Aerodynamic Shape Optimization with Time Spectral Flutter Adjoint

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 are an important consideration in commercial airliner design. Previous work with high-fidelity aerostructural optimization has shown a tendency for optimization algorithms to produce unrealistically high span designs, particularly when maximizing range or minimizing fuel burn, in an effort to maximize aerodynamic efficiency. In order to constraint this tendency, we propose to include a flutter constraint to these optimization problems. To be meaningful for this class of optimization problem, the flutter model should be able to predict the strong nonlinearity in the transonic regime, as commercial airliners largely operate in this regime. In addition, these problems typically include a large number of design variables, therefore we use gradient-based optimization algorithms, requiring the constraint formulation to be continuous, differentiable and have an efficient method for computing the gradients of the constraint. In this paper, we apply Euler time-spectral computational fluid dynamics methods to model the flutter constraint and we propose a coupled adjoint method to calculate the constraint sensitivity with respect to the design variables. The coupled adjoint method has the advantage that the gradient evaluation time is independent of the number of design variables. In the literature, the harmonic balance method has been used to model flutter constraints. A two-level adjoint formulation has been proposed to evaluate the sensitivity of flutter onset velocity with respect to the design variables. Such method requires solving the harmonic balance aerodynamic adjoint $\mathcal{O}(N_{CSD})$ times where N_{CSD} is the structural degree of freedom multiplied by number of time instances. On the contrary, we propose the coupled adjoint method which directly deals with the whole aeroelastic system and solves only 1 adjoint equation. We verify the adjoint sensitivity computation with the finite difference method. Finally, we present a MDO problem for the classic Isogai case in which we maximize the flutter velocity index with respect to aerodynamic shape design variables. We gain a 10.9% increase of the flutter velocity index through this optimization.

Nomenclature

$\mathcal{A}$	=	aerodynamic residual
α	=	pitching
$\alpha_{\text{1st mode}}$	=	the dominant pitching motion
$\alpha_1, \alpha_2 ...$	=	pitching for certain time instance
α^N	=	pitching history
b	=	airfoil half chord length $0.5c$
c	=	airfoil chord length
$c_0, c_1, ...$	=	pitching motion Fourier coefficients
$c_{1,r}, c_{1,i}, ...$	=	real and imaginary parts of certain pitching motion mode
C_c	=	coefficient for cos function in the dominant pitching mode
C_s	=	coefficient for sin function in the dominant pitching mode
$\boldsymbol{D}_Q$	=	second order temporal derivative matrix
f_A^n	=	aerodynamic nodal load
$\bar{f}^n$	=	dimensionless structural load
ϕ	=	structural residual adjoint
γ	=	heat capacity ratio
I	=	function of interest

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

J	=	the Jacobian matrix for the time spectral flutter equation residual with respect to state variables $\partial\mathcal{R}/\partial q$
k	=	constant $1, 2, ...$
$\boldsymbol{K}$	=	stiffness matrix
χ_{mag}	=	prescribed motion magnitude residual adjoint
χ_{pha}	=	prescribed motion phase residual adjoint
M	=	Mach number
$\boldsymbol{M}$	=	mass matrix
mag	=	dominant pitching motion magnitude
n	=	number of time instances
N	=	number of 3D grid elements
ω	=	flutter angular velocity
P	=	flutter adjoint preconditioner
$P_{\mathcal{A}}$	=	aerodynamic adjoint preconditioner
$P_{mot,\mathcal{S}}$	=	prescribed motion and structural residual preconditioner
P_∞	=	boundary static pressure
pha	=	dominant pitching motion phase
q	=	flutter states
R	=	universal gas constant
$\mathcal{R}$	=	flutter residual
$\mathcal{R}_{mag}$	=	prescribed motion magnitude residual
$\mathcal{R}_{pha}$	=	prescribed motion phase residual
ρ_{BC}	=	boundary static density
ρ_∞	=	boundary static density
$\mathcal{S}$	=	time spectral structural residual
ψ	=	aerodynamic residual adjoint
$\Psi_{\mathcal{A}}$	=	aerodynamic residual adjoint, ψ
$\Psi_{\mathcal{R}_{mot},\mathcal{S}}$	=	prescribed motion and structural residual adjoint, $\xi_{mag}, \xi_{pha}, \phi$
T_{BC}	=	boundary static temperature
T_∞	=	boundary static temperature
$\tau_{\mathcal{A}}$	=	aerodynamic adjoint solution of the preconditioned system of equations
$\tau_{mot,\mathcal{S}}$	=	motion and structural adjoint solution of the preconditioned system of equations
u^n	=	structural states
V_f	=	flutter velocity index
w_h, w_α	=	uncoupled natural frequencies of typical section in plunge and pitch respectively
x	=	(aerodynamic shape) design variables
$X_{S,0}$	=	undeformed aerodynamic surface coordinates
X_S^n	=	deformed aerodynamic surface coordinates
X_V^n	=	deformed aerodynamic volume coordinates
$X_{\mathcal{R}_{mag}}, X_{\mathcal{R}_{pha}}, X_{\mathcal{S}}, X_{\mathcal{A}}$	=	prescribed motion magnitude, phase, structural residual and aerodynamic residual seed
$X_{F_{\mathcal{A}^n}}, X_{X_S^n}$	=	aerodynamic nodal load and aerodynamic surface coordinate seed
$X_T, X_\rho, X_{T_0}, X_{T_\infty}$	=	static temperature, static density, time period and static temperature seed
y	=	FFD control points y coordinates
y_{min}, y_{max}	=	FFD control points y coordinates lower and upper bounds respectively
ζ^n	=	aerodyanmic states

I. Introduction

High-fidelity computational modeling and optimization of complex engineering systems has the potential to allow engineers to produce more efficient designs with fewer unforeseen design modifications late in the design process. However, in order to achieve this goal, the modeling and optimization methods need to include sufficient relevant physics to capture the important design drivers in the design space. As shown in Figure 1, previous work on aerostructural optimization has shown a tendency to produce unrealistically large spans, particularly when maximizing range or minimizing fuel burn. Since such a configuration is likely to be prone to flutter, due to its increased flexibility, inclusion of a flutter constraint becomes an important consideration in the design problem.

Adding flutter constraint to CFD-based optimization is a challenging problem. First, we need to solve for the flutter condition with a given geometry. This is the reverse of the typical process of simulating the aeroelastic response

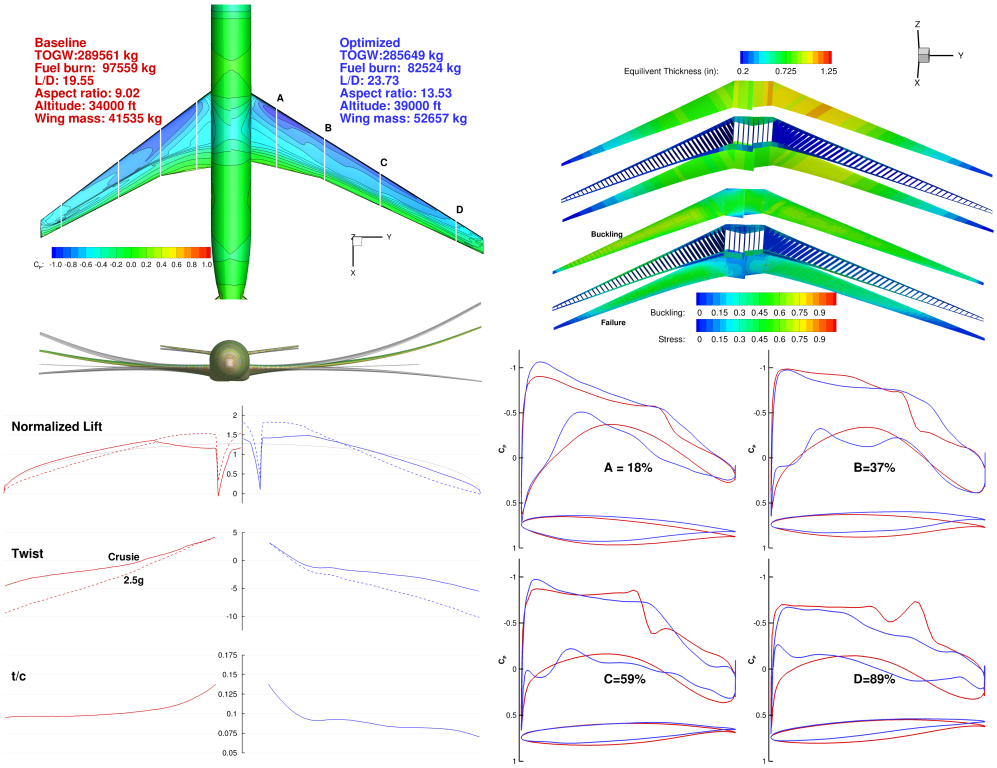


Figure 1: Aerostructural optimization result [1]: C_p and planform comparison with initial design (upper left); equivalent thickness distribution, stress and buckling KS failure criteria (upper right); comparison of initial and optimized lift distributions, twist distributions and thickness to chord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

of a given geometry under certain boundary conditions, for which numerical methods are mature. In this approach, we specify the aeroelastic response and solve for the conditions that produced that response. Second, to leverage gradient-based optimization's ability to handle large number of design variables, we need to find an efficient method to calculate the sensitivity of the flutter velocity index with respect to design variables from multiple disciplines.

The contribution of current paper is to deal with the challenge related with the gradient evaluation. We develop the formulation of the time-spectral flutter adjoint for gradient evaluation. By applying the adjoint method, the gradient evaluation time is independent of the number of design variables. This is especially beneficial for aerodynamic shape optimization problems which usually involve hundreds of variables. We extend the MACH framework of Kenway et al. [2, 3], which was originally developed for aerostructural problems, to the current aeroelastic problem. This work extends our previous work [4] which focused on the development of the flutter analysis used in this work.

II. Background

For the flutter analysis method, we limit the survey to only time-spectral and harmonic-balance methods since flutter analysis is not the focus of the current paper. For gradient evaluation methods, we cover topics with a broader scope. For more detailed references on flutter analysis and gradient evaluation methods, we refer the reader to a recent review paper of Jonsson et al. [5].

Time-spectral and harmonic-balance methods have the advantage that they can capture the aerodynamic nonlinearity in the transonic flow regime with a relatively low cost compared with unsteady CFD. This has been shown by several authors including Hall et al. [6], Gopinath and Jameson [7] and McMullen and Jameson [8]. This class of methods has been extended to compute flutter limits. In particular, Thomas et al. [9] apply Newton–Ralphson method to solve for flutter onset condition. More recently, Thomas and Dowell [10] apply fixed point iteration method to solve similar problem. He et al. [4] extend that work by developing a full space Newton–Krylov method. Li and Ekici [11] present a one-shot method for the flutter analysis. Prasad et al. [12] propose an alternative energy constraint instead of the small motion constraint proposed in [9] to capture the flutter onset condition. Yao and Marques [13] apply a pseudo-time stepping strategy to solve the harmonic balance flutter equation.

For flutter sensitivity evaluation method, there are multiple methods. There are non-CFD based approaches and CFD and low-fidelity mixed approaches. Stanford et al. [14] propose a pk method with nonlinear Euler solver and time-linearized transonic small disturbance (TSD) analysis. However, the sensitivity computation of the mode shape is ignored (i.e. fixed mode approach) which may cause issues when considering planform variables. Chen et al. [15] propose a method using an Euler CFD solver and a boundary layer code. The sensitivity is evaluated by complex-step method. The computational cost of this method scales with the number of design variables which makes it impractical for large number of design variables found in practical design problems. Bartels and Stanford [16] solve a structural optimization problem with the flutter constraint computed by eigenvalue analysis. Kennedy et al. [17] and Beran et.al [18] developed an adjoint for the flutter constraint which is formulated with Hopf bifurcation. Recently, Jonsson et al. [19] propose an adjoint method with an enhanced pk method which is able to track the change of flutter modes.

There is a handful of CFD-based methods for flutter analysis with sensitivities in the literature. Zhang et al. [20] formulate a time-accurate adjoint for flutter analysis with a high-fidelity Euler solver. Recently, Kiviaho et al. [21] develop a time-accurate adjoint by matrix-pencil method. Leveraging the efficient harmonic balance solver, Thomas and Dowell [22, 23] propose a harmonic balance adjoint for flutter. In that work, for every flutter sensitivity calculation, $\mathcal{O}(N_{CSD})$ CFD adjoint equations are solved. In the current work, we propose a coupled adjoint formulation which for each gradient evaluation only $\mathcal{O}(1)$ adjoint solution is required at a cost of solving a larger set of coupled equations.

III. Time Spectral Flutter Analysis

The time spectral flutter equation is given in Equation 1,

$$\mathcal{R}(q) := \begin{bmatrix} \mathcal{R}_{mag} \\ \mathcal{R}_{pha} \\ \mathcal{S} \\ \mathcal{A} \end{bmatrix}, q := \begin{bmatrix} V_f \\ \omega \\ u^n \\ \zeta^n \end{bmatrix}, \tag{1}$$

where V_f is flutter velocity index, ω is the flutter frequency, u^n is the displacement history and ζ^n is the aerodynamic states history. $\mathcal{R}_{mag}, \mathcal{R}_{mag}$ are the constraints for prescribed motion magnitude and phase respectively, $\mathcal{S}$ is the time spectral structural dynamic constraint and $\mathcal{A}$ is the time spectral aerodynamic constraint. In the method, a small pitching motion is prescribed, a flutter solution is found which has the corresponding pitching motion magnitude and phase. For a detailed description, we refer the readers to [4]. This set of equation is proposed in [22].

IV. Coupled Adjoint Derivative Computation

A. Coupled Adjoint Overview

The function (e.g. V_f) gradients with respect to design variables are important design information. We apply adjoint method to evaluate the sensitivity. For more general treatment for adjoint method, we refer the readers to Martins and Hwang [24]. The total derivatives of the function of interest with respect to the design variables are given in Equation 2. We set I as the objective function and x as the design variables. Other state variables are defined in Equation 1.

$$\begin{aligned} \frac{dI}{dx} &= \frac{\partial I}{\partial x} + \frac{\partial I}{\partial q}\frac{dq}{dx} \\ &= \frac{\partial I}{\partial x} + \begin{bmatrix} \frac{\partial I}{\partial V_f} & \frac{\partial I}{\partial \omega} & \frac{\partial I}{\partial u^n} & \frac{\partial I}{\partial \zeta^n} \end{bmatrix} \begin{bmatrix} \frac{dV_f}{dx} \\ \frac{d\omega}{dx} \\ \frac{du^n}{dx} \\ \frac{d\zeta^n}{dx} \end{bmatrix} \end{aligned} \tag{2}$$

The total derivatives of the state variables with respect to the design variables satisfies the Equation 3. This is based on the fact that no matter what values we set for the design variables, the residual should be zero for a physical solution.

$$\begin{gathered} \frac{d\mathcal{R}}{dx} = \frac{\partial \mathcal{R}}{\partial x} + \frac{\partial \mathcal{R}}{\partial q}\frac{dq}{dx} = 0 \\ \begin{bmatrix} \frac{d\mathcal{R}_{mag}}{dx} \\ \frac{d\mathcal{R}_{pha}}{dx} \\ \frac{d\mathcal{S}}{dx} \\ \frac{d\mathcal{A}}{dx} \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathcal{R}_{mag}}{\partial x} \\ \frac{\partial \mathcal{R}_{pha}}{\partial x} \\ \frac{\partial \mathcal{S}}{\partial x} \\ \frac{\partial \mathcal{A}}{\partial x} \end{bmatrix} + \begin{bmatrix} \frac{\partial \mathcal{R}_{mag}}{\partial V_f} & \frac{\partial \mathcal{R}_{mag}}{\partial \omega} & \frac{\partial \mathcal{R}_{mag}}{\partial u^n} & \frac{\partial \mathcal{R}_{mag}}{\partial \zeta^n} \\ \frac{\partial \mathcal{R}_{pha}}{\partial V_f} & \frac{\partial \mathcal{R}_{pha}}{\partial \omega} & \frac{\partial \mathcal{R}_{pha}}{\partial u^n} & \frac{\partial \mathcal{R}_{pha}}{\partial \zeta^n} \\ \frac{\partial \mathcal{S}}{\partial V_f} & \frac{\partial \mathcal{S}}{\partial \omega} & \frac{\partial \mathcal{S}}{\partial u^n} & \frac{\partial \mathcal{S}}{\partial \zeta^n} \\ \frac{\partial \mathcal{A}}{\partial V_f} & \frac{\partial \mathcal{A}}{\partial \omega} & \frac{\partial \mathcal{A}}{\partial u^n} & \frac{\partial \mathcal{A}}{\partial \zeta^n} \end{bmatrix} \begin{bmatrix} \frac{dV_f}{dx} \\ \frac{d\omega}{dx} \\ \frac{du^n}{dx} \\ \frac{d\zeta^n}{dx} \end{bmatrix} = 0 \end{gathered} \tag{3}$$

Combining Equation 2 and Equation 3, we obtain Equation 4. By associating $(\partial\mathcal{R}/\partial q)^{-\top}$ with $\partial I/\partial q$ and solving for Ψ, we get the adjoint equation. This form of the solution has the advantage that the number of linear solutions required to get the total derivatives scales with the dimension of the function of interest I rather than the dimension of the design variables. Multiplying Ψ with $\partial\mathcal{R}/\partial x$ scales with the dimension of design variables, but it only requires matrix vector products which are much cheaper to compute than a full linear solution. This is an advantage in aerodynamic shape design problems, since we typically have few functions of interest but hundreds of design variables. If there are more functions of interest than design variables, we should do the opposite: associating $(\partial\mathcal{R}/\partial q)^{-1}$ with $\partial\mathcal{R}/\partial x$.

$$\frac{dI}{dx} = \frac{\partial I}{\partial x} - \underbrace{\begin{bmatrix} \frac{\partial I}{\partial V_f} & \frac{\partial I}{\partial \omega} & \frac{\partial I}{\partial u^n} & \frac{\partial I}{\partial \zeta^n} \end{bmatrix} \begin{bmatrix} \frac{\partial\mathcal{R}_{mag}}{\partial V_f} & \frac{\partial\mathcal{R}_{mag}}{\partial \omega} & \frac{\partial\mathcal{R}_{mag}}{\partial u^n} & \frac{\partial\mathcal{R}_{mag}}{\partial \zeta^n} \\ \frac{\partial\mathcal{R}_{pha}}{\partial V_f} & \frac{\partial\mathcal{R}_{pha}}{\partial \omega} & \frac{\partial\mathcal{R}_{pha}}{\partial u^n} & \frac{\partial\mathcal{R}_{pha}}{\partial \zeta^n} \\ \frac{\partial\mathcal{S}}{\partial V_f} & \frac{\partial\mathcal{S}}{\partial \omega} & \frac{\partial\mathcal{S}}{\partial u^n} & \frac{\partial\mathcal{S}}{\partial \zeta^n} \\ \frac{\partial\mathcal{A}}{\partial V_f} & \frac{\partial\mathcal{A}}{\partial \omega} & \frac{\partial\mathcal{A}}{\partial u^n} & \frac{\partial\mathcal{A}}{\partial \zeta^n} \end{bmatrix}^{-1}}_{\Psi} \begin{bmatrix} \frac{\partial\mathcal{R}_{mag}}{\partial x} \\ \frac{\partial\mathcal{R}_{pha}}{\partial x} \\ \frac{\partial\mathcal{S}}{\partial x} \\ \frac{\partial\mathcal{A}}{\partial x} \end{bmatrix} \tag{4}$$

Separating out the adjoint equation from Equation 4, we get Equation 5 which is the the time-spectral flutter adjoint equation.

$$\begin{bmatrix} \frac{\partial\mathcal{R}_{mag}}{\partial V_f} & \frac{\partial\mathcal{R}_{mag}}{\partial \omega} & \frac{\partial\mathcal{R}_{mag}}{\partial u^n} & \frac{\partial\mathcal{R}_{mag}}{\partial \zeta^n} \\ \frac{\partial\mathcal{R}_{pha}}{\partial V_f} & \frac{\partial\mathcal{R}_{pha}}{\partial \omega} & \frac{\partial\mathcal{R}_{pha}}{\partial u^n} & \frac{\partial\mathcal{R}_{pha}}{\partial \zeta^n} \\ \frac{\partial\mathcal{S}}{\partial V_f} & \frac{\partial\mathcal{S}}{\partial \omega} & \frac{\partial\mathcal{S}}{\partial u^n} & \frac{\partial\mathcal{S}}{\partial \zeta^n} \\ \frac{\partial\mathcal{A}}{\partial V_f} & \frac{\partial\mathcal{A}}{\partial \omega} & \frac{\partial\mathcal{A}}{\partial u^n} & \frac{\partial\mathcal{A}}{\partial \zeta^n} \end{bmatrix}^{\top} \begin{bmatrix} \chi_{mag} \\ \chi_{pha} \\ \phi \\ \psi \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial V_f} & \frac{\partial I}{\partial \omega} & \frac{\partial I}{\partial u^n} & \frac{\partial I}{\partial \zeta^n} \end{bmatrix}^{\top}. \tag{5}$$

The total derivative can be simplified by introducing the adjoint variables into Equation 4 which results in Equation 6,

$$\frac{dI}{dx} = \frac{\partial I}{\partial x} - \chi_{mag}^{\top}\frac{\partial\mathcal{R}_{mag}}{\partial x} - \chi_{pha}^{\top}\frac{\partial\mathcal{R}_{pha}}{\partial x} - \phi^{\top}\frac{\partial\mathcal{S}}{\partial x} - \psi^{\top}\frac{\partial\mathcal{A}}{\partial x}. \tag{6}$$

In this work, the function of primary interest is $I = V_f$. Taking into account the fact that $\mathcal{R}_{mag} = \mathcal{R}_{mag}(u^n), \mathcal{R}_{pha} = \mathcal{R}_{pha}(u^n)$, we have

$$\frac{dI}{dx} = -\phi^{\top}\frac{\partial\mathcal{S}}{\partial x} - \psi^{\top}\frac{\partial\mathcal{A}}{\partial x}, \tag{7}$$

with the adjoint equation

$$\begin{bmatrix} 0 & 0 & \left(\frac{\partial\mathcal{S}}{\partial V_f}\right)^{\top} & \left(\frac{\partial\mathcal{A}}{\partial V_f}\right)^{\top} \\ 0 & 0 & \left(\frac{\partial\mathcal{S}}{\partial \omega}\right)^{\top} & \left(\frac{\partial\mathcal{A}}{\partial \omega}\right)^{\top} \\ \left(\frac{\partial\mathcal{R}_{mot,mag}}{\partial u^n}\right)^{\top} & \left(\frac{\partial\mathcal{R}_{mot,pha}}{\partial u^n}\right)^{\top} & \left(\frac{\partial\mathcal{S}}{\partial u^n}\right)^{\top} & \left(\frac{\partial\mathcal{A}}{\partial u^n}\right)^{\top} \\ 0 & 0 & \left(\frac{\partial\mathcal{S}}{\partial \zeta^n}\right)^{\top} & \left(\frac{\partial\mathcal{A}}{\partial \zeta^n}\right)^{\top} \end{bmatrix} \begin{bmatrix} \chi_{mag} \\ \chi_{pha} \\ \phi \\ \psi \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{8}$$

B. Coupled Adjoint Implementation

Equation 5 says nothing about the solution methodology. There are many ways to solve for the adjoint equation. The linear Gauss–Seidel method proposed in Kenway et al. [2] and Martins et al. [25]; the coupled Krylov adjoint solver [2]; the Monte Carlo method proposed in Wang et al. [26] which is developed mainly for unsteady adjoint though. For the current work, we apply a coupled Krylov method, since it has been demonstrated by Kenway et al. in [2] to be computationally more efficient than the linear Gauss–Seidel method.

One key component for the coupled Krylov is the matrix–vector products between the transpose of Jacobian matrix with certain seeds. To compute this accurately and efficiently, we apply the reverse AD method, which is precise up to machine precision, following Mader and Martins [27].

The coupled adjoint and function sensitivity evaluation involves four components as shown in Equation 2 and 3. $(\partial\mathcal{R}/\partial q)^{\top}\Psi$, $(\partial\mathcal{R}/\partial x)^{\top}\Psi$, $\partial I/\partial q$ and $\partial I/\partial x$. In this work, $\partial I/\partial q$ is a simple constant vector and $\partial I/\partial x$ is simply zero. We focus on the other two components in this section.

1. *Aerodynamic Residual Partial Derivatives*

The partial derivative of aerodynamic residual with respect to the design variables multiplied with an aerodynamic residual seed is expanded as

$$\left(\frac{\partial \mathcal{A}}{\partial x}\right)^\top \psi = \left(\frac{\partial X_{S,0}}{\partial x}\right)^\top \left(\frac{\partial X_S^n}{\partial X_{S,0}}\right)^\top \left(\frac{\partial X_V^n}{\partial X_S^n}\right)^\top \left(\frac{\partial \mathcal{A}^n}{\partial X_V^n}\right)^\top \psi, \tag{9}$$

where x represents design variables, $X_{S,0}$ represents the aerodynamic surface coordinates without structural displacement, X_S^n represents the aerodynamic surface coordinates with structural displacement for n time instances, X_V^n represents the deformed volume coordinates for n time instances. Notice that the $(\partial \mathcal{A}^n/\partial X_V^n)^\top$ is coupled for different time instances. On the contrary, $(\partial X_V^n/\partial X_S^n)^\top$ is decoupled for different time instances. $(\partial \mathcal{A}^n/\partial X_V^n)^\top$ is implemented by Mader et al. [28]. For each time instance from $(\partial X_V^n/\partial X_S^n)^\top$, we apply the reverse AD code developed by Kenway and Martins [29]. The original mesh deformation method is proposed by Luke et al. [30] which scales with $\mathcal{O}(Nlog(N))$ where N is the number of 3D elements.

The matrix–vector multiplication between aerodynamic residual partial derivatives with respect to structural displacement u^n and aerodynamic residual seeds is given as

$$\left(\frac{\partial \mathcal{A}}{\partial u^n}\right)^\top \psi = \left(\frac{\partial X_S^n}{\partial u^n}\right)^\top \left(\frac{\partial X_V^n}{\partial X_S^n}\right)^\top \left(\frac{\partial \mathcal{A}}{\partial X_V^n}\right)^\top \psi. \tag{10}$$

Similar with $(\partial X_V^n/\partial X_S^n)^\top$, $\partial X_S^n/\partial u^n$ is decoupled between time instances. The $(\partial \mathcal{A}/\partial u^n)^\top$ is dense – each displacement will affect all the aerodynamic residuals. This is observed in Figure 3 – the dense columns on the left side. The displacement affects the aerodynamic residual within its own time instance. It affects other time instances by affecting the spectral interpolated grid velocity.

The matrix–vector product between the aerodynamic residual partial derivative with respect to the flutter velocity index and an aerodynamic seed is given as

$$\left(\frac{\partial \mathcal{A}}{\partial V_f}\right)^\top \psi = \left(\frac{dT_\infty}{dV_f}\right)^\top \left(\frac{\partial \mathcal{A}}{\partial T_\infty}\right)^\top \psi, \tag{11}$$

where T_∞ is the boundary temperature. We are not free to pick T_∞, P_∞ and ρ_∞ all together, because the three variables are related with each other by the ideal gas law. In our simulation, we set pressure P_∞ as a constant and temperature as a variable determined by V_f through Equation 12. Then the density will be dependent on T_∞. In the notation, we make the distinction between the "∞" and "BC" subscriptions as the former represents the physical boundary condition satisfying the ideal gas law and the latter represents the one with free variables not constrained by the ideal gas law. As for the ADflow CFD solver, the solver does not enforce the ideal gas law – it is assumed that the boundary condition provided to the solver satisfies the ideal gas law.

$$\frac{dT_\infty}{dV_f} = \frac{2b^2 w_\alpha^2 \mu}{M^2 \gamma R} V_f \tag{12}$$

Since $P_\infty, T_\infty, \rho_\infty$ needs to satisfy the general gas law, by changing V_f, the boundary density will also be changed. This relationship is reflected in the following equations

$$\frac{\partial \mathcal{A}}{\partial T_\infty} = \frac{\partial \mathcal{A}}{\partial T_{BC}}\frac{\partial T_{BC}}{\partial T_\infty} + \frac{\partial \mathcal{A}}{\partial \rho_{BC}}\frac{\partial \rho_{BC}}{\partial T_\infty} \tag{13}$$

where,

$$\begin{aligned} \frac{\partial T_{BC}}{\partial T_\infty} &= 1, \\ \frac{\partial \rho_{BC}}{\partial T_\infty} &= -\frac{p_\infty}{R T_\infty^2}. \end{aligned} \tag{14}$$

The partials for angular velocity ω are given by

$$\begin{aligned} \left(\frac{\partial \mathcal{A}}{\partial \omega}\right)^\top \psi &= \left(\frac{\partial T_0}{\partial \omega}\right)^\top \left(\frac{\partial \mathcal{A}}{\partial T_0}\right)^\top \psi, \\ &= -\frac{2\pi}{\omega^2}\left(\frac{\partial \mathcal{A}}{\partial T_0}\right)^\top \psi, \end{aligned} \tag{15}$$

where T_0 is the time period. There is a new implementation related with $(\partial\mathcal{A}/\partial T_0)^\top$. With spectral interpolated grid velocity [4], the grid velocity will be dependent on the time period. This contributes the $(\partial\mathcal{A}/\partial T_0)^\top$.

There is one last partial: $\partial\mathcal{A}/\partial\zeta^n$ which has already been developed and discussed in detail by Mader and Martins [28].

2. *Structural Residual Partial Derivatives*

The expression for CSD equations is shown in Equation 16. For the detailed expression, we refer the readers to [4].

$$\mathcal{S} = \boldsymbol{M}\boldsymbol{D}_Q(\omega)u^n + \boldsymbol{K}u^n - \frac{V_f^2}{\pi}\bar{f}^n(f_A^n, X_S^n), \tag{16}$$

where $\bar{f}^n$ is the dimensionless structural dynamic load $((C_l, C_m)^n)^\top$, $\boldsymbol{M}$ is the mass matrix, $\boldsymbol{D}_Q$ is the second order time derivative matrix, and $\boldsymbol{K}$ is the stiffness matrix. Notice that $f_A^n = f_A^n(\zeta^n, X_S^n)$, $X_S^n = X_S^n(u^n, X_{S,0})$. The dependency of $\bar{f}$ with X_S^n is due to the load for this problem. For a general 3D case, this dependency may not appear.

The partial derivatives of structural residual with respect to design variables are given as:

$$\left(\frac{\partial\mathcal{S}}{\partial x}\right)^\top\phi = \left(\frac{\partial X_{S,0}}{\partial x}\right)^\top\left(\frac{\partial X_S^n}{\partial X_{S,0}}\right)^\top\left(\left(\frac{\partial X_V^n}{\partial X_S^n}\right)^\top\left(\frac{\partial f_A^n}{\partial X_V^n}\right)^\top\left(\frac{\partial\bar{f}^n}{\partial f_A^n}\right)^\top + \left(\frac{\partial\bar{f}^n}{\partial X_S^n}\right)^\top\right)\left(\frac{\partial\mathcal{S}}{\partial\bar{f}^n}\right)^\top\phi. \tag{17}$$

It is noted that $(\partial f_{\mathcal{A}}^n/\partial X_V^n)^\top$ $\left(\partial\bar{f}^n/\partial f_{\mathcal{A}}^n\right)^\top$ and $\left(\partial\bar{f}^n/\partial X_S^n\right)^\top$ decoupled for different time instances.

The partial derivatives of structural residual with respect to aerodynamic state variables are given as:

$$\begin{aligned}\left(\frac{\partial\mathcal{S}}{\partial\zeta^n}\right)^\top\phi &= \left(\frac{\partial\bar{f}^n}{\partial\zeta^n}\right)^\top\left(\frac{\partial\mathcal{S}}{\partial\bar{f}^n}\right)^\top\phi,\\ &= \left(\frac{\partial f_A^n}{\partial\zeta^n}\right)^\top\left(\frac{\partial\bar{f}^n}{\partial f_A^n}\right)^\top\left(\frac{\partial\mathcal{S}}{\partial\bar{f}^n}\right)^\top\phi.\end{aligned} \tag{18}$$

Notice that $\left(\partial\bar{f}^n/\partial f_{\mathcal{A}}^n\right)^\top$ is decoupled between different time instances.

The partial derivatives of structural residual with respect to structural state variables are given as:

$$\begin{aligned}\left(\frac{\partial\mathcal{S}}{\partial u^n}\right)^\top\phi &= (\boldsymbol{M}\boldsymbol{D}_Q(\omega)+\boldsymbol{K})^\top\phi + \left(\frac{\partial\bar{f}^n}{\partial u^n}\right)^\top\left(\frac{\partial\mathcal{S}}{\partial\bar{f}^n}\right)^\top\phi\\ &= (\boldsymbol{M}\boldsymbol{D}_Q(\omega)+\boldsymbol{K})^\top\phi + \left(\frac{\partial X_S^n}{\partial u^n}\right)^\top\left(\left(\frac{\partial X_V^n}{\partial X_S^n}\right)^\top\left(\frac{\partial f_A^n}{\partial X_V^n}\right)^\top\left(\frac{\partial\bar{f}^n}{\partial f_A^n}\right)^\top + \left(\frac{\partial\bar{f}^n}{\partial X_S^n}\right)^\top\right)\left(\frac{\partial\mathcal{S}}{\partial\bar{f}^n}\right)^\top\phi.\end{aligned} \tag{19}$$

The aerodynamic state will also affect the structural residual even though the aerodynamic state is frozen when calculating the partials. This because the surface nodes have changed and result in a different aerodynamic load.

The partial derivatives of structural dynamic residual with respect to V_f are given as:

$$\left(\frac{\partial\mathcal{S}}{\partial V_f}\right)^\top\phi = -\frac{2V_f}{\pi}\left(\bar{f}^n\right)^\top\phi. \tag{20}$$

The partial derivatives of structural dynamic residual with respect to ω are given as:

$$\begin{aligned}\left(\frac{\partial\mathcal{S}}{\partial\omega}\right)^\top\phi &= \left(\boldsymbol{M}\frac{\partial\boldsymbol{D}_Q(\omega)}{\partial\omega}u^n\right)^\top\phi,\\ &= \frac{2}{\omega}\left(\boldsymbol{M}\boldsymbol{D}_Q(\omega)u^n\right)^\top\phi,\end{aligned} \tag{21}$$

where we leverage on the fact that $\boldsymbol{D}_Q(\omega)\propto\omega^2$.

3. *Prescribed Motion Residual Partial Derivatives*

Since the dimension of $\partial \mathcal{R}_{mot}/\partial u^n$ is quite small ($2 \times n$), instead of giving the matrix–vector product form, we give the analytic expressions for the partials. This matrix is explicitly saved. We show the derivation of the matrix in this section.

At first, from frequency domain to the temporal domain (evaluated at kth time instance), we have the following transformation

$$\begin{aligned}
\alpha\left(\frac{kT}{N}\right) &= \frac{1}{N}\left(c_0 + c_1 e^{j\frac{2\pi}{T}\frac{kT}{N}} + c_2 e^{j\frac{2\pi}{T}\frac{2kT}{N}} + ... + c_{N-2} e^{j\frac{2\pi}{T}\frac{(N-2)kT}{N}} + c_{N-1} e^{j\frac{2\pi}{T}\frac{(N-1)kT}{N}}\right), \\
&= \frac{1}{N}\left(c_0 + c_1 e^{j\frac{2\pi k}{N}} + c_2 e^{j\frac{4\pi k}{N}} + ... + c_{N-2} e^{j\frac{2\pi(N-2)}{N}} + c_{N-1} e^{j\frac{2\pi(N-1)k}{N}}\right), \\
&= \frac{1}{N}\left(c_0 + c_1 e^{j\frac{2\pi k}{N}} + c_2 e^{j\frac{4\pi k}{N}} + ... + c_{N-2} e^{-j\frac{4\pi}{N}} + c_{N-1} e^{-j\frac{2\pi k}{N}}\right).
\end{aligned} \tag{22}$$

In addition, from the temporal domain to the frequency domain, with FFT, we have the following relation from Zhang [31]:

$$\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & w & \cdots & w^{N-1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & w^{N-1} & \cdots & w^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{bmatrix}, w = e^{-j\frac{2\pi}{N}}. \tag{23}$$

The two important states which relate with the dominant mode (for which we have the prescribed motion constraint) are the c_1 and c_{N-1}

$$\begin{aligned}
c_1 &= \alpha_0 + w\alpha_1 + \cdots + w^{N-1}\alpha_{N-1}, \\
c_{N-1} &= \alpha_0 + w^{N-1}\alpha_1 + \cdots + w^{(N-1)(N-1)}\alpha_{N-1}.
\end{aligned} \tag{24}$$

Notice that all α_i's are real and expand the equation with $w = e^{-j\frac{2\pi}{N}}$, we have

$$\begin{aligned}
c_1 &= \left(\alpha_0 + \cos\frac{2\pi}{N}\alpha_1 + \cdots + \cos\frac{2\pi(N-1)}{N}\alpha_{N-1}\right) + \left(-\sin\frac{2\pi}{N}\alpha_1 - \cdots - \sin\frac{2\pi(N-1)}{N}\alpha_{N-1}\right) j, \\
c_{N-1} &= \left(\alpha_0 + \cos\frac{2\pi(N-1)}{N}\alpha_1 + \cdots + \cos\frac{2\pi(N-1)^2}{N}\alpha_{N-1}\right) + \left(-\sin\frac{2\pi(N-1)}{N}\alpha_1 - \cdots - \sin\frac{2\pi(N-1)^2}{N}\alpha_{N-1}\right) j.
\end{aligned} \tag{25}$$

Define the real and imaginary components coefficients

$$\begin{aligned}
c_{1,r} &= \alpha_0 + \cos\frac{2\pi}{N}\alpha_1 + \cdots + \cos\frac{2\pi(N-1)}{N}\alpha_{N-1}, \\
c_{1,i} &= -\sin\frac{2\pi}{N}\alpha_1 - \cdots - \sin\frac{2\pi(N-1)}{N}\alpha_{N-1}, \\
c_{N-1,r} &= \alpha_0 + \cos\frac{2\pi(N-1)}{N}\alpha_1 + \cdots + \cos\frac{2\pi(N-1)^2}{N}\alpha_{N-1}, \\
c_{N-1,i} &= -\sin\frac{2\pi(N-1)}{N}\alpha_1 - \cdots - \sin\frac{2\pi(N-1)^2}{N}\alpha_{N-1}.
\end{aligned} \tag{26}$$

We have

$$\begin{aligned}
\frac{\partial c_{1,r}}{\partial \alpha^N} &= \begin{bmatrix} 1 & \cos\frac{2\pi}{N} & \cdots & \cos\frac{2\pi(N-1)}{N} \end{bmatrix}^\top, \\
\frac{\partial c_{1,i}}{\partial \alpha^N} &= \begin{bmatrix} 0 & -\sin\frac{2\pi}{N} & \cdots & -\sin\frac{2\pi(N-1)}{N} \end{bmatrix}^\top, \\
\frac{\partial c_{N-1,r}}{\partial \alpha^N} &= \begin{bmatrix} 1 & \cos\frac{2\pi(N-1)}{N} & \cdots & \cos\frac{2\pi(N-1)(N-1)}{N} \end{bmatrix}^\top, \\
\frac{\partial c_{N-1,i}}{\partial \alpha^N} &= \begin{bmatrix} 0 & -\sin\frac{2\pi(N-1)}{N} & \cdots & -\sin\frac{2\pi(N-1)(N-1)}{N} \end{bmatrix}^\top.
\end{aligned} \tag{27}$$

Moreover, the first mode can be expressed as

$$\begin{aligned}
\alpha_{\text{1st mode}} &= C_c \cos\frac{2\pi}{N} + C_s \sin\frac{2\pi}{N} + \text{pure imaginary number} \\
&= \frac{1}{N}(c_{1,r} + c_{N-1,r})\cos\frac{2\pi}{N} + \frac{1}{N}(-c_{1,i} + c_{N-1,i})\sin\frac{2\pi}{N} + \text{pure imaginary number}.
\end{aligned} \tag{28}$$

So we have:

$$\begin{bmatrix} \frac{\partial C_c}{\partial(c_{1,r}, c_{1,i}, c_{N-1,r}, c_{N-1,i})} \\ \frac{\partial C_s}{\partial(c_{1,r}, c_{1,i}, c_{N-1,r}, c_{N-1,i})} \end{bmatrix} = \begin{bmatrix} \frac{1}{N} & 0 & \frac{1}{N} & 0 \\ 0 & -\frac{1}{N} & 0 & \frac{1}{N} \end{bmatrix}. \tag{29}$$

We also know that the magnitude and phase can be written as

$$\begin{aligned} \text{mag} &= \sqrt{C_c^2 + C_s^2}, \\ \text{phase} &= \sin^{-1} \frac{C_c}{\sqrt{C_c^2 + C_s^2}}. \end{aligned} \tag{30}$$

So we have

$$\begin{bmatrix} \frac{\partial \text{mag}}{\partial(C_c, C_s)} \\ \frac{\partial \text{phase}}{\partial(C_c, C_s)} \end{bmatrix} = \begin{bmatrix} \frac{C_c}{\sqrt{C_c^2+C_s^2}} & \frac{C_s}{\sqrt{C_c^2+C_s^2}} \\ \frac{|C_s|}{C_c^2+C_s^2} & -\frac{C_c \text{sgn}(C_s)}{C_c^2+C_s^2} \end{bmatrix}. \tag{31}$$

Finally, to recap everything, we have

$$\begin{aligned} \begin{bmatrix} \frac{\partial \mathcal{R}_{mag}}{\partial \alpha^N} \\ \frac{\partial \mathcal{R}_{pha}}{\partial \alpha^N} \end{bmatrix} &= \begin{bmatrix} \frac{\partial \text{mag}}{\partial(C_c, C_s)} \\ \frac{\partial \text{phase}}{\partial(C_c, C_s)} \end{bmatrix} \begin{bmatrix} \frac{\partial C_c}{\partial(c_{1,r}, c_{1,i}, c_{N-1,r}, c_{N-1,i})} \\ \frac{\partial C_s}{\partial(c_{1,r}, c_{1,i}, c_{N-1,r}, c_{N-1,i})} \end{bmatrix} \begin{bmatrix} \frac{\partial c_{1,r}}{\partial \alpha^N} \\ \frac{\partial c_{1,i}}{\partial \alpha^N} \\ \frac{\partial c_{N-1,r}}{\partial \alpha^N} \\ \frac{\partial c_{N-1,i}}{\partial \alpha^N} \end{bmatrix}, \\ &= \begin{bmatrix} \frac{C_c}{\sqrt{C_c^2+C_s^2}} & \frac{C_s}{\sqrt{C_c^2+C_s^2}}, \\ \frac{|C_s|}{C_c^2+C_s^2} & -\frac{C_c \text{sgn}(C_s)}{C_c^2+C_s^2} \end{bmatrix} \begin{bmatrix} \frac{1}{N} & 0 & \frac{1}{N} & 0 \\ 0 & -\frac{1}{N} & 0 & \frac{1}{N} \end{bmatrix} \begin{bmatrix} 1 & \cos\frac{2\pi}{N} & \cdots & \cos\frac{2\pi(N-1)}{N} \\ 0 & -\sin\frac{2\pi}{N} & \cdots & -\sin\frac{2\pi(N-1)}{N} \\ 1 & \cos\frac{2\pi(N-1)}{N} & \cdots & \cos\frac{2\pi(N-1)(N-1)}{N} \\ 0 & -\sin\frac{2\pi(N-1)}{N} & \cdots & -\sin\frac{2\pi(N-1)(N-1)}{N} \end{bmatrix}. \end{aligned} \tag{32}$$

C. Coupled Adjoint Solution

1. *Coupled Krylov Solver*

To solve the coupled adjoint equation, we apply the Krylov subspace method. This is the first use of a monolithic solution method for the coupled, time-spectral flutter adjoint equation. Krylov subspace method has the advantage that it is not required to store the matrix explicitly and only the matrix–vector products are required for the solution. Since the matrix–vector products are between the transpose of Jacobian matrices and vectors, we apply backward propagation to this operation. The pseudocode for the operation is given in Alg 1.

To improve the convergence of the Krylov method, we apply a block Jacobi preconditioner. The reason we choose block Jacobi preconditioner is that it will allow the structural and aerodynamic preconditioning to be carried out in parallel and it allows the reuse of the time-spectral aerodynamic preconditioner developed in [28]. The preconditioner is given by

$$\begin{aligned} (J^\top P^{-1})\tau &= \frac{\partial I}{\partial u}, \\ P^{-1}\tau &= \Psi, \end{aligned} \tag{33}$$

where P is the preconditioner, τ is the solution of the preconditioned system. To be more specific, the second equation is expanded as

$$\begin{bmatrix} P_{mot,\mathcal{S}}^{-1} & 0 \\ 0 & P_{\mathcal{A}}^{-1} \end{bmatrix} \begin{bmatrix} \tau_{mot,\mathcal{S}} \\ \tau_{\mathcal{A}} \end{bmatrix} = \begin{bmatrix} \Psi_{\mathcal{R}_{mot},\mathcal{S}} \\ \Psi_{\mathcal{A}} \end{bmatrix}. \tag{34}$$

As mentioned before, the CFD preconditioner $P_{\mathcal{A}}^{-1}$ has been implemented previously in [28]. The prescribed motion and CSD preconditioner is a direct inversion of the approximate diagonal term, notice that it is the transpose of the preconditioner from the forward solve constructed in our previous work [4]. It is given as:

$$P_{mot,\mathcal{S}}^{-1} = \begin{bmatrix} 0 & 0 & -\frac{2V_f}{\pi}(f^n)^\top \\ 0 & 0 & \left(\boldsymbol{M}\frac{d\boldsymbol{D}_Q}{d\omega}u^n\right)^\top \\ \frac{\partial|\alpha_{\text{1st mode}}|}{\partial u^n} & \frac{\partial \phi}{\partial u^n} & (\boldsymbol{M}\boldsymbol{D}_Q + \boldsymbol{K})^\top \end{bmatrix}^{-1}. \tag{35}$$

Algorithm 1 Coupled Krylov method linear operator

function MULT(X)

 $(X_{\mathcal{R}_{mag}}, X_{\mathcal{R}_{pha}}, X_{\mathcal{S}}, X_{\mathcal{A}}) \leftarrow X$ ▷ Extract flutter velocity index, frequency, structural and aerodynamic components

 $X_{F_{\mathcal{A}}^n} \leftarrow \frac{\partial \bar{f}^n}{\partial F_{\mathcal{A}}^n}^\top \frac{\partial \mathcal{S}}{\partial f^n}^\top X_{\mathcal{S}}$ ▷ Off-diagonal contribution to aerodynamic states

 $X_{X_{\mathcal{S}}^n} \leftarrow \frac{\partial F_{\mathcal{S}}^n}{\partial X_{\mathcal{S}}^n}^\top \frac{\partial \mathcal{S}}{\partial f^n}^\top X_{\mathcal{S}}$ ▷ Direct contribution of structural residual seed to aerodyanmic surface coordinates

 $Y_{\zeta^n} \leftarrow \frac{\partial \mathcal{A}}{\partial \zeta^n}^\top X_{\mathcal{A}} + \frac{\partial F_{\mathcal{A}}^n}{\partial \zeta^n}^\top X_{F_{\mathcal{A}}^n}$ ▷ Summation of the diagonal and off-diagonal aerdynamic states seeds

 $X_{X_{\mathcal{S}}^n} \leftarrow X_{X_{\mathcal{S}}^n} + \frac{\partial \mathcal{A}}{\partial X_{\mathcal{S}^n}}^\top X_{\mathcal{A}} + \frac{\partial F_{\mathcal{A}^n}}{\partial X_{\mathcal{S}^n}}^\top X_{F_{\mathcal{A}}^n}$ ▷ Sum structural and aerodynamic contribution to the surface coordinates

 $X_{T_{BC}} \leftarrow \frac{\partial \mathcal{A}}{\partial T_{BC}}^\top X_{\mathcal{A}}, X_{\rho_{BC}} \leftarrow \frac{\partial \mathcal{A}}{\partial \rho_{BC}}^\top X_{\mathcal{A}}, X_{T_0} \leftarrow \frac{\partial \mathcal{A}}{\partial T_0}^\top X_{\mathcal{A}}$ ▷ Aerodynamic contribution to intermediate states for V_f and ω

 $X_{T_\infty} \leftarrow \frac{\partial T_{BC}}{\partial T_\infty}^\top X_{T_{BC}} + \frac{\partial \rho_{BC}}{\partial T_\infty}^\top X_{\rho_{BC}}$ ▷ Aerodynamic contribution to intermediate states for V_f

 $Y_{Vf} \leftarrow \frac{\partial T_\infty}{\partial V_f}^\top X_{T_\infty}$ ▷ Aerodynamic contribution to V_f

 $Y_\omega \leftarrow \frac{\partial T_0}{\partial \omega}^\top X_{T_0}$ ▷ Aerodynamic contribution to ω

 $Y_{u^n} \leftarrow (\boldsymbol{M}\boldsymbol{D}_Q(\omega) + \boldsymbol{K})^\top X_{\mathcal{S}} + \frac{\partial \mathcal{R}_{mag}}{\partial u^n}^\top X_{\mathcal{R}_{mag}} + \frac{\partial \mathcal{R}_{pha}}{\partial u^n}^\top X_{\mathcal{R}_{pha}}$ ▷ Structural and prescribed motion contribution to u^n

 $Y_{V_f} \leftarrow Y_{V_f} + \frac{\partial \mathcal{S}}{\partial V_f}^\top X_{\mathcal{S}}, Y_\omega \leftarrow Y_\omega + \frac{\partial \mathcal{S}}{\partial \omega}^\top X_{\mathcal{S}}$ ▷ Structural contribution to V_f and ω

 $Y_{u^n} \leftarrow Y_{u^n} + \frac{\partial X_{\mathcal{S}}^n}{\partial u^n}^\top X_{X_{\mathcal{S}}^n}$ ▷ Add the aerodynamic contribution to u^n

return $(Y_{V_f}, Y_\omega, Y_{u^n}, Y_{\zeta^n})$

end function

It is an “approximate” block Jacobi preconditioner in the sens that the aerodynamic load contribution is dropped from the diagonal term: $\partial\mathcal{S}/\partial u^n$,

$$
\begin{aligned}
\frac{\partial \mathcal{S}}{\partial u^n} &= (\boldsymbol{M}\boldsymbol{D}_Q(\omega) + \boldsymbol{K})^\top - \frac{V_f^2}{\pi}\left(\frac{\partial \bar{f}^n}{\partial u^n}\right)^\top, \\
&\approx (\boldsymbol{M}\boldsymbol{D}_Q(\omega) + \boldsymbol{K})^\top .
\end{aligned} \tag{36}
$$

V. Result

In this work, we use the NACA 64A010 two-dimensional wing section model as described in [32]. A schematic of the configuration is shown in Figure 2. The detailed mesh information can be found in [4].

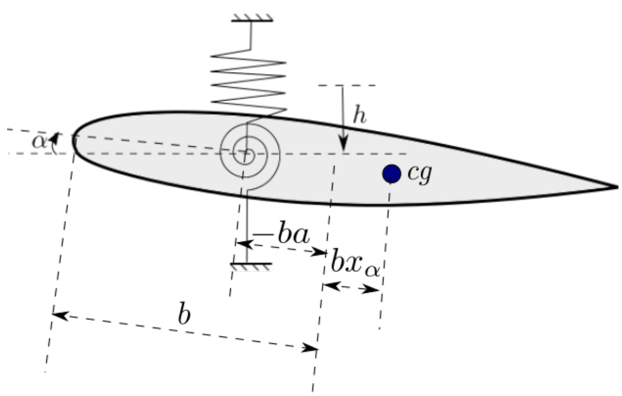


Figure 2: Wing section model

A. $\partial\mathcal{R}/\partial q$ Matrix Sparsity

The $\partial\mathcal{R}/\partial q$ matrix sparsity pattern is given in Figure 3 and a zoom-in view is shown in Figure 4. For clarity, we use a coarsened CFD mesh for this. The left dense columns indicating that the variables V_f, ω and u^n affects almost all residuals. The relatively short dense rows at the top are the $\partial\mathcal{S}/\partial\zeta^n$ which is only nonzero for the elements on the wall. The zeros on the diagonal block for $\partial\mathcal{R}_{mot}/\partial V_f, \partial\mathcal{R}_{mot}/\partial\omega$, the dense off-diagonal pattern for $\partial\mathcal{S}/\partial u^n$ $\partial\mathcal{S}/\partial V_f$ and $\partial\mathcal{S}/\partial\omega$ and also the off-diagonal terms from CFD component itself resulting from temporal derivative terms will make the system of equation less diagonally dominant and more difficult to solve.

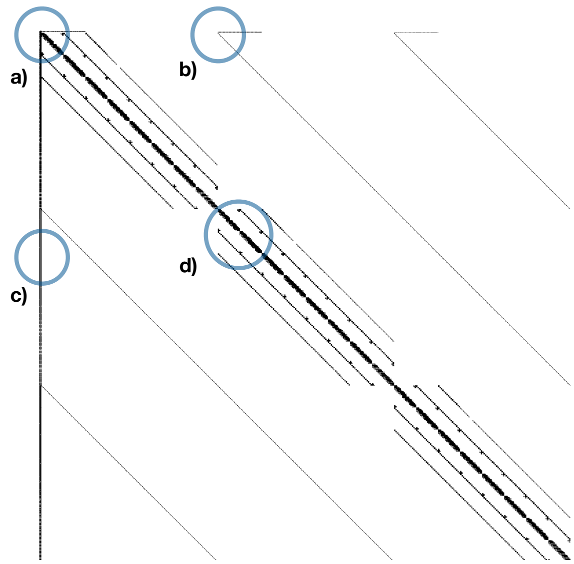


Figure 3: $\partial\mathcal{R}/\partial q$ sparsity pattern (a zoom-in view is given in Figure 4). Notice the dense column on the left edge. This indicates a strong coupling between the aerodynamic residual with respect to the structural variable, V_f and ω.

B. Partial Derivative Verification

To verify the correctness of our partial derivative implementations. We conduct the following two tests: dot product test for the forward automatic differentiation (FAD) and reverse automatic differentiation (RAD), and the directional partial derivative between finite difference (FD) and FAD. The former test verifies the consistency of RAD and FAD implementation, and the latter verifies the implementation of FAD. By the conduction of the two tests, we indirectly verified the RAD implementation.

1. *Dot Product Test*

We use the partial derivative of the time-spectral flutter residual with respect to states as an example. The FAD is given as

$$d\mathcal{R}_{\text{FAD}} = \frac{\partial\mathcal{R}}{\partial q} dq_{\text{FAD}}, \tag{37}$$

where dq_{FAD} is the forward seed, and $d\mathcal{R}_{\text{FAD}}$ is the forward output.

The RAD is given as

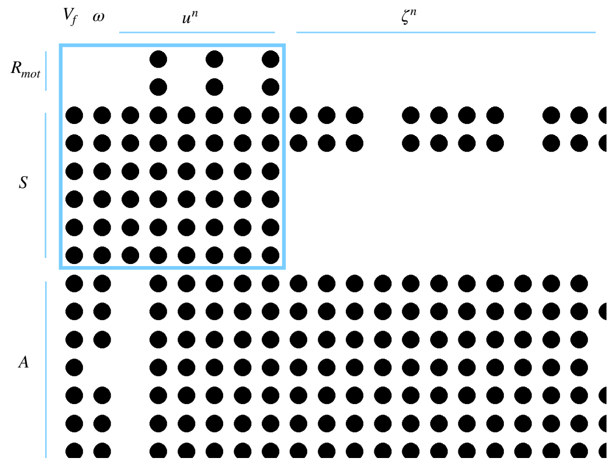
$$dq_{\text{RAD}} = \left(\frac{\partial\mathcal{R}}{\partial q}\right)^{\top} d\mathcal{R}_{\text{RAD}}. \tag{38}$$

where $d\mathcal{R}_{\text{RAD}}$ is the reverse seed and dq_{RAD} is the reverse output.

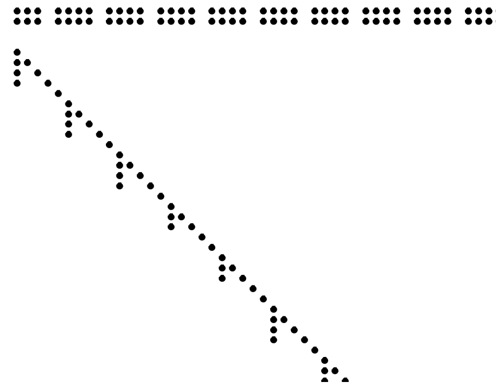
For any given dq_{FAD} and $d\mathcal{R}_{\text{RAD}}$, correctly implemented FAD and RAD should given outputs such that

$$d\mathcal{R}_{\text{FAD}}^{\top} d\mathcal{R}_{\text{RAD}} = dq_{\text{RAD}}^{\top} dq_{\text{FAD}} \tag{39}$$

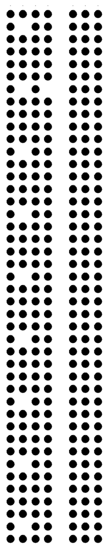
which can be verified by a substitution of results from Equation 37 and Equation 38. For verification purpose, we chose the seed as random vectors: $dy_{\text{FAD}} \sim \mathcal{U}(0, \mathbf{1})$, $d\mathcal{A}_{\text{RAD}} \sim \mathcal{U}(0, \mathbf{1})$. The result of a dot product test is given in Table ??. The outputs from FAD and RAD match with each other up to 13 digits indicating the consistence of the FAD and RAD implementations.



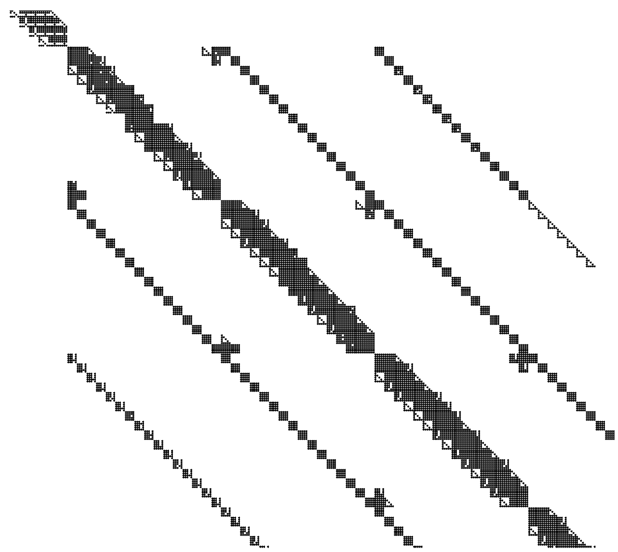
a) Zoom in for $\partial\mathcal{R}_{mot}/\partial V_f$, $\partial\mathcal{R}_{mot}/\partial\omega$, $\partial\mathcal{R}_{mot}/\partial u^n$, $\partial\mathcal{S}/\partial V_f$, $\partial\mathcal{S}/\partial\omega$ and $\partial\mathcal{S}/\partial u^n$



b) Zoom in for $\partial\mathcal{S}/\partial\zeta^n$



c) Zoom in for $\partial\mathcal{A}/\partial V_f$,$\partial\mathcal{A}/\partial\omega$ and $\partial\mathcal{A}/\partial u^n$



d) Zoom in for $\partial\mathcal{A}/\partial\zeta^n$

Figure 4: A zoom-in view for $\partial\mathcal{R}/\partial q$ sparsity pattern. The Figure 4a, Figure 4b,Figure 4c and Figure 4d correspond with the blocks from Figure 3.

2. *Directional Derivative Test*

We verify the FAD with FD with the directional test. The formulas are given as

$$\begin{aligned} d\mathcal{R}_{\text{FAD},v} &= \frac{\partial\mathcal{R}}{\partial q}v, \\ d\mathcal{R}_{\text{FD},v} &= \frac{\mathcal{R}(q+\epsilon v)-\mathcal{R}(q)}{\epsilon}, \end{aligned} \tag{40}$$

where ϵ is a small number and v is the direction the Jacobian is projected to. To test the implementation, we choose $v \sim \mathcal{U}(0,\mathbf{1})$. An example is show in Figure 5. The FAD and FD qualitatively match each other. Together with the dot product test which verifies the RAD and FAD give consistent outputs, it is expected the RAD is implemented correctly. RAD is what is required when we solve the adjoint equations.

Table 2: Dot product test

$d\mathcal{R}_{\text{FAD}}^{\top} d\mathcal{R}_{\text{RAD}}$	$-18794.27783914\mathbf{8148}$
$dq_{\text{RAD}}^{\top} dq_{\text{FAD}}$	$-18794.27783914\mathbf{9660}$

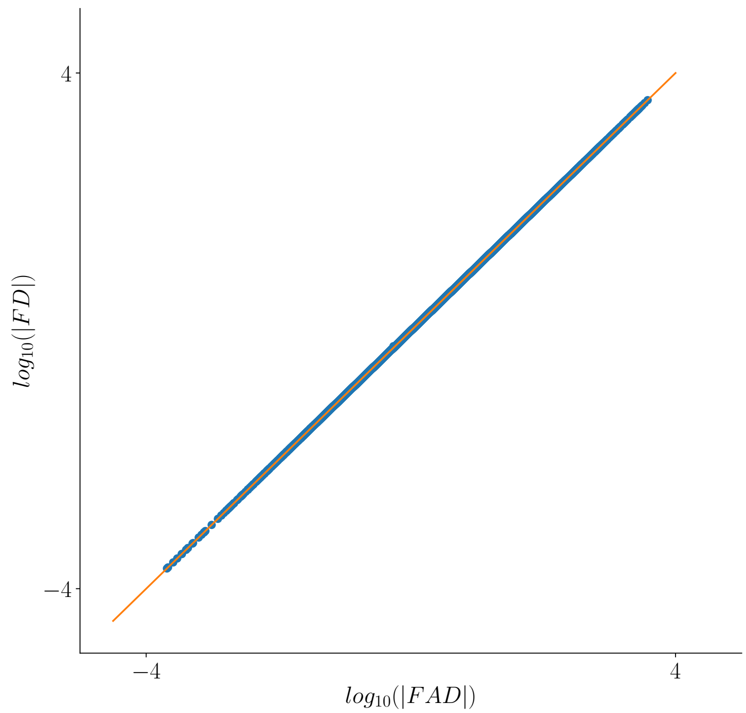


Figure 5: $d\mathcal{R}_{\text{FAD},v}$ and $d\mathcal{R}_{\text{FD},v}$ comparison, the line corresponding with $y = x$

C. Gradient Verification

To verify our adjoint solution, we run a simulation with $\text{M} = 0.825$. The vertical coordinates of 6 FFD points are the design variables for the verification case as shown in Figure 5. We want to solve for the flutter velocity sensitivity with respect to the design variables, i.e. dV_f/dx. We compare the adjoint method with the FD under different step sizes and the best match is with the step size set to be 10^{-6}. The result for one of the design variables is shown in Table 3. ADjoint and finite difference method give similar results. But since the finite difference is in general not of machine

Table 3: Accuracy validation of TS flutter adjoint

	ADjoint	FD	Difference	Step size
dV_f/dx	0.49633733	0.52626755	$3.0\text{E}-2$	$1\text{E}-3$
		0.50346179	$7.1\text{E}-3$	$1\text{E}-4$
		0.50098210	$4.6\text{E}-3$	$1\text{E}-5$
		0.49458700	$\mathbf{1.8E-3}$	$1\text{E}-6$
		0.47008000	$2.6\text{E}-2$	$1\text{E}-7$

precision, for a more careful verification of our adjoint method, we need to implement the complex step method [33].

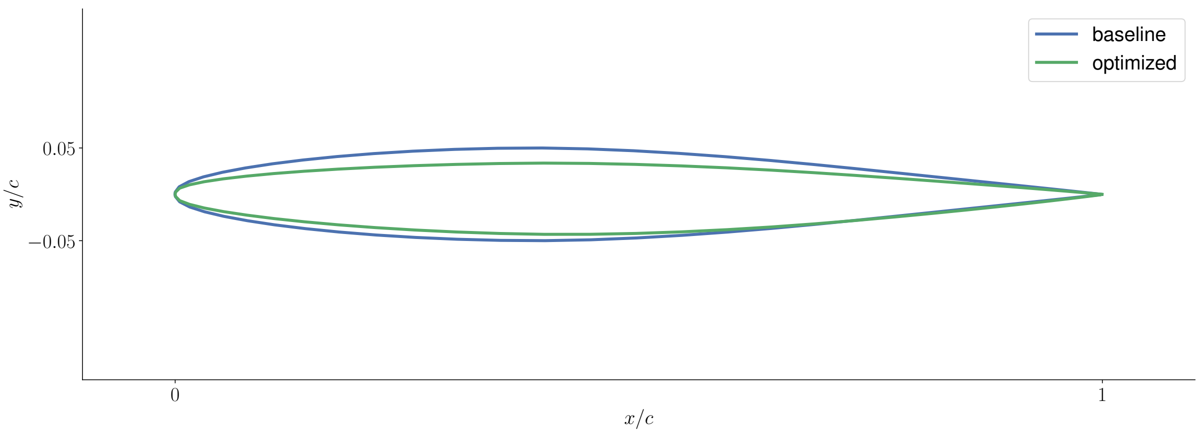
D. Flutter Velocity Index Optimization

We conduct an optimization with the goal to maximize the flutter velocity index. We have the “y” coordinates of FFD points as our design variables. The FFD is shown in Figure 6. We only have geometry constraints which are the upper and lower bounds (0.02 and -0.02 respectively) of displacement of the “y” coordinates of FFD points. By constraining the FFD points to be symmetric about the chord, the design variable number is reduced to 4. The Mach number for this case is 0.75 and the airfoil has a $2°$ angle of attack. The problem is set up for demonstration purpose and for a more realistic case the flutter velocity index should be a constraint rather than an objective function. We use SNOPT [34] as the optimizer which has a python interface from pyOptsparse [35]. The detailed settings are given in Table 4.

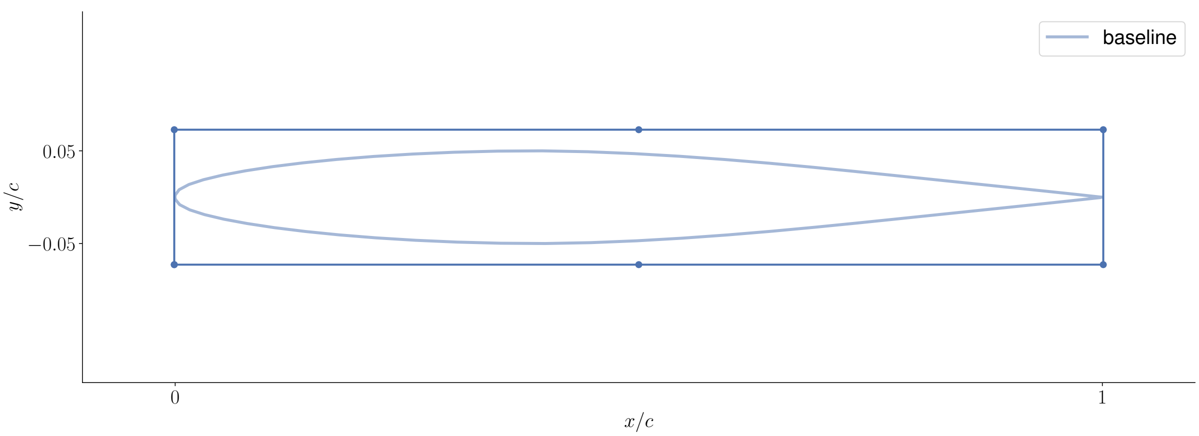
Table 4: Aerodynamic shape optimization problem

	Function/variable	Description	Quantity
maximize	V_f	flutter velocity index	
with respect to	y	FFD control points y coordinates	4
subject to	$y_{min} \leq y \leq y_{max}$	upper and lower bounds on FFD control points y coordinates	4

The baseline and optimized airfoils are shown in Figure 6. It is observed that the thickness of the airfoil is reduced and the curvature of the upper surface is reduced. It is observed that all the FFD points hit the bounds.



a) Baseline and optimized airfoils



b) FFD points

Figure 6: Baseline (NACA 64A010) and optimized airfoils with the FFD points (Notice that the LE and TE FFD points are constrained to be symmetric with $y = 0$, so there are 4 independent variables.)

The baseline and optimized V_f's are given in the Table 5. The optimized airfoil gets an improved flutter velocity index by about 10.9%. Due to the simplicity of the problem, the optimizer is able to find the solution in 3 major iterations with feasibility and merit function both 0.0 indicating that the case is feasible and optimal.

We also compare the flutter boundary in the range of M $= 0.75$ to 0.898 as shown in Figure 7. It is found that although the V_f is improved significantly for the subsonic regime, as the Mach number approaches 0.9, the optimized solution has a lower V_f compared with baseline NACA $64A010$ airfoil. A multipoint optimization is necessary to guarantee the airfoil gains better aeroelastic performance in the whole operation domain.

Table 5: V_f optimization result

	baseline	optimized	improvement
V_f	1.136	1.260	10.9%

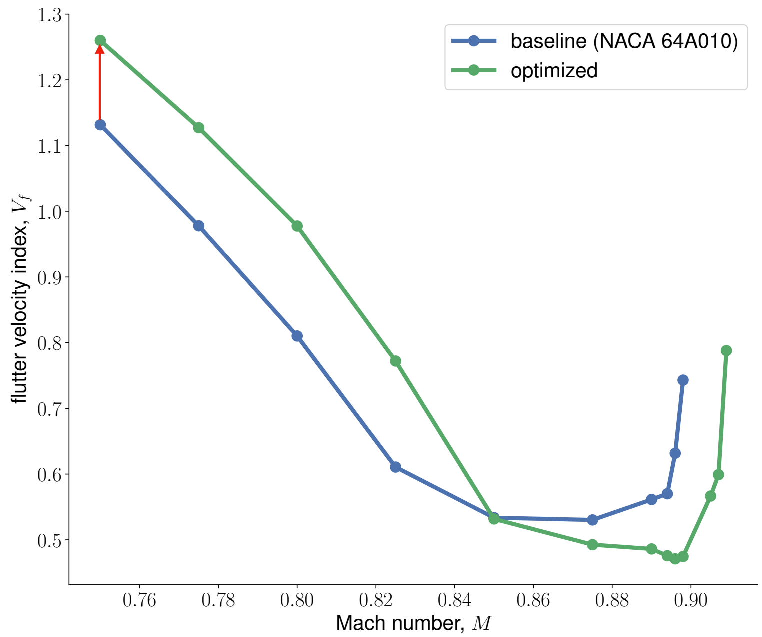


Figure 7: Flutter boundary for baseline and optimized case. The optimization is conducted for $\mathrm{M} = 0.75$ as shown by the red arrow. In the subsonic regime V_f is increased, but in the transonic regime, V_f is decreased.

VI. Conclusion

The flutter constraint is a challenging constraint to implement for aircraft design. We demonstrate a method for computing a flutter constraint using CFD by developing the coupled-adjoint for the time-spectral flutter equations. A coupled, Krylov solver is applied to solve this coupled-adjoint equation. We verify the gradients computed with adjoint method with finite difference gradients. Finally we conduct an aerodynamic shape optimization to maximize the flutter velocity index with respect to the aerodynamic shape variables. The optimized result has increased the flutter velocity index by about 10.9%.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] Jonsson, E., Lupp, C. A., Riso, C., Cesnik, C. E. S., Epureanu, B. I., and Martins, J. R. R. A., "Flutter and Post-Flutter Constraints in Aircraft Design Optimization," *Progress in Aerospace Sciences*, , No. 12, 2018, (In press).

[6] 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.

[7] Gopinath, A. K. and Jameson, A., "Time Spectral Method for Periodic Unsteady Computations over Two- Three Dimensional Bodies," *AIAA Paper* 2005-1220, 2005.

[8] McMullen, M. and Jameson, A., "The computational efficiency of non-linear frequency domain methods," *Journal of Computational Physics*, Vol. 212, No. 2, mar 2006, pp. 637–661. doi:10.1016/j.jcp.2005.07.021.

[9] 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.

[10] 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.

[11] 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.

[12] Prasad, R., Kim, H., Choi, S., and Yi, S., "High Fidelity Prediction of Flutter/LCO using Time Spectral Method," *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, American Institute of Aeronautics and Astronautics, jan 2018. doi:10.2514/6.2018-0459.

[13] Yao, W. and Marques, S. P., "Prediction of Transonic LCO using an Harmonic Balance Method," *44th AIAA Fluid Dynamics Conference*, American Institute of Aeronautics and Astronautics, jun 2014. doi:10.2514/6.2014-2310.

[14] Stanford, B., Wieseman, C. D., and Jutte, C., "Aeroelastic Tailoring of Transport Wings Including Transonic Flutter Constraints," *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Kissimmee, FL, January 5–9 2015.

[15] Chen, P. C., Zhang, Z., and Livne, E., "Design-Oriented Computational Fluid Dynamics-Based Unsteady Aerodynamics for Flight-Vehicle Aeroelastic Shape Optimization," *AIAA Journal*, Vol. 53, No. 12, 2015, pp. 3603–3619. doi:10.2514/1.J054024.

[16] Bartels, R. E. and Stanford, B., "Economical Unsteady High Fidelity Aerodynamics in a Structural Optimization with a Flutter Constraint," *35th AIAA Applied Aerodynamics Conference*, Denver, CO, June 5–9 2017.

[17] Kennedy, G. J., Kenway, G. K. W., and Martins, J. R. R. A., "Towards Gradient-Based Design Optimization of Flexible Transport Aircraft with Flutter Constraints," *Proceedings of the 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Atlanta, GA, June 2014. doi:10.2514/6.2014-2726, AIAA 2014-2726.

[18] Beran, P., Stanford, B. K., and Wang, K. G., "Fast Prediction of Flutter and Flutter Sensitivities," *17th International Forum on Aeroelasticity and Structural Dynamics (IFASD)*, Como, Italy, June 25–28 2017. doi:10.2514/6.2017-1350.

[19] Jonsson, E., Mader, C. A., Kennedy, G. J., and Martins, J. R. R. A., "Computational Modeling of Flutter Constraint for High-Fidelity Aerostructural Optimization," *Proceedings of the AIAA Science and Technology Forum and Exposition (SciTech)*, San Diego, CA, January 2019.

[20] Zhang, Z., Chen, P. C., Yang, S., Wang, Z., and Wang, Q., "Unsteady Aerostructure Coupled Adjoint Method for Flutter Suppression," *AIAA Journal*, Vol. 53, No. 8, 2015, pp. 2121–2129. doi:10.2514/1.J053495.

[21] Kiviaho, J. F., Jacobson, K., Smith, M. J., and Kennedy, G., "Application of a Time-Accurate Aeroelastic Coupling Framework to Flutter-Constrained Design Optimization," *2018 Multidisciplinary Analysis and Optimization Conference*, American Institute of Aeronautics and Astronautics, jun 2018. doi:10.2514/6.2018-2932.

[22] Thomas, J., Dowell, E., and Hall, K. C., "Discrete Adjoint Method for Nonlinear Aeroelastic Sensitivities for Compressible and Viscous Flows," *54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Boston, MA, April 8–11 2013.

[23] Thomas, J. and Dowell, E., "Discrete Adjoint Method for Aeroelastic Design Optimization," *15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, American Institute of Aeronautics and Astronautics, jun 2014. doi:10.2514/6.2014-2298.

[24] Martins, J. R. R. A. and Hwang, J. T., "Review and Unification of Methods for Computing Derivatives of Multidisciplinary Computational Models," *AIAA Journal*, Vol. 51, No. 11, November 2013, pp. 2582–2599. doi:10.2514/1.J052184.

[25] Martins, J. R. R. A., Alonso, J. J., and Reuther, J. J., "High-Fidelity Aerostructural Design Optimization of a Supersonic Business Jet," *Journal of Aircraft*, Vol. 41, No. 3, May 2004, pp. 523–530. doi:10.2514/1.11478.

[26] Wang, Q., Gleich, D., Saberi, A., Etemadi, N., and Moin, P., "A Monte Carlo method for solving unsteady adjoint equations," *Journal of Computational Physics*, Vol. 227, No. 12, jun 2008, pp. 6184–6205. doi:10.1016/j.jcp.2008.03.006.

[27] Mader, C. A., Martins, J. R. R. A., Alonso, J. J., and van der Weide, E., "ADjoint: An Approach for the Rapid Development of Discrete Adjoint Solvers," *AIAA Journal*, Vol. 46, No. 4, April 2008, pp. 863–873. doi:10.2514/1.29123.

[28] Mader, C. A. and Martins, J. R. R. A., "Derivatives for Time-Spectral Computational Fluid Dynamics Using an Automatic Differentiation Adjoint," *AIAA Journal*, Vol. 50, No. 12, December 2012, pp. 2809–2819. doi:10.2514/1.J051658.

[29] Kenway, G. K. W. and Martins, J. R. R. A., "Buffet Onset Constraint Formulation for Aerodynamic Shape Optimization," *AIAA Journal*, Vol. 55, No. 6, June 2017, pp. 1930–1947. doi:10.2514/1.J055172.

[30] Luke, E., Collins, E., and Blades, E., "A Fast Mesh Deformation Method Using Explicit Interpolation," *Journal of Computational Physics*, Vol. 231, No. 2, Jan. 2012, pp. 586–601. doi:10.1016/j.jcp.2011.09.021.

[31] Zhang, X.-D., *Matrix Analysis and Applications*, Cambridge University Press, oct 2017. doi:10.1017/9781108277587.

[32] Alonso, J. and Jameson, A., "Fully-implicit time-marching aeroelastic solutions," *32nd Aerospace Sciences Meeting and Exhibit*, American Institute of Aeronautics and Astronautics, jan 1994. doi:10.2514/6.1994-56.

[33] Martins, J. R. R. A., Sturdza, P., and Alonso, J. J., "The Complex-Step Derivative Approximation," *ACM Transactions on Mathematical Software*, Vol. 29, No. 3, September 2003, pp. 245–262. doi:10.1145/838250.838251.

[34] Gill, P. E., Murray, W., and Saunders, M. A., "An SQP algorithm for large-scale constrained optimization," *Society for Industrial and Applied Mathematics*, Vol. 47, No. 1, 2005.

[35] Perez, R. E., Jansen, P. W., and Martins, J. R. R. A., "pyOpt: A Python-Based Object-Oriented Framework for Nonlinear Constrained Optimization," *Structural and Multidisciplinary Optimization*, Vol. 45, No. 1, January 2012, pp. 101–118. doi:10.1007/s00158-011-0666-3.