

An Approach to Modeling Aeroservoelasticity in MPhys

Andrew S. Thelen,^{*} Bret K. Stanford,[†] Kevin E. Jacobson,[‡]
NASA Langley Research Center, Hampton, VA, 23681

Aeroelastic design frameworks must be able to include the presence of control surfaces in order to design for aeroservoelastic (ASE) systems. These types of systems (e.g., maneuver load alleviation, active flutter suppression, or gust load alleviation) have the potential to improve fuel efficiency as well as passenger safety and comfort. While many papers in the literature demonstrate the design of these systems with panel-based aerodynamics, doing so with higher fidelity aerodynamics (i.e., with Computational Fluid Dynamics (CFD)) is much less common, especially when unsteady aerodynamics are involved. The present work describes the recent implementation of ASE design tools developed at NASA Langley Research Center. Through the use of the MPhys library for OpenMDAO, panel- and CFD-based aerodynamic solvers may be easily interchanged. To demonstrate static ASE, an optimization with FUN3D Stabilized Finite Element (SFE) aerodynamics is demonstrated, with static control surface deflections being modeled at several off-design cruise conditions as well as at a 2.5g pull-up maneuver. As a demonstration of dynamic ASE, optimizations with doublet lattice method and FUN3D/SFE linearized frequency-domain aerodynamics are shown, where an active flutter suppression controller is designed simultaneously with the other wing design variables.

I. Introduction

In recent years, aerodynamic and aerostructural optimization toolsets in the literature have usually neglected the presence of control surfaces; this is especially true when using Computational Fluid Dynamics (CFD). These aeroservoelastic (ASE) systems, however, are important to consider when designing an aircraft as Maneuver Load Alleviation (MLA), Active Flutter Suppression (AFS), and Gust Load Alleviation (GLA) systems have the potential to increase safety and reduce weight and fuel burn of an aircraft. Modern production aircraft, in fact, are often designed with MLA and maybe even GLA in mind, with AFS likely on the horizon. In the MPhys [1] library for OpenMDAO [2], developed in part by the authors, a standardized way to include such systems is still lacking. While such a standard is not strictly necessary, as one could envision a model-specific way in OpenMDAO to include this, formalizing this standard may expedite the addition of this discipline when setting up a new optimization problem. Nonetheless, the requirements of such an implementation include the ability to model the same wing or aircraft design at different flight conditions with different control surface deflections, as part of or after an Outer Mold Line (OML) geometry has been adjusted by an optimizer. When considering dynamic ASE, controller design and assessment of the system's stability adds additional complexity to the design process, especially if analytic derivatives are desired for gradient-based optimization.

This paper therefore presents an implementation of static and dynamic ASE modeling for optimization with MPhys/OpenMDAO. Being based around MPhys, the implementation allows for either panel- or CFD-based aerodynamics to be used interchangeably. This is done through the use of aerodynamic and structural analysis software developed at NASA Langley Research Center, including FUN3D's Stabilized Finite Element (SFE) [3, 4] and Linearized Frequency Domain (LFD) [5, 6] solvers as well as in-house Vortex Lattice Method (VLM), Doublet Lattice Method (DLM), and structural solver (known as PyShell). The Matching-based Extrapolation of Loads and Displacements (MELD) [7] transfer scheme is used to transfer loads, displacements, and structural mode shapes between aerodynamic and structural meshes. Thus far, static ASE has been used for MLA and multi-point design, while dynamic ASE has been used for designing a wing with an AFS system.

In the remainder of the paper, the control surface implementation is first discussed. A description of the Rogers Rational Function Approximation (RFA) implementation used for dynamic ASE stability calculations is then described; details on controller design, which use Linear-Quadratic Regulator (LQR) of a reduced system, are subsequently given. Sample analysis results of the implementation are then provided, including both static and dynamic ASE. Optimization

^{*}Research Aerospace Engineer, Aeroelasticity Branch, andrew.s.thelen@nasa.gov, AIAA Member.

[†]Research Aerospace Engineer, Aeroelasticity Branch, AIAA Associate Fellow.

[‡]Research Aerospace Engineer, Aeroelasticity Branch, AIAA Member.

demonstrations are then shown, including static ASE optimization with MLA as well as dynamic ASE with an AFS system. The AFS design problem is solved using both panel- and CFD-based aerodynamics.

II. Modeling and Control Methodology

This section describes the current implementation of ASE analyses in MPhys/OpenMDAO. Section II.A provides details of how control surfaces are defined. Sections II.B and II.C subsequently provide details on the Rogers RFA flutter solver and LQR controller design.

A. Blended control surface implementation

The present control surface implementation utilizes a few basic parameters, primarily the inboard and outboard hinge line locations as fractions of local chord and span. To simplify mesh motion with control surface deflection, applications have thus far blended the control surfaces with the adjacent undeformed wing regions rather than physically modeling the gap around control surfaces. Modeling the gaps could be done with the right mesh, although deforming the mesh would presumably be challenging for the mesh deformation algorithm without a sliding boundary. Nonetheless, to achieve better mesh quality at these span-wise edges, an s-shaped blended region is applied in the adjacent regions on the wing; the width of the blended region is set by the user and is given as a fraction of current span.

Being based around MPhys, an unofficial MPhys builder for controls is used to add pre- and post-coupling subsystems to an aerodynamic or aerostructural scenario. Typically, the pre-coupling subsystem takes in aerodynamic and structural nodal coordinates from a geometry builder and then applies deformations due to control surface deflections before the MPhys coupling group is solved. A post-coupling subsystem, when carrying out dynamic ASE analyses, combines structural mode shapes with those of the control surfaces; the subsequent DLM or LFD analysis then simply solves for the flow around the total set of modes without knowing about the presence of control surfaces. The post-coupling subsystem also includes a hinge moment calculator, which has so far only been tested a limited amount. An OpenMDAO-based Schur complement solver [8, 9], while out of scope in the present work, has been used to allow all or a subset of control surface angles (in addition to angle of attack) to iteratively be trimmed within the coupling group based on a desired lift or moment. Generally, this can incur a considerable computational cost increase per design evaluation, although it potentially allows an optimization to complete in fewer iterations due to the smaller number of nonlinear constraints to satisfy.

With the control surface layout being defined based on hinge lines spanning specified fractions of the local chord and span, another unofficial MPhys builder, for now coined the “planform builder”, is currently required. This builder defines the planform outline based on a VLM/DLM mesh of the lifting body. The outline, which must also move through the geometry parameterization’s deformation process, is duplicated on all Message Passing Interface (MPI) ranks in the MPhys scenario so that a hinge line radius (i.e., the x-distance from the hinge line) may be computed for a given aerodynamic or structural node regardless of surface node parallelization. Each node’s x- and z- coordinate is then rotated about the hinge line using the node’s hinge radius, i.e.,

$$\Delta x = r (\cos(\theta) - 1) \quad (1)$$

and

$$\Delta z = r \sin(\theta), \quad (2)$$

where r and θ are the hinge radius and control surface deflection angle, respectively. Figure 1 shows an illustration of hinge line radius on the VLM mesh of the AGARD 445.6 wing, where the hinge line is positioned at 75% chord, spanning from 50% to 90% span, while the blend width is set to 5% of the span. Overall, there are a few limitations with the current implementation: (1) a panel mesh is required for the initial geometry, (2) control surfaces are assumed to be on horizontal (x-y plane) lifting bodies with nodal deformations being applied only in the x-z plane, and (3) the control surfaces are presumed to rotate the trailing edge and not the leading edge. The first issue requires slightly more work for the user but is not much of a hindrance otherwise; the latter two issues are likely resolveable although modifying analytic radius derivatives may be somewhat cumbersome.

When considering multiple control surfaces on the same lifting body, and when the optimizer is allowed to change control surface location, one must ensure that multiple control surfaces do not lie on top of each other. While chord fractions may be handled directly by an optimizer, the hinge line lengths along the span in such a case are instead

handled by an intermediate OpenMDAO component. The component uses control surface span and gap inputs; the summed spans and gaps are then normalized by the lifting body's span to provide hinge line span ranges without any overlap. Geometric constraints on maximum control surface span (e.g., 10% span) and minimum control surface gap (set so that blended regions do not overlap) can then be enforced by the optimizer.

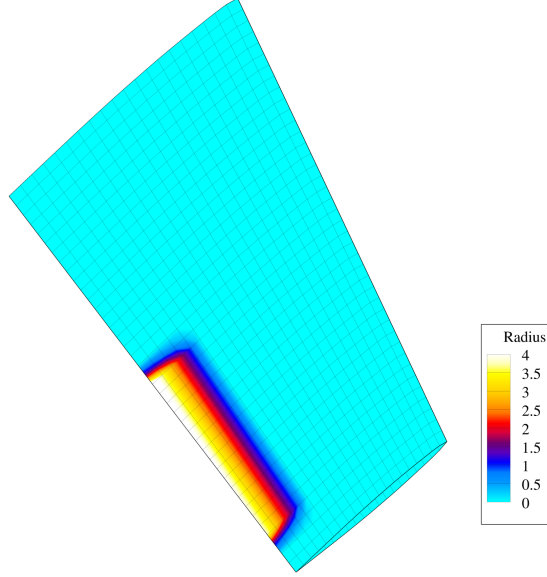


Fig. 1 Sample control surface radius (i.e., distance from hinge line) on the AGARD 445.6 wing with hinge line at 75% chord spanning from 50% to 90% span. The s-shaped blend function on either side of the control surface is set to 5% span.

B. Flutter analysis via Rogers Rational Function Approximation and the p -method

When dynamic ASE analysis is desired, an implementation of Rogers RFA and the p -method are used to represent the state-space system. The equations of motion (EOM) can be written as

$$\mathbf{M}\ddot{\eta} + \mathbf{C}\dot{\eta} + \mathbf{K}\eta = q\mathbf{A}^{ss}\eta + q\mathbf{A}^{cs}\theta, \quad (3)$$

where $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$, are the generalized mass, damping, and stiffness matrices, η are the generalized displacements of the modes, $\mathbf{A}^{ss}$ are the structure-on-structure Generalized Aerodynamic Forces (GAFs), $\mathbf{A}^{cs}$ are the controls-on-structure GAFs, and q is the dynamic pressure. η and θ are presumed to take the form

$$\eta = \bar{\eta}e^{pUt/b} \quad (4)$$

and

$$\theta = \bar{\theta}e^{pUt/b}, \quad (5)$$

where p , U , t , and b are the nondimensional Laplacian, flow velocity, time, and reference semichord length. Meanwhile, the overline symbol represents the initial or steady state value. The EOM can therefore be written as

$$\left[\frac{p^2 U^2}{b^2} \mathbf{M} + \frac{pU}{b} \mathbf{C} + \mathbf{K} - q\mathbf{A}^{ss} \right] \bar{\eta} = q\mathbf{A}^{cs} \bar{\theta}. \quad (6)$$

Rogers approximation [10] is given as

$$\mathbf{A}^{ss} = \mathbf{A}_0^{ss} + \mathbf{A}_1^{ss}p + \mathbf{A}_2^{ss}p^2 + \sum_{i=1}^{N_s} \left(\frac{\mathbf{A}_{i+2}^{ss}p}{p + \gamma_i^{ss}} \right) \quad (7)$$

and

$$\mathbf{A}^{\text{cs}} = \mathbf{A}_0^{\text{cs}} + \mathbf{A}_1^{\text{cs}} p + \mathbf{A}_2^{\text{cs}} p^2 + \sum_{i=1}^{N_c} \left(\frac{\mathbf{A}_{i+2}^{\text{cs}} p}{p + \gamma_i^{\text{cs}}} \right), \quad (8)$$

where these approximate GAFs are computed using a least squares procedure, with γ being the user-defined reduced frequency lag points (chosen to be $[0.2, 0.4, 0.6, 0.8, 1, 1.2]$ in later results). With the aerodynamic lag states being described as

$$\eta_i^{\text{A}} = \bar{\eta} \frac{p}{p + \gamma_i^{\text{ss}}} \quad (9)$$

and

$$\theta_i^{\text{A}} = \bar{\theta} \frac{p}{p + \gamma_i^{\text{cs}}}, \quad (10)$$

and conversion into the Laplace domain with

$$s = \frac{pU}{b}, \quad (11)$$

the EOM can be written as

$$\mathbf{A}\dot{\mathbf{x}} + \mathbf{B}\mathbf{x} = \mathbf{C}_1\theta + \mathbf{C}_2\dot{\theta} + \mathbf{C}_3\ddot{\theta}, \quad (12)$$

where

$$\mathbf{x} = \begin{pmatrix} \bar{\eta} \\ \bar{\eta}s \\ \bar{\eta}_1^{\text{A}} \\ \vdots \\ \bar{\eta}_N^{\text{A}} \\ \bar{\theta}_1^{\text{A}} \\ \vdots \\ \bar{\theta}_N^{\text{A}} \end{pmatrix}. \quad (13)$$

These state-space matrices are

$$\mathbf{A} = \begin{bmatrix} \frac{b}{U} \mathbf{I} & & & & & & & \\ & \mathbf{M} - \frac{1}{2} \rho b^2 \mathbf{A}_2^{\text{ss}} & & & & & & \\ & & \frac{b}{U} \mathbf{I} & & & & & \\ & & & \ddots & & & & \\ & & & & \frac{b}{U} \mathbf{I} & & & \\ & & & & & \frac{b}{U} \mathbf{I} & & \\ & & & & & & \ddots & \\ & & & & & & & \frac{b}{U} \mathbf{I} \end{bmatrix}, \quad (14)$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} & -\frac{b}{U} \mathbf{I} & & & & & & \\ \mathbf{K} - q \mathbf{A}_0^{\text{ss}} & \mathbf{C} - \frac{1}{2} \rho b U \mathbf{A}_1^{\text{ss}} & -q \mathbf{A}_3^{\text{ss}} & \dots & -q \mathbf{A}_N^{\text{ss}} & -q \mathbf{A}_3^{\text{cs}} & \dots & -q \mathbf{A}_N^{\text{cs}} \\ & -\frac{b}{U} \mathbf{I} & \gamma_1^{\text{ss}} \mathbf{I} & & & & & \\ & \vdots & & \ddots & & & & \\ & -\frac{b}{U} \mathbf{I} & & & \gamma_N^{\text{ss}} \mathbf{I} & & & \\ & & & & & \gamma_1^{\text{cs}} \mathbf{I} & & \\ & & & & & & \ddots & \\ & & & & & & & \gamma_N^{\text{cs}} \mathbf{I} \end{bmatrix}, \quad (15)$$

$$\mathbf{C}_1 = \begin{bmatrix} \mathbf{0} \\ q\mathbf{A}_0^{\text{cs}} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{C}_2 = \begin{bmatrix} \mathbf{0} \\ \frac{1}{2}\rho b U \mathbf{A}_1^{\text{cs}} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \frac{b}{U} \mathbf{I} \\ \vdots \\ \frac{b}{U} \mathbf{I} \end{bmatrix}, \quad \text{and} \quad \mathbf{C}_3 = \begin{bmatrix} \mathbf{0} \\ \frac{1}{2}\rho b^2 \mathbf{A}_2^{\text{cs}} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}. \quad (16)$$

One can approximate the relationship between the control surface command input δ and the actual control surface deflections θ as

$$\ddot{\theta} + \frac{a^3}{4} \omega_{\max} \ddot{\theta} + \frac{a^3}{2} \omega_{\max}^2 \dot{\theta} + a^3 \omega_{\max}^3 \theta = a^3 \omega_{\max}^3 \delta, \quad (17)$$

where $\omega_{\max}$ is the largest natural frequency of the structure and a is some multiplier (e.g., 3). One can then write

$$\dot{\mathbf{y}} = \mathbf{A}_y \mathbf{y} + \mathbf{B}_y \delta, \quad (18)$$

where $\theta \approx \delta$ if the actuation frequency of the command input is much less than $\omega_{\max}$. Here,

$$\mathbf{A}_y = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \\ -a^3 \omega_{\max}^3 & -\frac{a^3}{2} \omega_{\max}^2 & -\frac{a^3}{4} \omega_{\max} \end{bmatrix}, \quad \mathbf{B}_y = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ a^3 \omega_{\max}^3 \end{bmatrix}, \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} \theta \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix}. \quad (19)$$

Combining with the previous EOM yields

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{y}} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} -\mathbf{B} & \mathbf{C}_1 \mathbf{T}_1 + \mathbf{C}_2 \mathbf{T}_2 + \mathbf{C}_3 \mathbf{T}_3 \\ \mathbf{0} & \mathbf{A}_y \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} + \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{B}_y \end{bmatrix} \delta, \quad (20)$$

where

$$\mathbf{T}_1 = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \mathbf{T}_2 = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix}, \quad \text{and} \quad \mathbf{T}_3 = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}. \quad (21)$$

These EOM take the form

$$\dot{\mathbf{z}} = \bar{\mathbf{A}} \mathbf{z} + \bar{\mathbf{B}} \delta, \quad (22)$$

with the command inputs presumed to be proportional to the gain matrix $\mathbf{G}$,

$$\delta = -\mathbf{G} \mathbf{z}, \quad (23)$$

hence

$$\dot{\mathbf{z}} = (\bar{\mathbf{A}} - \bar{\mathbf{B}} \mathbf{G}) \mathbf{z}. \quad (24)$$

An eigenvalue solve subsequently yields stability of the closed-loop ASE system, with eigenvalues corresponding to structural modes, aerodynamic lag states, and actuation terms.

C. Controller design via linear-quadratic regulator

The controller's gain matrix ($\mathbf{G}$ in Eqs. 23 and 24) is computed using the Linear-Quadratic Regulator (LQR) method at a particular flow condition (although gain scheduling could be added without much additional effort). Before carrying this out, the RFA method's lag states are removed from the state-space system so that only structural modes

remain. Doing so provides a more well-conditioned system for LQR while also removing the non-physical, and therefore immeasurable, lag states. This model reduction is done using the approach of Hjartarson et al. [11], which partitions the full state-space system in Eq. 22 as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \quad (25)$$

and

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \end{bmatrix}. \quad (26)$$

Here, the first $2N_{\text{modes}}$ rows and columns of $\mathbf{A}$ correspond to flexible mode states and their time derivatives (see Eq. 13) and the first $2N_{\text{modes}}$ rows of $\mathbf{B}$ similarly corresponding to these modes. The reduced state-space matrices are then computed as

$$\mathbf{A}_{\text{re}} = \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21} \quad (27)$$

and

$$\mathbf{B}_{\text{re}} = \mathbf{B}_1 - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{B}_2. \quad (28)$$

Note that when computing analytic derivatives, partial derivatives of the matrices with respect to an arbitrary input z are readily available as

$$\mathbf{A}_{\text{re}z} = \mathbf{A}_{11z} - \mathbf{A}_{12z}\mathbf{A}_{22}^{-1}\mathbf{A}_{21} - \mathbf{A}_{12} \left(-\mathbf{A}_{22}^{-1}\mathbf{A}_{22z}\mathbf{A}_{22}^{-1} \right) \mathbf{A}_{21} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21z} \quad (29)$$

and

$$\mathbf{B}_{\text{re}z} = \mathbf{B}_{1z} - \mathbf{A}_{12z}\mathbf{A}_{22}^{-1}\mathbf{B}_2 - \mathbf{A}_{12} \left(-\mathbf{A}_{22}^{-1}\mathbf{A}_{22z}\mathbf{A}_{22}^{-1} \right) \mathbf{B}_2 - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{B}_{2z}. \quad (30)$$

After computing the reduced system to remove lag states, LQR then computes the optimal feedback controller that minimizes the quadratic cost function

$$J = \int_0^\infty \left(x_{re}(t)^T \mathbf{Q} x_{re}(t) + u_{re}(t)^T \mathbf{R} u_{re}(t) \right) dt, \quad (31)$$

where $\mathbf{Q}$ and $\mathbf{R}$ are diagonal matrices containing state and input weights (both potentially designable by the optimizer) while the reduced state-space system is described by

$$\dot{x}_{re}(t) = \mathbf{A}_{\text{re}}x_{re}(t) + \mathbf{B}_{\text{re}}u_{re}(t). \quad (32)$$

The system can ultimately be reduced down to an Algebraic Riccati Equation (ARE) with a solution $\mathbf{X}$,

$$\mathbf{A}_{\text{re}}^T \mathbf{X} + \mathbf{X} \mathbf{A}_{\text{re}} - \mathbf{X} \mathbf{B}_{\text{re}} \mathbf{R}^{-1} \mathbf{B}_{\text{re}}^T \mathbf{X} + \mathbf{Q} = \mathbf{0}, \quad (33)$$

which yields $\mathbf{X}$ (the upper bound on the cost function J being the sum of $\mathbf{X}$'s diagonals) as well as the resulting gain matrix, $\mathbf{G}_{\text{re}} = \mathbf{B}_{\text{re}}^T \mathbf{X}$. The full system's gain matrix, used in the Rogers RFA/ p -method solution, can be obtained by padding this reduced gain matrix with zeros.

The sensitivity of $\mathbf{Q}$ with respect to an arbitrary parameter z can be written as [12]

$$\begin{aligned} & \mathbf{X}_z (\mathbf{A}_{\text{re}} - \mathbf{B}_{\text{re}} \mathbf{G}_{\text{re}}) + (\mathbf{A}_{\text{re}} - \mathbf{B}_{\text{re}} \mathbf{G}_{\text{re}})^T \mathbf{X}_z \\ & + \{ \mathbf{X} \mathbf{A}_{\text{re}z} + \mathbf{A}_{\text{re}z}^T \mathbf{X} + \mathbf{Q}_z - \mathbf{X} (\mathbf{B}_{\text{re}z} \mathbf{R}^{-1} \mathbf{B}_{\text{re}}^T - \mathbf{B}_{\text{re}} \mathbf{R}^{-1} \mathbf{R}_z \mathbf{R}^{-1} \mathbf{B}_{\text{re}}^T + \mathbf{B}_{\text{re}} \mathbf{R}^{-1} \mathbf{B}_{\text{re}z}^T) \mathbf{X} \} = \mathbf{0} \end{aligned} \quad (34)$$

For parameters besides input and state weights, the sensitivity of $\mathbf{Q}$ with respect to the parameter z simplifies to

$$\mathbf{Q}_z = \mathbf{A}_{\text{re}z}^T \mathbf{X} + \mathbf{X} \mathbf{A}_{\text{re}z} - \mathbf{X} \left(\mathbf{B}_{\text{re}z} \mathbf{R}^{-1} \mathbf{B}_{\text{re}}^T + \mathbf{B}_{\text{re}} \mathbf{R}^{-1} \mathbf{B}_{\text{re}z}^T \right) \mathbf{X}. \quad (35)$$

The sensitivity of $\mathbf{X}$ can then be found by solving the Lyapunov equation

$$(\mathbf{A}_{\text{re}} - \mathbf{B}_{\text{re}} \mathbf{G}_{\text{re}})^T \mathbf{X}_z + \mathbf{X}_z (\mathbf{A}_{\text{re}} - \mathbf{B}_{\text{re}} \mathbf{G}_{\text{re}}) + \mathbf{Q}_z = \mathbf{0}. \quad (36)$$

Finally, the sensitivity of the reduced gain matrix with respect to the parameter z is

$$\mathbf{G}_{\text{re}z} = \mathbf{R}^{-1} \left(\mathbf{B}_{\text{re}z}^T \mathbf{X} + \mathbf{B}_{\text{re}}^T \mathbf{X}_z \right) \quad (37)$$

The sensitivity of the controller cost function bound can also be computed as the sum of the diagonals of $\mathbf{X}_z$. Implemented in an OpenMDAO explicit component with its matrix-free API, sensitivities of the gain matrix and cost function bound are computed for all inputs: reference length, generalized structural matrices, GAFs, state and input weights, and the dynamic pressure at which the gain matrix is computed. This therefore requires numerous solutions of the ARE and Lyapunov equations; this could be somewhat time-consuming if one was to use the full system, especially if very many lag states are included in the analysis, but fortunately the model reduction reduces this cost significantly.

III. Baseline Results of the AGARD 445.6 Wing

This section covers several results of the baseline (i.e., non-optimized) AGARD 445.6 wing to illustrate the current implementation of static and dynamic ASE in MPhys/OpenMDAO. Doing so demonstrates the implementation for a particular design without optimization. Firstly, static aeroelastic solutions with a deflected control surface are shown in Sect. III.A. To solve for the deflected structure, PyShell is coupled to either an in-house VLM or FUN3D/SFE's inviscid solver. Section III.B, meanwhile, illustrates flutter results. Open-loop flutter results from the current Rogers RFA flutter solver are compared to the authors' more well-established p - k method implementation [5]. Closed-loop flutter results are also shown using the LQR controller design detailed in the previous section.

A. Static control surface deflection

Figure 2 illustrates static aeroelastic deformation of the AGARD 445.6 wing at a 3 degree angle of attack and Mach 0.975. The control surface geometry matches that of Fig. 1, with the hinge line being positioned at 75% chord, spanning 50% to 90% of the span, and the width of the s-shaped blend function set to 5% of the span; the rotation here is set to 10 degrees downward. Relative to the case with no control surface, this control surface deflection ultimately increases tip displacement by 98% and a Kreisselmeier-Steinhauser (KS) [13, 14] stress aggregate by 69%. For context, KS aggregates condense numerous values into a single, continuous response with a scaling parameter providing a tradeoff between conservativeness and smoothness; in this case, the KS stress aggregate is approximately equal to the maximum von Mises stress. Nonetheless, one can imagine that during an optimization, an upward deflection could potentially help reduce outboard wing loading during a maneuver, thus allowing for a more lightweight structure. It should be noted that the internal structure of the control surface mechanism is obviously not realistic, as it excludes any sort of hydraulic system. A later static aeroelastic optimization with MLA neglects this structure entirely and simply models the wingbox structure.

B. Open- and closed-loop flutter analyses

Figure 3 illustrates open-loop flutter analysis results using an in-house DLM implementation coupled to PyShell, where Mach number is set to 0.95. In particular, damping and frequency curves versus flow dynamic pressure are compared between the present Rogers RFA p -method and the authors' p - k method implementation described in Jacobson et al. [5]. As expected, the flutter onset dynamic pressure matches well between the two methods, but the discrepancy between them becomes larger away from the x-axis. This is to be expected, as the p - k method presumes neutrally damped harmonic oscillations whereas the p -method does not; RFA approximation, in addition, may lead to differences depending on the number of aerodynamic lag roots chosen. Regardless, this close match of predicted flutter onset demonstrates that the p -method implementation is sufficient for analysis and optimization.

Figure 4 illustrates the results of a closed-loop flutter analysis using the previously described LQR controller design compared to the equivalent open-loop results. Here, there are two control surfaces that span from 20-40% and 60-80%

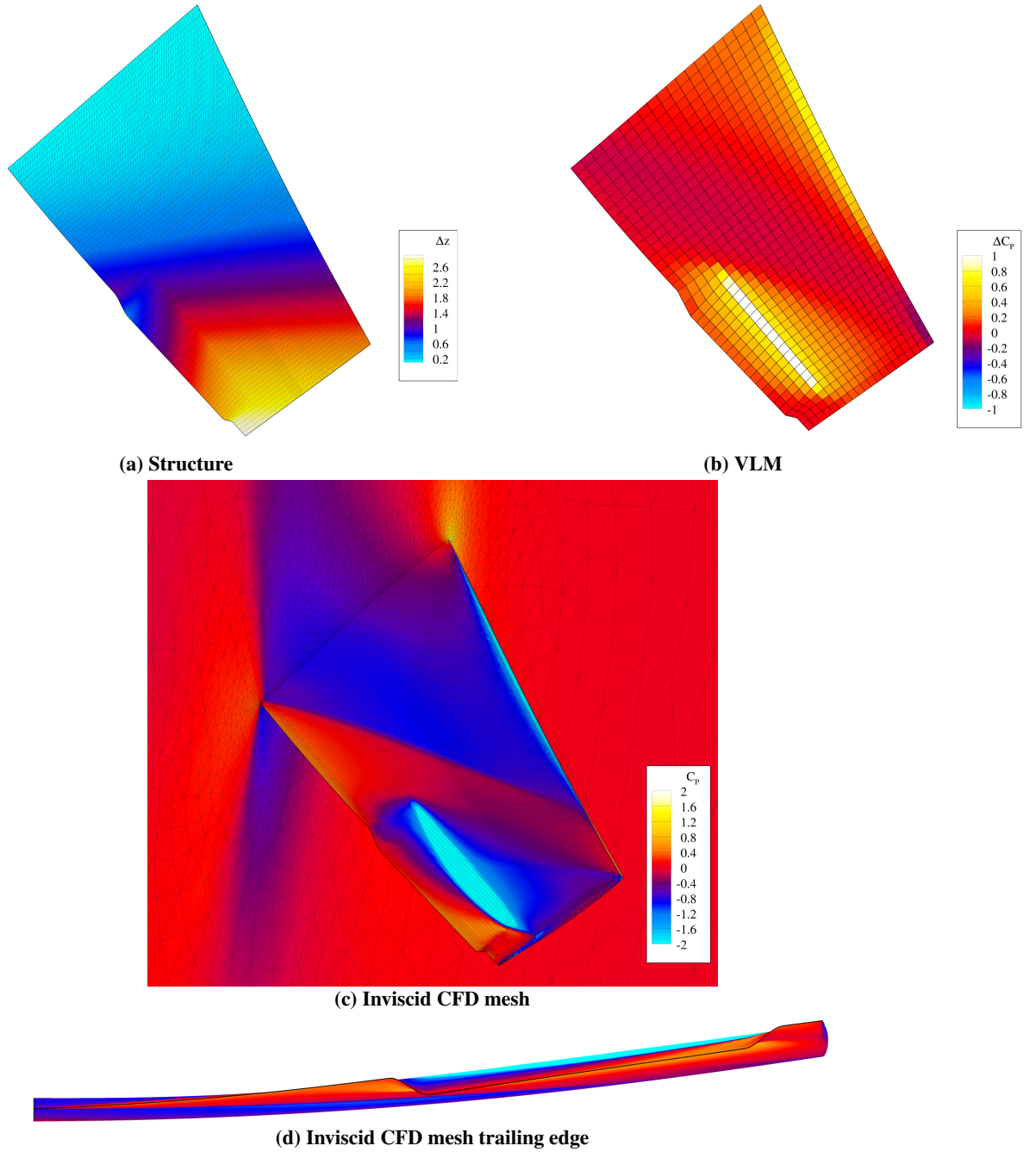


Fig. 2 Static aeroelastic deflection with an arbitrary control surface, deflected by -10° , at a Mach number of 0.975 and angle of attack of 3° . The structure shown is coupled to VLM aerodynamics.

span. The hinge line is positioned at 80% chord while the blend width is set to 4% of the span. The controller gain matrix is computed at a dynamic pressure of 0.6 psi, with state and input weights set to ones and 10^{-4} , respectively. Doing so ensures a stable response of the reduced system at that particular dynamic pressure, but makes no guarantees at other flow conditions; in theory, there is also only a guarantee of stability of the reduced and not full system (i.e., the system containing aerodynamic lag and actuation states). Thus, the closed-loop flutter constraint is still required to ensure a stable response in the full state-space system and the full dynamic pressure range of interest.

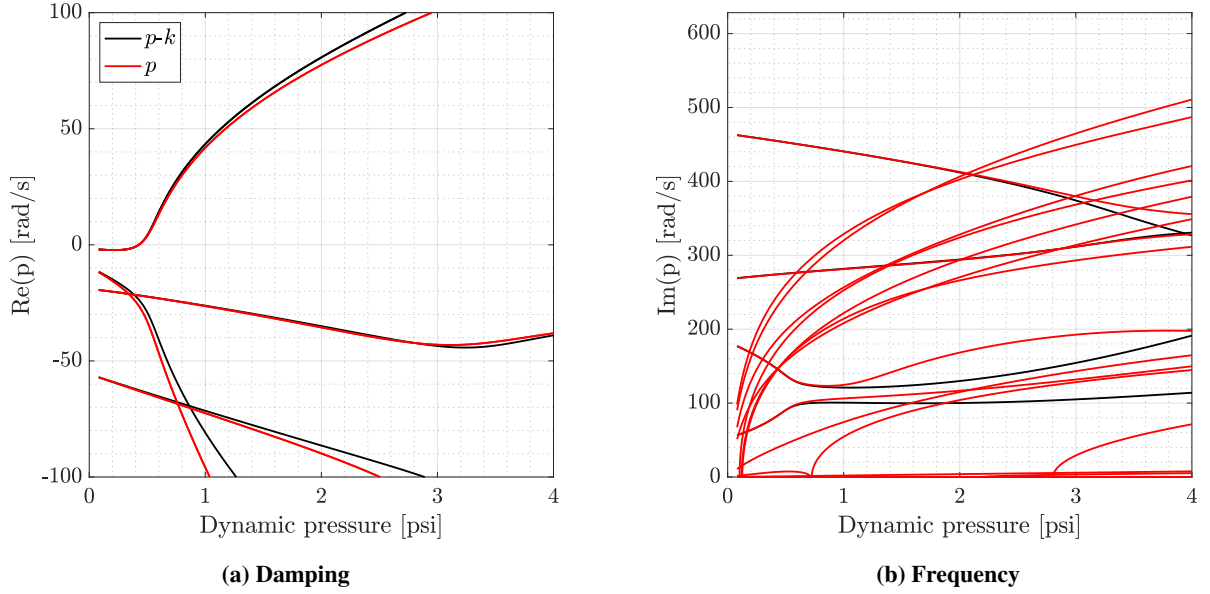


Fig. 3 Comparison between the p - k method from past literature and the Rogers RFA p -method from the present work, where p is the system's eigenvalues. Note that the p -method here includes aerodynamic lag roots, in addition to structural modes, whereas the p - k method has no notion of lag roots.

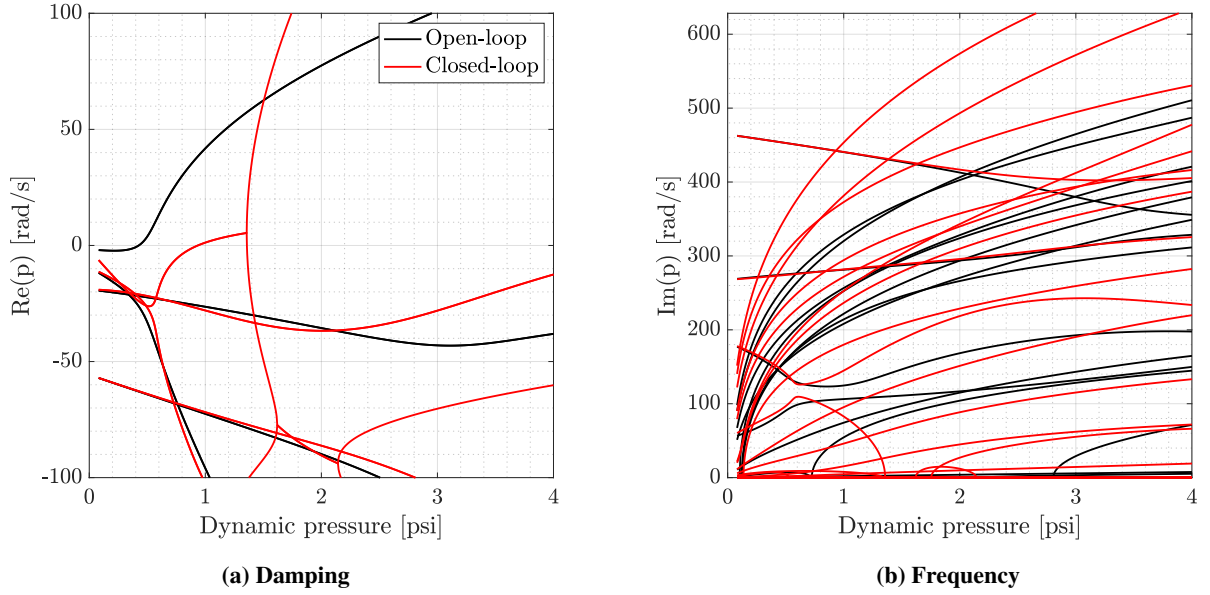


Fig. 4 Open- and closed-loop flutter analyses at Mach 0.95 using DLM aerodynamics. The closed-loop LQR controller is computed at a dynamic pressure of 0.6 psi.

IV. Sample Optimizations with Static and Dynamic Aeroservoelastic Modeling

This section presents aeroservoelastic optimizations using the described MPhys/OpenMDAO-based methodologies. Section IV.A shows a static multi-point optimization of the Common Research Model (CRM) wing with MLA, where FUN3D's Euler solver is coupled to PyShell. Section IV.B then illustrates an AGARD wing optimization subject to open- and closed-loop flutter constraints, where DLM or LFD aerodynamics are coupled to the same structural solver.

A. Static aeroelastic multi-point optimization of the CRM wing with maneuver load alleviation

The CRM optimization case of Thelen et al. [15], modeled after previous literature by Brooks et al. [16], is adapted such that the fuel burn objective function is now based on the average fuel burn across several flight conditions: the on-design cruise condition (which excludes control surfaces) and several off-design cruise conditions with designable deflections of two control surfaces. Stress and load factor constraints are now enforced at two different 2.5g pull-up maneuver conditions: one with an MLA system (i.e., including designable control surface deflections) and one without, where the latter’s KS stress constraint is relaxed but still present in case the MLA system fails. These two stress constraints, in particular, have upper limits that are based on material safety factors of 50 and 33% with and without the MLA system, respectively. Table 1 summarizes these different flight conditions.

MPhys remote components [15] are used to parallelize the two pull-up maneuver conditions and the cruise conditions across three different High-performance Computing (HPC) jobs. The HPC job containing on- and off-design cruise conditions itself uses the MPhys MultipointParallel class to evaluate them in parallel. A full function or derivative evaluation requested by the optimizer is evaluated on a login node (with 3 MPI ranks, to make use of an OpenMDAO ParallelGroup) so that the optimization can span across numerous HPC jobs with 72-hour time limits on NASA Langley’s K-cluster.

The flutter and gust constraints mentioned in Thelen et al. [15] are not included in this particular optimization. Additional design variables include each flight condition’s two control surface deflections (at three off-design cruise conditions and the 2.5g pull-up maneuver condition), with associated angles of attack also being designable. Control surface positioning design variables (span, gap, and chord fraction) are also included so that the two control surfaces may be positioned by the optimizer. With the blend width being set to 4% span, the geometric constraints on gaps are limited to 8% or more, while span constraints are set so that the surfaces cannot span more than 10% of the wingspan. Chord fraction design variables have lower limits of 0.75 and initial values of 0.8, so that they cannot exceed 25% of the local chord. Table 2 summarizes the design variables and constraints

Figure 5 shows the convergence histories of the objective function, maximum constraint violation, and first-order optimality during the optimization. The SLSQP algorithm in pyOptSparse [17] is used as the optimizer here, with the objective convergence tolerance set to 10^{-5} (chosen rather arbitrarily based on previous optimizations). Figure 6

Table 1 Flight conditions used for the multi-point CRM optimization.

Description	Mach number	Altitude	Includes control surfaces?
1g cruise, on-design	0.85	37,000 ft	No
1g cruise, off-design 1	0.825	35,000 ft	Yes
1g cruise, off-design 2	0.85	33,000 ft	Yes
1g cruise, off-design 3	0.875	39,000 ft	Yes
2.5g pull-up (safety factor 0.33)	0.64	Sea level	No
2.5g pull-up, MLA (safety factor 0.5)	0.64	Sea level	Yes

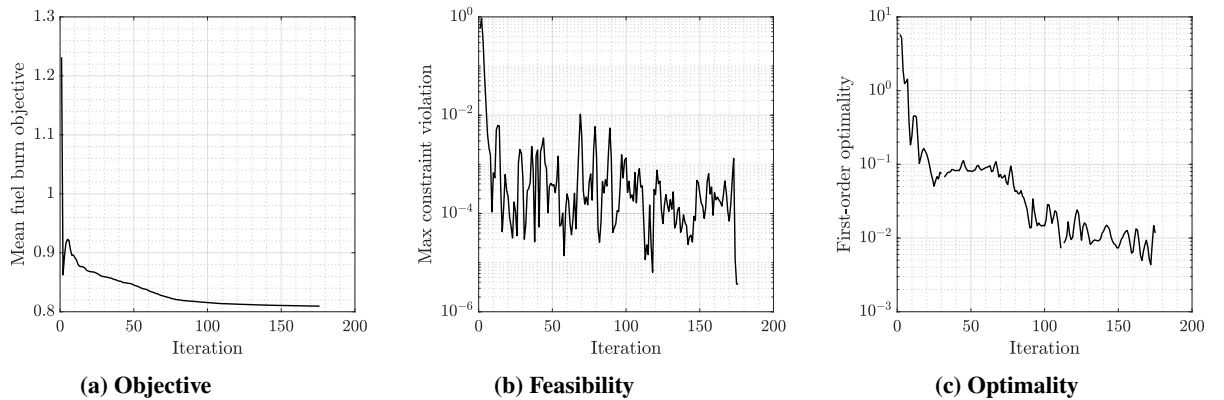


Fig. 5 Summary of CRM optimization results with static maneuver load alleviation.

Table 2 Objective, design variables, and constraints for the CRM optimization problem.

	Description	Specification	Quantity
minimize:	Average cruise fuel burn		
with respect to:	Root chord differential	$\in [-3, 3]$ m	1
	Tip chord differential	$\in [-1, 1]$ m	1
	Span differential	$\in [-10, 10]$ m	1
	Sweep differential	$\in [-10, 10]$ deg	1
	Wing thickness	$t_{wing} \geq 0.5 \times t_{wing,0}$	5
	Wing twist differential	$\in [-10, 10]$ deg	5
	Fuel scaler	$\in [0, 1]$	1
	Shell thickness	$\in [0.003, 0.03]$ m	240
	Stiffener thickness	$\in [0.003, 0.03]$ m	240
	Stiffener height	$\in [0.03, 0.15]$ m	240
	Static angle of attack	$\in [-10, 10]$ deg	6
	Control surface deflection	$\in [-10, 10]$ deg	8
	Control surface chord fraction	$\in [0.75, 0.95]$	4
	Control surface gap	$\in [0, 1]$	3
	Control surface span	$\in [0, 1]$	2
	Total		758
subject to:	Fuel mismatch	$= 0$	1
	Static load factor	$= [1, 1, 1, 1, 2.5, 2.5]^T$	6
	Wing area	$\geq 205.8 \text{ m}^2$	1
	Available fuel mass	$\geq 63,464.28 \text{ kg}$	1
	Trailing edge spar depth	$\geq 0.19 \text{ m}$	1
	Stiffener aspect ratio	$KS_{AR} \leq 1$	1
	Stiffener vs. shell thickness	$KS_{t/t_s} \leq 1$	1
	Pull-up stress	$1.33KS_{\sigma} \leq 1$	1
	Pull-up stress, MLA	$1.5KS_{\sigma} \leq 1$	1
	Control surface gap	$\geq 8\% \text{ span}$	3
	Control surface span	$\leq 10\% \text{ span}$	2
	Total		19

shows convergence histories of design variables and constraints directly relating to control surfaces. Based on these convergence histories, the optimization stopping tolerance may have been set slightly too high, as many control surface metrics appear to still be changing somewhat. Nonetheless, the change in the normalized objective function has changed by less than the set tolerance, with the maximum constraint violation being below a similar value. The optimality has dropped by around three orders of magnitude by this point. Based on Fig. 6, the span design variables follow similar paths, with both control surface spans stopping at the maximum span of 10%. The chord fractions, in contrast, are not maximized while the gap constraints are also not active (which is perhaps not surprising when using only two control surfaces).

Finally, Fig. 7 illustrates the final control surface positioning on the baseline and optimized planform outlines. Also shown is a summary of all the control surface deflection and angle of attack variables. Based on this figure, the optimizer has chosen to utilize inboard and outboard control surfaces differently. In particular, the outboard control surface is deflected upwards so as to shift more of the wing loading inboard; the inboard control surface is deflected downwards to increase the lift to meet the 2.5g load factor constraint (while not increasing bending moment much due to the short moment arm). Interestingly, the MLA stress constraint is just slightly inactive, although this may change if given more time for the optimizer to converge.

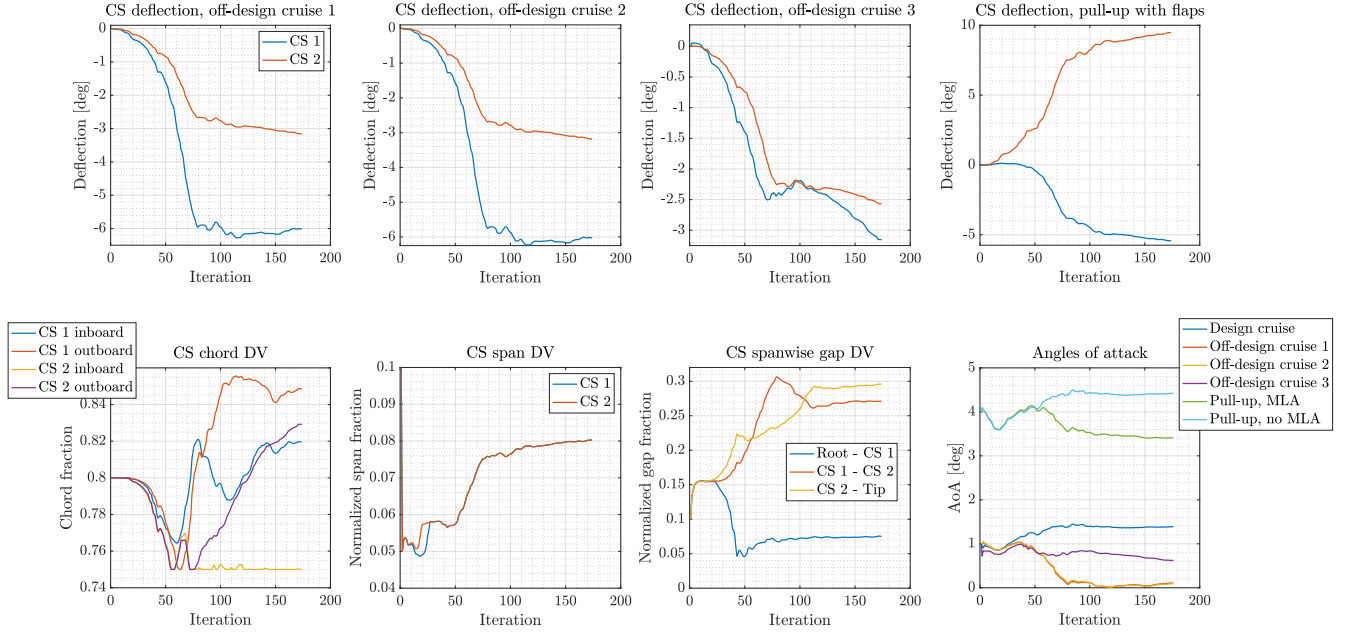


Fig. 6 Convergence histories of control surface (CS)-related design variables (DV) and angles of attack.

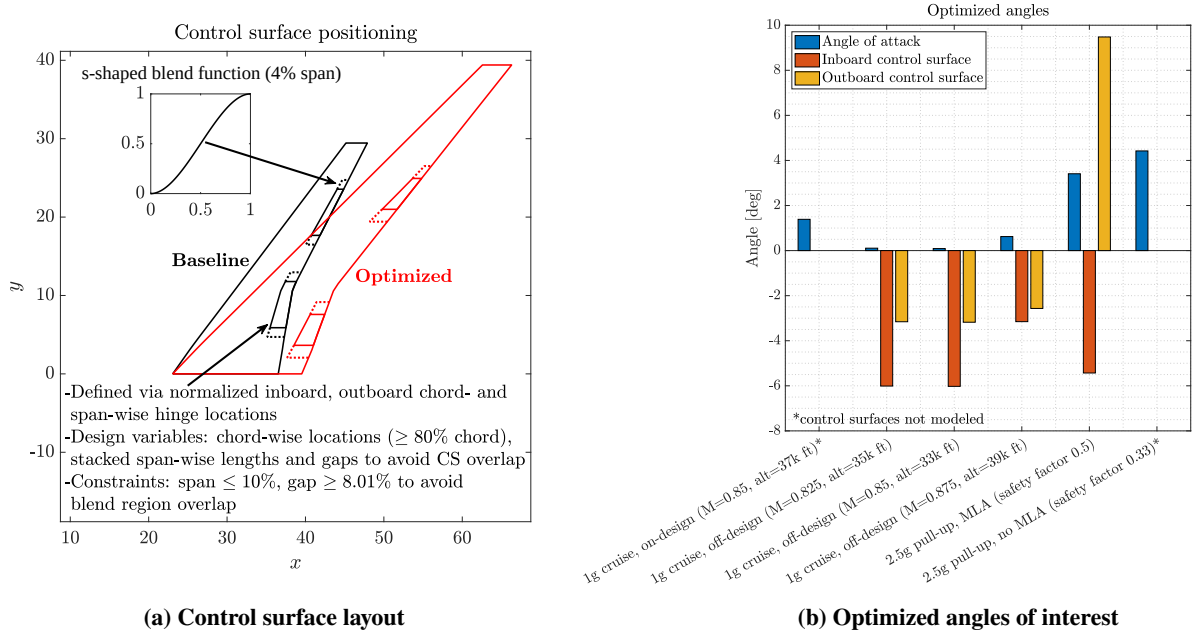


Fig. 7 Summary of control surface layout and angles from the static ASE optimization. Dotted lines in the first figure indicate the region where the s-shaped blend function is applied.

B. AGARD wing optimization including open- and closed-loop flutter constraints

The AGARD 445.6 wing optimization is based on that of Jacobson and Stanford [5], which minimizes the structural mass at Mach 0.5 subject to 100 structural thickness design variables in addition to four planform variables: root chord, tip chord, sweep, and span. A minimum flutter onset dynamic pressure q_f is enforced, while an equality constraint ensures that the planform area is the same as the initial design's.

The primary difference is the presence of an AFS system, which utilizes two control surfaces that can be moved by the optimizer. The closed-loop minimum q_f is set to 0.6 psi while an open-loop value of 0.5 psi is chosen; the open-loop constraint is thus somewhat relaxed (compared to a CS-free case with $q_f \geq 0.6$ psi) but still present in case the AFS system fails. These minimum flutter constraints are enforced using KS damping aggregates via a polynomial “keep-out zone” as described in Jacobson and Stanford [5]. For the controller design, the LQR input and state weights, in addition to dynamic pressure, may be designed by the optimizer with log-scale bounds of $[-6, 0]$ and linear-scale bounds of $[0.1, 2.0]$, respectively. An arbitrary upper limit on the LQR controller cost function bound is set to 0.005. Like the static ASE CRM case, control surfaces are allowed to be up to 25% of the local chord and 10% of the total span; meanwhile, the blended regions adjacent to the control surfaces are set to 4% span so the gaps between them are constrained to be 8% or larger.

The optimization problem is solved using both DLM and inviscid LFD aerodynamics. To obtain the LFD solution more cheaply, the optimization is first evaluated using a very coarse CFD mesh; the optimization is subsequently restarted from the final design using a finer mesh. For context, the DLM, coarse mesh LFD, and finer mesh LFD cases are carried out on 1, 2, and 6 HPC compute nodes, respectively. Each design iteration (including all responses and derivatives) using coarse and finer mesh LFD is roughly 1 and 2 orders of magnitude more costly than a DLM design iteration, respectively.

Figure 8 shows the convergence histories of this dynamic ASE optimization case with open- and closed-loop flutter constraints, including the objective function, maximum constraint violation, and first-order optimality. The DLM objective function levels off at around 20 iterations, at which point the optimality steeply drops and the objective tolerance of 10^{-4} is immediately satisfied. The coarse mesh LFD result shows a similar trend in optimality, corresponding to a slightly larger mass objective reduction. Switching to the finer mesh incurs a slight mass penalty while quickly finding a feasible design; optimality is also quickly reduced to a similar order of magnitude as the coarse mesh result.

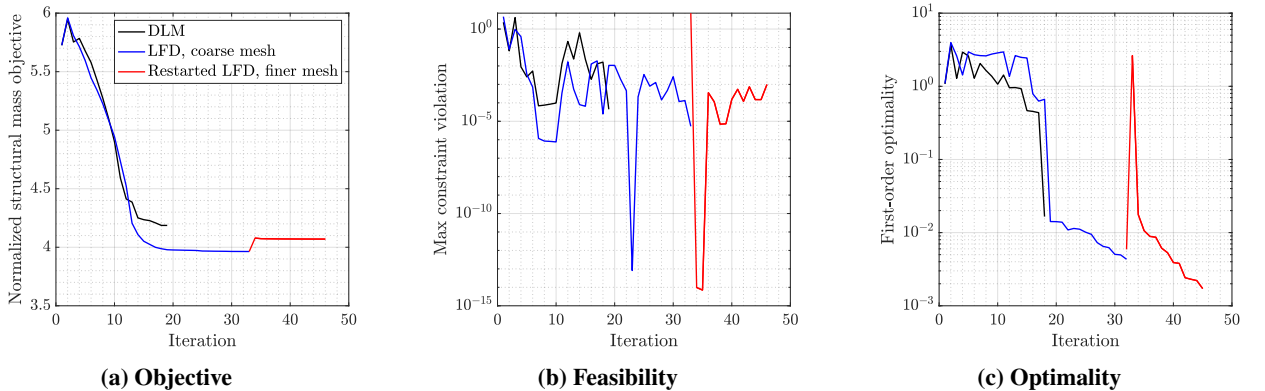


Fig. 8 Summary of AGARD optimization results with open- and closed-loop flutter constraints.

Figure 9 shows convergence histories of selected inequality constraints, including open- and closed-loop flutter responses as well as the LQR controller cost. Meanwhile, Fig. 10 shows the final flutter responses using the three different aerodynamic fidelities. Figure 11, in addition, shows the convergence histories of the LQR-related design variables. The DLM optimization yields a design with only an active closed-loop flutter constraint. In the corresponding damping diagram, the open- and closed-loop responses are essentially the same, with the open-loop response satisfying the closed-loop dynamic pressure limit of 0.6 psi. This essentially means that the control surfaces can go unused and that there will be no benefit of including the AFS system. This may occur due to the various parameters chosen for the optimization case, including the q_f limits, LQR input and state weight bounds, and the controller cost limit. Further studies are needed to characterize the tradeoff between final structural mass and the limiting cost function bound.

The LFD optimizations, on the other hand, produce designs with active open-loop constraints and just slightly

inactive closed-loop constraints, with the controller cost constraint also being slightly inactive. Note, however, that the reference value for the cost constraint was left as 1.0 due to an oversight; normalizing the constraint to be $O(1)$ may have led to a converged constraint value of closer to 0, although this was not observed to be the case with a separate DLM optimization. Nonetheless, as seen in other studies by the authors (excluded for brevity), there is generally a trade-off between controller cost limit and state weights, with lower cost limits requiring smaller weights. Once the input weights reach their lower bounds (which is nearly true for LFD cases as shown in Fig. 11), the closed-loop flutter constraint tends to become more active. At the same time, the cost constraint tends to incur an increasing mass penalty with a decreasing upper limit, at least until the open-loop response coincides with the closed-loop one, as is seen with DLM. The state weights, of course, also have an effect on the closed-loop response; fixing these parameters and only varying input weights may have simplified the problem and caused all three constraints shown to be active. Realistic controller design may also involve other requirements (e.g., maximum rate limits); inclusion of such additional constraints may further restrict the design space.

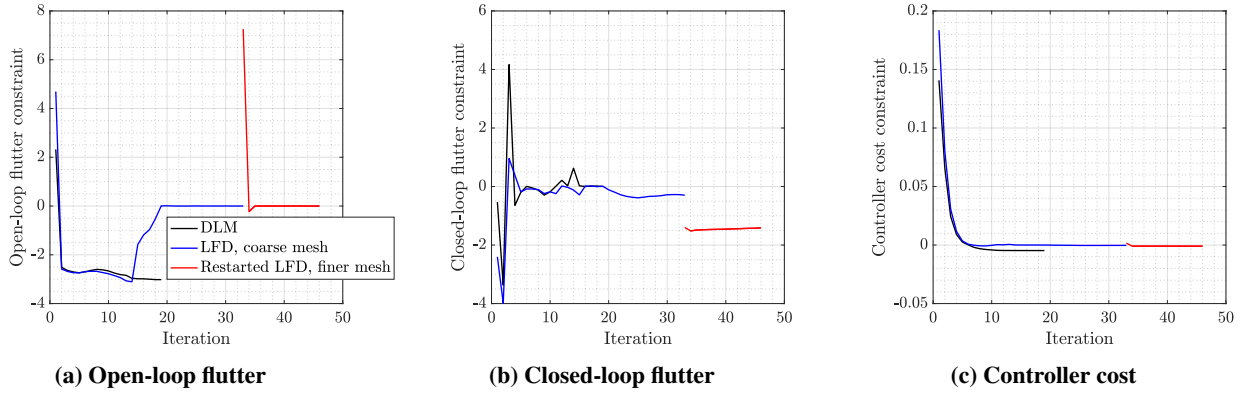


Fig. 9 Selected constraint convergence histories for the AGARD optimizations.

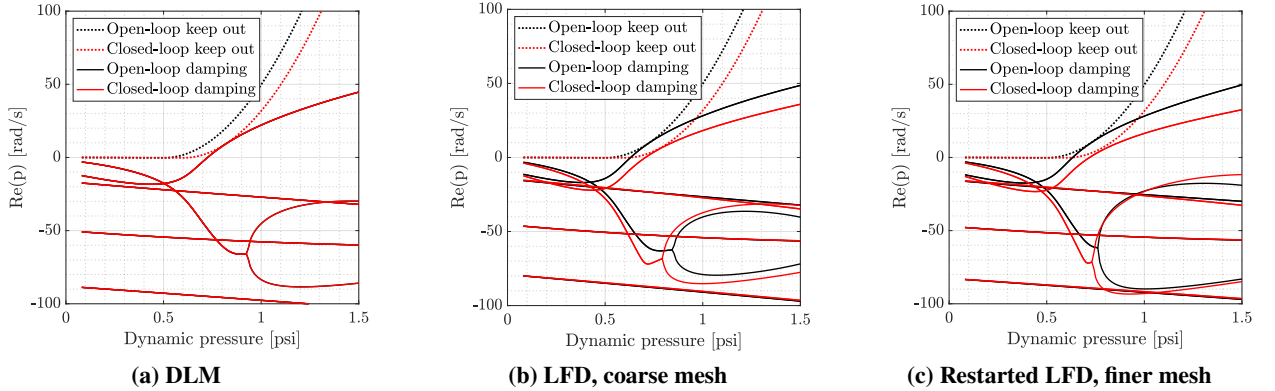


Fig. 10 Open- and closed-loop flutter responses of the optimized AGARD designs.

For reference, Fig. 12 illustrates the final closed-loop flutter mode shape from the restarted LFD optimization. Baseline and optimized planform shapes are indicated as well. Interestingly, the inboard control surface, which is positioned perhaps surprisingly close to the wing root, appears to be most active among the two control surfaces. The outboard control surface, in contrast, is barely visible in the flutter mode's imaginary part. It is unclear whether it is perhaps the cost constraint that pushes the most active control surface inboard, or if this phenomenon can be explained by other means. Further studies will need to be conducted to characterize how the optimizer tends to shift control surface positioning and movement.

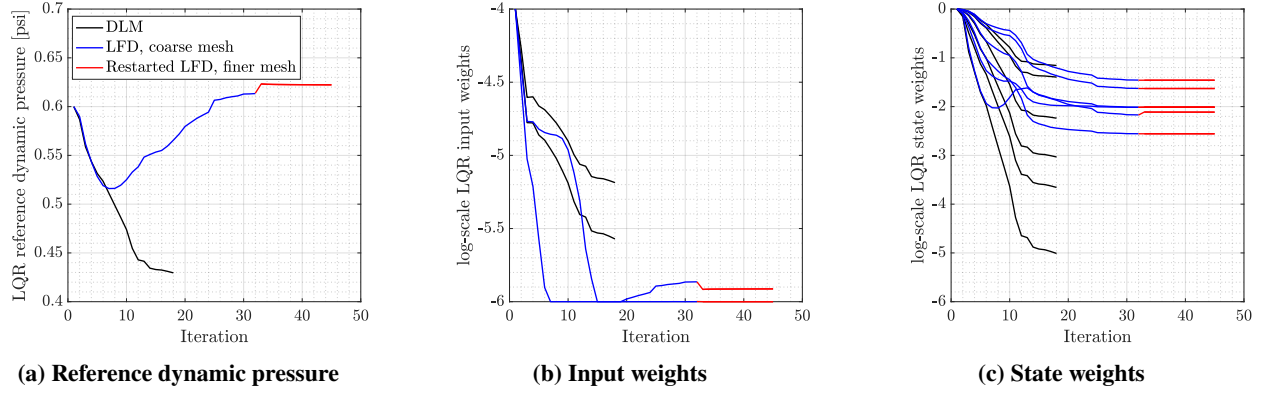


Fig. 11 Convergence histories of the LQR-related design variables for the AGARD optimizations.

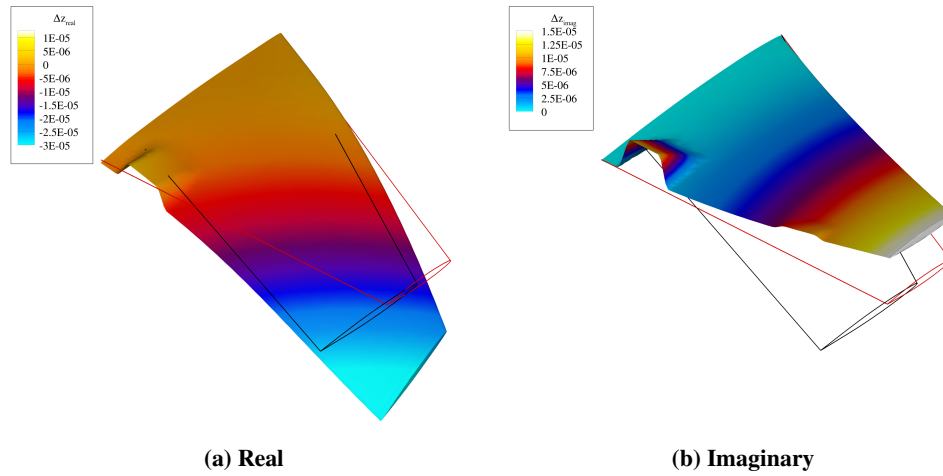


Fig. 12 Closed-loop flutter mode shape (scaled by 2.5×10^5) for the restarted LFD optimization. Black and red wing outlines indicate the baseline and optimized planforms, while the contours indicate the out-of-plane modal displacement.

V. Conclusion

This work describes the recent implementation of static and dynamic ASE systems. Using the modularity of the MPhys library for OpenMDAO, panel- and CFD-based aerodynamic solvers may be interchanged; this is demonstrated with in-house VLM and DLM solvers, as well as the steady and LFD solvers in FUN3D/SFE. The static ASE optimization uses CS to minimize the average fuel burn at several cruise conditions, while also including an MLA system to allow for a relaxed stress constraint at a 2.5g pull-up maneuver. The dynamic ASE case optimizes an AFS system, where the controller is designed using LQR of a reduced form of the state-space system. A Rogers RFA flutter solver has been implemented to solve for open- and closed-loop flutter responses. With all analytic derivatives implemented, the gradient-based optimizations are carried out with an SLSQP optimization algorithm.

There are several potential areas of improvement for the present work. For either static or dynamic ASE cases, one could refine the aerodynamic mesh around the CS boundaries to better resolve the flow field. This would introduce challenges if the CS locations were to change, as the refined regions would need to move as the CS move. Likewise, the implementation could be tested with non-blended CS boundaries, with gaps physically being modeled. This would introduce further mesh-related challenges when CS deflections are large, as a valid volume mesh (i.e., one without negative volumes) may be difficult to solve for. Another aspect that could provide better realism would be to include the internal structure of the CS; this aspect is not defined for the CRM and AGARD cases chosen. Enforcing hinge moment constraints could serve as a rough surrogate for preventing internal structural failure, as this quantity is computed using only the aerodynamic surface mesh. For the CRM case specifically, benefits of including CS in the optimization problem

could be quantified; this could involve separate studies investigating either the off-design or pull-up scenarios without CS. The optimization, in addition, could have been more deeply converged, although doing so would presumably only yield relatively minor performance and design changes.

For the dynamic ASE case, enforcing additional constraints on controller performance could potentially be added for better realism. Likewise, the controller cost constraint could be replaced with something more meaningful and less arbitrary. Either modification could potentially cause both open- and closed-loop flutter constraints to be active simultaneously; since they are not in the cases shown, the optimizer may currently have an unreasonable amount of freedom on the controller design. Finally, the Rogers RFA solver could be expanded to include GLA calculations; ultimately, this would aim to enforce a stress constraint as CS are moved during transient gust loads.

Acknowledgments

This work is supported by the Transformational Tools and Technologies (TTT) project of the NASA Transformative Aeronautics Concepts Program (TACP).

References

- [1] Yildirim, A., Jacobson, K., Anibal, J., Stanford, B., Gray, J., Mader, C., Martins, J., and Kennedy, G., “MPhys: A Modular Multiphysics Simulation Package using the OpenMDAO Framework,” *Structural and Multidisciplinary Optimization*, Vol. 68, No. 1, 2025. doi:10.1007/s00158-024-03900-0.
- [2] Gray, J. S., Hwang, J. T., Martins, J. R. R. A., Moore, K. T., and Naylor, B. A., “OpenMDAO: An Open-Source Framework for Multidisciplinary Design, Analysis, and Optimization,” *Structural and Multidisciplinary Optimization*, Vol. 59, 2019, pp. 1075–1104. doi:10.1007/s00158-019-02211-z.
- [3] Anderson, W. K., Biedron, R. T., Carlson, J.-R., Derlaga, J. M., Diskin, B., Druyor, C. T., Gnoffo, P. A., Hammond, D. P., Jacobson, K. E., Jones, W. T., Kleb, B., Lee-Rausch, E. M., Liu, Y., Lohry, M. W., Nastac, G. C., Nielsen, E. J., Padway, E. M., Park, M. A., Rumsey, C. L., Thomas, J. L., Thompson, K. B., Walden, A. C., Wang, L., Wood, S. L., Wood, W. A., and Zhang, X., “FUN3D 14.2 Manual,” Tech. Rep. TM-20250002308, NASA, April 2025. URL <https://fun3d.larc.nasa.gov/>.
- [4] Anderson, W. K., Newman, J. C., and Karman, S. L., “Stabilized Finite Elements in FUN3D,” *55th AIAA Aerospace Sciences Meeting*, 2017.
- [5] Jacobson, K. E., and Stanford, B. K., “Flutter-Constrained Optimization with the Linearized Frequency-Domain Approach,” *AIAA SciTech Forum*, San Diego, California, USA, 2022.
- [6] Jacobson, K., Thelen, A., and Stanford, B., “Linearized Frequency-Domain Gust Analysis and Adjoint-Based Sensitivities,” *AIAA SciTech*, Orlando, Florida, USA, 2024. doi:10.2514/6.2024-2591.
- [7] Kiviaho, J. F., and Kennedy, G. J., “Efficient and Robust Load and Displacement Transfer Scheme Using Weighted Least Squares,” *AIAA Journal*, Vol. 57, No. 5, 2019, pp. 2237–2243.
- [8] Yildirim, A., Gray, J. S., and Martins, J. R. R. A., “A Nonlinear Schur Complement Solver for CFD-Based Multidisciplinary Models,” *Eleventh International Conference on Computational Fluid Dynamics*, 2022. URL https://www.iccfd.org/iccfd11/assets/pdf/papers/ICCFD11_Paper-0702.pdf, iccfd11-0702.
- [9] Abdul-Kaiyoom, M. A. S., Yildirim, A., and Martins, J. R. R. A., “Robust Schur-Complement Solvers and Large-Scale Multidisciplinary Design Optimization,” *34th Congress of the International Council of the Aeronautical Sciences*, Florence, Italy, 2024.
- [10] Roger, K. L., “Airplane math modeling methods for active control design.” 1977.
- [11] Hjartarson, A., Seiler, P., and Balas, G., “LPV Aeroservoelastic Control using the LPVTools Toolbox,” *AIAA Atmospheric Flight Mechanics (AFM) Conference*, Boston, MA, 2013.
- [12] Gilbert, M., “An Analytical Sensitivity Method for use in Integrated Aeroservoelastic Aircraft Design,” *Mechanical Systems and Signal Processing*, Vol. 4, No. 3, 1990, pp. 215–231. doi:10.1016/0888-3270(90)90016-E.
- [13] Kreisselmeier, G., and Steinhauser, R., “Systematic Control Design by Optimizing a Vector Performance Index,” *International Federation of Active Controls Symposium on Computer-Aided Design of Control Systems*, Zurich, Switzerland, 1979.

- [14] Kennedy, G. J., and Hicken, J. E., “Improved Constraint-Aggregation Methods,” *Computer Methods in Applied Mechanics and Engineering*, Vol. 289, 2015, pp. 332–354.
- [15] Thelen, A., Stanford, B., and Jacobson, K., “Flutter and Stochastic Gust Constraint Studies of the Undelected CRM using MPhys,” *AIAA SciTech Forum*, Orlando, Florida, USA, 2024. doi:10.2514/6.2024-2774.
- [16] Brooks, T. R., Kennedy, G., and Martins, J., “High-fidelity Aerostructural Optimization of a High Aspect Ratio Tow-steered Wing,” *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2016, p. 1179.
- [17] Wu, N., Kenway, G., Mader, C. A., Jasa, J., and Martins, J. R. R. A., “pyOptSparse: A Python Framework for Large-Scale Constrained Nonlinear Optimization of Sparse Systems,” *Journal of Open Source Software*, Vol. 5, No. 54, 2020, p. 2564. doi:10.21105/joss.02564.