function in this case is always a negative number, it would be a good choice to assign zero to the cost function in case of constrains violation;
    2. *way 2*: pass the absolute value of the parameter to the simulation: this choice is very simple compared to *way 1*, but it is not able to handle near-zero values of the constrained parameters (that usually produce poor results and are difficult to handle, as the solver may have to handle large deformations);

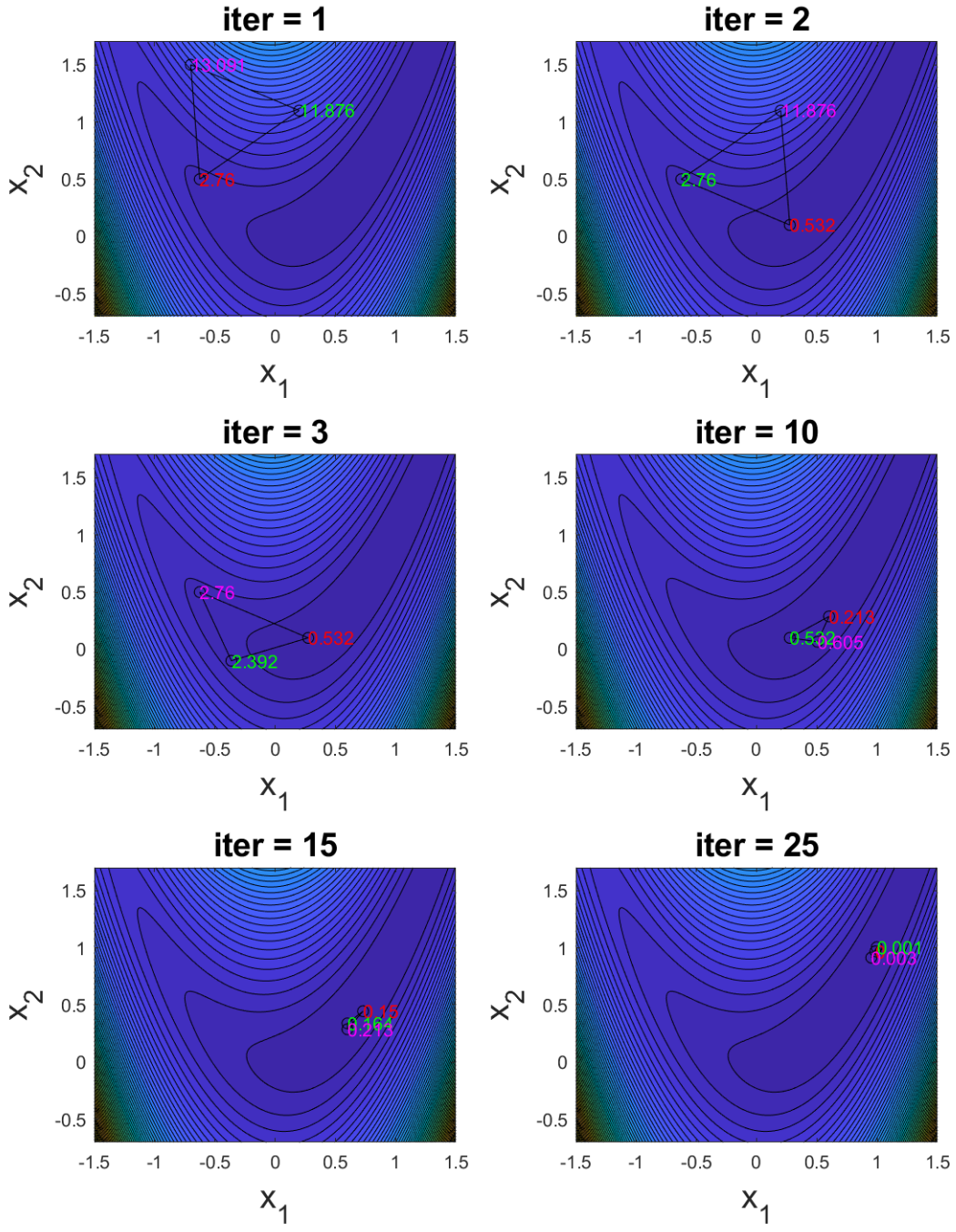


Fig. 3.4: Downhill Simplex method applied to the Rosenbrock function in MatLab. The values of $f(x)$ are marked on the corresponding point of the simplex. The algorithm converges to the real minimum of the function with a precision of about 0.001 in 26 iterations.

- *constraint on amplifying signals:* only amplifying signals are acceptable to compute the cost function, as in damping signals the system cannot sustain its oscillation and in neutral signals the removal of energy from the system would generate a damped behaviour. For this reason, the algorithm is constrained to search only for amplifying solutions using *way 1*;

- if the constraint implementation is made using *way 1* and if the choice of the initial simplex is not made after a proper series of attempts, it may happen that the last k elements of the simplex are associated with a zero-valued cost function (e.g. because they violate the constraint on amplifying signals): following algorithm 1, this would result in a recursive choice of the option in *line 25*, which would repeat the calculations of the last two simplices until the maximum number of iterations is reached. This problem has several possible solutions:

    1. *sol*.1 : check at every new simplex if the actual simplex is repeated or not and eliminate the problem by hand, restarting the calculation and substituting the failed points with new points;
    2. *sol*.2 : check if in the simplex cost function values one or more zeros are present and exclude them in the computation of the centroid $\bar{\boldsymbol{x}}$ (*line 8*). This operation guarantees that $\boldsymbol{x'}$ is then calculated nearer to the points that have non-zero cost function values.

In the present implementation, the constraint on ω is implemented using *way 1* (the *runner* detects the violation and returns the zero cost function value without running the simulation) and the control on multiple zeros is made using *sol*.2. Another variation is made on the stop-criterion: since the calculations are inviscid (and thus they do not detect the stall), the algorithm will be stopped when C_L values are above 1.8 (the static C_L curve for a NACA 64-A010 airfoil reaches about 1.1 for the given Mach) and the simplex is sufficiently confined to a value. The check on this criterion is made manually. The reason for this correction is that it is not worth proceeding further with calculations that probably predict high outputs because of stall-violation, but at the same time clear trends in the solution are desired.

3.2.3 Results

The results of the optimization process for both cost function are reported in fig. 3.12. The charts should be read starting from the bottom figure, where the relative cost (i.e. cost divided by the initial value of the cost, which is the reference case presented in section 2) is presented as a function of the iteration k. The cost monotonically decreases, which means that the algorithm shows no obvious error in its implementation. The colors help reconstruct the path followed by the algorithm in the parameter space. The optimal values are $AoA = 5.1°$ $x_\alpha = 0.066$ $\omega_h = 101.2 rad/s$ $\omega_\alpha = 122.1 rad/s$ for the case with cost function proportional to the derivative of plunge and $AoA = -0.015°$ $x_\alpha = 0.093$ $\omega_h = 146.3 rad/s$ $\omega_\alpha = 173.7 rad/s$ for the other case. Some conclusions can be drawn:

- large improvements can be expected in both cases with respect to the initial configuration;

- the free parameters have a nonlinear relationship: for example, when the cost function is the plunge derivative, non-zero AoA values initially result in non-optimal solutions or stable responses, while they have a beneficial effect when ω_α is increased. An explanation for this beahviour may be that a non-zero AoA changes the mean position of the airfoil, thus changing the mean amplitude of the oscillations in pitch and plunge. As a result, the springs are kept in tension, generating a stiffening effect that is superimposed to the real stiffness of the springs;

- for both definitions of the cost function, the optimum is reached when the elastic axis is slightly upstream of the center of gravity of the airfoil. The distance between the center of gravity and the elastic axis is related to the sweep angle of a wing [13]. In the author's opinion, it is not clear how Isogai [13] defined the equivalence between the wings he considered and the reduced order model he used. Qualitatively, a small distance between the elastic axis and the center of gravity is typical of low-sweep wings. Provided that it is appropriate to reduce a wing problem to a two-dimensional reduced-order model, the fact that the optimal wing would have a low sweep seems to be in contrast with the fact that back-swept wings are more unstable in the transonic regime. This apparent contrast can be solved considering that

  1. back-sweep in wings is used for stability and aerodynamic purposes: in particular it is used to increase the critical Mach of the wing (i.e. delay the presence of shock-waves as speed is increased). A wing with low sweep develops shock waves more easily than a high-sweep wing, thus one can think that a spanwise element of the wing (i.e. an airfoil) should run into compressibility effects more easily if the sweep is low. As described in section 1, compressibility plays a role in the development of flutter;
  2. in swept wings the component of the air velocity acting on the airfoil is the projection of the air velocity in the direction orthogonal to the wing spanwise principal axis of inertia. Thus the velocity seen by the airfoil decreases as sweep increases (fig. 3.5);
  3. the objective function does not measure the ability of the wing to enter an unstable regime, but its ability to exchange energy with the air flow once the instability has already begun. Thus, if a wing is less stable and experiences flutter more easily than another wing, this does not automatically implies that the second is less able of exchanging energy with the air flow once it has started to show an unstable behaviour.

- the pitch and plunge natural frequencies are related to the spring stiffnesses in the pitch and plunge direction. These features mimic the torsional and bending stiffnesses of a wing, which are related to the wing box panels stiffness and to the spars stiffness, respectively. Thus, with respect to the initial configuration, the optimal configuration is torsionally more stiff.

Some examples of aeroelastic responses are presented in figs. 3.7, 3.8 and 3.9 . From

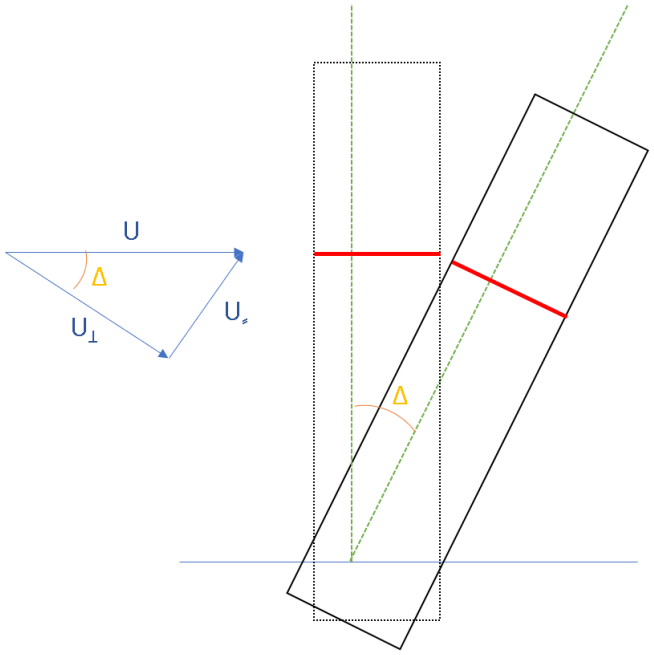


Fig. 3.5: Velocity decomposition in a wing swept of an angle Δ with respect to an unswept wing. The green lines represent the spanwise principal axes of inertia, while the red lines represent a spanwise section of the wing (i.e. an airfoil).

these time-traces, one can make a classification of the type of responses observed in this optimization procedure:

- damped response: after the initial perturbation, the system stabilizes around a value for plunge and pitch and after some time no oscillations can be observed;
- neutral response: the initial perturbation amplitude is preserved even after a long time;
- amplifying response: after the initial perturbation, the system initially diverges. After some time, this solution may originate
  1. limit-cycle oscillations (LCOs) at one frequency;
  2. limit-cycle oscillations at one frequency with small modulation;
  3. bump-like oscillations (the oscillations amplify, reach a maximum of amplification and then decrease and finally behave as LCOs)
- amplifying/damped responses: the oscillations initially amplify, reach a maximum of amplification and then decrease, without producing a LCO. This case is the rarest, but also the most demanding in terms of computational cost, since to distinguish this case from case 3 of the previous point, it is necessary to run for a long time.

In fig. 3.13 a collection of all the points used for the present study is presented, distinguishing between unstable and neutral/stable responses. This classification helps to reconstruct the flutter boundary in the explored parameter space.

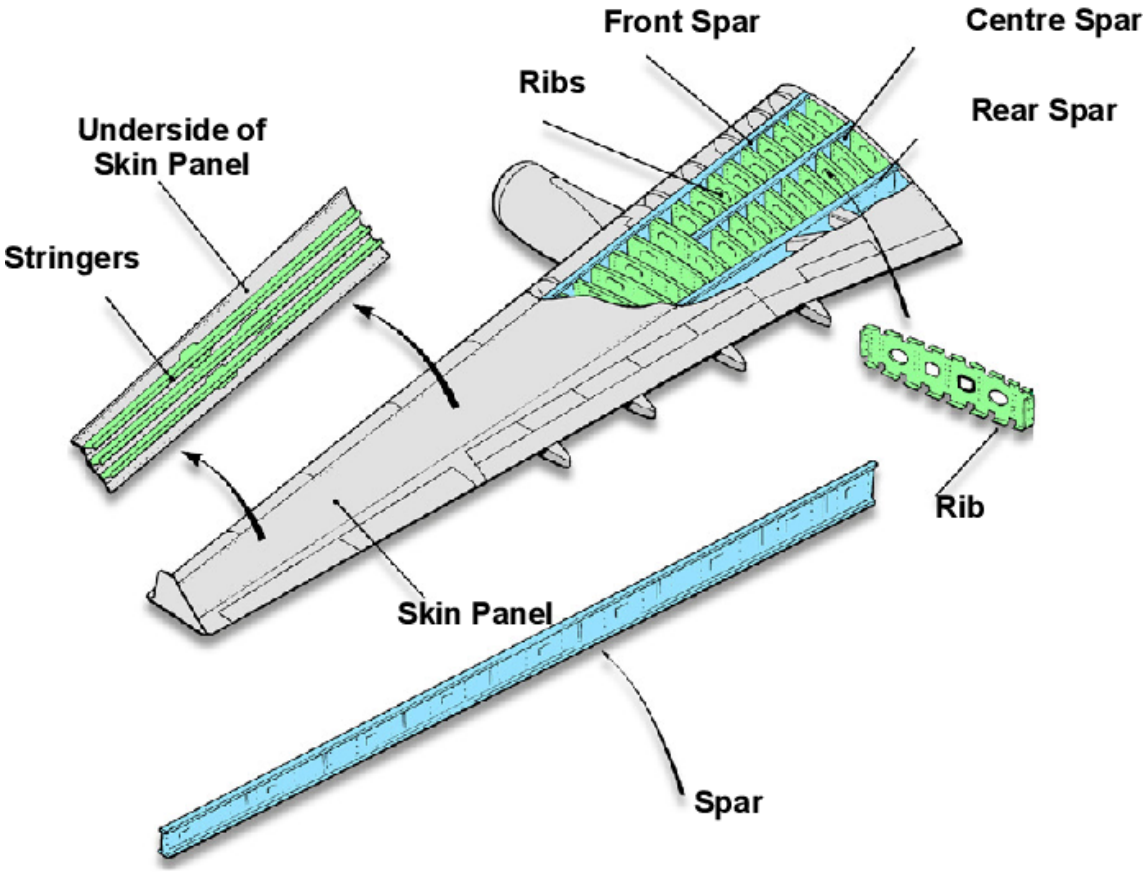
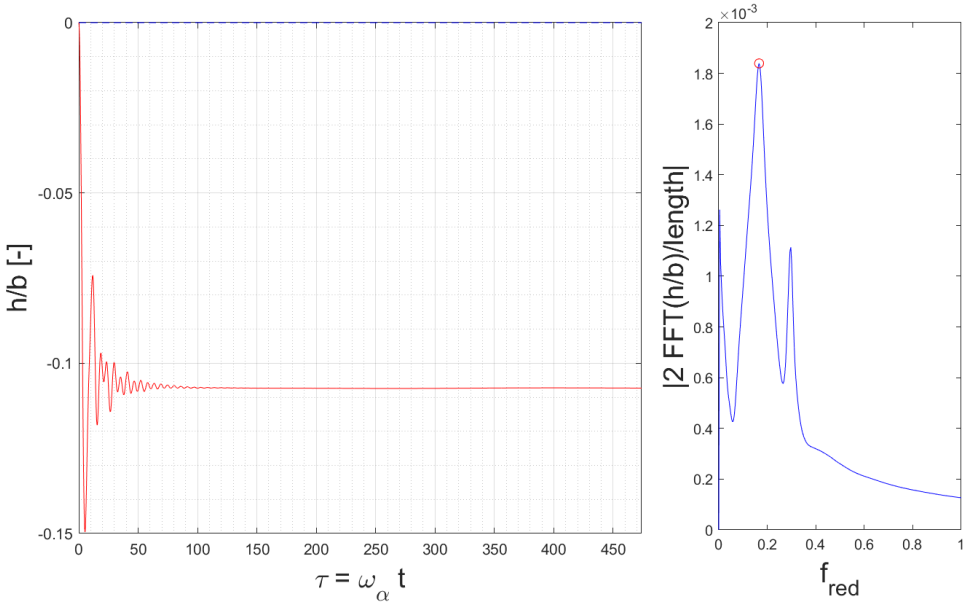


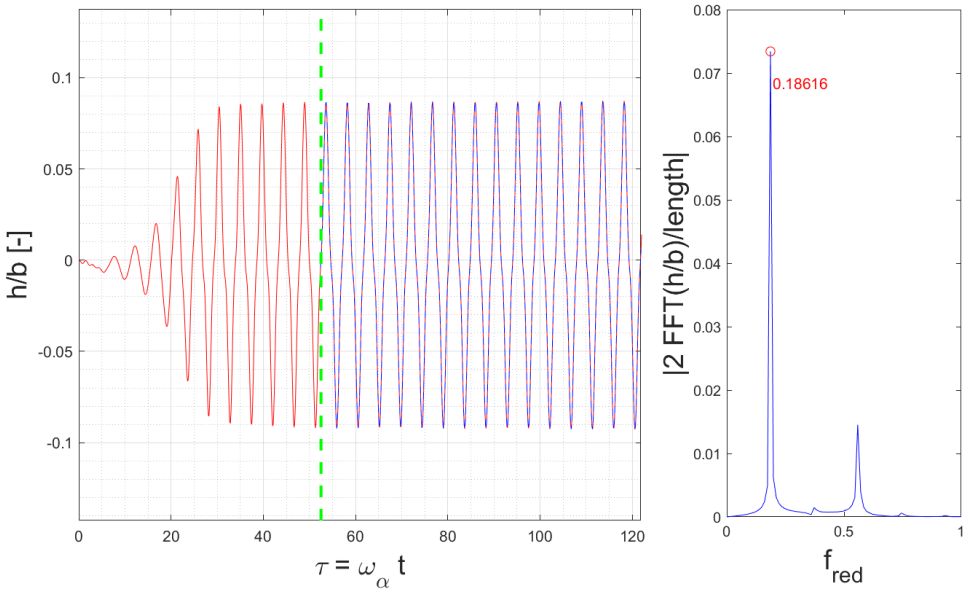
Fig. 3.6: Wing structure of a civil aircraft [56].

3.2.4 Robust design study

The aeroelastic pitch-plunge airfoil model parameters AoA, x_α, ω_h and ω_α were optimized to obtain the maximum of the cost functions described in section 3.2. In this process the free-stream Mach number and pressure are fixed. The aeroelastic solution depends in general also from these parameters, so it is natural to describe the stability of the optimal solution with respect to a "small perturbation" of these parameters, that were not included in the optimization process. In this case a variation of $\pm 2\%$ from the optimal solution will be used. A 2% perturbation on the Mach is, in fact, enough to move the mean location of the shock-waves, which are responsible for flutter in transonic regime. From the results are presented in fig. 3.14, it is clear that the optimum found in the derivative-cost case is more stable than the one found in the power cost case. The former seems also to be quite insensitive to pressure variations, in contrast to the latter. The trends seems also to be different for the two cases: the former shows an increase in the cost function as Mach is increased, while the latter does the opposite.

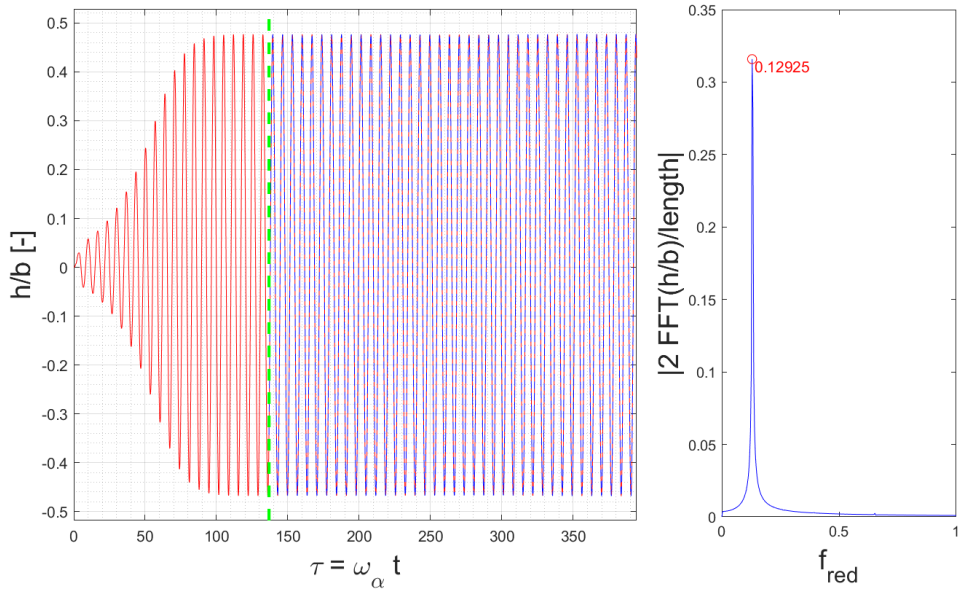


(a)

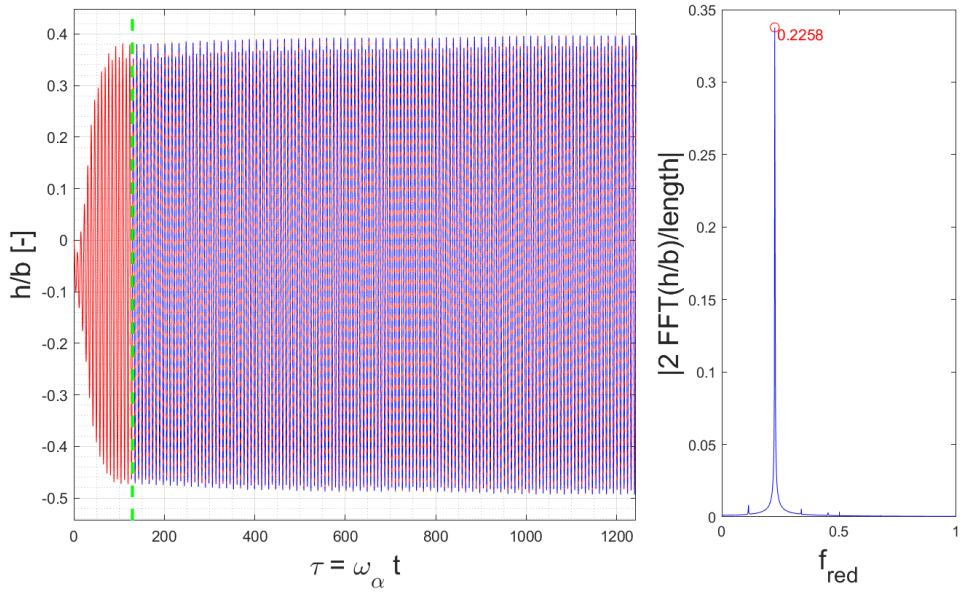


(b)

Fig. 3.7: 3.7a: example of damped response ($AoA = 6.029°$, $x_\alpha = -0.133$, $\omega_h = 115.673 rad/s$, $\omega_\alpha = 135.845 rad/s$). *3.7b: example of neutral response* ($AoA = 1°$, $x_\alpha = 0.9$, $\omega_h = 150 rad/s$, $\omega_\alpha = 70 rad/s$).

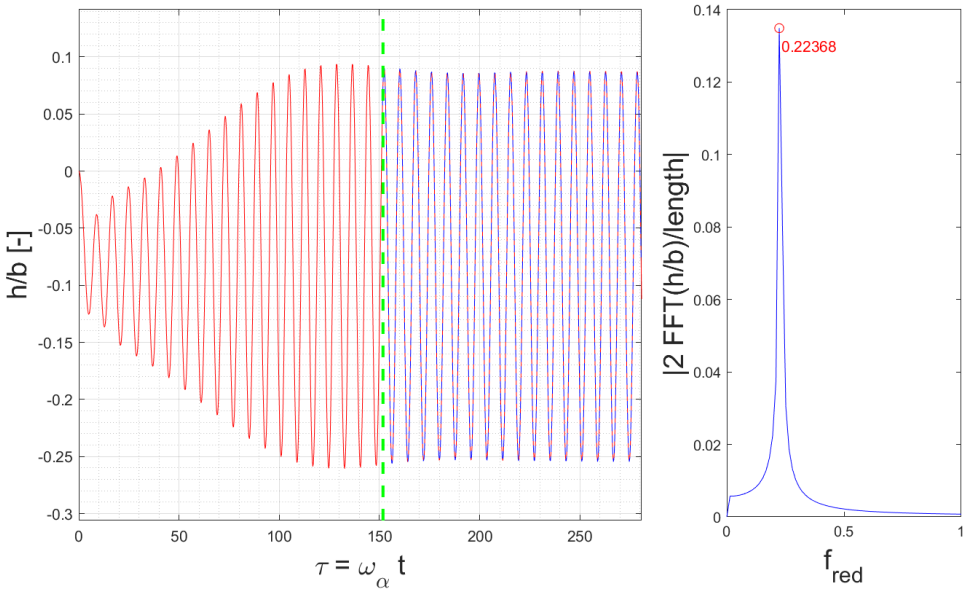


(a)

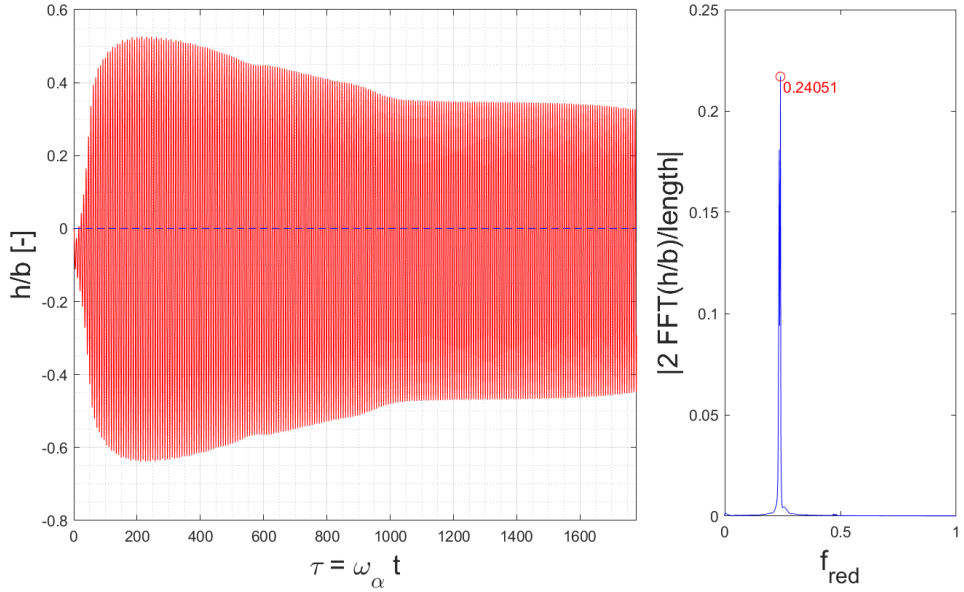


(b)

Fig. 3.8: 3.8a: example of LCO response ($AoA = -0.207°$, $x_\alpha = 1.626$, $\omega_h = 61.533 rad/s$, $\omega_\alpha = 72.927 rad/s$). 3.8b: example of LCO response with small modulation ($AoA = 5.048°$, $x_\alpha = 0.169$, $\omega_h = 104.131 rad/s$, $\omega_\alpha = 142.952 rad/s$).



(a)



(b)

Fig. 3.9: 3.9a: example of "bump" amplifying response ($AoA = 5.265°$, $x_\alpha = 0.056$, $\omega_h = 102.952 rad/s$, $\omega_\alpha = 146.412 rad/s$). 3.9b: example of amplifying/damped response ($AoA = 5.437°$, $x_\alpha = 0.076$, $\omega_h = 109.5 rad/s$, $\omega_\alpha = 145.606 rad/s$).

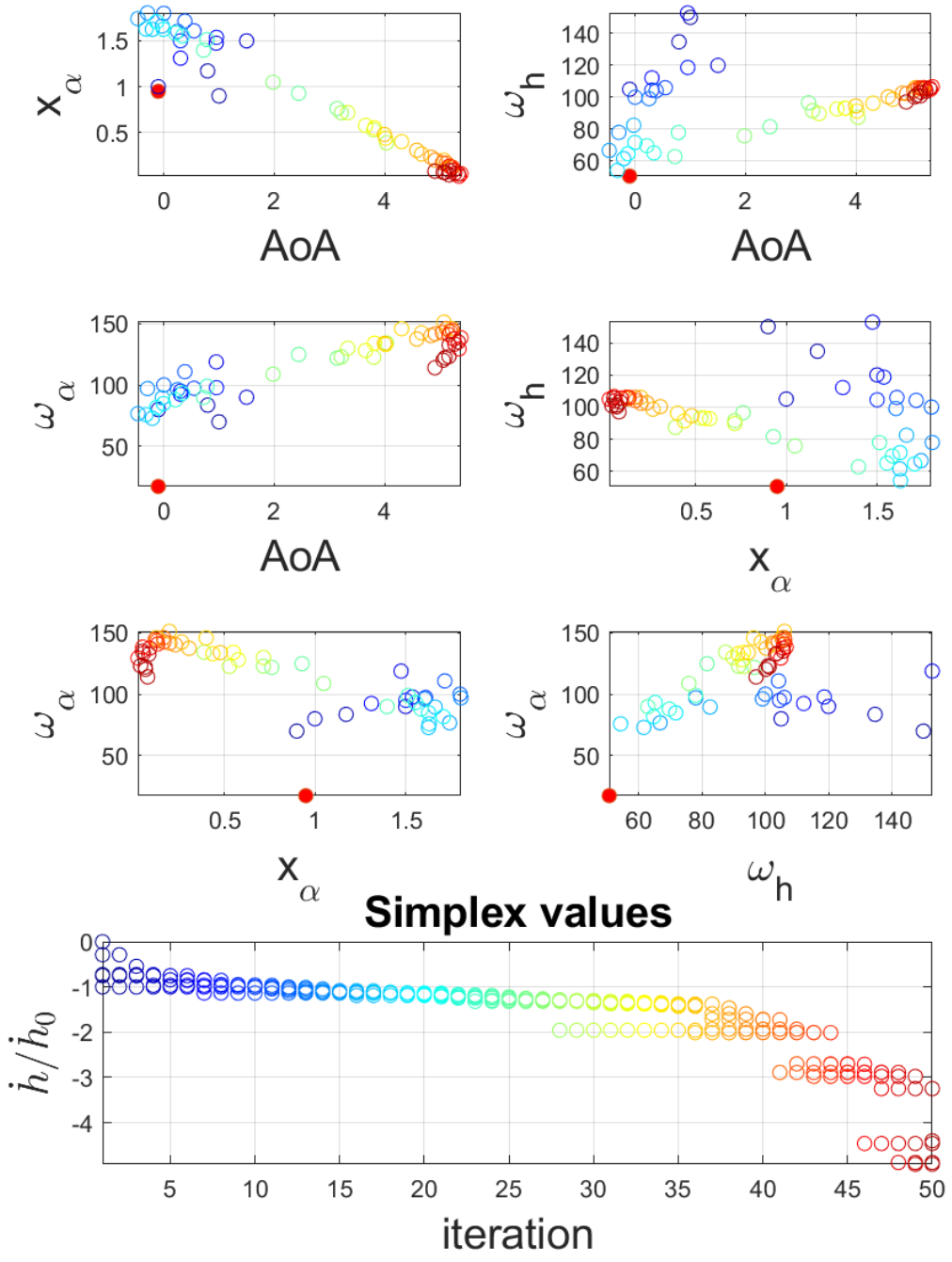


Fig. 3.10: History of the optimization process for the derivative-cost function case. The bottom frame shows the cost as a function of the iteration number. The colors help to reconstruct the position of the simplex in the parameter space, represented by the upper figures. Full red dots represent points that violate a constraint.

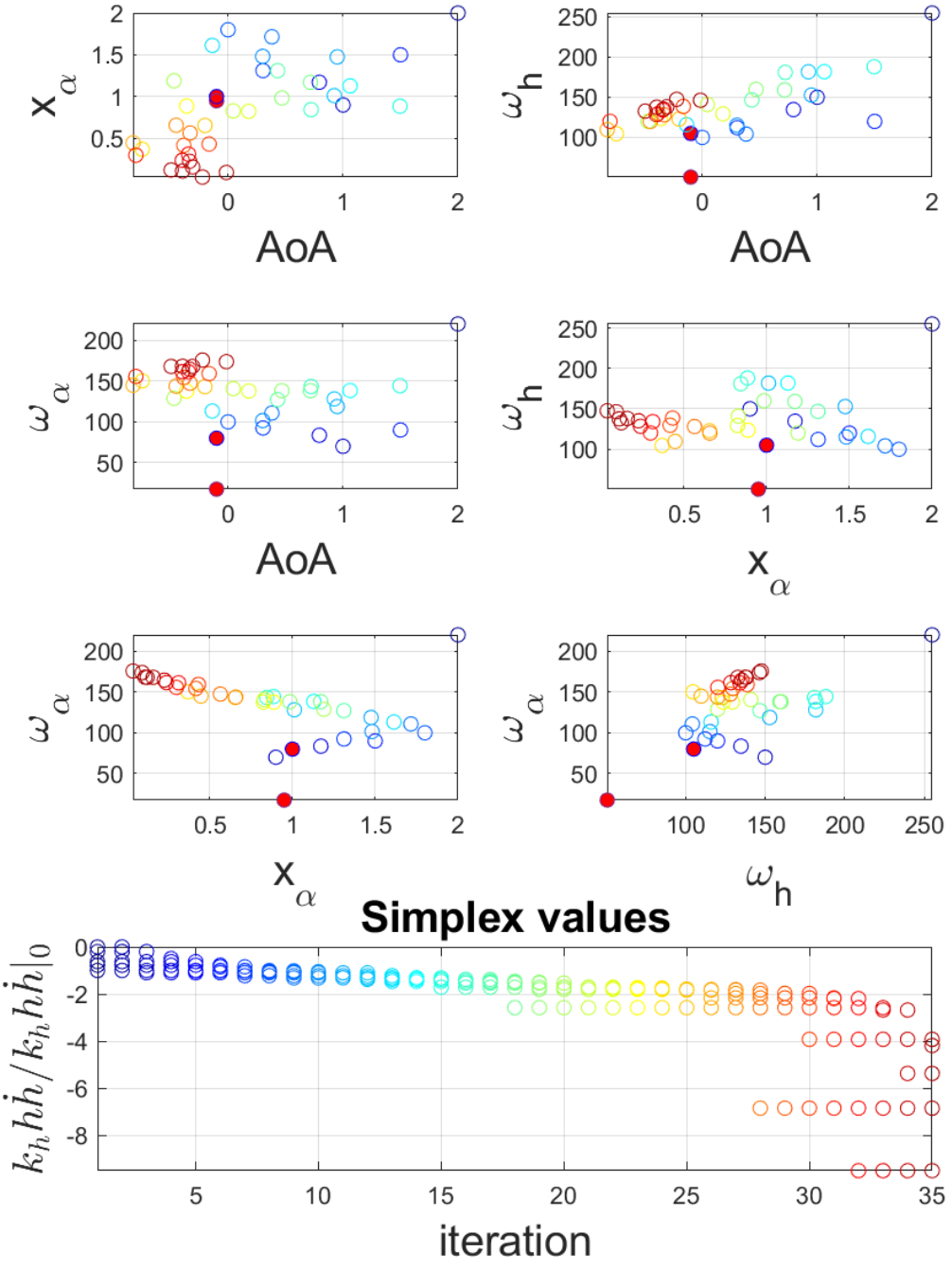
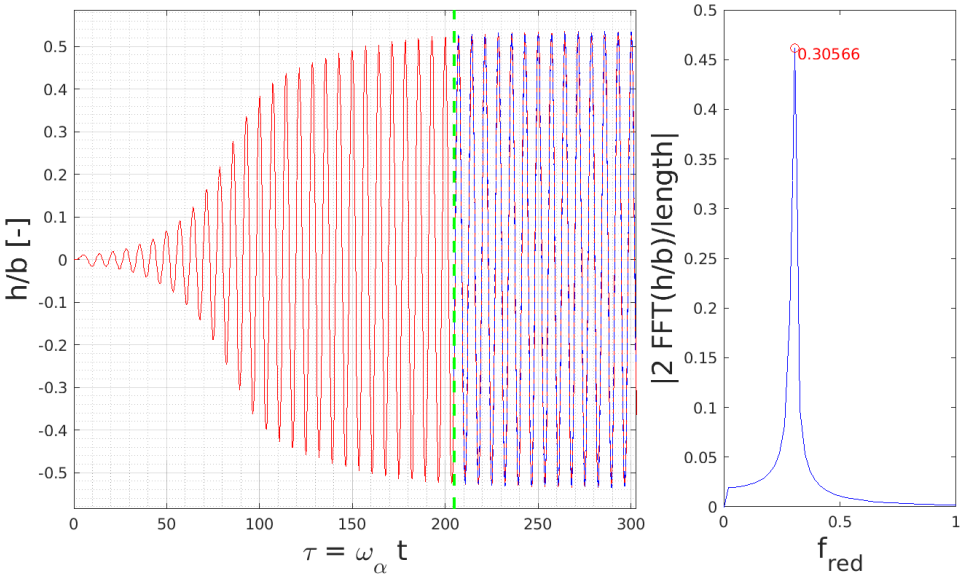
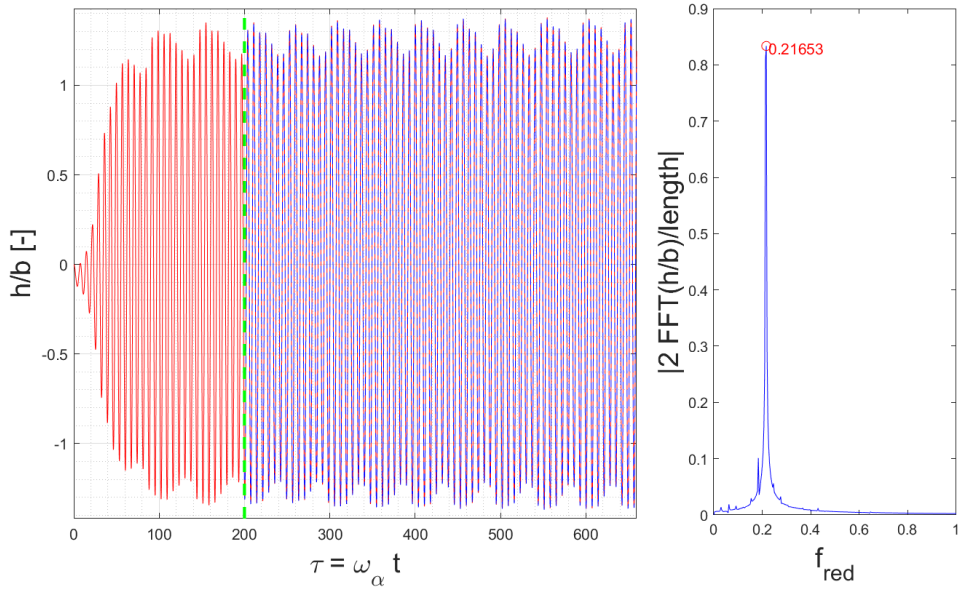


Fig. 3.11: History of the optimization process for the power-cost function case. The bottom frame shows the cost as a function of the iteration number. The colors help to reconstruct the position of the simplex in the parameter hyperspace, represented by the upper figures. Full red dots represent points that violate a constraint.



(a)



(b)

Fig. 3.12: Time histories and spectra of the optimal responses for the power-cost (3.12a) and derivative-cost case (3.12b). Spectra are computed using the statistically-stationary part of the signal (blue line).

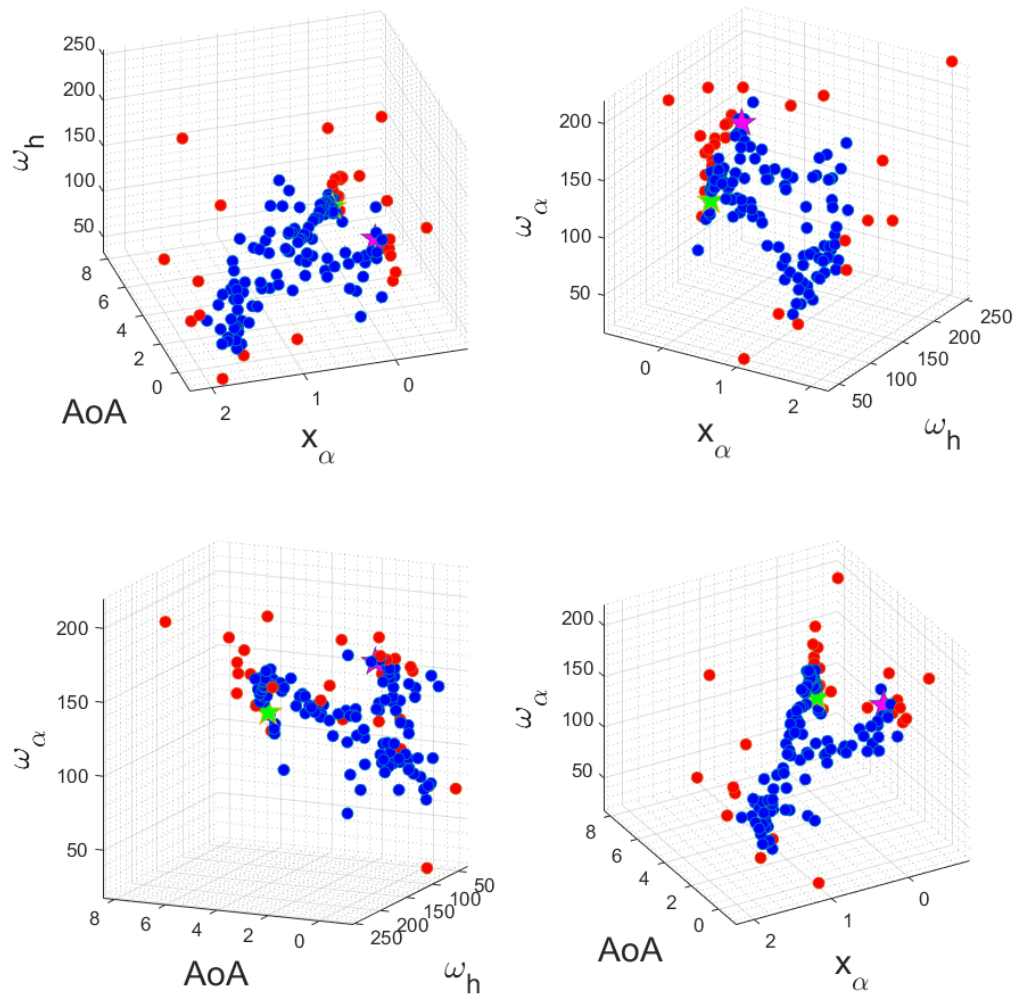


Fig. 3.13: classification of the response type (blue=unstable, red=neutral/stable) in the parameter space. Stars represent the optimal solutions: green for derivative-cost and magenta for power-cost case.

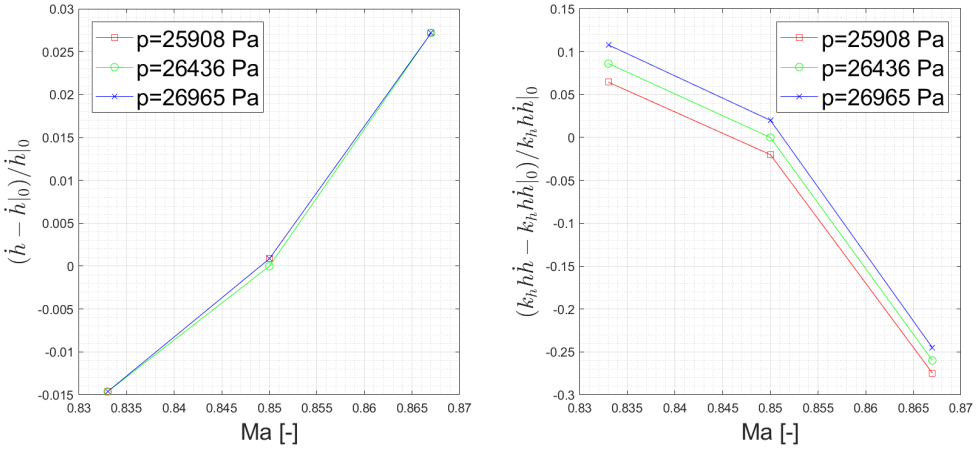


Fig. 3.14: Percentage variations of the cost function with respect to the optimal configuration for small variations of the free-stream pressure and Mach number. Left: derivative-cost case. Right: power-cost case.

4 Conclusions and future development

In this thesis, a pitch-plunge airfoil model has been examined and modified to optimize the maximum useful output (either the electromotive force that can be generated using the plunging motion or the power exchanged at the plunging spring). The mechanical parameters that were modified in this process are the angle of attack, the distance between the elastic axis (EA) and the center of gravity (CG), and the pitch and plunge natural frequencies. The optimization was run using inviscid cases, which have a computational cost lower than turbulent cases. At the end of the process, for two cost functions, the optimal values of the parameters show that the distance between the EA and the CG must be small but positive (i.e. the EA must stay slightly upstream of the CG). This configuration is typical of wings with very low sweep. Optimal solutions show also higher natural frequencies in the pitch direction than in the plunge direction, suggesting that, if energy is extracted only from the plunging motion, wings that are torsionally (rather than bending) more stiff are optimal for energy harvesting purpose. As far as the useful output is concerned, it has been shown that large improvements of the energy-harvesting characteristics of this model are possible and that large improvements can be obtained with relatively small changes of the parameters values, near the optimum. To confirm these, stall dynamics has to be taken into account, thus high-resolution turbulent calculations (URANS or even LES) will be needed in future study. Then, having defined the cost function as the power exchanged in the plunging spring, no significant assumption is made on the way with which energy is converted from mechanical to electric. This makes the developed model general: instead of evaluating the ability of the system to output energy through a particular process, the ability of the system to exchange energy to and from the air flow is evaluated, disregarding the mean of conversion. Specializing for one mean of conversion, as done with the derivative-cost function case (which considers electromagnetic coupling) could lead to the addition of en extra damping term in the mechanical equation of the system. The optimal solution found in this case could be a good starting point in the parameter space for future and more advanced calculations that take this damping term into account. It has been shown also that the the parameters have a strongly nonlinear trend. For example, the AoA has a stiffening effect that superimposes with the real stiffness. As previously described, this could provide either a detrimental or a favourable effect on the cost function value and this effect depends on the stiffness values themselves (fig. 3.10). From a classification study, where the aeroelastic responses are divided into unstable (useful) and stable/neutral (not useful), one can see that the optima are very close to the flutter boundary in the parameter space, suggesting that the quasi-optimal values of the parameters should be chosen in the unstable zone (in fig. 3.13 marked in blue) of the space, taking a margin from the flutter boundary. The final goal, that of introducing flutter energy harvesting devices in aeronautics, is still a theoretical exercise after this work. The final power output of this kind of devices greatly depends on the technology used to extract power from the oscillations. This is the reason why this study was not tailored on a single type of extracting technology in particular. For the same reason, the question "how much power can

a flutter energy harvester device make available?" still needs to be answered, as it would strongly depend on the power-conversion technology. However, this thesis has identified a good starting point from which more advanced and technological studies can move towards the search for an optimal configuration. Then, it has also made a contribution towards the understanding of aeroelastic responses in the transonic regime, where the high non-linearities produce, for small variations of the external parameters, large variations of the type or response.

References

[1] A. R. Collar. *The first fifty years of aeroelasticity.* Aerospace, 2, vol. 5, 1978.

[2] Decreto ministeriale 14 gennaio 2008 «nuove norme tecniche per le costruzioni». `https://people.dicea.unifi.it/gianni.bartoli/normative/DM_14.01.08_NTC.pdf`, 2008. Accessed: February 2021.

[3] Dlr, aeroelasticity: whirl flutter. `https://www.dlr.de/ae/en/desktopdefault.aspx/tabid-9667/16675_read-40599/`, 2020. Accessed: February 2021.

[4] J. Quill. *Spitfire - A Test Pilot's Story.* Crecy Publishing, 1998.

[5] Iwm collections. `https://www.iwm.org.uk/collections`, 2021. Accessed: February 2021.

[6] H. Halfman R.L. Bisplinghoff, R.L. Ashley. *Aeroelasticity.* Dover Publications, 1977.

[7] A. Abdelkefi. *Aeroelastic energy harvesting: A review.* International Journal of Engineering Science 100, 2015.

[8] P.M. Daynes, S. Weaver. *Review of shape-morphing automobile structures: concepts and outlook.* Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 2013.

[9] 2021 formula 1 technical regualtions. `https://www.fia.com/sites/default/files/2021_formula_1_technical_regulations_-_2019-10-31.pdf`, 2021. Accessed: February 2021.

[10] Aeroelasticity and experimental aerodynamics course, prof. g. dimitriadis, université de liège. `http://www.ltas-aea.ulg.ac.be/cms/index.php?page=aeroelasticity-course`, 2021. Accessed: February 2021.

[11] H. Wagner. *Über die Entstehung des dynamischen Auftriebes von Tragflügeln.* Z. Angew. Math. Mech. 5, 17–35., 1925.

[12] P.W. Farmer, M.G. Hanson. *Comparison of supercritical and conventional wing flutter characteristics.* 17th Structures, Structural Dynamics, and Materials Conference, 1976.

[13] K. Isogai. *On the transonic-dip mechanism of flutter of a swept back wing.* AIAA Journal, Vol. 17, No. 7, 1979.

[14] S.R. Maler, H.G. King. *Transonic Flutter Model Tests, Part I. 45 Degree Swept Wings.* WADC Technical Report 56-214, Part I. Wright Air Development Center, Wright-Patterson Air Force Base, Ohio, 1957.

[15] W. J. Mykytow. *A brief overview of transonic flutter problems.* AGARD Conference Proceeding n. 226, 1977.

[16] K. Isogai. *Transonic Dip Mechanism of Flutter of a Sweptback Wing: Part II.* AIAA Journal 81-4229, 1981.

[17] I. Theodorsen, T. Garrick. *Mechanism of flutter: a theoretical and experimental investigation of the flutter problem.* NACA Report 685, 1940.

[18] D.J. Anton, S.R. Inman. *Vibration energy harvesting for unmanned aerial vehicles.* Proceedings of SPIE - The International Society for Optical Engineering 6982, 2008.

[19] Vortex bladeless. `https://vortexbladeless.com/`, 2016. Accessed: February 2021.

[20] New concept of affordable wind energy generators without blades. `https://cordis.europa.eu/project/id/726776/it`, 2016. Accessed: February 2021.

[21] G. Mazzino A. Boragno C. Olivieri, S. Boccalero. *FLuttering Energy Harvester for Autonomous Powering (FLEHAP): aeroelastic characterisation and preliminary performance evaluation.* Procedia Engineering, Volume 199, 2017.

[22] E. Bryant, M. Garcia. *Modeling and Testing of a Novel Aeroelastic Flutter Energy Harvester.* J. Vib. Acoust. 133(1), 2011.

[23] M.E. Roccia B.A. Valdez M.F. Verstraete M.L. Preidikman S. Beltramo, E. Pérez Segura. *Constructive Aerodynamic Interference in a Network of Weakly Coupled Flutter-Based Energy Harvesters.* Aerospace, 2020.

[24] K. Soon-Dock K. Kincho H. Park, J.K. Kyoung-Min. *An aero-elastic flutter based electromagnetic energy harvester with wind speed augmenting funnel.* AIAA Journal 81-4229, 2012.

[25] S. Gasnier P. Willemin J. Reboud J.L. Perez, M. Boisseau. *An electret-based aeroelastic flutter energy harvester.* Smart Mater. Struct. 24, 035004, 2015.

[26] S. R. Copeland T. W. Lukaczyk Economon, F. Palacios and J. J. Alonso. *SU2: An Open-Source Suite for Multiphysics Simulation and Design.* AIAA JOURNAL Vol. 54, No. 3, 2015.

[27] A. Alonso, J.J. Jameson. *Fully Implicit Time-Marching Aeroelastic Solutions.* AIAA Journal 94-0056, 1994.

[28] Governing equations in su2. `https://su2code.github.io/docs_v7/Theory/#compressible-euler`, 2019. Accessed: December 2020.

[29] F. Marti, F. Liu. *Flutter Study of NACA 64A010 Airfoil Using URANS and e^N Transition Models Coupled with an Integral Boundary Layer Code.* 55th AIAA Aerospace Science Meeting, 2017.

[30] Su2 testcases. `https://github.com/su2code/TestCases/tree/master/aeroelastic`, 2020. Accessed: December 2020.

[31] J.P. Dowell E.H. Hall, K.C. Thomas. *Proper Orthogonal Decomposition Technique for Transonic Unsteady Aerodynamic Flows.* AIAA Journal vol. 38, 2000.

[32] M.R. Bennett. *Test cases for flutter of the benchmark models rectangular wings on the pitch and plunge apparatus.* Defense Technical Information Center, 2000.

[33] X. Banerjee J.R. Kassem, H.I. Liu. *Transonic flutter analysis using a fully coupled density based solver for inviscid flow.* Advances in Engineering Software 95, 2016.

[34] K. Li, H. Ekici. *A novel approach for flutter prediction of pitch–plunge airfoils using an efficient one-shot method.* Journal of fluid and structures 82, 2018.

[35] matlab file exchange: Grabit. `https://it.mathworks.com/matlabcentral/fileexchange/7173-grabit`, 2018. Accessed: December 2020.

[36] griddata. `https://it.mathworks.com/help/matlab/ref/griddata.html`, 2021. Accessed: February 2021.

[37] P.P. McNamara, J.J. Friedmann. *Flutter-Boundary Identification for Time-Domain Computational Aeroelasticity.* AIAA Journal vol. 45 n. 7, 2007.

[38] K. Timme, S. Badcock. *Oscillatory behavior of transonic aeroelastic instability boundaries.* AIAA J. 47 (6), 2009.

[39] O. Kaynak U. Cakmakcioglu, S.C. Bas. *A correlation-based algebraic transition model.* Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 2017.

[40] S.K. Kachra, F. Nadarajah. *Viscous Aeroelastic Solutions Using the Non-Linear Frequency Domain Method.* AIAA, 24th Applied Aerodynamics Conference 5 - 8 June 2006, San Francisco, California, 2006.

[41] G.-C. Hu Z. Yang M.-T. Chen, X. Zha. *Flutter Prediction Based on Fully Coupled Fluid-Structural Interactions.* 9th National Turbine Engine High Cycle Fatigue Conference, 2004.

[42] Bohbot J. and D. Darracq. *Time Domain Analysis of Two D.O.F. Airfoil Flutter Using an Euler/Turbulent Navier-Stokes Implicit Solver.* International Forum on Aeroelasticity and Structural Dynamics, Madrid, Spain, June 5-7, 2001.

[43] T. Pető, S. Bilicz, L. Szűcs, S. Gyimóthy, and J. Pávó. The radar cross section of small propellers on unmanned aerial vehicles. In *2016 10th European Conference on Antennas and Propagation (EuCAP)*, pages 1–4, 2016.

[44] Next generation jammer mid-band. `https://www.raytheonintelligenceandspace.com/capabilities/products/ngj`, 2021. Accessed: January 2021.

[45] G. Norris. *Boeing's seventh wonder.* IEEE Spectrum, 1995.

[46] Naval air systems command: Alq-99 tactical jamming system. https://www.navair.navy.mil/product/ALQ-99-Tactical-Jamming-System, -. Accessed: February 2021.

[47] D.C. Carel. *The history of the aerial gatling gun.* Air Command and Staf College, Air University Maxwell AFB, 1987.

[48] A. Leitch. *V-Force Arsenal: Weapons for the Valiant, Victor and Vulcan.* Air Enthusiast No. 107, 2003.

[49] Ram air turbine. https://it.wikipedia.org/wiki/Ram_air_turbine, 2019. Accessed: January 2021.

[50] S.B. Zulkafli M.F. Saad, M.M.M. Mohd. *A survey on the use of ram air turbine in aircraft.* AIP Conference Proceedings 1831, 020048, 2017.

[51] W. Späte. *Der streng geheime Vogel Me 163.* Dörfler Verlag GmbH, 2003.

[52] K. Schubert H. von Gersdorff, K. Grasmann. *Flugmotoren und Strahltriebwerke.* Bernard und Graefe Verlag, 1981.

[53] R. Nelder, J. A. Mead. *A Simplex Method for Function Minimization.* The Computer Journal, Volume 7, Issue 4, January 1965.

[54] Basics on continuous optimization: Downhill simplex. http://www.brnt.eu/phd/node10.html#SECTION00622200000000000000, 2011. Accessed: January 2021.

[55] Mathworks help: fminsearch. https://it.mathworks.com/help/matlab/ref/fminsearch.html, 2006. Accessed: January 2021.

[56] L. Krog, A. Tucker, Airbus Uk, and R. Boyd. Topology optimisation of aircraft wing box ribs. 2004.

[57] Multigrid methods in cfd-online forum. https://www.cfd-online.com/Wiki/Multigrid_methods, 2019. Accessed: December 2020.

[58] The multigrid method. https://www.wias-berlin.de/people/john/LEHRE/MULTIGRID/multigrid_6.pdf.

[59] Gradient computation in cfd-online forum. https://www.cfd-online.com/Wiki/Gradient_computation, 2019. Accessed: December 2020.

[60] E.F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics.* Springer-Verlag Berlin Heidelberg GmbH, 1997.

[61] M.H. Saad, Y. Schultz. *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems.* SIAM J. Sci. Stat. Comput., 7:856–869, 1986.

[62] R.P. Selim, M.M. Koomullil. *Mesh Deformation Approaches – A Survey.* Journal of Physical Mathematics 7(2), 2016.

[63] Langley research center turbulence modeling resource. https://turbmodels.larc.nasa.gov/, 2020. Accessed: December 2020.

[64] Ansys fluent theory guide: Sst k-ω model. https://www.afs.enea.it/project/neptunius/docs/fluent/html/th/node67.htm, 2009. Accessed: December 2020.

[65] Su2 configuration template on github. https://github.com/su2code/SU2/blob/master/config_template.cfg, 2020. Accessed: December 2020.

A Appendix

A.1 Multi-Grid methods

Standard solvers (Gauss-Seidel, Jacobi...) experience a slower convergence rate as meshes are increasingly refined, that may result in a convergence stall. It is possible to demonstrate that the convergence rate is function of the error gradient from node to node: the higher the spatial frequency of the error, the faster the convergence. Since in many iterative methods to reach a proper convergence the information has to travel back and forth several times but -at the same time- it can travel only one cell per iteration, it is possible to define a grid that is coarser than the actual grid, in such a way that low frequency errors are seen as high frequency ones on the coarse grid and thus the convergence rate of linear solvers is improved.
In this context, *restriction* is the interpolation method used to inject the residual from the fine to the coarse grid, while *prolongation* is the opposite. Typically, a multi-grid cycle starts at the finest level, the solution is restricted to the coarser one, some relaxation cycles are performed and then the solution is restricted to the next coarser level until the coarsest level reaches convergence. At that point, the solution is prolongated back to the finest level until convergence is reached [57].
The hierarchy of the grids defines the name of the multigrid algorithm: if the restriction steps are all subsequent and then only prolongation steps are performed, that procedure takes the name of "V-cycle"; changing the sequence of restriction and prolongation steps results in W and F multigrid cycles [58]. In the full-multigrid cycle, the initial guess for the first pre-smoothing step on the finest grid can be obtained by a nested iteration. In the nested iteration, the system is first solved (or smoothed) on a very coarse gird, then smoothed on the next finer grid and so on, until the finest grid is reached.

A.2 Gradient computation

For grids of general shape, the usual approach to compute the gradient of a given scalar function ϕ is by mean of the Green-Gauss Theorem [59]:

$$\int_V \nabla\phi dV = \int_{\partial S} \phi\hat{n} dS$$

where V is a volume of boundary S and $\hat{n}$ is its external normal. Assuming that $\nabla\phi$ is constant in V,

$$\int_V \nabla\phi dV = \nabla\phi_P V \quad \text{that can be approximated using} \quad \nabla\phi_P \approx \frac{1}{V}\sum_{faces} \phi_f \vec{S_f}$$

where the average face value ϕ_f can be computed using

- *cell based* methods: the face value is computed using the values at its straddling cells using a weighted interpolation. Very simple to implement but also very sensitive to grid skewness.

- *node based* methods: the face value is computed using the values at its straddling nodes. The value on the faces is computed as the mean of the nodes defining the face and the properties at the nodes are calculated using a weighted average within the cells adjacent to that node.

A.3 Roe solver

The Roe scheme is a first order scheme originally studied for solving Euler equations in presence of shock waves. Starting from the general Initial Boundary Value Problem (IBVP)

$$\begin{cases} \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}(\boldsymbol{U})}{\partial x} = 0 \\ \boldsymbol{U}(x,0) = \boldsymbol{U}^{(0)}(x) \\ \boldsymbol{U}(0,t) = \boldsymbol{U_{left}(t)}; \quad \boldsymbol{U}(L,t) = \boldsymbol{U}_{right}(t) \end{cases}$$

The Roe method resolves the Riemann problem approximately by introducing a Jacobian matrix

$$\boldsymbol{A(U)} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}$$

and using the chain rule, the conservation law becomes

$$\boldsymbol{U}_t + \boldsymbol{F}(\boldsymbol{U})_x = \boldsymbol{0} \quad \Rightarrow \quad \boldsymbol{U}_t + \boldsymbol{A(U)}(\boldsymbol{U})_x = \boldsymbol{0}$$

where the subscripts denote the derivatives in time or space. The Roe method approximates $\boldsymbol{A(U)}$ with a constant Jacobian matrix $\tilde{\boldsymbol{A}}$:

$$\begin{cases} \frac{\partial \boldsymbol{U}}{\partial t} + \tilde{\boldsymbol{A}} \frac{\partial \boldsymbol{U}}{\partial x} = \boldsymbol{0} \\ \boldsymbol{U}(x,0) = \begin{cases} \boldsymbol{U}_{left} & if \quad x < 0 \\ \boldsymbol{U}_{right} & if \quad x > 0 \end{cases} \end{cases}$$

This linear system with constant coefficients is then solved exactly. For a general hyperbolic system of conservation laws the Roe Jacobian matrix $\tilde{\boldsymbol{A}}$ has to satisfy some properties (hyperbolicity of the system, consistency with the exact Jacobian and conservation across discontinuities). The present description of the Roe scheme is taken from [60].

A.4 FMGRES

The *Flexible Generalized Minimal Residual* algorithm is a generalization of the GMRES algorithm [61], whose structure is briefly summarized as follows:

- let $Ax = b$ be a square linear system, let A be invertible and b normalized.
- GMRES approximates the exact solution of $Ax = b$ using a vector $x_n \in K_n$, where K_n is the n-th Krilov subspace for this problem: $K_n = K_n(A, b)$. x_n is the solution that minimizes $||Ax_n - b||$
- the Arnoldi iteration is used to avoid problems with quasi-linear dependent vectors and the problem reduces to linear least square problem.

Iterative solvers are more unstable than direct solvers and they do not always converge, for this reason, an appropriate preconditioner has to be selected. The $FMGRES$ method allows the possibility of using a different right preconditioner at each step in solving the preconditioned system, is numerically more stable than $GMRES$ with floating point data but requires more memory.

A.5 Mesh deformation: Inverse volume method

the Inverse Volume Method for mesh deformation is a Linear-Elasticity model where the mesh elements are treated as solid element, thus mesh deformation is accomplished by solving the linear elasticity equations for the mesh point displacements throughout the field. Since a linear solid model requires a modulus of elasticity of the material E and a Poisson's ratio ν, it is usual to take E as inversely proportional to the cell volume and ν as a constant. The linear elasticity equations are then FEM-discretized and solved using GMRES method. Among other methods/choices, the described one is very beneficial for avoiding invalid mesh cells near the boundaries. [62]

A.6 Spalart-Allmaras (SA) turbulence model

The Spalart-Allmaras [63] is a one-equation turbulence model originally developed for aerospace applications and thus suitable for external flows. The equation that is solved in this model is

$$\frac{D\tilde{\nu}}{Dt} = c_{b1}(1-f_{t2})\tilde{S}\tilde{\nu} - \left[c_{w1}f_w - \frac{c_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2 + \frac{1}{\sigma}\left[\frac{\partial}{\partial x_j}\left((\nu+\tilde{\nu})\frac{\partial \tilde{n}u}{\partial x_j}\right) + c_{b2}\frac{\partial\tilde{\nu}}{\partial x_i}\frac{\partial\tilde{\nu}}{\partial x_i}\right]$$

where "tilde" denoted the turbulence field variable. The turbulent eddy viscosity id computed from

$$\mu_t = \rho\tilde{\nu}f_{v1} \qquad\qquad f_{v1} = \frac{(\tilde{\nu}/\nu)^3}{(\tilde{\nu}/\nu)^3 + c_{v1}^3}$$

A complete definition of the model constants and auxiliary relations is present in [63]. The recommended boundary conditions are

$$\tilde{\nu_{wall}} = 0 \qquad\qquad 3\nu_\infty < \tilde{\nu_{far field}} < 5\nu_\infty$$

To avoid possible numerical problems the term $\tilde{S}$ is generally limited to positive values. When transition detection is important, the use of the "turbulence index" is recommended:

$$0 < i_t = \frac{1}{\kappa u_T}\frac{\partial\tilde{\nu}}{\partial n} < 1$$

where n is the wall-normal and $u_T \approx \sqrt{\nu\Omega}$. the lower bound corresponds to a laminar region and the upper bound to a fully turbulent one.
In SU2 the standard SA model and some variants are present (SA-E, SA-NEG and SA-COMP). The SA-NEG was developed to address the problems of the SA with under resolved grids and yields very similar results in the other cases; the SA-COMP has a correction for compressible mixing layers and the SA-Edwards was developed to improve convergence near walls. Combinations of the aforementioned models are also available in SU2.

A.7 Menter SST $k-\omega$ turbulence model

The Shear-Stress Transport (SST) $k-\omega$ model by Menter [63] is a 2-equation model widely used for internal and external flow problems. The governing equations are summed up as follows:

$$\frac{D\rho k}{Dt} = \underbrace{\tau_{ij}\frac{\partial u_i}{\partial x_j}}_{\text{Production}} - \underbrace{\beta^*\rho\omega k}_{\text{Dissipation}} + \underbrace{S_k}_{\text{Source}} + \underbrace{\frac{\partial}{\partial x_j}\left[(\mu+\sigma_k\mu_t)\frac{\partial k}{\partial x_j}\right]}_{\text{Diffusion}}$$

$$\frac{D\rho\omega}{Dt} = \underbrace{\frac{\gamma}{\nu_t}\tau_{ij}\frac{\partial u_i}{\partial x_j}}_{\text{Production}} - \underbrace{\beta\rho\omega^2}_{\text{Dissipation}} + \underbrace{S_\omega}_{\text{Source}} + \underbrace{\frac{\partial}{\partial x_j}\left[(\mu+\sigma_\omega\mu_t)\frac{\partial \omega}{\partial x_j}\right]}_{\text{Diffusion}} + \underbrace{2\rho(1-F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}}_{\text{Cross-Diffusion}}$$

are the transport equations for k and ω, where D/Dt is the material derivative, S are user-defined source terms [64], and

$$\tau_{ij} = \mu_t\left(2S_{ij} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) - \frac{2}{3}\rho k\delta_{ij} \qquad S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$$

The turbulent viscosity is computed as

$$\mu_t = \frac{\rho k}{\omega}\frac{1}{max[\frac{1}{\alpha^*}, \frac{SF_2}{a_1\omega}]}$$

and each model constant is a blend of an inner (1) and outer (2) constant:

$$\phi = F_1\phi_1 + (1-F_1)\phi_2$$

a complete description of the model constants σ, β, γ, α, a_1, F_2 is present in [63] and the same reference suggests to limit the production terms in the k and ω transport equations.
The recommended boundary conditions are

$$\frac{U_\infty}{L} < \omega_{farfield} < 10\frac{U_\infty}{L} \qquad \frac{10^{-5}U_\infty^2}{Re_L} < k_{farfield} < \frac{0.1U_\infty^2}{Re_L}$$

$$\omega_{wall} = 10\frac{6\nu}{\beta_1(\Delta d_1)^2} \qquad k_{wall} = 0$$

where L is the approximate length of the computational domain and Δd_1 is the wall cell thickness.
In SU2 the Menter SST and the *Sustainable SST* are implemented [65], both models are present in [63].

A.8 SU2 configuration file for inviscid cases

Note: setting a free-stream temperature as described in the SU2 tutorials [28] (i.e. by setting INIT_OPTION: TD_CONDITIONS and FREESTREAM_OPTION: TEMPERATURE_FS in the configuration file, for example) would result in an incorrect calculation, as the forces would be computed using the given V^* while the flow field would be computed using the free-stream temperature and pressure. The

settings presented in this section work on SU2 version 7.0.8. and are based on [30]

```
SOLVER= EULER
MATH_PROBLEM= DIRECT
WRT_BINARY_RESTART= NO
READ_BINARY_RESTART= NO
MACH_NUMBER= 0.8
AOA= 0.0
FREESTREAM_PRESSURE= 101325.0

REF_ORIGIN_MOMENT_X = -0.5
REF_ORIGIN_MOMENT_Y = 0.00
REF_ORIGIN_MOMENT_Z = 0.00
REF_LENGTH= 1.0
REF_AREA= 1.0
REF_DIMENSIONALIZATION= DIMENSIONAL

TIME_DOMAIN=YES
TIME_MARCHING= DUAL_TIME_STEPPING-2ND_ORDER
TIME_STEP= 0.00174532925199
INNER_ITER= 150

SURFACE_MOVEMENT= AEROELASTIC
MACH_MOTION= 0.8
MARKER_MOVING= ( airfoil )
FLUTTER_SPEED_INDEX = 0.8
PLUNGE_NATURAL_FREQUENCY = 100
PITCH_NATURAL_FREQUENCY = 100
AIRFOIL_MASS_RATIO = 60
CG_LOCATION = 1.8
RADIUS_GYRATION_SQUARED = 3.48
AEROELASTIC_ITER = 3

WIND_GUST = YES
GUST_TYPE = SINE
GUST_DIR = Y_DIR
GUST_WAVELENGTH= 15.8175
GUST_PERIODS= 1.0
GUST_AMPL= 5.462
GUST_BEGIN_TIME= 0.0
GUST_BEGIN_LOC= -15.8175

MARKER_EULER= ( airfoil )
MARKER_FAR= ( farfield )
MARKER_PLOTTING = ( airfoil )
MARKER_MONITORING = ( airfoil )
 NUM_METHOD_GRAD= GREEN_GAUSS
```

```
CFL_NUMBER= 4.0

LINEAR_SOLVER= FGMRES
LINEAR_SOLVER_PREC= LU_SGS
LINEAR_SOLVER_ERROR= 1E-4
LINEAR_SOLVER_ITER= 2

MGLEVEL= 3
MGCYCLE= W_CYCLE
MG_PRE_SMOOTH= ( 1, 2, 3, 3 )
MG_POST_SMOOTH= ( 0, 0, 0, 0 )
MG_CORRECTION_SMOOTH= ( 0, 0, 0, 0 )
MG_DAMP_RESTRICTION= 0.75
MG_DAMP_PROLONGATION= 0.75

CONV_NUM_METHOD_FLOW= JST
MUSCL_FLOW= YES
SLOPE_LIMITER_FLOW= VENKATAKRISHNAN
ENTROPY_FIX_COEFF= 0.001
JST_SENSOR_COEFF= ( 0.5, 0.02 )
TIME_DISCRE_FLOW= EULER_IMPLICIT

DEFORM_LINEAR_SOLVER= FGMRES
DEFORM_LINEAR_SOLVER_PREC= LU_SGS
DEFORM_LINEAR_SOLVER_ITER= 500
DEFORM_NONLINEAR_ITER= 1
DEFORM_CONSOLE_OUTPUT= NO
DEFORM_LINEAR_SOLVER_ERROR= 1E-14
DEFORM_STIFFNESS_TYPE= INVERSE_VOLUME

TIME_ITER= 720
CONV_CRITERIA= RESIDUAL
CONV_RESIDUAL_MINVAL= -8
CONV_STARTITER= 0
CONV_CAUCHY_ELEMS= 100
CONV_CAUCHY_EPS= 1E-10

MESH_FILENAME= mesh.su2
MESH_FORMAT= SU2
SOLUTION_FILENAME= solution_flow.dat
TABULAR_FORMAT= CSV
CONV_FILENAME= history
RESTART_FILENAME= restart_flow.dat
VOLUME_FILENAME= flow
SURFACE_FILENAME= surface_flow
WRT_SOL_FREQ= 1000
WRT_SOL_FREQ_DUALTIME= 1000
WRT_CON_FREQ= 1
```

WRT_CON_FREQ_DUALTIME= 1
SCREEN_OUTPUT=(TIME_ITER,INNER_ITER,RMS_DENSITY, RMS_ENERGY, LIFT, DRAG_ON_SURFACE, PLUNGE, PITCH)
OUTPUT_FILES= (CSV,SURFACE_CSV, SURFACE_PARAVIEW_ASCII, PARAVIEW_ASCII)
HISTORY_OUTPUT= (ITER, TIME_DOMAIN, REL_RMS_RES,RMS_RES, AERO_COEFF,TAVG_AERO_COEFF,CAUCHY,AEROELASTIC)

A.9 Problems in SU2 7.0.8

Hereafter a list of the main reported problems in SU2 is reported:

- flow and surface output files are written in *.vtu* or *.vtk* files (where the former is more efficient than the latter) and both should be readable in Paraview 5.8: a reported problem is that if SU2 simulation is run on MS Win10, *.vtu* files are corrupted and *.vtk* are readable, while the opposite happens sometimes in Linux-Ubuntu.

- the aeroelastic package in SU2 is not updated to the latest version and the restart option doesn't work because the aeroelastic package never saves its state.