A Coupled Newton–Krylov Time Spectral Solver for Wing Flutter and LCO Prediction

Sicheng He,*
Eirikur Jonsson†
Charles A. Mader‡
Joaquim R. R. A. Martins§
University of Michigan, Department of Aerospace Engineering, Ann Arbor, MI

Flutter onset characteristic is an important consideration for transport aircraft design. Previous work with high fidelity aerostructural optimization has shown a tendency for optimization algorithms to produce unrealistically high aspect ratio designs, particularly when maximizing range or minimizing fuel burn, in an effort to maximize aerodynamic efficiency. In this work, we propose an efficient high fidelity flutter solution method. The flow and the structural dynamics have been modeled with time-spectral (TS) method. The TS method is designed to capture the dominant modes of periodic behaviors efficiently. We develop a coupled Newton–Krylov solver to solve this motion-fluid-structure coupled problem. In the literature, Newton-based TS flutter methods have been proposed in a segregated form. In the current work, the whole coupled system is directly solved. By doing so, one motion-fluid-structure solution is required instead of $\mathcal{O}(N_{\text{CSD}}) \times \mathcal{O}(N_{\text{iter}})$ CFD solutions where N_{CSD} is the structural degree-of-freedom for all time instances and N_{iter} is the number of Newton steps. We demonstrate the method on the classic AGARD 445.6 case.

I. Introduction

High-fidelity computational modeling and optimization of complex engineering systems has the potential to allow engineers to produce more efficient designs and to reduce the occurrence of unforeseen late stage design modifications. In particular, for transonic wing design, the simultaneous optimization of both the aerodynamic design and the internal structural sizing can yield significant fuel burn savings. However, when conducting optimization on an aircraft, all the relevant physics must be represented in the optimization problem, otherwise the result generated by the optimization procedure may not be meaningful or physical. Previous optimization results obtained by Kenway et al. [1, 2, 3, 4], without flutter constraints, produced optimized wings with large aspect ratio as shown in Fig. 1. Such configurations may be prone to flutter, which calls into question the usefulness of the result. Therefore, flutter and limit cycle oscillation (LCO) should be modeled and constrained during the optimization.

The goal of this work is to develop an efficient and robust solution methodology for finding the flutter onset. Although more emphasis is placed on flutter in this work, the proposed method is also capable of computing LCO conditions. Background on high fidelity CFD based methods modeling flutter is presented in Section II. The proposed flutter analysis method, presented in Section III, is a preconditioned, Jacobian-free, coupled Newton–Krylov method. It directly deals with all CFD, CSD, flutter velocity index V_f and flutter frequency variables without any black box computation. In one coupled Newton solution procedure, the residual of the coupled system is driven to zero and the flutter velocity index V_f is found. In Section IV the proposed method is demonstrated on the AGARD 445.6 where the flutter boundary as well as LCO map is computed. Section V concludes the paper.

II. Background

Since flutter is a certification-critical phenomenon, it is important to be able to predict it accurately. Conservative design approaches may lead to overly-stiff and hence high mass designs, while unconstrained optimization approaches, such as those shown in Fig. 1 may lead to overly-flexible wings that may cause problems when certifying the aircraft. Further, it is currently not unusual for flutter issues to be identified only at the final design and flight testing stages, at which point design changes are extremely costly. Accurate flutter prediction methods will therefore lead directly to

*PhD Student, AIAA student member
†PhD Student, AIAA student member
‡Research Investigator, AIAA Senior Member
§Professor, AIAA Associate Fellow

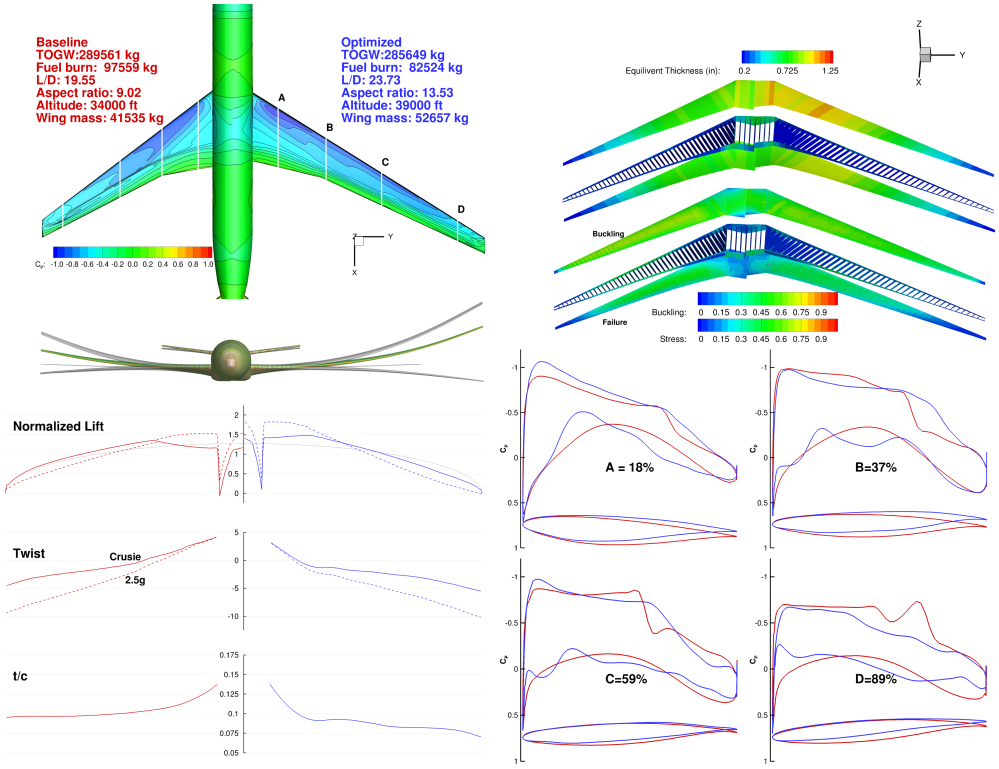


Figure 1. Aerostructural optimization result [3]: C_p and planform comparison with initial design (upper left); equivalent thickness distribution, stress and buckling KS failure criteria (upper right); comparison of initial and optimized lift distributions, twist distributions and thickness to chord ratio (t/c) (lower left); four airfoils with corresponding C_p distributions (lower right). (notice the increased span ratio)

significant cost savings. In this section, we give an overview of flutter prediction methods. For more details, we refer readers to a recent review paper on this topic by Jonsson et al. [5].

The current standard for flutter prediction in industry are methods based on panel codes, doublet-lattice method (DLM) [6], and linearized transonic small disturbance (TSD) equations [7, 8, 9, 10]. The DLM has gained remarkable success and is found in commercial software products such as MCS/Nastran and Astros [11, 12] and has become the aeroelasticians method of choice in industry. Recently, Jonsson et al. [13] developed a flutter constraint, using DLM aerodynamics, suitable for high fidelity gradient based optimization including wing planform variables. One major limit of some of those low fidelity linear aerodynamic methods is that they are unable to predict the occurrence of shock waves in the transonic flow. A consequence of this is that the prediction of the flutter boundaries can become inaccurate. In the transonic regime, there is a significant reduction in the flutter speed, called the *transonic dip (or flutter bucket)*. The bottom of the dip defines the minimum velocity at which flutter can occur across the flight envelope. Corrections using wind tunnel experimental data can however be applied to panel method aerodynamic influence coefficients (AIC). Unfortunately, for aerostructural design optimization this data is unavailable. Consequently, these low fidelity method are not applicable for commercial airliner design.

In the research community, the standard method for predicting a flutter boundary is to analyze the wing with a time-accurate coupled CFD-CSD solver similar as proposed by Liu et al. [14]. However, these methods incur a high computational cost since hundreds if not thousands of time steps are often required to simulate the flutter motion. This high computational cost makes the full time-accurate method ill-suited to optimization. Recently, various efforts use time-accurate results to train surrogate models on airfoils or flat plates and use the surrogate models to substitute high fidelity CFD in flutter simulation to reduce the computational cost [15, 16, 17, 18]. But the surrogate may not be accurate to predict aerodynamics on a more complex geometry e.g. a swept wing where the 3D effect is no longer negligible. Like in many periodic problems, much of the computational time is spent resolving the decay of the initial transients in the unsteady problem [19]. Fortunately, in problems where the periodic steady state is of primary interest, time periodic simulation methods such as the Harmonic Balance (HB) presented by Hall et al. [20], TS by Gopinath and Jameson [21] or the non-linear frequency domain method by McMullen [19] can all be used to accelerate the solution process. The basic idea of time periodic methods is to represent all the state variables in the system with a Fourier series. This allows the time-dependent problem to be transformed into a series of coupled steady state

problems. Among the methods, TS method [21] has the advantage of having state variables in time domain and is can be relatively more easily adapted from a steady state solver.

There also has been significant effort put to prediction of the flutter boundary using spectral methods [22, 23, 24, 25, 26, 27, 28]. Thomas et al. [29, 30] proposed flutter equations including a prescribed motion and a solution methodology with Newton–Raphson method. With their method, the flutter velocity index is a state variable to be solved rather than an external parameter used as an input for a general unsteady solver. That gives the method an advantage of being automatic which is critical in MDO. However, the method is hindered by scalability: $O(N_{\text{CSD}} \times N_{\text{iter}})$ CFD evaluations are required for flutter simulation where N_{CSD} is the number of CSD state variables and N_{iter} is the number of Newton steps. The root cause of this limitation is that the CFD module is treated as a black box.

To overcome these shortcomings, He et al. [26, 31] propose a Jacobian-free, coupled Newton–Krylov method which resolves both CSD and CFD equations without any black box limitations for airfoils. Since all the states are directly resolved, the $O(n_{\text{CSD}})$ CFD evaluations are gone, making the method more efficient. The use of Krylov iterative methods guarantees fast solution for each linear equation encountered in Newton iterations. While the Jacobian-free nature of the method further cuts the computational and memory cost of the method by eliminating the need to evaluate and store the full Jacobian. The current paper is an extension of previous work by the authors to a wing case [26].

III. Time Spectral Flutter Equation

We solve the TS flutter equation in this research [29, 30, 26]. The TS flutter equation is composed of three components: a prescribed motion magnitude constraints, CSD equations and CFD equations An aerostructural problem usually contains CFD and CSD components. The additional prescribed motion constraints in the TS flutter equation are added to make the choice of speed and frequency unique as indicated by Fig. 2. Without the prescribed motion constraints, every point on both branches of LCO response satisfy the CSD and CFD equations. In this work, we treat flutter as a special LCO with small motion amplitude (ϵ small).

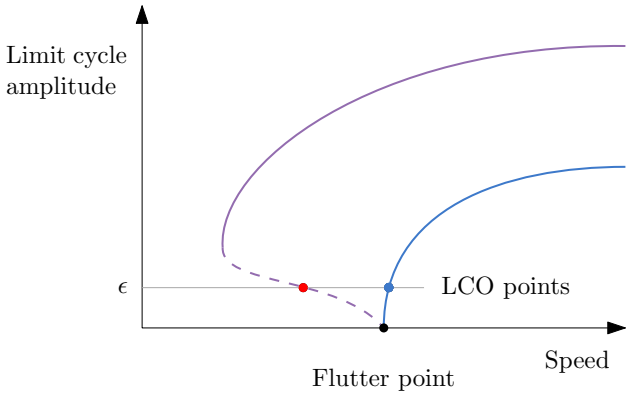


Figure 2. LCO and flutter prediction. The purple line shows a subcritical response and the blue curve shows a supercritical response. The red point is an unstable LCO point and the blue point is a stable LCO point corresponding with prescribed motion magnitude ϵ. The black point is a flutter point where can be obtained by extrapolation of the LCO response with $\epsilon = 0$.

The time spectral flutter equation written in residual form is given in Eq. (1), for details we refer the readers to our previous work [26].

$$\boldsymbol{\mathcal{R}}(\mathbf{q}) \coloneqq \begin{bmatrix} \mathcal{R}_{\text{magnitude}} \\ \mathcal{R}_{\text{phase}} \\ \boldsymbol{\mathcal{S}}_{\text{TS}} \\ \boldsymbol{\mathcal{A}}_{\text{TS}} \end{bmatrix}, \mathbf{q} \coloneqq \begin{bmatrix} V_f \\ \omega \\ \bar{\boldsymbol{\eta}}^n \\ \boldsymbol{\zeta}^n \end{bmatrix}, \tag{1}$$

where V_f is flutter velocity index, ω is the flutter frequency, $\bar{\boldsymbol{\eta}}^n$ is the normalized generalized coordinates and $\boldsymbol{\zeta}^n$ is the aerodynamic states history for n time instances. $\mathcal{R}_{\text{magnitude}}, \mathcal{R}_{\text{phase}}$ are the constraints for prescribed motion magnitude and phase respectively, $\boldsymbol{\mathcal{S}}_{\text{TS}}$ is the TS structural dynamic constraint and $\boldsymbol{\mathcal{A}}_{\text{TS}}$ is the TS aerodynamic constraint.

A. Prescribed Motion Equations

In this section we present derivation of the two equations constraining the magnitude and phase of the natural mode similar with [26]. States from different time instances are defined as $\bar{\eta}^1, \bar{\eta}^2, \ldots, \bar{\eta}^N$. Defining $\omega = 2\pi/T$, where T is

the time period, a harmonic motion for the jth mode can be described by the following expression,

$$\bar{\eta}_{,j}(t) \approx c_{0,j} + c_{1,j}e^{i\omega t} + c_{2,j}e^{i2\omega t} + \ldots + c_{-2,j}e^{-i2\omega t} + c_{-1,j}e^{-i\omega t}, \tag{2}$$

where $\bar{\eta}_j(t)$ denotes the jth structural mode coefficient. The 1st harmonic in temporal space of the jth structural mode is given as

$$\begin{aligned}\bar{\eta}_{\text{1st harmonic},j} &= c_{1,j}e^{i\omega t} + c_{-1,j}e^{-i\omega t} \\ &= [\Re(c_{1,j}) + \Re(c_{-1,j})]\cos(\omega t) + [\Im(-c_{1,j}) + \Im(c_{-1,j})]\sin(\omega t) + \text{pure imaginary number} \\ &= C_{c,j}\cos(\omega t) + C_{s,j}\sin(\omega t) + \text{pure imaginary number}\end{aligned} \tag{3}$$

and by dropping the imaginary part and using trigonometric identities we obtain

$$\bar{\eta}_{\text{1st harmonic},j} = |\bar{\eta}_{\text{1st harmonic},j}|\sin(\omega t + \theta_{\text{1st harmonic},j}), \tag{4}$$

where dominant mode magnitude and phase are given as

$$\begin{aligned}|\bar{\eta}_{\text{1st harmonic},j}| &= \sqrt{C_{c,j}^2 + C_{s,j}^2}, \\ \theta_{\text{1st harmonic},j} &= \sin^{-1}\left(\frac{C_{c,j}}{\sqrt{C_{c,j}^2 + C_{s,j}^2}}\right).\end{aligned} \tag{5}$$

and the coefficients $C_{c,j}$ and $C_{s,j}$ are given as

$$\begin{aligned}C_c &= \Re(c_{1,j}) + \Re(c_{-1,j}), \\ C_s &= -\Im(c_{1,j}) + \Im(c_{-1,j}).\end{aligned} \tag{6}$$

Given c_i, the magnitude and the motion can be calculated from the equations above. The c_i coefficients are calculated by fast-Fourier-transform (FFT) with for a given structural mode shape history as the input:

$$[c_{0,j}, c_{1,j}, c_{2,j}, ..., c_{-2,j}, c_{-1,j}] = \frac{1}{n}\text{FFT}(\bar{\eta}_{1,j}, \bar{\eta}_{2,j}, ..., \bar{\eta}_{n,j}). \tag{7}$$

Finally, the residual of the two constraints are written out as following:

$$\begin{aligned}\mathcal{R}_{\text{magnitude}} &:= |\bar{\eta}_{\text{1st harmonic},j}| - \epsilon_{0,j}, \\ \mathcal{R}_{\text{phase}} &:= \theta_{\text{1st harmonic},j} - \theta_{0,j},\end{aligned} \tag{8}$$

where $\epsilon_{0,j}, \theta_{0,j}$ are prescribed small motion magnitude and its phase, for the jth structural mode. In this work the prescribed motion is applied to the first natural mode i.e. $j = 1$.

B. TS CSD Equations with Mode Shapes

The CSD equations are

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{f}, \tag{9}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{K}$ is the stiffness matrix, $\mathbf{u}$ is the displacement and $\mathbf{f}$ is the external load. We construct the CSD equations using the structural natural mode shapes. At first, we conduct a modal analysis,

$$\omega_j^2\mathbf{M}\boldsymbol{\phi}_j = \mathbf{K}\boldsymbol{\phi}_j, \tag{10}$$

where $\omega_j, \boldsymbol{\phi}_j$ are the jth natural frequency and mode shape, respectively. We define

$$\begin{aligned}\boldsymbol{\Omega} &\coloneqq \text{Diag}(\omega_1, \cdots, \omega_r), \\ \boldsymbol{\phi} &\coloneqq \left[\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_r\right].\end{aligned} \tag{11}$$

where r is the number of modes and mode shapes computed which is typically much smaller than the structural degrees of freedom. Assuming that displacements can be approximated by $\mathbf{u} \approx \boldsymbol{\phi}\boldsymbol{\eta}$ we write the equations of motion as,

$$\begin{aligned}\boldsymbol{\phi}^\top\mathbf{M}\boldsymbol{\phi}\ddot{\boldsymbol{\eta}} + \boldsymbol{\phi}^\top\mathbf{K}\boldsymbol{\phi}\boldsymbol{\eta} - \boldsymbol{\phi}^\top\mathbf{f} = 0, \\ \mathbf{M}_r\ddot{\boldsymbol{\eta}} + \mathbf{K}_r\boldsymbol{\eta} - \mathbf{f}_r = 0,\end{aligned} \tag{12}$$

where $\boldsymbol{\eta}$ is the general coordinate and subscript $\mathbf{M}_r, \mathbf{K}_r, \mathbf{f}_r$ denote the reduced or generalized mass, stiffness, and force matrices. This equation can be further simplified by writing Eq. (10) in matrix form and pre-multiplying $\boldsymbol{\phi}^\top$ to obtain

$$\mathbf{K}_r = \mathbf{M}_r \boldsymbol{\Omega}^2 \tag{13}$$

where $\boldsymbol{\Omega}^2$ is defined in Eq. (11) with reduced set of natural frequencies. Equation (12) can then be written as,

$$\mathbf{M}_r \ddot{\boldsymbol{\eta}} + \mathbf{M}_r \boldsymbol{\Omega}^2 \boldsymbol{\eta} - \mathbf{f}_r = 0. \tag{14}$$

The TS form of Eq. (12) including the with mode shapes can then be written as,

$$\mathbf{M}_r^n \left(\mathbf{P}^\top \mathbf{D}_{t,t} \mathbf{P}\right) \boldsymbol{\eta}^n + \mathbf{M}_r (\boldsymbol{\Omega}_r^n)^2 \boldsymbol{\eta}^n - \mathbf{f}_r^n = 0, \tag{15}$$

where

$$\begin{aligned}
\mathbf{M}_r^n &\coloneqq \text{Diag}(\underbrace{\boldsymbol{\phi}^\top \mathbf{M} \boldsymbol{\phi}, \ldots, \boldsymbol{\phi}^\top \mathbf{M} \boldsymbol{\phi}}_{n}) = \text{Diag}(\underbrace{\mathbf{M}_r \ldots, \mathbf{M}_r}_{n}), \\
\boldsymbol{\Omega}^n &\coloneqq \text{Diag}(\underbrace{\boldsymbol{\Omega}, \cdots, \boldsymbol{\Omega}}_{n}), \\
\mathbf{P}_{i,j} &= \begin{cases} 1 & \text{if mod}(j, r) = \lceil i/n \rceil), \\ 0 & \text{otherwise,} \end{cases} \\
\mathbf{D}_{t,t} &\coloneqq \text{Diag}(\underbrace{\mathbf{D}_t^2, \ldots, \mathbf{D}_t^2}_{r}) = \omega^2 \text{Diag}(\underbrace{\mathbf{D}^2, \ldots, \mathbf{D}^2}_{r}) = \omega^2 \bar{\mathbf{D}}^2, \\
\mathbf{f}_r^n &\coloneqq \left[\boldsymbol{\phi}^\top \mathbf{f}_1, \quad \ldots \quad, \boldsymbol{\phi}^\top \mathbf{f}_n\right]^\top = \left[\mathbf{f}_{r,1}, \quad \ldots \quad, \mathbf{f}_{r,n}\right]^\top.
\end{aligned} \tag{16}$$

Here $\mathbf{P}$ is a permutation matrix and ω is the flow frequency as defined Section A. Together with the second order spectral derivative matrix $\mathbf{D}_{t,t}$, the second time derivatives of state variables for different modes are obtained,

$$\ddot{\boldsymbol{\eta}}^n = \left(\mathbf{P}^\top \mathbf{D}_{t,t} \mathbf{P}\right) \boldsymbol{\eta}^n. \tag{17}$$

1. Dimensionless form

Here we present the TS equations of motion in a nondimensional form. The aerodynamic forces are normalized by the dynamic pressure $q_\infty = 1/2 \rho_\infty U_\infty^2$ and a reference are S_{ref} and can be written as

$$\bar{\mathbf{f}} = \frac{\mathbf{f}}{\frac{1}{2}\rho_\infty U_\infty^2 S_{\text{ref}}}. \tag{18}$$

It follows that the normalized generalized aerodynamic forces are then written as

$$\bar{\mathbf{f}}_r^n \coloneqq \left[\boldsymbol{\phi}^\top \bar{\mathbf{f}}_1, \quad \cdots \quad, \boldsymbol{\phi}^\top \bar{\mathbf{f}}_n\right]^\top = \left[\bar{\mathbf{f}}_{r,1}, \quad \ldots \quad, \bar{\mathbf{f}}_{r,n}\right]^\top. \tag{19}$$

To nondimensionalize Eq. (15) we use the wing mass m_0, the semi chord b and the first torsion mode, $\omega_\alpha = \omega_2$, which in this case is the second natural mode. The dimensionless CSD equation can then be written in residual form as,

$$\begin{aligned}
\boldsymbol{\mathcal{S}}_{\text{TS}} &\coloneqq \left(\frac{\mathbf{M}_r^n}{m_0}\right)\left(\frac{\omega^2}{\omega_\alpha^2}\right)\left(\mathbf{P}^\top \bar{\mathbf{D}}^2 \mathbf{P}\right)\left(\frac{\boldsymbol{\eta}^n}{b}\right) + \left(\frac{\mathbf{M}_r^n}{m_0}\right)\left(\frac{\boldsymbol{\Omega}^2}{\omega_\alpha^2}\right)\left(\frac{\boldsymbol{\eta}^n}{b}\right) - \frac{1}{2}\frac{\rho_\infty U_\infty^2 S_{\text{ref}}}{m_0 \omega_\alpha^2 b}\bar{\mathbf{f}}_r^n, \\
&= \left(\frac{\mathbf{M}_r^n}{m_0}\right)\left(\frac{\omega^2}{\omega_\alpha^2}\right)\left(\mathbf{P}^\top \bar{\mathbf{D}}^2 \mathbf{P}\right)\bar{\boldsymbol{\eta}}^n + \left(\frac{\mathbf{M}_r^n}{m_0}\right)\left(\frac{\boldsymbol{\Omega}^2}{\omega_\alpha^2}\right)\bar{\boldsymbol{\eta}}^n - \frac{1}{2}\left(\frac{S_{\text{ref}} b}{V_0}\right) V_f^2 \bar{\mathbf{f}}_r^n = 0,
\end{aligned} \tag{20}$$

where the following nondimensional coefficients have been introduced and are defined as,

$$\begin{aligned}
\mu &\coloneqq \frac{m_0}{\rho_\infty V_0}, \\
V_f &\coloneqq \frac{U_\infty}{\sqrt{\mu}\omega_\alpha b}, \\
\bar{\boldsymbol{\eta}}^n &\coloneqq \frac{\boldsymbol{\eta}^n}{b}.
\end{aligned} \tag{21}$$

Here μ is the mass ratio, V_0 is the volume of a conical frustum having root chord as lower base diameter, tip chord as upper base diameter, and panel span as height and V_f is the flutter speed index.

C. TS CFD Equations

In this work, we use the open-source CFD solver ADflow[a], a parallel, finite-volume, cell-centered, multi-block solver, which solves the Euler and the Reynolds averaged Navier-Stokes (RANS) equations in either steady, unsteady or TS modes by Kenway et al. [32]. A robust ANK solver proposed by Yildirim et al. [33] is implemented to reduce the residual to several orders of magnitude. Subsequently, a Jacobian-free NK solver is used to get the final solution. In this work, we solve TS Euler equations with ANK and NK solvers. The Euler equations can be written as:

$$\frac{\partial (V\boldsymbol{\zeta})}{\partial t} + \nabla \cdot \mathbf{F} = 0, \tag{22}$$

where

$$\boldsymbol{\zeta} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho e_t \end{bmatrix}, \mathbf{F}_k = \begin{bmatrix} \rho u_k - \rho w_k \\ \rho u_k u_1 - \rho w_k u_1 + p\delta_{k,1} \\ \rho u_k u_2 - \rho w_k u_2 + p\delta_{k,2} \\ \rho u_k u_3 - \rho w_k u_3 + p\delta_{k,3} \\ \rho u_k (e_t + p) - \rho w_k e_t \end{bmatrix}, \tag{23}$$

$\boldsymbol{\zeta}$ is the state vector, $\mathbf{F}_k$ is the inviscid flux term in kth coordinate, ρ is the density, V is the cell volume, u_ks is flow velocity in kth coordinate, w_k is the grid velocity in kth coordinate, p is the static pressure and, e_t is the total energy.

The TS Euler equation is given as

$$\boldsymbol{\mathcal{A}}_{\text{TS}} \coloneqq \mathbf{D}_t(V^n \boldsymbol{\zeta}^n) + \mathbf{R}(\boldsymbol{\zeta}^n), \tag{24}$$

where n denotes number of time instances, $V^n = (V_1, ..., V_n)$, V_i denoting the volume of a cell from ith time instance, $\boldsymbol{\zeta}^n = (\boldsymbol{\zeta}_1, ..., \boldsymbol{\zeta}_n)$, $\boldsymbol{\zeta}_i$ is the state variable from ith time instance, $\mathbf{R}(\boldsymbol{\zeta}^n) = \left[(\nabla \cdot \mathbf{F})_1, \quad \cdots, \quad (\nabla \cdot \mathbf{F})_n\right]^\top$ where $(\nabla \cdot \mathbf{F})_i$ is the inviscid flux in ith time instance and $\mathbf{D}_t$ is the spectral derivative operator. The spectral derivative operator maps the state variables to their time derivatives

$$\mathbf{D}_t \boldsymbol{\zeta}^n = \omega \mathbf{D} \boldsymbol{\zeta}^n = \omega \sum_{i=1}^{n} \mathbf{D}_{i,j} \boldsymbol{\zeta}_i \tag{25}$$

where

$$\mathbf{D}_{i,j} = \begin{cases} \frac{1}{2} \frac{(-1)^{(j-i)}}{\sin[\pi(j-i)/n]} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}. \tag{26}$$

The grid velocity w_k^n is computed by spectral differentiation

$$w_k^n = \mathbf{D}_t x^n_{\text{grid, k}} + U_{\infty,k}, \tag{27}$$

where $x^n_{\text{grid, k}}$ is the grid kth coordinate for all time instances, $U_{\infty,k}$ is the main stream velocity in kth coordinate.

1. *Boundary condition*

In ADflow, the boundary condition can be set through the triplet (T_∞, p_∞, M). In this analysis we define the problem in terms of (M, μ, V_f) and thus need to compute (T_∞, p_∞). Here we detail the procedure to compute the boundary conditions. Flow density can be computed from previously defined nondimesnional coefficients in Eq. (21),

$$\rho_\infty = \frac{m_0}{\mu V_0}. \tag{28}$$

Similarly, from Eq. (21) and the dynamic pressure, the static pressure, p_∞ is obtained as

$$\begin{aligned} U_\infty &= V_f b \omega_\alpha \sqrt{\mu}, \\ q_\infty &= \frac{1}{2} \rho_\infty U_\infty^2, \\ p_\infty &= \frac{2 q_\infty}{\gamma M^2}. \end{aligned} \tag{29}$$

Finally, the temperature T_∞ is found by usind the ideal gas law,:

$$T_\infty = \frac{p_\infty}{\rho_\infty R}, \tag{30}$$

[a] https://github.com/mdolab/adflow.git

where R is the gas constant for air.

Here M and μ are taken as parameters whereas V_f is taken as an independent state variable. Subsequently, we can define the static pressure and temperatures as a function of the flutter speed index,

$$\begin{aligned} T_\infty &= T_\infty(V_f), \\ p_\infty &= p_\infty(V_f). \end{aligned} \tag{31}$$

2. Load calculation

The aerodynamic load is computed at each nodes for each time instance i

$$\mathbf{f}_i = \mathbf{f}_i(\mathbf{X}_{S,i}, \boldsymbol{\zeta}_i), \tag{32}$$

where $\mathbf{X}_{S,i}$ is the surface mesh for time instance i. The dimensionless aerodynamic load is as previously defined

$$\bar{\mathbf{f}}_i = \frac{1}{q_\infty S_{\text{ref}}} \mathbf{f}_i(\mathbf{X}_{S,i}, \boldsymbol{\zeta}_i).$$

Furthermore, since $U_\infty = U_\infty(V_f)$, we finally have

$$\bar{\mathbf{f}}_i = \bar{\mathbf{f}}_i(V_f, \mathbf{X}_{S,i}, \boldsymbol{\zeta}_i). \tag{33}$$

D. CFD–CSD load and displacement transfer

1. Displacement transfer

In general the aerodynamic and structural grids do not have the same topology or match in terms of surface grid point locations. Thus, an interpolation scheme is needed to transfer both loads and displacements the is suitable for not matching grids. In this work a relatively simple strategy is adapted. Given the structure mode shapes $\boldsymbol{\phi}$ at coordinates $\mathbf{X}_{\text{mode}}$ we fit $\boldsymbol{\phi}$ with a fourth order polynomial which we denote as $\hat{\boldsymbol{\phi}}$. The aerodynamic mode shapes can then be computed from this function by evaluating it using the jig shape i.e. $\hat{\boldsymbol{\phi}}_3(\mathbf{X}_J)$.

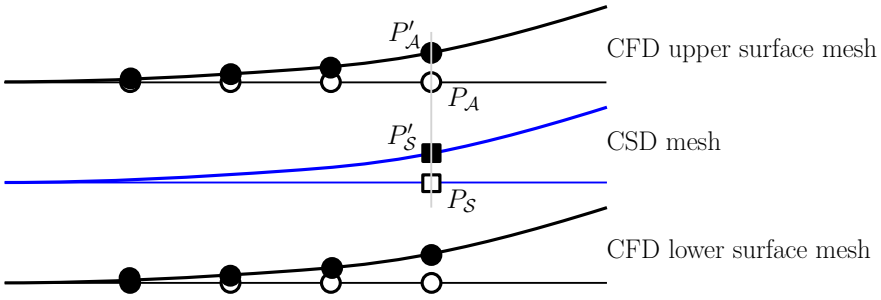


Figure 3. Transfer illustration. $P_{\mathcal{S}}$ is a projection of $P_{\mathcal{A}}$ on the structure mode surface. $|P_{\mathcal{A}}P'_{\mathcal{A}}| = |P_{\mathcal{S}}P'_{\mathcal{S}}|$.

The displacement of the CFD nodes in the ith time instance can be defined in term of the aerodynamic modes shapes as,

$$\bar{\mathbf{u}}_{3,i} = \hat{\boldsymbol{\phi}}_3(y_1, y_2)\bar{\boldsymbol{\eta}}_i. \tag{34}$$

Here we ignore any displacement in u_1 and u_2 since the magnitude of these two components is much smaller than compared with u_3. The surface deformation of the aerodynamic mesh can then be written as

$$\mathbf{X}_{S,i} = \mathbf{X}_J + \begin{bmatrix} 0 \\ 0 \\ \hat{\boldsymbol{\phi}}_3(\mathbf{X}_J) \end{bmatrix} \bar{\boldsymbol{\eta}}_i b. \tag{35}$$

where $\mathbf{X}_J$ surface coordinates of the undeformed aerodynamic mesh i.e. the jig shape coordinates.

2. Load transfer

The virtual work on the CFD mesh after a small deformation can be written as

$$\begin{aligned}\delta\overline{W}_{\text{CFD},i} &= -\left(\bar{\mathbf{f}}_{3,i}(V_f, \boldsymbol{\zeta}_i, \mathbf{X}_{S,i})\right)^\top \delta\bar{\mathbf{u}}_{3,i} \\ &= -\left(\bar{\mathbf{f}}_{3,i}(V_f, \boldsymbol{\zeta}_i, \mathbf{X}_{S,i})\right)^\top \hat{\boldsymbol{\phi}}_3(\mathbf{X}_J)\delta\bar{\boldsymbol{\eta}}_i,\end{aligned} \tag{36}$$

where Eq. (34) was applied. The virtual work on the CSD mesh is given as

$$\delta\overline{W}_{\text{CSD},i} = \left(\bar{\mathbf{f}}_{r,i}(\mathbf{X}_J, V_f, \boldsymbol{\zeta}_i, \mathbf{X}_{S,i})\right)^\top \delta\bar{\boldsymbol{\eta}}_i. \tag{37}$$

To make the transfer consistent, we have

$$\delta\overline{W}_{\text{CFD},i} + \delta\overline{W}_{\text{CSD},i} = 0, \tag{38}$$

for all virtual displacement. This gives

$$\bar{\mathbf{f}}_{r,i}(\mathbf{X}_J, V_f, \boldsymbol{\zeta}_i, \mathbf{X}_{S,i}) = \hat{\boldsymbol{\phi}}_3(\mathbf{X}_J)^\top \left(\bar{\mathbf{f}}_{3,i}(V_f, \boldsymbol{\zeta}_i, \mathbf{X}_{S,i})\right). \tag{39}$$

E. Coupled Newton–Krylov Solver for the TS Flutter Equation

Now we present the solution methodology for the TS flutter equation, Eq. (1). The method used is a preconditioned coupled Jacobian-Free Newton-Krylov method. Equation (1) is solved using Newton's method, which results in the following linear system:

$$\begin{aligned}\mathbf{J}\Delta\mathbf{q} &= -\boldsymbol{\mathcal{R}}(\mathbf{q}^{(k)}), \\ \mathbf{q}^{(k+1)} &= \mathbf{q}^{(k)} + \alpha\Delta\mathbf{q},\end{aligned} \tag{40}$$

where $\mathbf{J}$ is the Jacobian, $\frac{\partial\boldsymbol{\mathcal{R}}(q)}{\partial\mathbf{q}}|_{\mathbf{q}=\mathbf{q}^{(k)}}$, $\Delta\mathbf{q}$ is the increment step, $\mathbf{q}^{(k)}$ and $\mathbf{q}^{(k+1)}$ are the current states and the states for the next step, and α is a positive step size determined by either line search or trust region methods. Each increment step, $\Delta\mathbf{q}$, is solved iteratively up to a tolerance determined by Eisenstat–Walker algorithm [34]. It is solved with FGMERES method [35], a Krylov subspace method, by minimizing the residual $\mathbf{J}\Delta\mathbf{q} + \boldsymbol{\mathcal{R}}(\mathbf{q}^{(k)})$ norm in the span of $\{\boldsymbol{\mathcal{R}}(q^{(k)}), \mathbf{J}\boldsymbol{\mathcal{R}}(\mathbf{q}^{(k)}), ..., \mathbf{J}^{m-1}\boldsymbol{\mathcal{R}}(\mathbf{q}^{(k)})\}$, where m is the Krylov subspace size. The most computational demanding steps of this process are those related with matrix vector products, i.e. $\mathbf{Jv}$ computed when conducting Arnoldi iteration within FGMRES method. Instead of evaluating all the terms in the Jacobian, saving it explicitly and directly applying matrix vector product, we apply the following approximation, which is more economic in terms of both computational time and memory:

$$\mathbf{Jv} \approx \frac{\boldsymbol{\mathcal{R}}(\mathbf{q}^{(k)} + \epsilon\mathbf{v}) - \boldsymbol{\mathcal{R}}(\mathbf{q}^{(k)})}{\epsilon}. \tag{41}$$

The step size ϵ is determined based on Brown and Saad [36],

$$\epsilon = \begin{cases} e_{\text{rel}}\mathbf{v}^\top\mathbf{q}^{(k)}/\|\mathbf{v}\|_2^2 & \text{if } \left|\mathbf{v}^\top\mathbf{q}^{(k)}\right| > u_{\min}\|\mathbf{v}\|_1 \\ e_{\text{rel}}u_{\min}\,\text{sign}\left(\mathbf{v}^T\mathbf{q}^{(k)}\right)\|\mathbf{v}\|_1/\|\mathbf{v}\|_2^2 & \text{else,}\end{cases} \tag{42}$$

where $u_{\min}$ and e_{rel} are 10^{-6} and 10^{-8} respectively. Finally, the step size α is selected with a cubic line search option. The details for Jacobian-Free Newton Krylov method can be found in [37]. The residual evaluation is described in Algorithm 1:

We implement the solver through PETSc [38]. A relatively large subspace iteration number of 150 to 300 are used to enhance the robustness of the solver with a penalty on the solution speed. This solution method is an extension of Kenway et al. [1] which is a steady state aeroelastic solver.

1. Preconditioner

When solving Eq. (40), one critical requirement for good performance of an iterative method is that the eigenvalues of $\mathbf{J}$ be close with each other. To guarantee that, we need to carefully design a preconditioner. Similar to the previous steady aerostructural work, we use a block-Jacobi preconditioner. Here we implement a right preconditioner,

$$\begin{aligned}(\mathbf{JP}^{-1})\Delta\mathbf{y} &= -\boldsymbol{\mathcal{R}}(\mathbf{q}^{(k)}), \\ \mathbf{P}^{-1}\Delta\mathbf{y} &= \Delta\mathbf{q},\end{aligned} \tag{43}$$

Algorithm 1 Coupled nonlinear residual computation

function $\mathcal{R}(V_f, \omega_f, \bar{\boldsymbol{\eta}}^n, \boldsymbol{\zeta}^n)$
  $\mathbf{X}_S^n \leftarrow \hat{\boldsymbol{\Phi}}\bar{\boldsymbol{\eta}}^n b + \mathbf{X}_J$ ▷ Transfer displacements
  $\mathbf{X}_V^n \leftarrow \mathcal{W}(\mathbf{X}_S^n)$ ▷ Deform volume mesh to match surface
  $\boldsymbol{\mathcal{A}}_{\text{TS}} \leftarrow \boldsymbol{\mathcal{A}}_{\text{TS}}(V_f, \omega_f, \boldsymbol{\zeta}^n, \mathbf{X}_V^n)$ ▷ Evaluate CFD residuals
  $\bar{\mathbf{f}}_A^n \leftarrow \bar{\mathbf{f}}_A^n(\boldsymbol{\zeta}^n, \mathbf{X}_S^n)$ ▷ Evaluate aerodynamics forces
  $\bar{\mathbf{f}}^n \leftarrow \hat{\boldsymbol{\phi}}^\top \bar{\mathbf{f}}_A^n$ ▷ Transfer forces
  $\boldsymbol{\mathcal{S}}_{\text{TS}} \leftarrow \boldsymbol{\mathcal{S}}_{\text{TS}}(V_f, \omega_f, \bar{\boldsymbol{\eta}}^n, \bar{\mathbf{f}}^n)$ ▷ Evaluate CSD residuals
  $\boldsymbol{\mathcal{R}}_{\text{motion, magnitude}} \leftarrow \boldsymbol{\mathcal{R}}_{\text{motion, magnitude}}(\bar{\boldsymbol{\eta}}^n)$ ▷ Evaluate prescribed motion magnitude residual
  $\boldsymbol{\mathcal{R}}_{\text{motion, phase}} \leftarrow \boldsymbol{\mathcal{R}}_{\text{motion, phase}}(\bar{\boldsymbol{\eta}}^n)$ ▷ Evaluate prescribed motion phase residual
  $\boldsymbol{\mathcal{R}} \leftarrow (\boldsymbol{\mathcal{R}}_{\text{motion, magnitude}}, \boldsymbol{\mathcal{R}}_{\text{motion, phase}}, \mathcal{S}_{\text{TS}}, \mathcal{A}_{\text{TS}})$ ▷ Combine residuals
return $\mathcal{R}$
end function

where $\mathbf{P}$ is the preconditioner. To be more specific, the second equation can be expanded as,

$$\begin{pmatrix} \mathbf{P}^{-1}_{\text{motion,CSD}} & 0 \\ 0 & \mathbf{P}^{-1}_{\text{CFD}} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{y}_{V_f,\omega,\text{CSD}} \\ \Delta\mathbf{y}_{\text{CFD}} \end{pmatrix} = \begin{pmatrix} \Delta\mathbf{q}_{V_f,\omega,\text{CSD}} \\ \Delta\mathbf{q}_{\text{CFD}} \end{pmatrix}. \tag{44}$$

The preconditioner already implemented by Kenway et al. [1] for CFD solver $\mathbf{P}^{-1}_{\text{CFD}}$ (solution of CFD with a smaller stencil) is reused here. A new preconditioner for the motion equations and the CSD equations needs to be implemented. Denoted as $\mathbf{P}^{-1}_{\text{motion,CSD}}$, a direct inversion of the Jacobian is used as the preconditioner for the motion equations and the CSD equations. For relatively small problem size, such as of a 2D case, a direct factorization is reasonable. However, with a larger structure, such as required for a full 3D wing box, more careful research should be devoted to the preconditioner design. The resulting preconditioner can then be written as,

$$\mathbf{P}^{-1}_{\text{motion,CSD}} = \begin{pmatrix} 0 & 0 & \frac{\partial|\bar{\eta}_{\text{1st harmonic},j}|}{\partial\bar{\boldsymbol{\eta}}^n} \\ 0 & 0 & \frac{\partial\theta_{\text{1st harmonic},j}}{\partial\bar{\boldsymbol{\eta}}^n} \\ -\frac{V_f S_{\text{ref}} b}{V_0}\mathbf{f}_r^n & \left(\frac{\mathbf{M}_r^n}{m_0}\right)\left(2\frac{\omega}{\omega_\alpha^2}\right)\left(\mathbf{P}^\top\bar{\mathbf{D}}^2\mathbf{P}\right)\bar{\boldsymbol{\eta}}^n & \left(\frac{\mathbf{M}_r^n}{m_0}\right)\left(\frac{\omega^2}{\omega_\alpha^2}\right)\left(\mathbf{P}^\top\bar{\mathbf{D}}^2\mathbf{P}\right) + \left(\frac{\mathbf{M}_r^n}{m_0}\right)\left(\frac{\boldsymbol{\Omega}^2}{\omega_\alpha^2}\right) \end{pmatrix}^{-1}. \tag{45}$$

IV. Result

A. Model description

1. *Structural model*

At first, we conduct a conversion of the dimensionless structural mode. In the AGARD 445.6 case, the modes are given as displacements at points. The generalized mass matrix, $\tilde{\boldsymbol{\phi}}^\top\mathbf{M}\tilde{\boldsymbol{\phi}}$, in the original AGARD report is normalized to give unit mass in English units ($\text{lbf}\cdot\text{sec}^2\cdot\text{in}^{-1}$). Here the original matrix can be written in SI units as,

$$\begin{aligned} \tilde{\boldsymbol{\phi}}^\top\mathbf{M}\tilde{\boldsymbol{\phi}} &= \begin{bmatrix} 1\text{lbf}\,\text{sec}^2\,\text{in}^{-1} & & \\ & \ddots & \\ & & 1\text{lbf}\,\text{sec}^2\,\text{in}^{-1} \end{bmatrix}, \\ &= \begin{bmatrix} \frac{1\text{slug ft}}{\text{sec}^2}\,\text{sec}^2\,\text{in}^{-1} & & \\ & \ddots & \\ & & \frac{1\text{slug ft}}{\text{sec}^2}\,\text{sec}^2\,\text{in}^{-1} \end{bmatrix}, \\ &= \begin{bmatrix} 12\text{slug} & & \\ & \ddots & \\ & & 12\text{slug} \end{bmatrix}, \\ &= \begin{bmatrix} 175.127\text{kg} & & \\ & \ddots & \\ & & 175.127\text{kg} \end{bmatrix}. \end{aligned} \tag{46}$$

However, in this work we require this to be nondimensional from such that.

$$\boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} = \mathbf{I}. \tag{47}$$

We thus seek to find a scaling factor c such that,

$$(c\sqrt{m_0}\tilde{\boldsymbol{\phi}}^\top) \frac{\mathbf{M}}{m_0} (c\sqrt{m_0}\tilde{\boldsymbol{\phi}}) = \mathbf{I}. \tag{48}$$

Thus by expanding

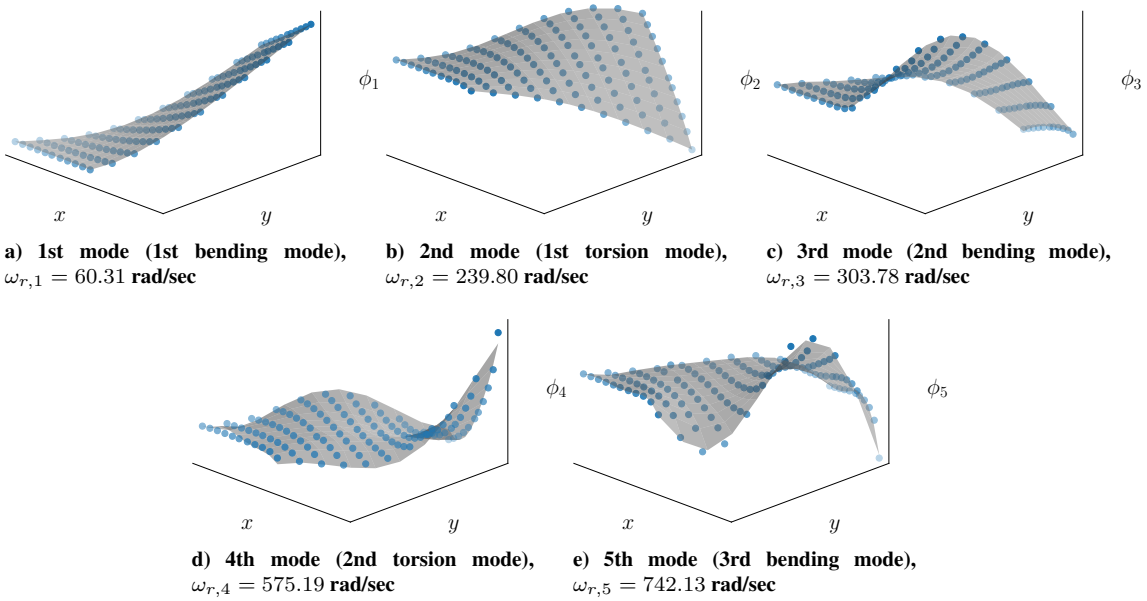
$$(c\sqrt{m_0}\tilde{\boldsymbol{\phi}}^\top) \frac{\mathbf{M}}{m_0} (c\sqrt{m_0}\tilde{\boldsymbol{\phi}}) = c^2 \tilde{\boldsymbol{\phi}}^\top \mathbf{M} \tilde{\boldsymbol{\phi}} = c^2 \begin{bmatrix} 175.127\text{kg} & & \\ & \ddots & \\ & & 175.127\text{kg} \end{bmatrix} = \mathbf{I}. \tag{49}$$

we can find the coefficient to be $c = 1/\sqrt{175.127} = 0.075565$. Thus, to obtain a nondimensional form we use the following scaling,

$$\boldsymbol{\phi} = 0.075565\sqrt{m_0}\tilde{\boldsymbol{\phi}}. \tag{50}$$

where m_0 is the initial weight of the wing.

The first 5 mode shapes $\boldsymbol{\phi}$ are shown in Fig. 4. We use scikit-learn [39] to construct a 4th order polynomial approximation $\hat{\boldsymbol{\phi}}$ for each structural mode. In Figure 4 $\hat{\boldsymbol{\phi}}$ is shown as a gray surfaces which demonstrate acceptable fit with respect to the structural mode shapes, $\boldsymbol{\phi}$, shown as blue dots. We then use $\hat{\boldsymbol{\phi}}$ to evaluate aerodynamic nodal displacements as given in Eq. (35).



a) 1st mode (1st bending mode), $\omega_{r,1} = 60.31$ **rad/sec**

b) 2nd mode (1st torsion mode), $\omega_{r,2} = 239.80$ **rad/sec**

c) 3rd mode (2nd bending mode), $\omega_{r,3} = 303.78$ **rad/sec**

d) 4th mode (2nd torsion mode), $\omega_{r,4} = 575.19$ **rad/sec**

e) 5th mode (3rd bending mode), $\omega_{r,5} = 742.13$ **rad/sec**

Figure 4. First 5 modes of AGARD 445.6 case weakened mode 3 [40]. The coordinates of the blue points are from AGARD report. The gray surfaces are a polynomial regression of those blue points.

Note that in the AGARD report [40], a sixth mode is also included but is ignored here. This is because the sixth mode is a lateral motion (in-plane) mode and its z direction displacement is no longer the dominant motion, which is in contrary with our transfer class assumptions. Further, other work has indicated that the sixth mode is insignificant to flutter boundary prediction [28].

2. *Aerodynamic model*

We generate the geometry based on AGARD report [40] using the open-source package *pyLayout*[b] which is an inhouse built geometry engine. The wing planform is shown in Fig. 5 and the detailed geometry parameters are given in Table 1. The wing airfoil cross section is a NACA 65A004.

[b] https://github.com/mdolab/pylayout.git

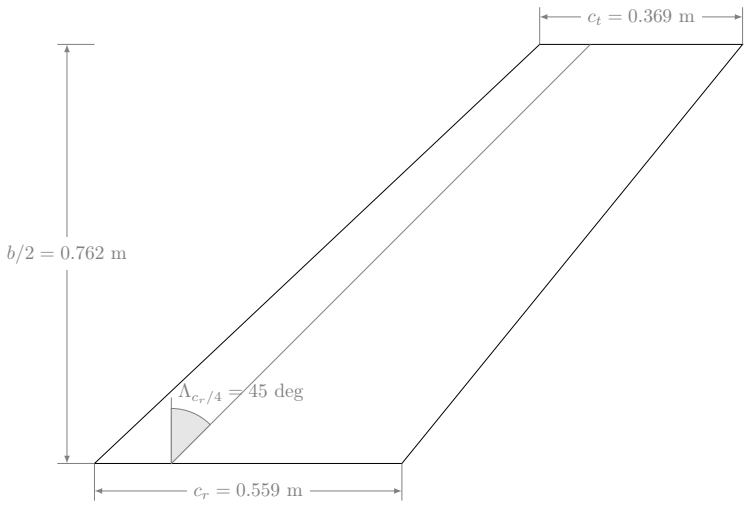


Figure 5. Geometry of AGARD 445.6 case

Table 1. AGARD 445.6 wing geometric properties

Description	Symbol	Value	Unit
Sweep	$\Lambda_{c_r/4}$	45	deg
Aspect ratio	AR	1.65	-
Taper ratio	λ	0.66	-
Semi span	$b/2$	0.762	m
Root chord	c_r	0.559	m
Tip chord	c_t	0.369	m
Area	A	0.353	m^2

The surface mesh is generated by ICEM [41]. We then apply the open-source package *pyHyp*[c], an inhouse hyperbolic mesh generator, to generate the volume mesh from the surface mesh. The mesh we use for the work is a "O" mesh as shown in Fig. 6. In this work a relatively coarse mesh of 11428 elements is used. The tip of the wing is rounded which makes it easier for the solver to reduce the residual.

[c] https://github.com/mdolab/pyhyp.git

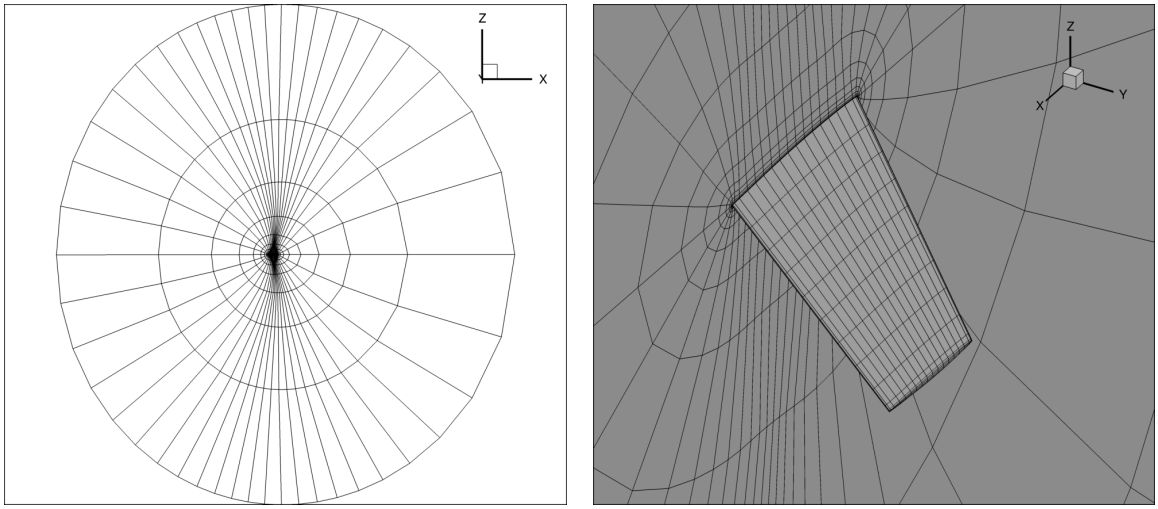


Figure 6. CFD mesh used in this study

B. Flutter boundary results

In this section we compute the flutter boundary of the proposed method and compare it to experimental and other CFD results. The M chosen here to compute the flutter boundary are the same as used in the AGARD report. At first, we compute the flutter onset velocity at $M = 0.499$. We then use this result to initialize the neighboring states for $M = 0.654$ as we expect the solution to be close. We continue with this initialization strategy and then obtain the whole flutter boundary shown in Fig. 7. The transonic dip due to the nonlinear dynamics around $M = 0.95$ is captured.

In the figure, we compare current result using different number of mode shapes with experimental results from Yates [40] and numerical results from Thomas et al. [30] who solved the Euler equations with a harmonic balance approach. The current results with 5 structural modes match better with Thomas et al. CFD results [30] than the experimental results of Yates [40]. This may be caused by the missing viscosity effects in Euler equations. Also, we find that as the number of modes considered increases, in the subsonic regime ($M < 0.7$) the flutter onset velocity is not significantly affected. For high transonic and especially supersonic Mach numbers when more modes are considered, the flutter onset velocity is increases.

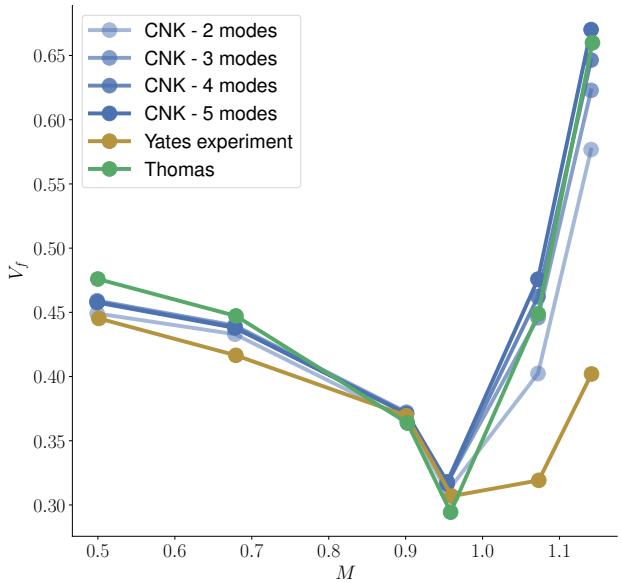


Figure 7. AGARD 445.6 flutter boundary with different structural modes considered. Current results match better with numerical results by Thomas et al. [30] than with experimental results by Yates [40]

The flutter onset frequency is shown in Fig. 8 under different M with the first 5 natural frequencies shown as well. Similar to the transonic dip for flutter velocity, we also observe a flutter frequency decrease around $M = 0.95$. In general the flutter onset frequency is between the 1st bending and 1st torsion structural natural frequencies under all M considered.

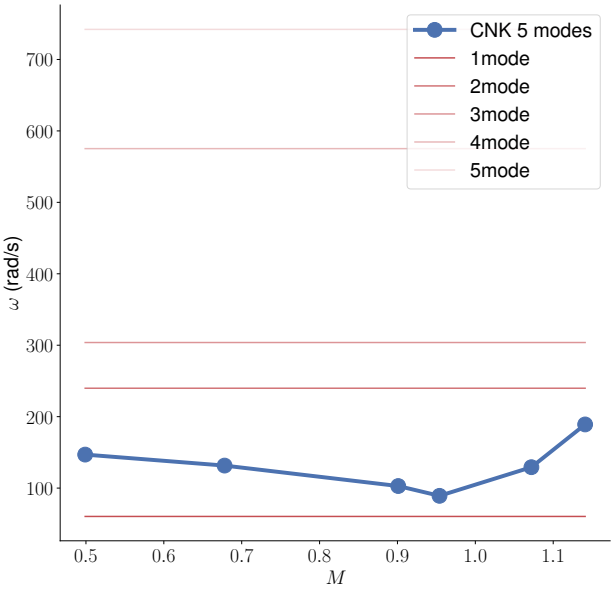


Figure 8. AGARD 445.6 flutter frequency shown as a function of Mach number. The frequency follows a similar trend with the V_f flutter velocity index. There is a dip around $M = 0.95$. For reference the natural frequencies are also shown as red lines.

We present the motion magnitude for different structural modes under all M on the flutter boundary in Fig. 9. The first mode, which is the 1st bending mode, is fixed to 0.01 as expected due to its prescribed motion. Further this mode has the largest amplitude. The second most significant with is the second mode or the 1st torsion mode. This indicates that the wing flutter mode is the classic torsion–bending mode. The 1st torsion mode relative to the 1st bending mode is the weakest around the transonic dip indicating the bending mode is dominant mode at the transonic dip for this case. This is consistent with the fact that at the transonic dip, the flutter onset frequency is closer to the natural frequency of the 1st bending mode than that of the 1st torsion mode.

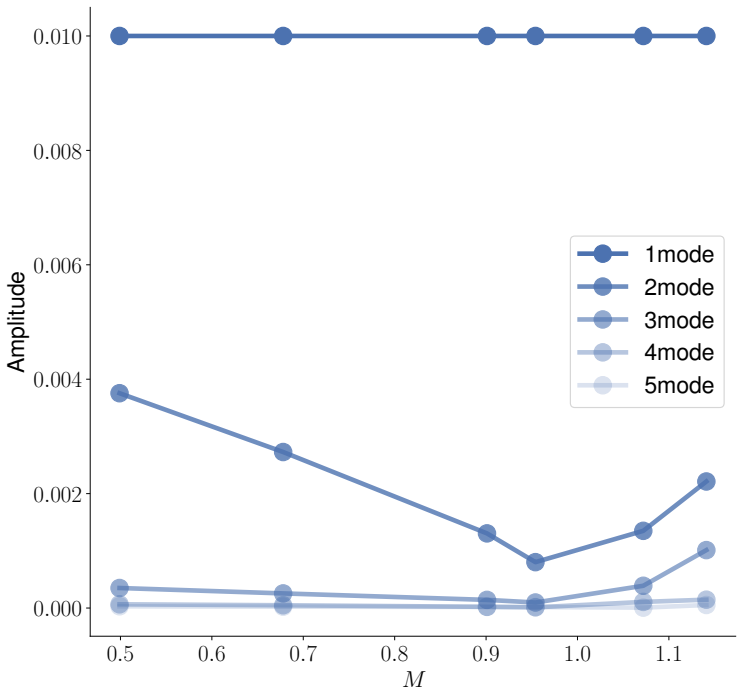


Figure 9. AGARD 445.6 mode amplitudes shown as a function of Mach number. Mode one, or the 1st bending mode, is shown as a constant mode due to its prescribed motion. Mode two is with greater amplitude compared with the other modes.

C. LCO results

The proposed method can also be used to predict LCO behavior using larger prescribed motion amplitudes. We present one case under $M = 1.072$, which is one of the experimental Mach numbers, in Fig. 10. Here the first mode is given a prescribed motion as indicated by the amplitude in the figure and then the flutter speed is solved for. With the LCO points computed, we conduct a linear regression to estimate the flutter onset point. The nature of the LCO bifurcation is very important, where it is preferable to have a supercritical LCO behavior i.e. as the speed is increased the motion magnitude is gradually increased. A subcritical behavior should be avoided as the system responds with a large amplitude LCO when it is perturbed beyond the flutter point. This sub- and supercritial bifurcations are depicted in Fig. 2. In this study Figure 10 demonstrates the preferable supercritical behavior.

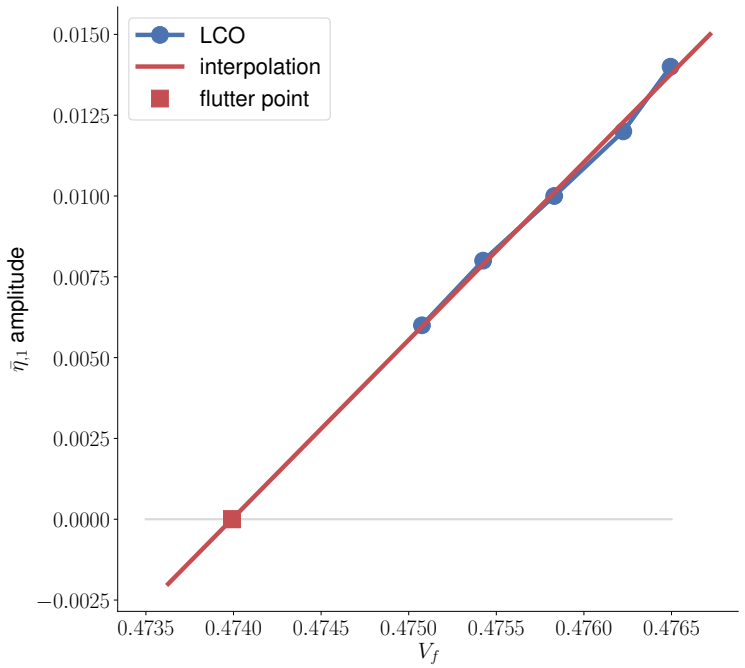


Figure 10. LCO behavior for AGARD 445.6 at $M = 1.072$

V. Conclusion

In this paper, we extend our previous work on airfoil flutter onset prediction to a wing case. The wing structure is described with modal shapes and we propose a nondimensional TS formulation for it. We present a CNK method to compute the flutter onset and LCO behavior of a wing. The proposed method has the adavantage of its better scalability with respect to structural degree of freedom than classic harmonic balance method in literature. The proposed method requires one solution of the coupled system instead of $O(N_{\text{CSD}} \times N_{\text{iter}})$ CFD simulations from the classic harmonic balance method. We predict the flutter boundary of the classic AGARD 445.6 case. The transonic dip of the flutter boundary is captured. We benchmark our results using experimental and numerical results from literature. We also conduct prediction of LCO behaviour and show that the proposed method is able to predict a LCO behaviour is subcritical or supercritical which is an important information for designers. The method is demonstrated on the classic AGARD 445.6 where the flutter boundary is obtained.

References

[1] Kenway, G. K. W., Kennedy, G. J., and Martins, J. R. R. A., "Scalable Parallel Approach for High-Fidelity Steady-State Aeroelastic Analysis and Derivative Computations," *AIAA Journal*, Vol. 52, No. 5, May 2014, pp. 935–951. doi:10.2514/1.J052255.

[2] Kenway, G. K. W. and Martins, J. R. R. A., "Multipoint High-Fidelity Aerostructural Optimization of a Transport Aircraft Configuration," *Journal of Aircraft*, Vol. 51, No. 1, January 2014, pp. 144–160. doi:10.2514/1.C032150.

[3] Kenway, G. W. K. and Martins, J. R. R. A., "High-fidelity aerostructural optimization considering buffet onset," *Proceedings of the 16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Dallas, TX, June 2015, AIAA 2015-2790.

[4] Brooks, T. R., Kenway, G. K. W., and Martins, J. R. R. A., "Benchmark Aerostructural Models for the Study of Transonic Aircraft Wings," *AIAA Journal*, Vol. 56, No. 7, July 2018, pp. 2840–2855. doi:10.2514/1.J056603.

[5] Jonsson, E., Riso, C., Lupp, C. A., Cesnik, C. E. S., Martins, J. R. R. A., and Epureanu, B. I., "Flutter and Post-Flutter Constraints in Aircraft Design Optimization," *Progress in Aerospace Sciences*, 2019. doi:10.1016/j.paerosci.2019.04.001.

[6] Albano, E. and Rodden, W. P., "A doublet-lattice method for calculating lift distributions on oscillating surfaces in subsonic flows." *AIAA Journal*, Vol. 7, No. 2, Februray 1969, pp. 279–285. doi:10.2514/3.5086.

[7] Batina, J. T., Bennett, R. M., Seidel, D. A., Cunningham, H. J., and Bland, S. R., "Recent advances in transonic computational aeroelasticity," *Computers & Structures*, Vol. 30, No. 1, 1988, pp. 29–37.

[8] Batina, J. T., Seidel, D. A., Bland, S. R., and Bennett, R. M., "Unsteady transonic flow calculations for realistic aircraft configurations," *Journal of Aircraft*, Vol. 26, No. 1, 1989, pp. 21–28.

[9] Bennett, R. M., Batina, J. T., and Cunningham, H. J., "Wing-Flutter Calculations With the CAP-TSD Unsteady Transonic Small-Disturbance Program," *Journal of Aircraft*, Vol. 26, No. 9, 1989, pp. 876–882. doi:10.2514/3.45854.

[10] Gibbons, M. D., "Aeroelastic calculations using CFD for a typical business jet model," 1996.

[11] MSC Software Corp., *MSC.Nastran Reference Manual*, 2001.

[12] Neill, D. J., Johnson, E. H., and Canfield, R., "ASTROS - A multidisciplinary automated structural design tool," *Journal of Aircraft*, Vol. 27, No. 12, Dec. 1990, pp. 1021–1027. doi:10.2514/3.45976.

[13] Jonsson, E., Mader, C. A., Kennedy, G. J., and Martins, J. R. R. A., "Computational Modeling of Flutter Constraint for High-Fidelity Aerostructural Optimization," *2019 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, American Institute of Aeronautics and Astronautics, San Diego, CA, January 2019. doi:10.2514/6.2019-2354.

[14] Liu, F., Cai, J., Zhu, Y., Tsai, H., and F. Wong, A., "Calculation of wing flutter by a coupled fluid-structure method," *Journal of Aircraft*, Vol. 38, No. 2, 2001, pp. 334–342.

[15] Opgenoord, M. M., Drela, M., and Willcox, K. E., "Towards a Low-Order Model for Transonic Flutter Prediction," *8th AIAA Theoretical Fluid Mechanics Conference*, American Institute of Aeronautics and Astronautics, June 2017. doi:10.2514/6.2017-4340.

[16] Opgenoord, M. M. J., Drela, M., and Willcox, K. E., "Influence of Transonic Flutter on the Conceptual Design of Next-Generation Transport Aircraft," *AIAA Journal*, March 2019, pp. 1–15. doi:10.2514/1.j057302.

[17] Huang, D., Rokita, T., and Friedmann, P., "Efficient Reduced-Order Modeling for Skin Panels in Hypersonic Flow and Its Application to Generating Aerothermoelastic Scaling Laws," *International Forum on Aeroelasticity and Structural Dynamics*, June 2017.

[18] Li, K., Kou, J., and Zhang, W., "Deep neural network for unsteady aerodynamic and aeroelastic modeling across multiple Mach numbers," *Nonlinear Dynamics*, Vol. 96, No. 3, April 2019, pp. 2157–2177. doi:10.1007/s11071-019-04915-9.

[19] McMullen, M., *The Application of Non-Linear Frequency Domain Methods to the Euler and Navier-Stokes Equations*, Ph.D. thesis, Stanford University, March 2003.

[20] Hall, K. C., Thomas, J. P., and Clark, W. S., "Computation of Unsteady Nonlinear Flows in Cascades Using a Harmonic Balance Technique," *AIAA Journal*, Vol. 40, No. 5, 2015/06/01 2002, pp. 879–886. doi:10.2514/2.1754.

[21] Gopinath, A. and Jameson, A., "Time spectral method for periodic unsteady computations over two-and three-dimensional bodies," 2005.

[22] Kachra, F. and Nadarajah, S. K., "Aeroelastic Solutions Using the Nonlinear Frequency-Domain Method," *AIAA Journal*, Vol. 46, 2008, pp. 9.

[23] McMullen, M. and Jameson, A., "The Computational Efficiency of Non-Linear Frequency Domain Methods," *Journal of Computational Physics*, Vol. 212, No. 2, 2006, pp. 637–661.

[24] Mundis, N. and Mavriplis, D., "Quasi-periodic Time Spectral Method for Aeroelastic Flutter Analysis," 2015/06/01 2013.

[25] Thomas, J. and Dowell, E., "A Fixed Point Iteration Approach for Harmonic Balance Based Aeroelastic Computations," *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, American Institute of Aeronautics and Astronautics, Jan. 2018. doi:10.2514/6.2018-1446.

[26] He, S., Jonsson, E., Mader, C. A., and Martins, J. R. R. A., "A Coupled Newton–Krylov Time Spectral Solver for Flutter Prediction," *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, American Institute of Aeronautics and Astronautics, Kissimmee, FL, January 2018. doi:10.2514/6.2018-2149.

[27] Li, H. and Ekici, K., "A novel approach for flutter prediction of pitch–plunge airfoils using an efficient one-shot method," *Journal of Fluids and Structures*, Vol. 82, Oct. 2018, pp. 651–671. doi:10.1016/j.jfluidstructs.2018.08.012.

[28] Li, H. and Ekici, K., "Aeroelastic Modeling of a Three-Dimensional Wing Using the Harmonic-Balance-Based One-shot Method," *AIAA Scitech 2019 Forum*, American Institute of Aeronautics and Astronautics, January 2019. doi:10.2514/6.2019-0607.

[29] Thomas, J. P., Dowell, E. H., and Hall, K. C., "Nonlinear Inviscid Aerodynamic Effects on Transonic Divergence, Flutter, and Limit-Cycle Oscillations," *AIAA Journal*, Vol. 40, No. 4, apr 2002, pp. 638–646. doi:10.2514/2.1720.

[30] Thomas, J. P., Dowell, E. H., and Hall, K. C., "A Harmonic Balance Approach for Modeling Three-Dimensional Nonlinear Unsteady Aerodynamics and Aeroelasticity," *5th International Symposium on Fluid Structure International, Aeroeslasticity, and Flow Induced Vibration and Noise*, ASME, 2002. doi:10.1115/imece2002-32532.

[31] He, S., Jonsson, E., Mader, C. A., and Martins, J. R. R. A., "Aerodynamic Shape Optimization with Time Spectral Flutter Adjoint," *2019 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, American Institute of Aeronautics and Astronautics, San Diego, CA, January 2019. doi:10.2514/6.2019-0697.

[32] Kenway, G. K. W., Mader, C. A., He, P., and Martins, J. R. R. A., "Effective Adjoint Approaches for Computational Fluid Dynamics," *Progress in Aerospace Sciences*, 2019, (Submitted).

[33] Yildirim, A., Kenway, G. K. W., Mader, C. A., and Martins, J. R. R. A., "A Jacobian-free approximate Newton–Krylov startup strategy for RANS simulations," *Journal of Computational Physics*, 2018, (Submitted).

[34] Eisenstat, S. C. and Walker, H. F., "Choosing the Forcing Terms in an Inexact Newton Method," *SIAM Journal on Scientific Computing*, Vol. 17, No. 1, January 1996, pp. 16–32. doi:10.1137/0917003.

[35] Saad, Y., "A flexible inner-outer preconditioned GMRES algorithm," *SIAM Journal on Scientific Computing*, Vol. 14, No. 2, 1993, pp. 461–469.

[36] Brown, P. N. and Saad, Y., "Hybrid Krylov Methods for Nonlinear Systems of Equations," *SIAM Journal on Scientific and Statistical Computing*, Vol. 11, No. 3, May 1990, pp. 450–481. doi:10.1137/0911026.

[37] Knoll, D. A. and Keyes, D. E., "Jacobian-free Newton–Krylov methods: a survey of approaches and applications," *Journal of Computational Physics*, Vol. 193, No. 2, 2004, pp. 357–397.

[38] Balay, S., Gropp, W. D., McInnes, L. C., and Smith, B. F., "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries," *Modern Software Tools for Scientific Computing*, edited by E. Arge, A. M. Bruaset, and H. P. Langtangen, Birkhäuser Press, 1997, pp. 163–202.

[39] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Rettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.

[40] Yates, E., "AGARD standard aeroelastic configuration for dynamic response, candidate configuration I—Wing 445.6," *NASA TM-100492*, 1987.

[41] Finlayson, M. A., *ANSYS ICEM CFD User's Manual*, ANSYS, Inc., Canonsburg, PA.

Appendix

A. Example for TS CSD equation

In order to aid the understanding of how the modal TS CSD system of equations is formed we present here an example using 3 time instances and 2 modes, i.e. $n = 3$ and $r = 2$. The state variables are

$$\bar{\boldsymbol{\eta}}^n = \frac{1}{b}\begin{bmatrix}\boldsymbol{\eta}_{1,1}\\ \boldsymbol{\eta}_{1,2}\\ \boldsymbol{\eta}_{2,1}\\ \boldsymbol{\eta}_{2,2}\\ \boldsymbol{\eta}_{3,1}\\ \boldsymbol{\eta}_{3,2}\end{bmatrix}. \tag{51}$$

The permutation matrix $\mathbf{P}$ and the time derivative matrix $\mathbf{D}$ for this case is given as

$$\begin{aligned}\mathbf{P} &= \begin{bmatrix}1&0&0&0&0&0\\0&0&1&0&0&0\\0&0&0&0&1&0\\0&1&0&0&0&0\\0&0&0&1&0&0\\0&0&0&0&0&1\end{bmatrix},\\ \mathbf{D} &= \begin{bmatrix}0&-\frac{1}{\sqrt{3}}&\frac{1}{\sqrt{3}}\\ \frac{1}{\sqrt{3}}&0&-\frac{1}{\sqrt{3}}\\ -\frac{1}{\sqrt{3}}&\frac{1}{\sqrt{3}}&0\end{bmatrix}.\end{aligned} \tag{52}$$

The structural dimensionless load is given as

$$\begin{bmatrix}\bar{\mathbf{f}}_{r,1,1}\\ \bar{\mathbf{f}}_{r,1,2}\\ \bar{\mathbf{f}}_{r,2,1}\\ \bar{\mathbf{f}}_{r,2,2}\\ \bar{\mathbf{f}}_{r,3,1}\\ \bar{\mathbf{f}}_{r,3,2}\end{bmatrix} = \frac{1}{\frac{1}{2}\rho_\infty U_\infty^2 S_{\text{ref}}}\begin{bmatrix}\boldsymbol{\phi}^\top\mathbf{f}_1\\ \boldsymbol{\phi}^\top\mathbf{f}_2\\ \boldsymbol{\phi}^\top\mathbf{f}_3\end{bmatrix}, \tag{53}$$

where

$$\boldsymbol{\phi} = \begin{bmatrix}\boldsymbol{\phi}_1 & \boldsymbol{\phi}_2\end{bmatrix}. \tag{54}$$

The resultant TS CSD equations are given as

$$\boldsymbol{\mathcal{S}}_{\text{TS}} = \begin{bmatrix} \boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} & & \\ & \boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} & \\ & & \boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} \end{bmatrix} \left(\frac{\omega^2}{\omega_\alpha^2} \right) \left(\mathbf{P}^\top \begin{bmatrix} \mathbf{D}^2 & \\ & \mathbf{D}^2 \end{bmatrix} \mathbf{P} \right) \frac{1}{b} \begin{bmatrix} \boldsymbol{\eta}_{1,1} \\ \boldsymbol{\eta}_{1,2} \\ \boldsymbol{\eta}_{2,1} \\ \boldsymbol{\eta}_{2,2} \\ \boldsymbol{\eta}_{3,1} \\ \boldsymbol{\eta}_{3,2} \end{bmatrix}$$
$$+ \begin{bmatrix} \boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} & & \\ & \boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} & \\ & & \boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} \end{bmatrix} \begin{bmatrix} \frac{\boldsymbol{\Omega}_r}{\omega_\alpha^2} & & \\ & \frac{\boldsymbol{\Omega}_r}{\omega_\alpha^2} & \\ & & \frac{\boldsymbol{\Omega}_r}{\omega_\alpha^2} \end{bmatrix} \frac{1}{b} \begin{bmatrix} \boldsymbol{\eta}_{1,1} \\ \boldsymbol{\eta}_{1,2} \\ \boldsymbol{\eta}_{2,1} \\ \boldsymbol{\eta}_{2,2} \\ \boldsymbol{\eta}_{3,1} \\ \boldsymbol{\eta}_{3,2} \end{bmatrix} \qquad . \tag{55}$$
$$- \frac{1}{2} \left(\frac{S_{\text{ref}} b}{V_0} \right) V_f^2 \begin{bmatrix} \bar{\mathbf{f}}_{r,1,1} \\ \bar{\mathbf{f}}_{r,1,2} \\ \bar{\mathbf{f}}_{r,2,1} \\ \bar{\mathbf{f}}_{r,2,2} \\ \bar{\mathbf{f}}_{r,3,1} \\ \bar{\mathbf{f}}_{r,3,2} \end{bmatrix}$$

As discussed in Section IV, Section A the mode shapes $\boldsymbol{\phi}$ are normalized such that

$$\boldsymbol{\phi}^\top \frac{\mathbf{M}}{m_0} \boldsymbol{\phi} = \mathbf{I}, \tag{56}$$

so the TS CSD equations can be further reduced to

$$\boldsymbol{\mathcal{S}}_{\text{TS}} = \left(\frac{\omega^2}{\omega_\alpha^2} \right) \left(\mathbf{P}^\top \begin{bmatrix} \mathbf{D}^2 & \\ & \mathbf{D}^2 \end{bmatrix} \mathbf{P} \right) \frac{1}{b} \begin{bmatrix} \boldsymbol{\eta}_{1,1} \\ \boldsymbol{\eta}_{1,2} \\ \boldsymbol{\eta}_{2,1} \\ \boldsymbol{\eta}_{2,2} \\ \boldsymbol{\eta}_{3,1} \\ \boldsymbol{\eta}_{3,2} \end{bmatrix} + \begin{bmatrix} \frac{\boldsymbol{\Omega}_r}{\omega_\alpha^2} & & \\ & \frac{\boldsymbol{\Omega}_r}{\omega_\alpha^2} & \\ & & \frac{\boldsymbol{\Omega}_r}{\omega_\alpha^2} \end{bmatrix} \frac{1}{b} \begin{bmatrix} \boldsymbol{\eta}_{1,1} \\ \boldsymbol{\eta}_{1,2} \\ \boldsymbol{\eta}_{2,1} \\ \boldsymbol{\eta}_{2,2} \\ \boldsymbol{\eta}_{3,1} \\ \boldsymbol{\eta}_{3,2} \end{bmatrix} - \frac{1}{2} \left(\frac{S_{\text{ref}} b}{V_0} \right) V_f^2 \begin{bmatrix} \bar{\mathbf{f}}_{r,1,1} \\ \bar{\mathbf{f}}_{r,1,2} \\ \bar{\mathbf{f}}_{r,2,1} \\ \bar{\mathbf{f}}_{r,2,2} \\ \bar{\mathbf{f}}_{r,3,1} \\ \bar{\mathbf{f}}_{r,3,2} \end{bmatrix} . \tag{57}$$