AIAA SciTech Forum
7-11 January 2019, San Diego, California
AIAA Scitech 2019 Forum

10.2514/6.2019-2354

Computational Modeling of Flutter Constraint for High-Fidelity Aerostructural Optimization

Eirikur Jonsson*
Charles A. Mader,†
Joaquim R. R. A. Martins‡
University of Michigan, Ann Arbor, Michigan, United States
Graeme J. Kennedy§
Georgia Institute of Technology, Atlanta, Georgia, United States

High fidelity computational modeling and optimization of aircraft has the potential to allow engineers to produce more efficient designs, requiring fewer unforeseen design modifications late in the design process. In order for the optimization algorithm to generate a useful design, all the relevant physics must be considered, including flutter. This is especially important for the high-fidelity aerostructural optimization of commercial aircraft, which is likely to result in wing designs that are prone to flutter. To address this issue, we develop a flutter constraint formulation suitable for gradient-based optimization. This paper investigates the feasibility of using a Doublet-Lattice Method (DLM) based flutter constraint for high-fidelity aerostructural optimization. An efficient non-iterative root finding method is developed to compute the flutter eigenvalues from a generalized eigenvalue problem, based on the p-k flutter equation. A mode tracking scheme is implemented at two levels that successfully tracks the mode migration. An effective constraint curve is then developed to define the flutter free flight envelope, in addition to preventing discontinuities in the flutter constraint. This method allows for minimum flutter velocity to be specified implicitly. The Kreisselmeier–Steinhauser (KS) function is used to aggregate the difference of the flutter damping eigenvalues and the constraint curve into a single value that is then used in an optimization. We compute accurate and efficient derivatives of the constraint value with respect to structural sizing variables, as well as wing planform variables. Derivatives are computed using a combination of analytic and automatic differentiation methods in reverse (adjoint) and validated using the complex-step method. To study the behavior of the flutter constraint using this method, we perform a design space analysis and optimize an idealized wing (flat plate) of a rectangular planform.

I. Introduction

Aircraft design often requires high-fidelity computational modeling and optimization to accurately model all the relevant physical behavior. This is necessary to develop effective designs and to reduce the occurrence of unforeseen design modifications during the later stages of development. In particular, for transonic wing design, the simultaneous optimization of both the aerodynamic design and the internal structure can yield significant reductions in fuel burn. However, all the relevant physics must be represented in the optimization problem in order to generate a physically realizable design. For example, previous optimization results carried out by Kenway et al. [1, 2, 3, 4], without flutter constraints, produced wings with large aspect ratios as shown in Fig. 1. Such configurations are prone to flutter, which calls into question the usefulness of the results.

Since flutter is a safety and certification-critical phenomenon, it is important to be able to account for it early in the design process. Conservative design approaches may lead to excessively stiff and hence heavy designs, while unconstrained optimization approaches, such as those shown in Fig. 1, may lead to excessively flexible wings. Further, it is not unusual for flutter issues to be identified only at the final design and flight testing stages, at which point design changes are very costly. Therefore, we seek to model the flutter characteristics of the aircraft and account for them in the optimization process.

In previous work [5], the authors developed and implemented an efficient flutter constraint to be used in a high-fidelity, gradient-based aerostructural optimization. Derivatives of the constraint were obtained efficiently using a

*PhD Candidate, AIAA student member
†Research Investigator, AIAA Senior Member
‡Professor, AIAA Associate Fellow
§Assistant Professor, School of Aerospace Engineering, AIAA Member

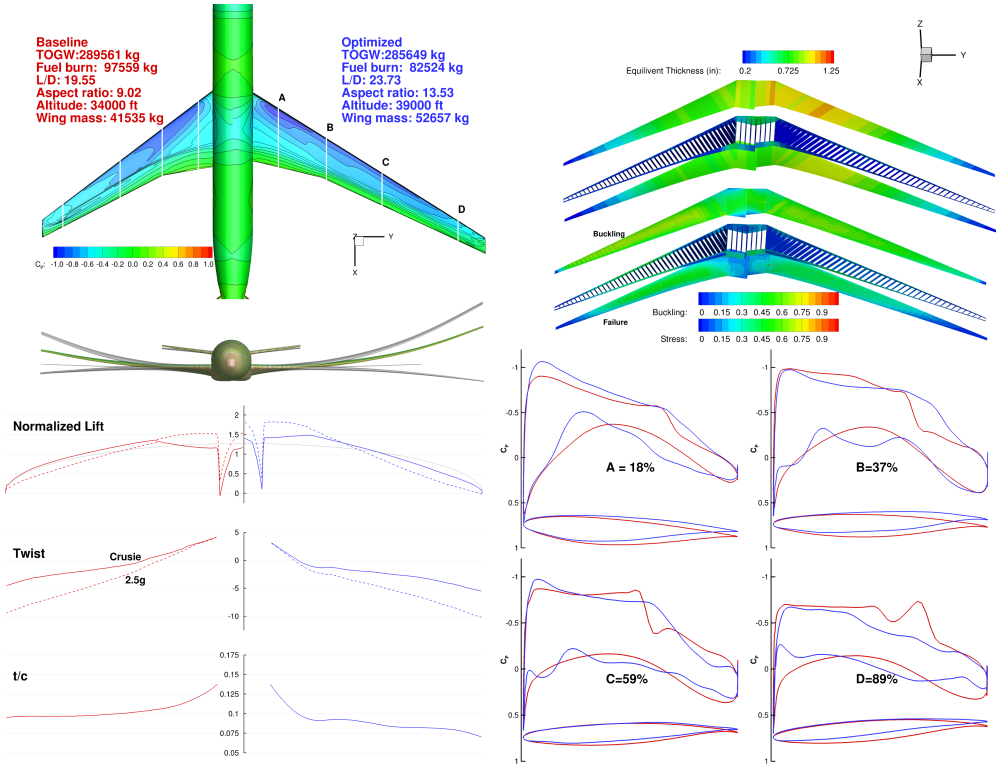


Figure 1. Aerostructural optimization result without flutter constraints [3]: Cp and planform comparison with initial design (upper left); equivalent thickness distribution, stress and buckling KS failure criteria (upper right); comparison of initial and optimized lift distributions, twist distributions and thickness to chord ratio (t/c) (lower left); four airfoils with corresponding Cp distributions (lower right). (notice the increased span ratio)

combination of analytic methods and Automatic Differentiation (AD) in reverse (adjoint) and were shown to be accurate [5]. Further, since planform is expected to change in the optimization process, no assumptions or simplifications are made on the mode shapes, unlike in other work, where fixed mode derivatives are employed [6, 7, 8]. Here, the mode shapes are taken to be functions of the design variables and are differentiated accordingly.

However, as illustrated in previous work [5] with a simple flat plate problem optimization, the determinant iterative root finding method, is not well suited for optimization, even for the simplest problems. Challenges encountered in previous work can be summarized as:

1. If an eigenvalue changes rapidly with increasing velocity the method might converge to non-structural root, or an aerodynamic lag root, due to a large step size in velocity. This is believed to be the problem moving from slice A to slice B, shown in the top plot of Fig. 2. This issue can be mitigated to some extent by substantially increasing the number of velocities analyzed, thus increasing computational burden.

2. If two eigenvalues have similar or same imaginary parts the determinant iterative method can "lock" onto an incorrect eigenvalue. This may cause discontinuity in the aggregated values which poses a big challenge for the optimizer. This issue was frequently observed in previous work and is shown in Fig. 2 at slice B. The determinant iterative method offers no direct way of tracking each mode. Substantially increasing the number of velocity points analyzed may sometimes alleviate this issue, although it would increase the computational burden if large number of modes and velocities are needed for the flutter analysis.

3. Further, the determinant iterative method offers no direct way of tracking two real roots that emerge from a bifurcated root where the imaginary part vanishes. Its only capable of tracking as many modes as there are structural modes. It therefore picks the first branch it finds. This issue is highlighted in Fig. 2 at slice C where the diverging branch is picked up over the more stable branch (not shown) that should continue onward to more stable (negative) damping values for higher velocities.

The purpose of this paper is to address the aforementioned issues with a more robust flutter analysis method and mode tracking and an improved flutter constraint formulation. As before, we implement accurate and efficient

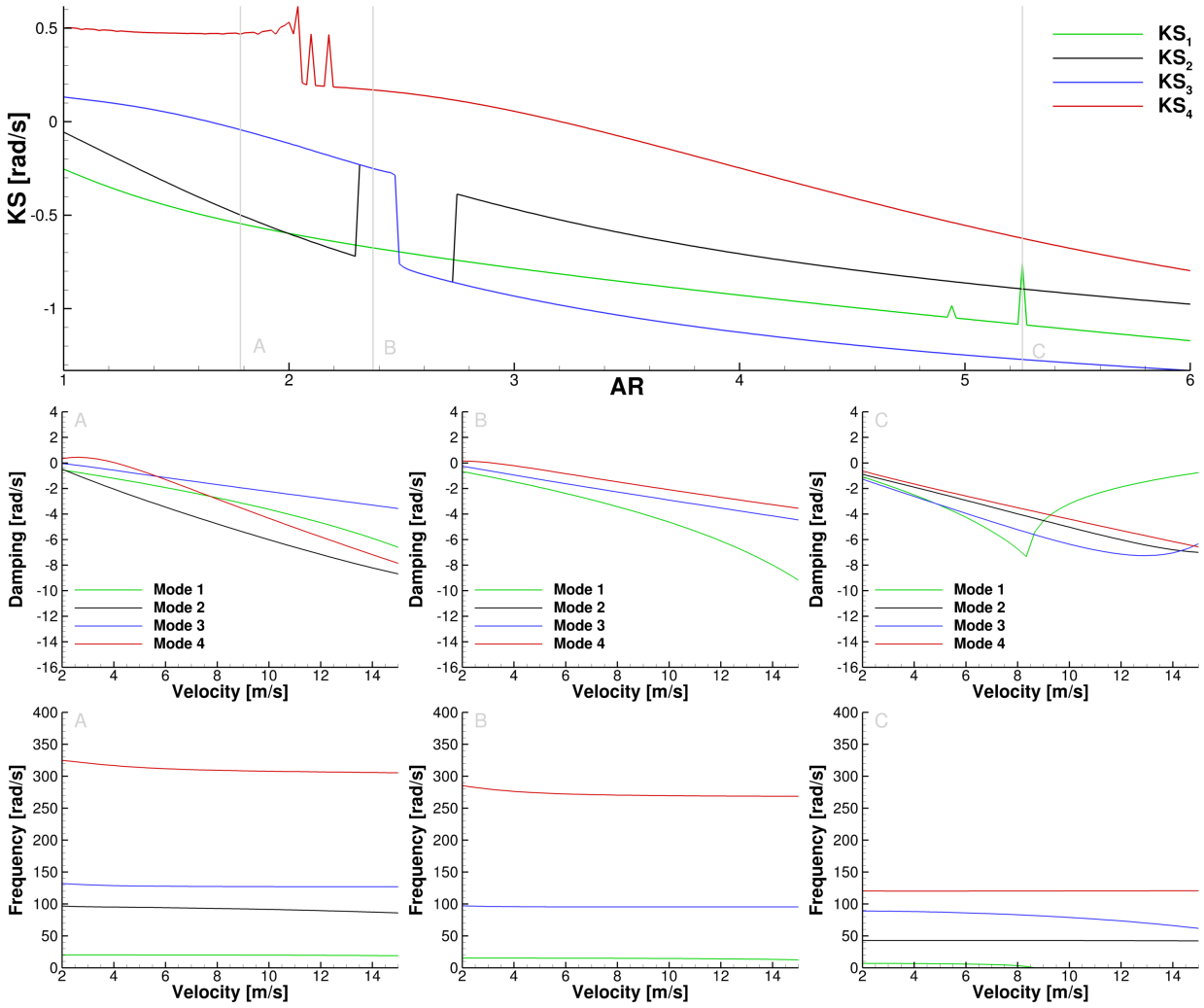


Figure 2. Results from [5] for a flat plate geometry showing discontinuities appearing in the constraint values (top plot) at slices A, B and, C when varying the aspect ratio. Damping and frequency plots are shown as well for slices A, B and, C. At slice B the determinant iterative method is unable to distinguish between the two flutter eigenvalues and predicts the same value twice for the two different modes, as shown in the corresponding damping and frequency plots for slice B. Similarly the determinant iterative method is incapable of tracking both real roots that emerge from a bifurcation and "locks" onto either branch. $1 \leq AR \leq 6$**, was sampled with 256 points with a fixed thickness.**

sensitivities of the proposed methodology to enable efficient optimization. The paper can be outlined as follows. In Section II we introduce the methods and tools implemented in order to perform efficient flutter analysis as well as give the formulation of a continuous flutter constraint suitable for gradient based optimization. In Section III sensitivities of the flutter constraint are discussed and verified. The proposed methodology is then applied in Section IV to the same rectangular wing planform which is analyzed and then optimized. Section V concludes the paper with a discussion and comparison of the proposed method and previously implemented method.

II. Methods

There are several techniques and components necessary to enable flutter computations. Figure 3 gives a high-level overview of the overall flutter analysis process. The green boxes represent processes or components and are all developed in-house. The off-diagonal terms are the variables that are passed between components. In the following section we will outline the major characteristics of these components.

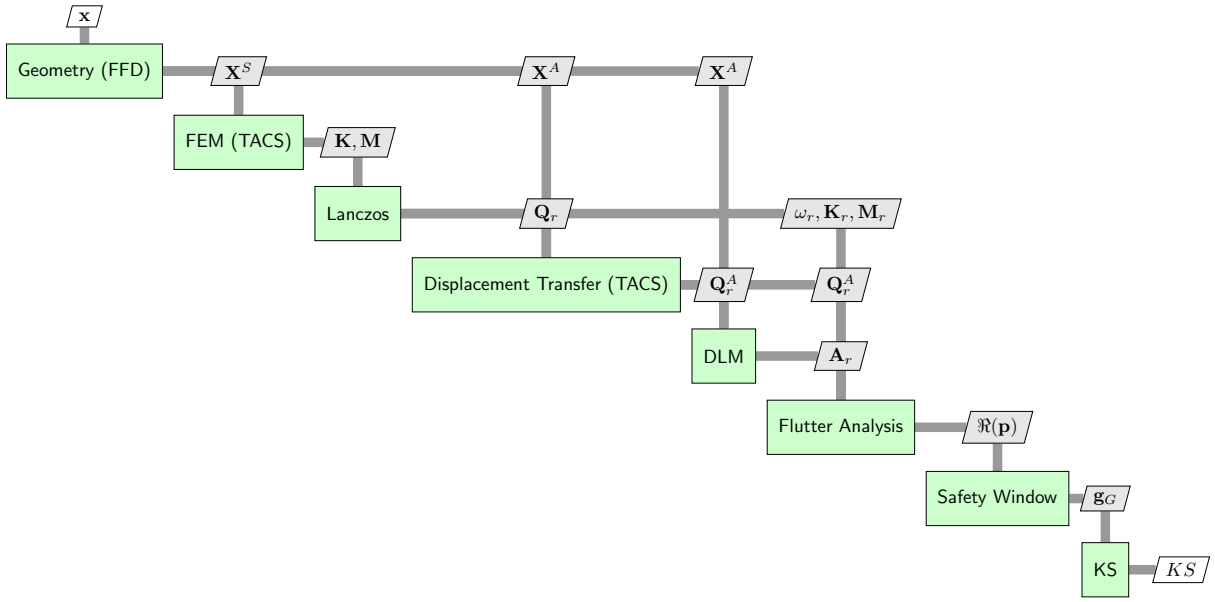


Figure 3. XDSM [9] of the proposed flutter analysis process and constraint formulation.

1. Geometric Parametrization

The wing shape is parametrized using a Free Form Deformation volume (FFD) approach [10]. This method has also been used with great success in computer graphics to deform solid geometries [11]. In this approach the geometry of interest is embedded within an FFD volume, which can be deformed using a number of control points. To create full-wing design variables such as span or sweep, the control points can be grouped together. Using this method, the structural and the aerodynamic meshes, $\mathbf{X}^S$, $\mathbf{X}^A$ respectively, can be handled the same way, minimizing the effort needed to update the internal structure as the optimizer changes the aerodynamic surface. A simple example of an FFD is shown in Fig. 4 where the red points represent the control points and the black lines connecting them represents the outer edges of the volume.

2. Finite Element Analysis Solver

The structural solver for this work is the Toolkit for the Analysis of Composite Structures (TACS) [12, 13]. TACS is a parallel finite-element solver with the capability to handle poorly conditioned problems, which is common in work involving thin-walled structures typically found in transport aircraft. For such cases, the stiffness matrix condition numbers may exceed $\mathcal{O}(10^9)$, but through the use of a Schur-complement based parallel direct solver, TACS is able to effectively solve these poorly conditioned problems. Sensitivities of structural functions of interest are also computed efficiently using the adjoint method with respect to structural and geometric design parameters. Load and displacement

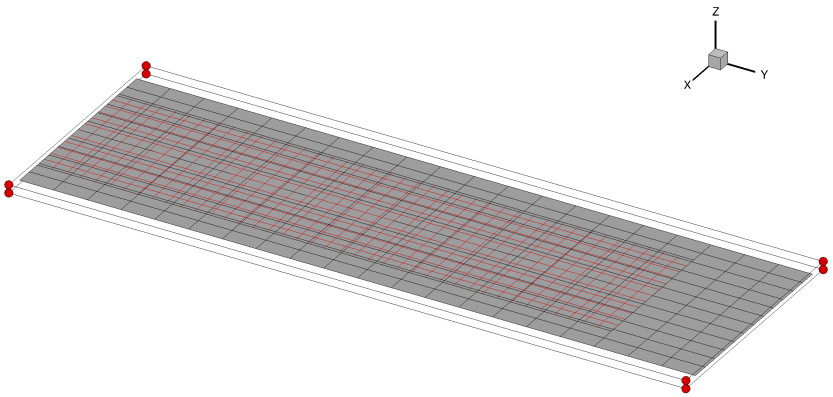


Figure 4. Flat plate structural and aerodynamic mesh shown in red and black respectively. The plate is cantilevered at the left edge. A Free-Form-Deformation (FFD) volume is also shown with 8 control points which are depicted as red spheres connected by solid black lines.

transfer scheme used here is the rigid link approach, which follows the work of Brown [14]. In this approach, rigid links are used to extrapolate the displacements from the structural surface to the aerodynamic surface.

3. Doublet Lattice Method

The doublet-lattice method (DLM) [15] consists of a lifting surface method that is formulated in the frequency domain. The DLM is based on the vortex-lattice method (VLM) where the VLM is extended to harmonically oscillating surfaces where a flat wake is assumed. A substantial body of literature exists on the DLM. An excellent reference worth mentioning is the work done by Blair [16]. The DLM has been widely adopted in the aeroelastic community and has been a valuable tool for the flutter analysis of subsonic aircraft. Commercial software tools such as MSC/Nastran have adopted the DLM [17]. In this work the implementation is based in part on the method of Albano and Rodden [15], and the extension by Rodden et al. [18].

4. Flutter Analysis

In this work, we use a *pk* type flutter analysis that takes the following form,

$$\left[\left(\frac{U}{b}\right)^2 p^2\mathbf{M} + \frac{U}{b}p\mathbf{C} + \mathbf{K} - \mathbf{F}_{\text{aero}}(ik)\right]\bar{\mathbf{u}} = 0 \tag{4.1}$$

where $p = g + ik$ is the eigenvalue (non-dimensional Laplace parameter s where $p = (b/U)s = (b/U)(\sigma + i\omega)$), g is the non-dimensional aerodynamic damping, and k is the reduced frequency. $\bar{\mathbf{u}}$ is the eigenvector in terms of structural degrees of freedom, U is the flow velocity, b is the semi reference chord and $\mathbf{M}, \mathbf{C}, \mathbf{K}, \mathbf{F}_{\text{aero}}(ik)$ are the mass, damping, stiffness, and aerodynamic matrices respectively, and are functions of the design variables $\mathbf{x}$. The mass and stiffness matrices are real and symmetric and the damping matrix is real and usually taken to be symmetric as well. Note that the aerodynamic forces are purely oscillatory (no damping) and are a nonlinear function of k. The aerodynamic forces can be written as

$$\mathbf{F}_{\text{aero}}(ik) = q_\infty \mathbf{A}(ik) \tag{4.2}$$

where $q_\infty = 1/2\rho_\infty U^2$ is the dynamic pressure [19]. Following Rodden et al. [20] the aerodynamic forces can be split into real and imaginary parts, $\mathbf{A} = \mathbf{A}^R + i\mathbf{A}^I$, in order to improve the approximation of the damping. Further, assuming small damping, $p/ik \approx 1$, we can write the forces as

$$\mathbf{F}_{\text{aero}}(ik) = q_\infty \left(\mathbf{A}^R(ik) + p\mathbf{A}^I(ik)/k\right). \tag{4.3}$$

Equation (4.1) can then be written as

$$\left[\left(\frac{U}{b}\right)^2 p^2\mathbf{M} + p\left(\frac{U}{b}\mathbf{C} - \frac{q_\infty}{k}\mathbf{A}^I\right) + \left(\mathbf{K} - q_\infty\mathbf{A}^R\right)\right]\bar{\mathbf{u}} = 0, \tag{4.4}$$

Using the trivial expression

$$\mathbf{I}\dot{\mathbf{u}} - \mathbf{I}\dot{\mathbf{u}} = 0, \tag{4.5}$$

where $\mathbf{I}$ is the identity matrix, the now can write it on a state space form (substituting $\mathbf{u} = \bar{\mathbf{u}}e^{pt}$)

$$p\begin{bmatrix}\mathbf{I} & 0\\ 0 & \left(\frac{U}{b}\right)^2\mathbf{M}\end{bmatrix}\begin{Bmatrix}\bar{\mathbf{u}}\\ p\bar{\mathbf{u}}\end{Bmatrix} - \begin{bmatrix}0 & \mathbf{I}\\ -(\mathbf{K} - q_\infty\mathbf{A}^R) & -\left(\frac{U}{b}\mathbf{C} - \frac{q_\infty}{k}\mathbf{A}^I\right)\end{bmatrix}\begin{Bmatrix}\bar{\mathbf{u}}\\ p\bar{\mathbf{u}}\end{Bmatrix} = 0. \tag{4.6}$$

Note that all matrices are real, but due to non-symmetric nature of the aerodynamic loads, Eq. (4.6) is a generalized nonlinear eigenvalue problem, resulting in complex eigenvalues and eigenvectors.

4.1. Reduced Eigenproblem

Solving Eq. (4.4) (or Eq. (4.6)) is prohibitive for any realistic structure due to the size of the matrices. Instead, a modal approach is adopted here where small number of natural mode shapes from the free vibration problem are used to reduce the computational effort of Eq. (4.4) by casting it into a small number of generalized degrees of freedom. The reduced mode shapes are the eigenvectors of the free vibration problem. Assuming simple harmonic motion $\mathbf{u} = \bar{\mathbf{u}}e^{i\omega t}$:

$$\left[\mathbf{K} - \omega_i^2\mathbf{M}\right]\bar{\mathbf{u}}_i = 0, \tag{4.7}$$

where ω_i, $\bar{\mathbf{u}}_i$ are the eigenvalues and eigenvectors or natural modes and mode shapes, respectively.

To obtain the lowest natural modes and mode shapes, Equation (4.7) is solved using a shift and invert Lanczos method previously developed [5]. The Lanczos algorithm uses an $\mathbf{M}$-orthonormal subspace, written as $\mathbf{V}_m \in \mathbb{R}^{n\times m}$, such that $\mathbf{V}_m^T\mathbf{M}\mathbf{V}_m = \mathbf{I}_m$, where n is the size of the square mass and stiffness matrices, and m the size of the subspace chosen. We use an expensive, but effective, full-orthonormalization procedure (Gram–Schmidt) that enforces $\mathbf{M}$-orthonormality. The Lanczos implementation is shown to produce good approximation and has verified using the commercial software MSC/Nastran [20]. For more details on the method, we refer the reader to [5].

The natural mode shapes $\bar{\mathbf{u}}$ are then computed from the constructed subspace $\mathbf{V}_m$. The $\bar{\mathbf{u}}_i$, $\forall\, i = 1, \ldots, r < m$, are then collected in the matrix $\mathbf{Q}_r \in \mathbb{R}^{n\times r}$. These eigenvectors are $\mathbf{M}$-orthonormal, such that $\mathbf{Q}_r^T\mathbf{M}\mathbf{Q}_r = \mathbf{I}_r$. To obtain good approximations for the first r number of natural modes and mode shapes, the subspace size m is chosen to be at least $m > 2r$. Introducing the generalized coordinates, $\bar{\mathbf{u}} = \mathbf{Q}_r\bar{\mathbf{q}}$, the reduced generalized eigenproblem can now be written as

$$p\begin{bmatrix}\mathbf{I}_r & 0\\ 0 & \left(\frac{U}{b}\right)^2\mathbf{M}_r\end{bmatrix}\begin{Bmatrix}\bar{\mathbf{q}}\\ p\bar{\mathbf{q}}\end{Bmatrix} - \begin{bmatrix}0 & \mathbf{I}_r\\ -(\mathbf{K}_r - q_\infty\mathbf{A}_r^R) & -\left(\frac{U}{b}\mathbf{C}_r - \frac{q_\infty}{k}\mathbf{A}_r^I\right)\end{bmatrix}\begin{Bmatrix}\bar{\mathbf{q}}\\ p\bar{\mathbf{q}}\end{Bmatrix} = 0, \tag{4.8}$$

where the reduced generalized matrices take the form:

$$\begin{aligned}\mathbf{M}_r &= \mathbf{Q}_r^T\mathbf{M}\mathbf{Q}_r = \mathbf{I}_r \in \mathbb{R}^{r\times r},\\ \mathbf{K}_r &= \mathbf{Q}_r^T\mathbf{K}\mathbf{Q}_r = \mathrm{diag}\{\omega_i^2\} \in \mathbb{R}^{r\times r},\\ \mathbf{A}_r(ik) &= \mathbf{Q}_r^T\mathbf{A}(ik)\mathbf{Q}_r \in \mathbb{C}^{r\times r}.\end{aligned}$$

Note that $\mathbf{A}_r$ has no sparsity structure and is a dense matrix in general, while $\mathbf{M}_r$ and $\mathbf{K}_r$ are diagonal. The structural damping can be approximated by Rayleigh damping, $\mathbf{C}_r = \alpha\mathbf{M}_r + \beta\mathbf{K}_r$ [21], similarly as in [6], but is omitted here for simplicity. The reduced generalized eigenvalue problem is solved with LAPACK [22].

4.2. Non-iterative Flutter Solution Method

In flutter analysis when solving Eq. (4.8), iterative procedures are commonly applied since the aerodynamic matrix depends on the reduced frequency k, the imaginary part of eigenvalue p. One such procedure is the determinant iteration [19] as used in the authors previous work [5]. Another popular method is found in commercial software such as MSC/Nastran [17, 20], where an eigenvalue problem is solved based on an assumed reduced frequency k. The resulting eigenvalue for the mode under study is identified and compared to the assumed reduced frequency k. If the difference exceeds some predefined tolerance the iteration continues with the imaginary part of the new eigenvalue used as the reduced frequency $k = \Im(p)$ to compute new aerodynamic loads.

As summarized in Section I, the aforementioned methods may experience convergence issues as well as converge to incorrect values. For example if two eigenvalues are close to each other, i.e. when the imaginary component (frequency) of the two modes is close or numerically identical, an incorrect mode may be picked up and iterated on.

This will result in *mode hopping*, such that a discontinuity may appear in the damping. Another related issue is when the flutter eigenvalue changes rapidly with dynamic pressure which can result in convergence issues or in an incorrect eigenvalue being picked up.

Furthermore, these methods do not distinguish aerodynamic lag roots from structural modes. If an aerodynamic lag roots become unstable, these methods may converge to the lag root over the structural mode or not converge at all. Neither of the aforementioned methods are able to add or remove aerodynamic roots as they appear when dynamic pressure is incremented. A more sophisticated root finding method is thus needed.

In a gradient based optimization setting, robustness of analysis methods, continuity, and smoothness of functions of interest are of the utmost importance. Thus, a flutter method that mitigates the aforementioned issues of convergence and damping discontinuities is chosen carefully. In this work we implement a non-iterative method proposed by van Zyl [23]. A similar method is also proposed by Chen [24]. The van Zyl method is outlined here:

1. At each dynamic pressure q_i increment, Eq. (4.8) is solved for a range of reduced frequencies k
2. Eigenvalues are valid roots of the flutter equation if the imaginary part of the eigenvalue equals the assumed k value i.e. a matched point solution $\Im(p) = k$
3. A change in sign of the difference $\Im(p) - k$ signifies the existence of a root
4. The root is then determined by a linear interpolation

For a hypothetical system with two modes (shown in orange and blue), Fig. 5 shows, qualitatively, the reduced frequency sweep for a single dynamic pressure q_i. The black dots represent an intersection of a mode with the black diagonal line, $\Im(p) = k$. There are 5 valid roots for this particular system, 4 from the orange mode and 1 from the blue mode. For the orange mode, there are two real roots at $k = k_0 = 0$ and two complex roots at $k = k_1$ and $k = k_2$. The blue mode is complex throughout the reduced frequency sweep and has only one root at $k = k_3$. An iterative method, such as the determinant iteration, would have issues converging to $k = k_1$ and would converge to $k = k_2$. Similar behavior is noted in [25] where the iterative method converges to a real root when it should have converged to oscillatory complex root. Furthermore, the non-iterative method places no limit on the number of roots that can be found.

4.3. Mode Tracking Method

In order to prevent mode swapping or mode hopping, the modes must be tracked at two stages in the analysis process:

1. During the reduced frequency sweep, which finds all the roots for a given dynamic pressure q_i
2. During the migration of the modes between dynamic pressure increments q_i and q_{i+1}

In this work the modes are tracked using the mode tracking method of [26]. The tracking of the modes during second stage, the mode migration, is more challenging and requires extra care. This is due to the fact that new modes can show up, as well as disappear. A correlation metric is implemented to determine if the dynamic pressure increment is too big. If the increment is too big, the dynamic pressure step is halved until its either accepted, moving on to the next dynamic pressure, or fails due to being halved too often, which is controlled by a minimum allowed dynamic pressure increment. In case of failure where minimum increment is reached, the roots are accepted and the dynamic pressure is incremented using a normal step-size.

4.4. Code Verification

Here we present a brief verification study of the methods used in our flutter analysis. The computation of natural modes and mode shapes using the Lanczos method were verified in [5]. Table 1 shows that the natural modes computed by the current implementation compare very well to MSC/Nastran, where the first 10 modes are withing 3% relative difference.

Verification of the flutter analysis code is performed using the flat plate geometry presented in Section IV. Specifically, in Fig. 6 we compare the flutter eigenvalues of the first 10 modes with MSC/Nastran. Overall trends of our implementation are good although some discrepancy is present. The first bifurcation is predicted at a very similar velocity but the flutter point of current implementation seems to be predicted at slightly higher velocities than what MSC/Nastran predicts. For higher velocities, the bifurcation velocity for the flutter mode is different between the two codes. Another interesting difference is that MSC/Nastran seems to confuse the damping values of the flutter bifurcated mode with the previously bifurcated root, whereas the current implementation shows no such issue. Figure 6b

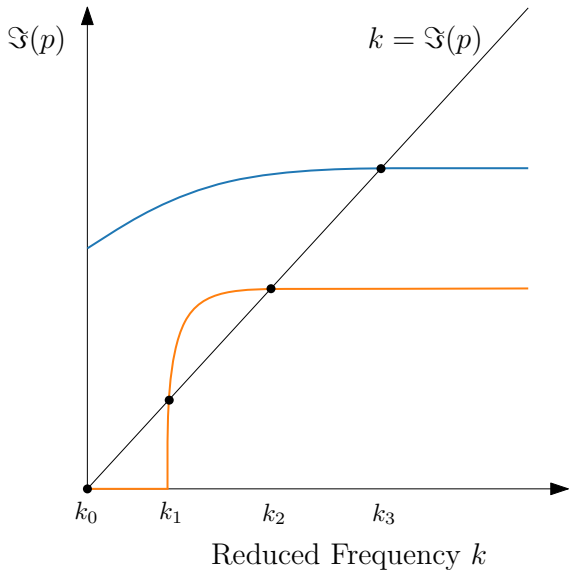


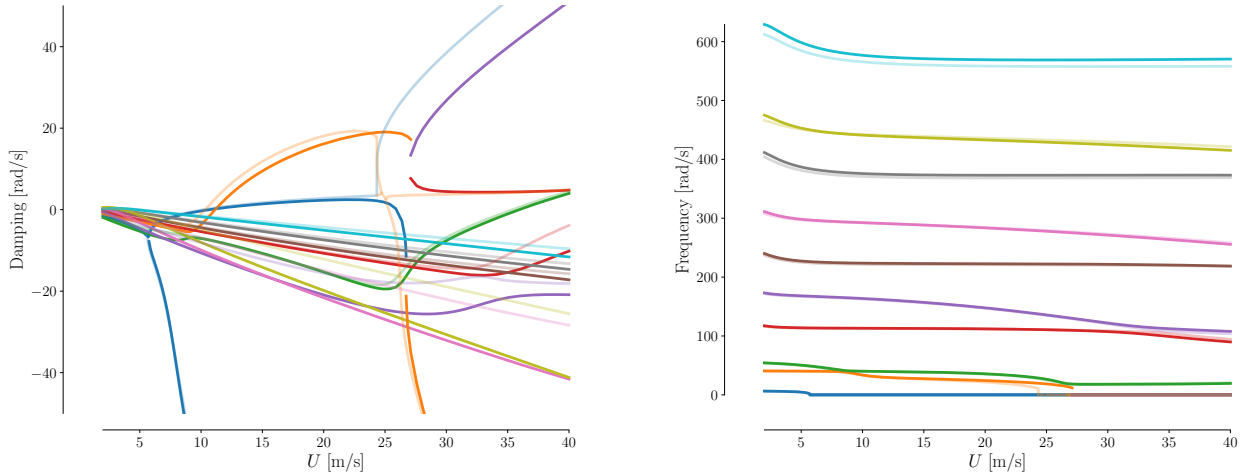
Figure 5. Hypothetical system with two modes shown in blue and orange. Black dots represent a match point solution, i.e., where the modes intersect the diagonal line, $\Im(p) = k$, depicted in black.

Table 1. Natural modes [rad/s] of current implementation matches very well compared to MSC/Nastran.

Mode #	MSC/Nastran [rad/s]	Current [rad/s]	Rel. Diff. [-]	Rel. Diff. [%]
1	7.13	7.13	5.79E-04	0.06
2	44.58	44.69	2.63E-03	0.26
3	57.70	58.27	9.69E-03	0.97
4	125.07	125.84	6.10E-03	0.61
5	177.92	179.95	1.13E-02	1.13
6	245.88	248.67	1.12E-02	1.12
7	311.83	316.43	1.45E-02	1.45
8	407.48	414.88	1.78E-02	1.78
9	466.87	476.03	1.92E-02	1.92
10	608.89	625.15	2.60E-02	2.60

shows fewer differences between the two codes and overall trend is good. The frequency values predicted for all modes of the given velocity range are very similar in numerical value except the previously mentioned bifurcation of the fluttering mode.

Note that the load and displacement transfer scheme used in MSC/Nastran is the FPS scheme [17]. This is different compared to current implementation and thus may cause some discrepancy between the two implementations.



a) Damping **b) Frequency**

Figure 6. Flutter solution of current work (solid lines) compared to MSC/Nastran (faded lines).

5. Flutter Constraint Formulation

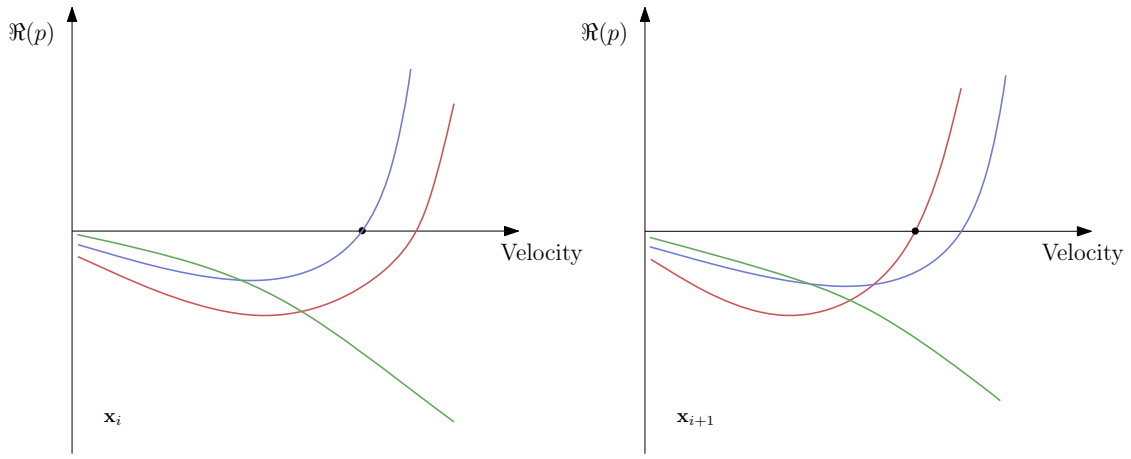
A critical aspect in constraint formulation is the constraint smoothness, which is paramount in gradient-based optimization. Jonsson et al. [27] discuss many of the considerations that are necessary to formulate an efficient and continuous flutter constraint suited for a gradient-based optimization framework. Here a short summary of those considerations is also given.

A naive or a simple approach is to specify the flutter point directly. This is defined as the flutter speed or dynamic pressure at which the wing flutters, that is, one of the modes becomes unstable [28]. However, enforcing the flutter point may introduce discontinuities in the constraint value between two consecutive design iterations $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. For instance, a discontinuity is observed when mode switching occurs or a hump mode becomes active at a significantly lower velocity. Additional steps are required to ensure a continuous constraint.

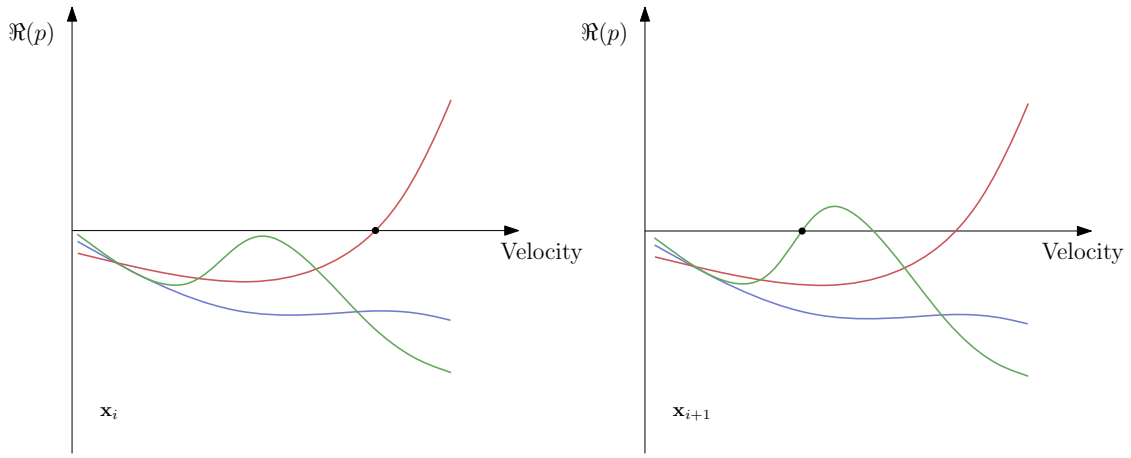
In the case of mode switching, as shown in Fig. 7a, two modes (showing only damping values) switch being the unstable mode, that is, the mode with the lowest dynamic pressure. This can cause discontinuity in the predicted flutter point, which is C^1 discontinuous at best, and thus poses a challenge to gradient-based optimizers [29]. The imaginary part of the eigenvalue (frequency) also switches and in many cases the frequencies coalesce, causing a mode to become unstable. A more serious problem is when a hump mode is present and it becomes the unstable mode, as shown in Fig. 7b. The constraint value experiences a C^0 discontinuity, which poses a serious problem to gradient-based optimizers.

Techniques exist to mitigate these problems and are summarized by Stanford et al. [30]. Frequency-separation constraints proposed by Langthjem and Sugiyama [31] and also by Odaka and Furuya [32] can be used to prevent the mode switching by enforcing the critical mode to remain the same. For a hypothetical system, this approach is shown in Fig. 8 (left). The disadvantage of this method is that for a system with N_m modes, $N_m - 2$ constraints are needed (expecting two modes to coalesce, hence no constraint is needed for two modes). Further, specifying the unstable mode may over-constrain the optimization process. A more serious flaw with this approach is the possibility of a hump mode becoming active is still present.

Other techniques exist to handle both mode switching and hump modes. One approach is to enforce the damping of each eigenvalue to remain below a preset bounding curve. Such bounding curve is proposed by Hajela [33], Ringertz [29], and later Stanford et al. [7, 34]. The boundary curve, $G(U)$ shown in Fig. 8, spans the operating conditions of interest from wind-off to some maximum velocity. The area above the curve is now defined as the



a) Critical flutter mode switching with small changes in the design. This behavior causes a C^1 discontinuity in the flutter point value.



b) Hump mode becoming active with small changes in the design. This behavior causes a C^0 discontinuity in the flutter point value.

Figure 7. Possible sources of discontinuities in the flutter constraint between two consecutive designs $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. (Adapted from Ref. [30]).

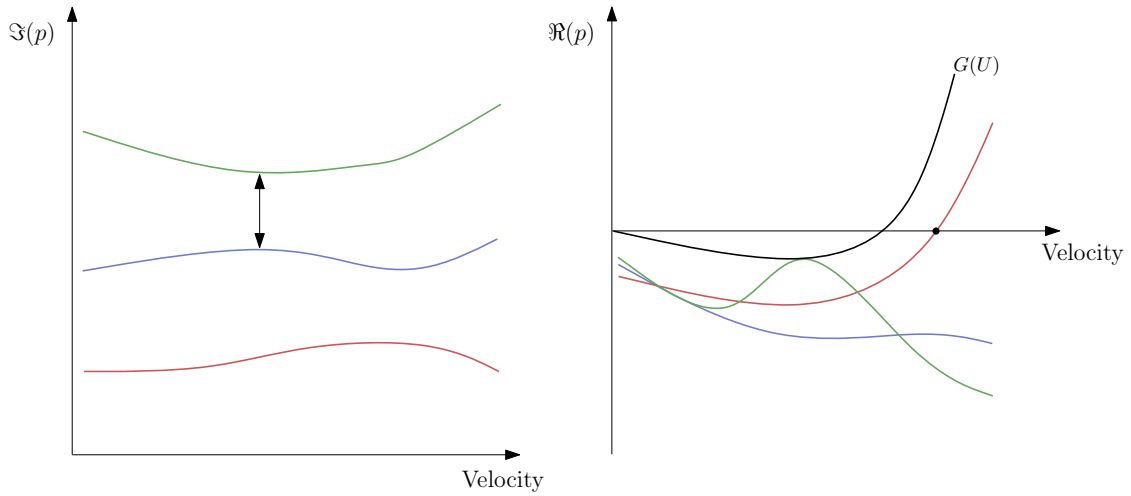


Figure 8. Two possible methods to prevent discontinuities in the flutter constraint: frequency-separation method (left) and damping boundary (right, black line). (Adapted from Ref. [30]).

unstable region and no modes should cross into that region. This approach mitigates the aforementioned issues and has further benefits: 1) No constraint is placed on the flutter point itself, so there is no need to compute it explicitly, but it can be set implicitly, 2) Modes that become unstable abruptly (hard flutter) and have steeper slope than the boundary are handled as well.

In this work we use a boundary shape proposed by Stanford et al. [34, 8] which is a modified version of a boundary proposed by Ringertz [29]. The boundary is defined as

$$G(U) = \begin{cases} g^* \left(3U^2U^* - 2U^3\right)/(U^*)^3 & 0 \leq U < U^* \\ \beta(U - U^*)^2 + g^* & U \geq U^* \end{cases} \tag{5.1}$$

where $g^* < 0$ rad/s, $U^* > 0$ m/s, and $\beta > 0$ rad $\cdot$ s/m^2 are chosen based on the criteria of the problem at hand. Note that here the coefficients are given in dimensional form, e.g. g^*, β, but they can be given in non-dimensional form as well. Similarly, for U which could be replaced with density, dynamic pressure, or equivalent airspeed. Figure 9 illustrates the boundary where we have chosen $g^* = -2$ rad/s, $U^* = 20$ m/s, and $\beta = 3$ rad $\cdot$ s/m^2 as an example. The flutter constraint is then written as

$$g_{ij} \leq G(U_i) \quad i = 1, \ldots, N_U \quad j = 1, \ldots, N_m \tag{5.2}$$

where, N_U, N_m are the total number of velocity increments and modes, respectively. The damping, $g_{ij} = \Re(p_{ij})$ has to be less than $G(U_i)$ for every mode j at every velocity of interest i. Thus, for stability of the system we require,

$$g_{G,ij} = g_{ij} - G(U_i) \leq 0 \quad \forall\, i, j. \tag{5.3}$$

In case the boundary is not used, $G(U_i) = 0$, such that, $g_{G,ij} = g_{ij} \leq 0 \quad \forall\, i, j$, which simply enforces all damping values to be less than zero.

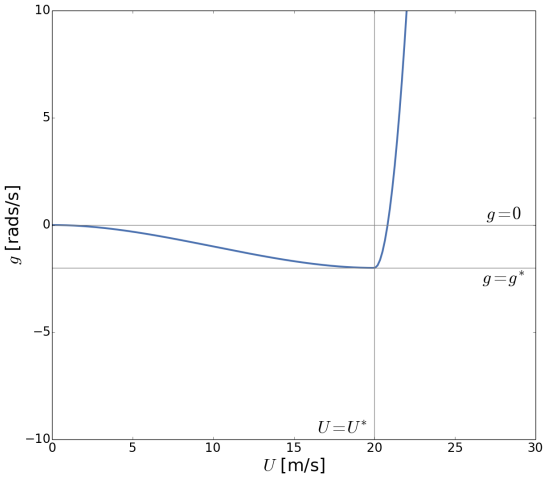


Figure 9. Example of a constraint curve that can be applied to mitigate issues with discontinuities due to flutter point changing and hump modes (Adapted from Ref. [34]).

5.1. Flutter Constraint Aggregation

Although the above approach mitigates the discontinuity issues, it requires one to apply a constraint for each velocity increment and mode. Thus, N_U velocity increments and N_m modes results in $N_m N_U$ constraints, increasing the total number of constraints dramatically. In these circumstances, the active set method could be used. This consists of considering the full set of points but, based on the constraint value, reducing them to a smaller set before evaluating derivatives. This approach was applied by Ringertz [29].

An alternate approach to reduce the number of constraints was proposed by Haftka [35], who suggested replacing parametric constraints by minimum-value constraints. This reduces the number of constraints to the total number of modes for the entire flight envelope.

To reduce the large number of constraints using the proposed method, we apply a different approach. The Kreisselmeier–Steinhauser (KS) function [36, 37, 38] can be used to aggregate constraints into a single composite function. The KS function is C_1 continuous and gives a conservative estimate of the maximum for a given set of constraints. The KS function can be written as:

$$KS(\mathbf{g}(\mathbf{x})) = \frac{1}{\rho_{\text{KS}}} \ln \left(\sum_{j=1}^{m} e^{\rho_{\text{KS}} g_j(\mathbf{x})} \right) \tag{5.4}$$

where $\mathbf{g}(\mathbf{x})$ is a set of constraints at a design point $\mathbf{x}$ and ρ_{KS} is the KS parameter. This parameter can be used as a buffer or tolerance and is analogous to a penalty parameter used in constrained optimization. As $\rho_{\text{KS}} \to \infty$ the KS function approaches the true maximum, but too large a value can cause sharp changes in the gradient. In this work $\rho_{\text{KS}} = 30$ is chosen based on experience. To avoid numerical difficulties due to overflow we use an alternate form of the KS function

$$KS(\mathbf{g}(\mathbf{x})) = g_{\max}(\mathbf{x}) + \frac{1}{\rho_{\text{KS}}} \ln \left(\sum_{j=1}^{m} e^{\rho_{\text{KS}} (g_j(\mathbf{x}) - g_{\max}(\mathbf{x})} \right) \tag{5.5}$$

where $g_{\max}(\mathbf{x})$ is the maximum of all constraints evaluated at current design $\mathbf{x}$ and is taken to be constant. As for the value, the first derivative with respect to the design variables $\mathbf{x}$ of both forms should be equal, subject to finite precision arithmetic,

$$\frac{\partial KS(\mathbf{g}(\mathbf{x}))}{\partial \mathbf{x}} = \frac{\sum_{j=1}^{m} e^{\rho_{\text{KS}} g_j(\mathbf{x})} \frac{\partial g_j(\mathbf{x})}{\partial \mathbf{x}}}{\sum_{j=1}^{m} e^{\rho_{\text{KS}} g_j(\mathbf{x})}} \tag{5.6}$$

$$\frac{\partial KS(\mathbf{g}(\mathbf{x}))}{\partial \mathbf{x}} = \frac{\sum_{j=1}^{m} e^{\rho_{\text{KS}} (g_j(\mathbf{x}) - g_{\max}(\mathbf{x}))} \frac{\partial g_j(\mathbf{x})}{\partial \mathbf{x}}}{\sum_{j=1}^{m} e^{\rho_{\text{KS}} (g_j(\mathbf{x}) - g_{\max}(\mathbf{x}))}} \tag{5.7}$$

Note that the KS function is indifferent to the order of the constraints in the aggregate. Here, we apply the KS function twice in a sequence, once for each mode over all velocities resulting in N_m KS constraints values, then over all the modes resulting in a single constraint value,

$$KS(g_{G,ij}, \rho_{KS}) = KS\left([KS(g_{G,i1}, \rho_{KS}), KS(g_{G,i2}, \rho_{KS}), \ldots, KS(g_{G,iN_m}, \rho_{KS})], \rho_{KS}\right) \leq 0. \tag{5.8}$$

Although aggregating all damping values into a single constraint hides information from the optimizer this allows for any number of modes to show up in the analysis process. In general we only need to know if any one mode is violating the constraint boundary. For the geometry under investigation to be flutter free the double aggregated KS value has to be less than zero, indicating that the constraint is inactive.

6. Optimization Algorithm

The optimization package used in this work is SNOPT (Sparse Nonlinear OPTimizer) [39]. SNOPT is a gradient based optimizer that implements a sequential quadratic programming (SQP) algorithm. SNOPT uses an augmented Lagrangian merit function, and the Hessian of the Lagrangian is approximated using a quasi-Newton approach. This optimizer is designed to perform well for optimization problems featuring many sparse nonlinear constraints and it requires a low number of function calls. SNOPT is wrapped with a sparse implementation of pyOpt [40].

III. Derivative Implementation

To produce accurate gradient information, the process presented in Fig. 3 uses a combination of the adjoint method, analytic as well as automatic differentiation (AD). The derivatives in direct (forward) mode are presented in figure Fig. 10 We now describe the derivative approach used in this work.

1. Automatic Differentiation (AD)

Automatic differentiation, also known as algorithmic differentiation, is a well established method that systematically applies the differentiation chain rule to source code. This method uses source transformation tools that takes in the original computer program, augments it, and generates a new code, such that it computes the analytical derivatives

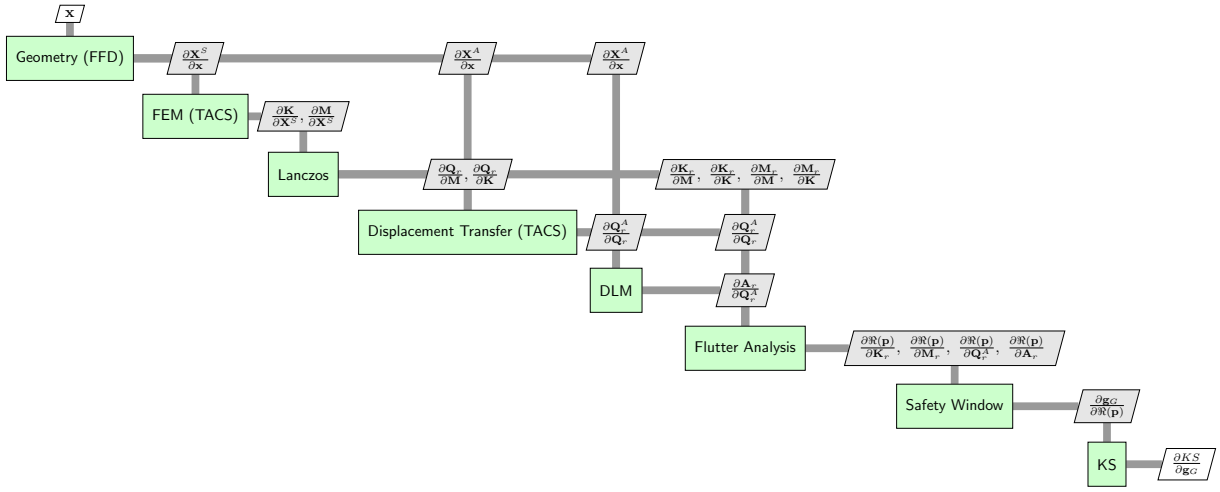


Figure 10. XDSM [9] of the derivatives computed by the proposed flutter analysis and constraint formulation.

along with the original program [41, 42]. Two modes exist, the forward mode and the reverse mode. For a generic system with scalar input x and output y we can write it as:

$$\begin{aligned} \text{System} \quad & x \rightarrow \boxed{F(x)} \rightarrow y \\ \text{Forward AD} \quad & \dot{x} \rightarrow \boxed{F'(x)} \rightarrow \dot{y} \\ \text{Reverse AD} \quad & \bar{x} \leftarrow \boxed{F'^{*}(x)} \leftarrow \bar{y} \end{aligned}$$

where the arrows represent the flow of information, the box represents the system or a function. The forward mode, know as the tangent, is denoted with a dot ($\dot{}$) over the variable. Given some small variations on the input (independent) variables $\mathbf{x}$ we can compute the resulting variations of the output (dependent) variables $\mathbf{y}$. The Jacobian matrix $\mathbf{J}$ contains the partial derivatives of each dependent variable y_j with respect to each independent variable x_i. The forward mode thus computes $d\mathbf{y} = \mathbf{J}d\mathbf{x}$ for each given $d\mathbf{x}$ or

$$\begin{aligned} d\mathbf{y} &= \mathbf{J}d\mathbf{x} \\ \begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \vdots \\ \dot{y}_n \end{bmatrix} &= \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_m} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_n}{\partial x_1} & \frac{\partial y_n}{\partial x_2} & \cdots & \frac{\partial y_n}{\partial x_m} \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_m \end{bmatrix}. \end{aligned}$$

Conversely the reverse mode, known as the adjoint, is denoted by a bar ($\bar{}$) over the variable. The order of operations reverses and we compute the transposed Jacobian product $d\mathbf{x} = \mathbf{J}^{*}d\mathbf{y}$ for each given $d\mathbf{y}$ or

$$\begin{aligned} d\mathbf{x} &= \mathbf{J}^{*}d\mathbf{y} \\ \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_n \end{bmatrix} &= \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_n}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_n}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_m} & \frac{\partial y_2}{\partial x_m} & \cdots & \frac{\partial y_n}{\partial x_m} \end{bmatrix} \begin{bmatrix} \bar{y}_1 \\ \bar{y}_2 \\ \vdots \\ \bar{y}_m \end{bmatrix}. \end{aligned}$$

In other words, the gradient of the independent variable is a linear combination of the variation in the dependent variable. This is a very important observation particularly in the case with fewer output variables than input variables. Since many aerodynamic optimization formulations contain many more design variables than outputs of interest (or $n_x >> n_I$) we use the reverse mode or the adjoint mode in order to obtain the derivatives as efficiently as possible. The AD source-transformation tool, Tapenade [43, 44] is used in this work.

2. Flutter Derivatives

A simplified flow of information for the flutter calculation, obtaining derivatives in forward and reverse mode can be written as:

$$\mathbf{x} \rightarrow \mathbf{K}, \mathbf{M} \rightarrow \mathbf{Q}_r, \mathbf{K}_r, \mathbf{M}_r, \mathbf{A}_r \rightarrow \mathbf{p} \rightarrow \mathbf{g}_G \rightarrow KS \tag{2.1}$$

$$\dot{\mathbf{x}} \rightarrow \dot{\mathbf{K}}, \dot{\mathbf{M}} \rightarrow \dot{\mathbf{Q}}_r, \dot{\mathbf{K}}_r, \dot{\mathbf{M}}_r, \dot{\mathbf{A}}_r \rightarrow \dot{\mathbf{p}} \rightarrow \dot{\mathbf{g}}_G \rightarrow \dot{KS} \tag{2.2}$$

$$\overline{\mathbf{x}} \leftarrow \overline{\mathbf{K}}, \overline{\mathbf{M}} \leftarrow \overline{\mathbf{Q}}_r, \overline{\mathbf{K}}_r, \overline{\mathbf{M}}_r, \overline{\mathbf{A}}_r \leftarrow \overline{\mathbf{p}} \leftarrow \overline{\mathbf{g}}_g \leftarrow \overline{KS} \tag{2.3}$$

where $\mathbf{K}, \mathbf{M}$ are the stiffness and mass matrices, and $\mathbf{Q}_r$ are the reduced natural modes shapes. $\mathbf{p}$ is the complex flutter eigenvalues for all modes and velocities and $\mathbf{g}_G$ are the damping values once the boundary in Section 5 has been applied. The KS value is the aggregated damping values computed using the KS function defined in Eq. (5.5). Note that in reverse the $\overline{\mathbf{K}}, \overline{\mathbf{M}}$ matrices are not explicitly computed, but the derivatives are directly accumulated onto the design variables.

Total derivatives of the flutter constraint $dKS/d\mathbf{x}$ is generated with a combination of AD and analytically differentiated code. Matrix operation such as LAPACK's [22] complex and real linear solves, matrix inverses, and eigenvalue solvers are not automatically differentiated, but instead must be differentiated analytically and implemented as such.

3. Derivative Verification

To demonstrate correct and accurate derivatives we perform a rigorous verification. Each function in the code is unit tested where sensitivities are computed using a second order central finite-difference stencil, a complex-step [45], as well as forward, and reverse mode AD. For a function of interest $I(x)$, the finite-difference stencil used here is

$$\frac{dI}{dx} = \frac{I(x+h) - I(x-h)}{2h} + O(h^2),$$

with a step-size ranging from $h = 10^{-3}$ to $h = 10^{-6}$, depending on the function under consideration. Note that in order to get a reasonable prediction with finite-difference, a step-size study was performed to get the most accurate gradient possible. For the complex-step method the sensitivity of a function is computed as

$$\frac{dI}{dx} = \frac{\Im[I(x+ih)]}{h} + O(h^2),$$

where $i = \sqrt{-1}$. The complex-step method does not suffer from a subtractive cancellation errors unlike the finite-difference method. The step-size can be made very small, e.g. $h = 10^{-40}$, hence the $O(h^2)$ truncation error becomes negligible.

One drawback of the complex-step method is that it cannot be used without modifying programs that already contain complex numbers and perform complex arithmetic. Such programs need to be modified such that the complex number is represented by two real numbers, one for the real part and one for the imaginary part. Complex arithmetic thus needs to be performed using manually defined functions and cannot be done using intrinsic functions. In this work, all code that uses complex number, such as the DLM and flutter analysis method, has been modified such that it utilizes only real numbers for complex calculations and is complex-step safe.

Since we have many more design variables than functions of interest $I(x)$, we employ the adjoint or the reverse mode AD to compute the gradient used in the optimization. One technique to verify the reverse mode is to implement the forward mode as well and perform a *dot product test*. The forward mode and the complex-step should match close to machine precision so implementing the forward mode as well is an important step. The *dot product test* [46] can be written as:

$$\begin{aligned}
\bar{\mathbf{x}}^* \dot{\mathbf{x}} &= \left(\mathbf{J}^* \bar{\mathbf{y}}\right)^* \dot{\mathbf{x}} \\
&= \bar{\mathbf{y}}^* \left(\mathbf{J} \dot{\mathbf{x}}\right) \\
&= \bar{\mathbf{y}}^* \dot{\mathbf{y}} \qquad set \quad \bar{\mathbf{y}} = \dot{\mathbf{y}} \\
&= \dot{\mathbf{y}}^* \dot{\mathbf{y}}
\end{aligned}$$

This equality should be exact to machine precision.

3.1. *Intermediate Derivatives*

Here we present derivative results of the flutter constraint with respect to the reduced stiffness matrix $\mathbf{K}_r$, the aerodynamic mesh nodes $\mathbf{X}^A$ and $\mathbf{Q}_r^A$ which are the reduced mode shapes, $\mathbf{Q}_r$, transferred on to the aerodynamic mesh. This includes derivatives of computation of the generalized loads and the newly developed flutter analysis code. As shown in Table 2 the reverse AD sensitivities developed match to machine precision when compared to complex-step. As expected, the second order finite difference method does not perform as well, and is sensitive to variation in step-size. In order to get the best finite difference derivative the step-size was varied from $h = 10^{-3}$ to $h = 10^{-7}$ depending on which derivative was being calculated.

Table 2. Intermediate sensitivities of the aggregated flutter constraint, KS, with respect to a single value in the reduced stiffness $\mathbf{K}_r$ matrix, aerodynamic mesh points $\mathbf{X}^A$ matrix and the reduced aerodynamic mode shapes $\mathbf{Q}_r^A$. Finite difference step size of $h = 10^{-6}$ gave overall the best results.

	$\frac{\partial KS}{\partial \mathbf{K}_r}$	$\frac{\partial KS}{\partial \mathbf{X}^A}$	$\frac{\partial KS}{\partial \mathbf{Q}_r^A}$
Finite Difference	0.00220121232353	0.0921885550120	-0.000394003940585
Complex-step	0.00220122045797	0.0921885466706	-0.000394004582749
AD (Reverse)	0.00220122045797	0.0921885466706	-0.000394004582749

3.2. *Total Derivatives*

Total sensitivities for a flat plate geometry presented in Section IV are given in Table 3. These are sensitivities of the entire analysis process. As before, finite difference offers less accuracy compared to AD or complex-step despite of performing a step-size study. The step-size chosen range from $h = 10^{-3}$ to $h = 10^{-7}$. The number of matching digits for the full derivative chain is somewhat less than what is presented in Table 2. Once passed through the reverse implementation of TACS and the Lanczos method some accuracy appears to be lost. The reasons behind this are not obvious at this point and requires further investigation. However, similar behavior is reported in literature (c.f. Table 2 in [47]). Accuracy of the implemented adjoint sensitivities is however sufficient for high-fidelity aerostructural optimization.

Table 3. Sensitivities of flutter constraint, KS, with respect to design variables, chord, span and plate thickness. Finite difference step size of $h = 10^{-3}$ for geometric variables (chord, span)and $h = 10^{-6}$ for structure variables gave overall the best results.

	$\frac{dKS}{dx_{\text{chord}}}$	$\frac{dKS}{dx_{\text{span}}}$	$\frac{dKS}{dx_{\text{thickness}}}$
Finite Difference	0.536986799217	-1.04625065236	177.140039513
Complex-step	0.536985041094	-1.04624172821	177.200299680
AD (Reverse)	0.536985160000	-1.04624205000	177.200209304

IV. Flat Plate Flutter Analysis and Optimization

1. Baseline Model Description

The flat plate geometry shown in Fig. 4 is chosen for verification purposes due to its simplicity and short optimization turnaround. This geometry was used in previous study [5] and is repeated here for completeness. The flat plate structure, shown in red, is embedded within a larger flat aerodynamic mesh, shown in gray. The red spheres are control points of the FFD volume, which both meshes have been embedded in. The black lines connecting the spheres show the outer edges of the FFD. The structural model consists of 12 elements in the streamwise direction and 40 elements in the spanwise direction a total of 480 finite elements. The finite element used here is the MITC4 shell element. Initial element thickness is chosen to be $t = 0.0012$ m. The aerodynamic model consists of 12 elements in the streamwise direction and 20 elements in the spanwise direction. Material properties, dimensions, and discretization for the baseline flat plate are summarized in Table 4.

Table 4. Flat plate mechanical properties, dimensions, and discretization of the structure and the aerodynamic surface.

	Variable	Symbol	Value
Mechanical properties	Density	ρ_s	2800 kg/m^3
	Modulus of elasticity	E	70 GPa
	Poisson ratio	ν	0.3
	Yield stress	σ_y	400 KPa
FEM	Thickness	t	0.0012 m
	Structure span	b_s	0.85 m
	Structure chord	c_s	0.21 m
	Finite elements, streamwise	n_x^{FEM}	12
	Finite elements, spanwise	n_y^{FEM}	40
DLM	Span	b	1.0 m
	Chord	c	0.3 m
	DLM elements, streamwise	n_x^{DLM}	12
	DLM elements, spanwise	n_y^{DLM}	20
	Planform area	A_{init}	0.3 m^2

For the flat plate analysis and optimization the air density is kept fixed and the Mach number is set to zero (incompressible flow). The velocity range to be analyzed is from 2 to 15 m/s. The flight conditions are summarized in Table 5. The aerodynamic loads generated by the DLM are generated using 50 values for the reduced frequency,k. These values are sampled non-uniformly using a quadratic or cubic stencil. In order to be consistent with previous work [5], we apply the constraint aggregation presented in Section 5 to the first 4 flutter modes. The first $r = 6$ natural modes and mode shapes are computed using the Lanczos algorithm, with a subspace of size $m = 20$, and are used as a basis for the flutter solution. Here we present only the lowest 4. Figure 11 shows the first 4 natural mode shapes.

Table 5. Flat plate operating conditions under investigation used in the baseline analysis.

	Variable	Symbol	Value
Operating conditions	Mach	M_∞	0
	Lift coefficient	C_L	0.5
	Air density	ρ_∞	1.225 kg/m^3
	Air speed range	U_∞	2 – 15 m/s
	Reduced frequency range	k	0 – 20
Range equation coeff.	Flight speed (range eqn.)	V	1.0 m/s
	Thrust specific fuel consumption	c_T	1.0 lb/(lbf · h)
	Fixed weight	W_{fixed}	1.0 kg
	Fuel weight	W_{fuel}	0.25 kg

1.1. Baseline Model Flutter Analysis

We investigate the flutter and divergence characteristics of the baseline geometry under the given operating conditions. Figure 12 shows the flutter and divergence characteristics for the baseline geometry where damping and frequency are given in dimensional units. The unstable region is highlighted in a pink color. In Fig. 12a the unstable region is the entire positive damping region. Divergence and flutter occur at $U_d = 13.99$ and $U_f = 14.07$ m/s, respectively.

In Fig. 12b the boundary in Eq. (5.1) has been applied with $g^* = -1$ rad/s, $U^* = 13$ m/s, and $\beta = 1$ rad · s/m^2.

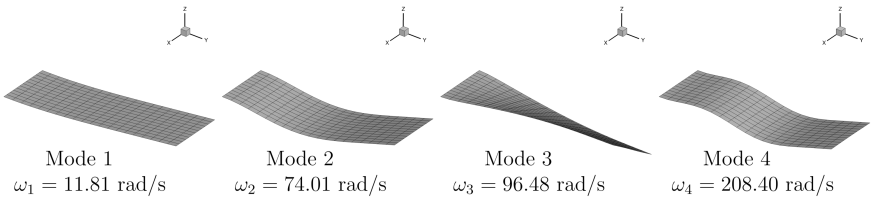


Figure 11. First 4 modes shapes for the baseline geometry.

The selection of current parameters sets the minimum implicit allowable flutter speed to be 14 m/s. The boundary is shown in light gray and defines the start of the unstable damping region. The boundary divergence speed occurs at $U_{G,d} = 12.00$ m/s, but the boundary flutter speed has increased and is $U_{G,f} = 14.30$ m/s. Note that these are not the physical divergence or flutter velocities, but the values where modes intersect the constraint boundary $G(U)$. The physical flutter and divergence speeds U_f, U_d are the ones reported above. Frequency migration is shown in Fig. 12c. The 1st bending mode diverges whereas the 2nd bending is the fluttering mode.

2. Problem Statement

For this flat plate problem, the design variables chosen here are as follows: plate thickness of the entire plate $x_{\text{thickness}}$, span x_{span} and chord length x_{chord}. No sweep, taper or dihedral is considered here. The objective is to maximize range using the Breguet range equation,

$$R = \frac{V}{c_T}\frac{C_L}{C_D}\ln\left(\frac{W_{\text{init}}}{W_{\text{final}}}\right), \tag{2.1}$$

where R is range, V/c_T is the flight speed to thrust-specific fuel consumption ratio, C_L/C_D is the lift to drag ratio, and $W_{\text{init}}/W_{\text{final}}$ is the initial to final cruise weight ratio. For simplicity we assume that $V/c_T = 1$, and we define the cruise weights as

$$\begin{aligned} W_{\text{final}} &= W_{\text{fixed}} + W_{\text{plate}}, \\ W_{\text{init}} &= W_{\text{final}} + W_{\text{fuel}}, \end{aligned} \tag{2.2}$$

where the W_{fixed} is a fixed weight and W_{fuel} is the fuel weight. The lift coefficient is fixed at $C_L = 0.5$ and the drag coefficient is calculated assuming it consists only of the lift induced drag,

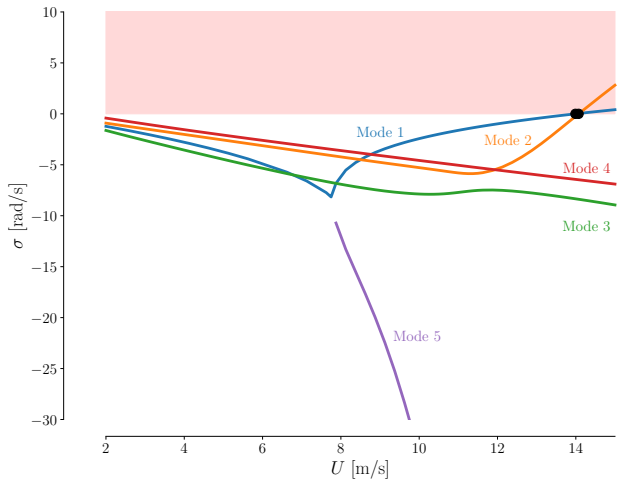
$$C_D = \frac{C_L^2}{\pi e AR}, \tag{2.3}$$

where the wing span efficiency factor is set to $e = 1$ for simplicity. AR is the aspect ratio defined as $AR = b^2/S$ where b is the span and S is the planform area. The range can thus be increased by reducing the drag coefficient and by reducing the weight, i.e. the thickness of the plate. The drag coefficient is reduced by increasing the aspect ratio as other parameters are fixed.

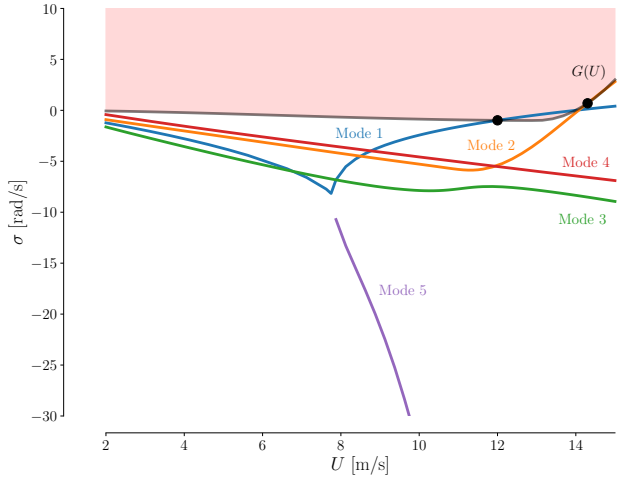
The chord and span directly affect the aspect ratio so we want to formulate the problem in terms of the aspect ratio rather than directly the chord and the span. By adding an area equality constraint we ensure that there is a link between the chord and span, allowing us to instead look at the trade-off between thickness and aspect ratio. Further, this allows for visualization, which can provide valuable information about the design space . The initial area is given in Table 4.

We consider two scenarios, one without applying the constraint curve, and a second one where the constraint curve is active, using the same parameters as presented in the baseline analysis. In the former we require that no flutter or divergence must occur for the entire velocity range of 2-15 m/s. This case allows for higher flutter or divergence velocities but may produce a less robust wing design. For the latter case, minimum flutter or divergence speed is set implicitly by the parameters chosen for the constraint curve in the baseline analysis. The implicit minimum flutter or divergence speed is 14 m/s which is computed from the parabolic part of the constraint curve i.e. where it intersects the zero axis. Hence, the range 2-14 m/s must then be flutter and divergence free. In addition to pushing modes further into the design space making for a more robust design, it also controls how rapidly modes can flutter up until a velocity of 15 m/s, making for a more robust design.

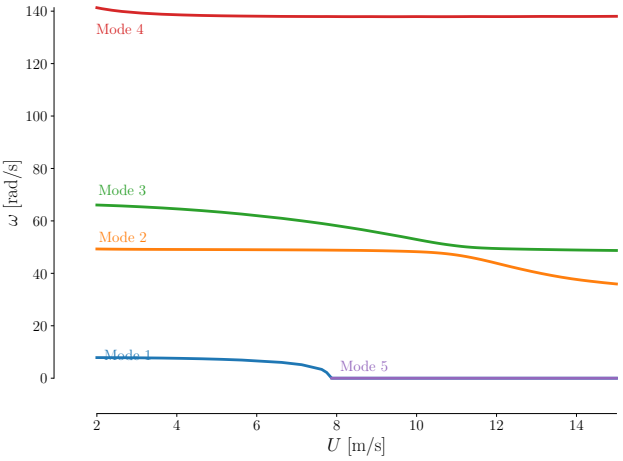
The side constraints are as follows, chord and span are specified such that the aspect ratio is allowed to vary from $1 \leq AR \leq 6$, and the material thickness of the plate is allowed to vary from $0.0012 \leq t \leq 0.0025$ m. The optimization problem is summarized in Table 6 and Fig. 13 shows the flutter analysis implementation as it is applied in the optimization.



a) Damping shown without constraint boundary. Divergence and flutter occur at $U_d = 13.99$ and $U_f = 14.07$ m/s, respectively.



b) Damping shown with constraint boundary applied on figure (shown in gray). Divergence and flutter (boundary intersection) occur at $U_{G,d} = 12.00$ and $U_{G,f} = 14.30$ m/s, respectively.



c) Frequency migration

Figure 12. Flutter analysis of the flat plate baseline geometry. The unstable area is highlighted with a pink color in a), b). Frequency migration is shown in c).

Table 6. Optimization formulation of the flat plate problem

	Function/variable	Description	Quantity
maximize	Range	Breguet equation	
with respect to	x_{span}	Plate span	1
	x_{chord}	Plate chord	1
	$x_{\text{thickness}}$	Plate thickness	1
		Total design variables	**3**
subject to	A - $A_{\text{init}} = 0.0$	Fixed plate area	1
	$\text{KS}(g_{G,ij}) \leq 0$	KS aggregate of damping values for all modes	1
		Total constraints	**2**

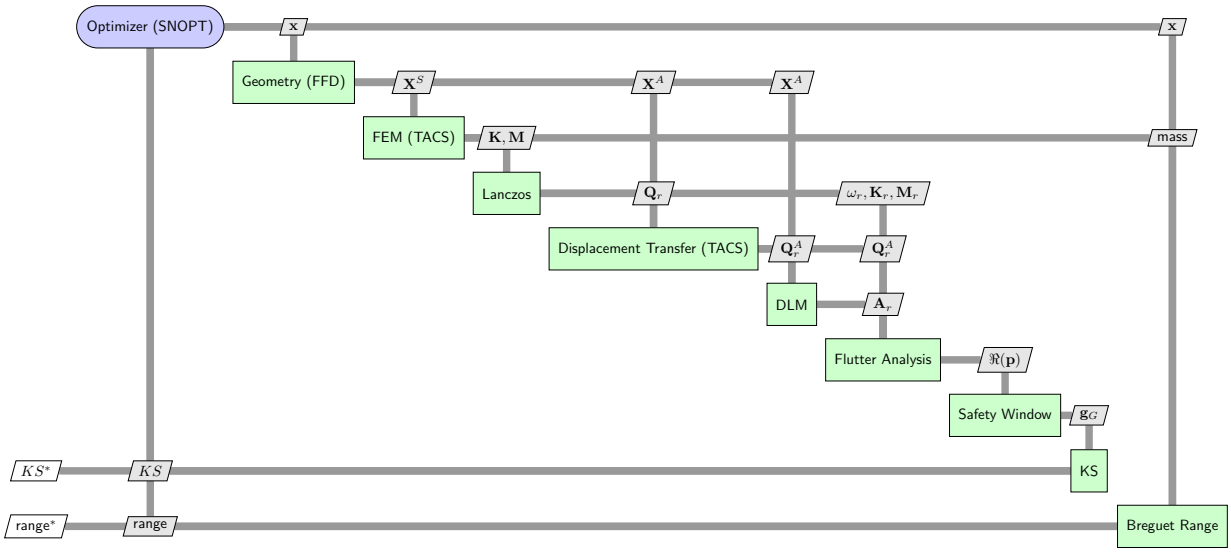
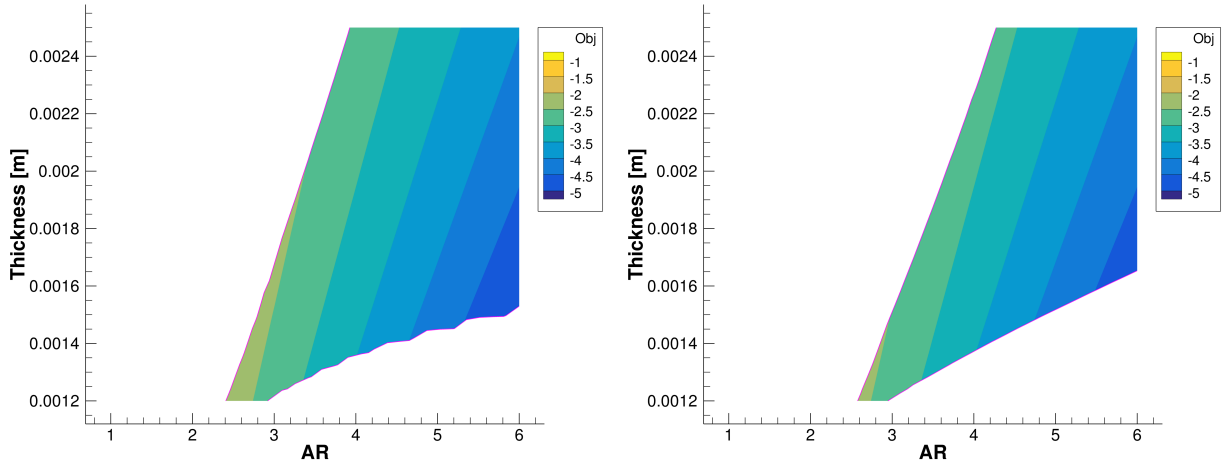


Figure 13. XDSM [9] showing the flutter constraint as applied in the optimization.

3. Design space analysis

Before running an optimization, its valuable to investigate the design space. A contour plot is generated by sweeping over both the aspect ratio range $1 \leq AR \leq 6$ and the thickness range $0.0012 \leq t \leq 0.0025$ m. A grid of 32 steps in each variable is used giving a total of 1024 points. Figure 14 shows contour plots of the objective function in the feasible design space with and without the contraint curve $G(U)$ active. Where the aggregated flutter constraint value is greater than zero, $KS > 0$, the objective function value has been blanked out, as this part of the design space is infeasible. Comparing Figs. 14a and 14b it is evident that with the constraint curve active the feasible region is smaller. Furthermore, the constraint curve results in a smoother feasible design space. The objective function is smooth with a clear maximum. By visual inspection the aspect ratio is $AR^* \approx 6$, with a plate thickness of $t^* \approx 0.00153$ m and $t^* \approx 0.00165$ m without and with the constraint curve active, respectively. Note that here the range is given in nondimensional units and is presented as a negative quantity, due to objective being a maximization. In these cases the maximum range is approximately -5.0 and -4.8, without and with the constraint curve active, respectively.



a) Without constraint curve **b) With constraint curve**

Figure 14. Contour plot of the objective function, range, shown with the flutter constraint applied to the contour plot. Blanked out regions represent values of where the constraint is violated or $KS > 0$. To generate the contour, the design space is sampled using 32 points in both variables for a total of 1024 design points.

In both figures the infeasible design space consists of two disjoint regions, one with low aspect ratio, spanning the entire thickness bounds, and the second one at higher aspect ratio with relatively thin plate thickness. These regions indicate that either a mode is fluttering or has bifurcated and diverged. In order to investigate the infeasible regions further and determine the cause of these regions being infeasible, we fix the thickness of the plate to $t = 0.001619$ m and perform sweep over the aspect ratio $1 \leq AR \leq 6$, using 256 points. This thickness value is chosen since it is close to the optimum thickness value. Here we perform the analysis *without* the constraint curve active. Inspecting Fig. 15 we observe that the aggregated constraint value is smooth and continuous, except when the wing design changes from aspect ratio 2.176 to 2.196. Another observation is at low aspect ratio the damping is positive, thus unstable. With larger aspect ratio the wing becomes stable at $AR \approx 3$ and continues until $AR \approx 5$ when the damping starts to increase again. This increment is investigated later. At the discontinuity, Figs. 16a and 16b show the damping and frequency migration for each design. Solid and dashed lines represent the wing design at aspect ratio 2.176 and 2.196 respectively. We observe that the damping characteristics of mode 4 seem to have changed drastically despite that the frequency changes only marginally for all modes. Other damping modes change marginally between designs, as expected.

This issue of discontinuity is due to a swap in natural mode shapes computed for each design. As the wing design changes a higher frequency natural mode approaches and then becomes lower than natural mode 4, i.e. natural mode 5 and mode 4 swap places between these two designs. The associated mode shapes also swap places between these two designs. This becomes evident in Fig. 17 where the first five natural mode shapes are shown for these two designs. Since the natural modes are applied in the computation of the generalized matrices, this affects the flutter modes directly such that a different damping behavior may appear. Another interesting observation in Fig. 16 is that mode 4

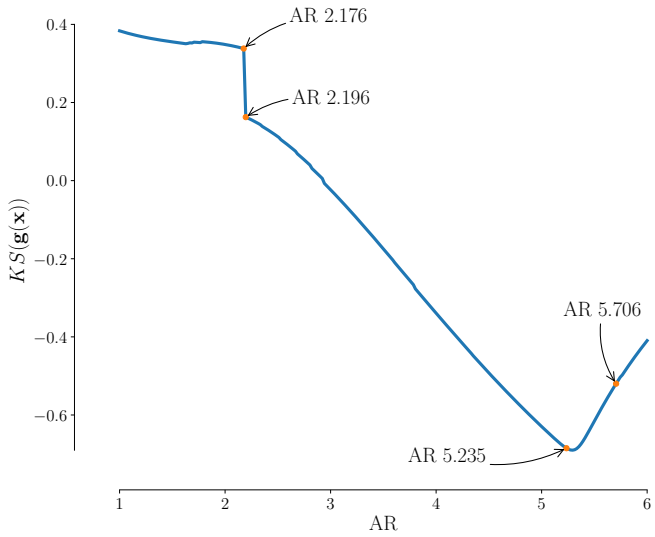


Figure 15. Aggregated damping values for a slice through the design space at $t = 0.001619$ m. Small change in the design i.e. increasing the aspect ratio from 2.176 to 2.196 results in a discontinuity.

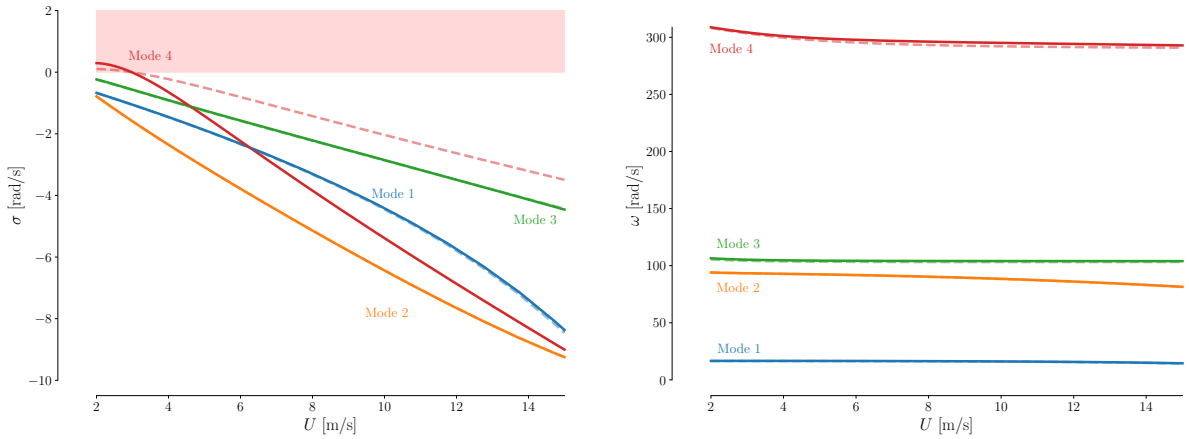
has fluttered at a very low velocity. This is believed to be caused by the insufficient DLM resolution as the reduced frequency at such low flow velocities can be large for the higher natural modes.

To investigate the increase in the aggregated damping value at higher aspect ratio in Fig. 15 we select two designs at aspect ratio 5.235 and 5.706. Damping and frequency plots for these designs are shown in Figs. 18a and 18c, respectively. The increase in damping is caused by the bifurcated mode 1 where, for the higher aspect ratio design, bifurcation occurs at a lower velocity. Additionally, the damping is shown with the constraint curve in Fig. 18b where the bifurcated mode 1, at the higher aspect ratio design, is close to crossing into the unstable region. This is why the blanked out region in Fig. 14b at higher aspect ratio is larger than the one without it active. The effect of the constraint curve on the design is that is will push the bifurcated mode further into the damping region. Thus, in order to be flutter free (as indicated by Fig. 14b) a thicker plate is needed, which will increases the wing stiffness resulting in a more robust design.

4. Optimization results

In this section we present the results from the optimization problem presented in Section 2. The baseline and optimized plate flutter and divergence characteristics are presented in Fig. 19 and the optimized modes and mode shapes are shown in Fig. 20. Inspecting the mode migration in Fig. 19a we see that mode 1 is the critical mode, the active mode, where it lies close up to the constraint curve. However, no mode is close to being active for the chosen flight conditions, and has been pushed out of the range considered. This indicates that the diverging mode is driving the design in the optimization. Similarly, Fig. 19b shows that the optimized wing has better overall frequency separation than the baseline. The optimized mode shapes given in Fig. 20 are found to be the same as the baseline i.e. in order, 1st bending, 2nd bending, 1st torsion and 3rd bending.

Figure 21 shows the major iterations that the optimizer took in the left figure. The optimum aspect ratio and thickness is $AR^* = 6$ and $t^* = 0.00166$ m, respectively. The maximum range (objective) is found to be -4.8192 units and has increased by 60%. The right plot in the figure compares the baseline and optimized planforms are compared. Table 7 compares the numerical values of the thickness, aspect ration, and range. The flutter and divergence free optimized geometry has an increased aspect ration of 82%, increasing the range by 60%, despite making the structure (plate) 38% thicker. Figure 22 shows the convergence history for of the optimization. Minimum feasibility and optimality was specified as $< 1e^{-6}$ and was reached in both cases. Exceptional feasibility is demonstrated as it reaches machine zero.



a) Damping b) Frequency

Figure 16. Small change in the design from aspect ratio from 2.176 to 2.196, shown in solid and dashed lines respectively, results in a large change in the damping. Frequency changes minimally as expected.

Table 7. Numerical comparison of baseline and optimized wing.

	Aspect ratio AR [-]	Thickness t [m]	Range obj [-]
Baseline	3.29522	0.00120	-3.00636
Optimized	6.00000	0.00166	-4.81924
Rel. Diff. (%)	0.82	0.38	0.60

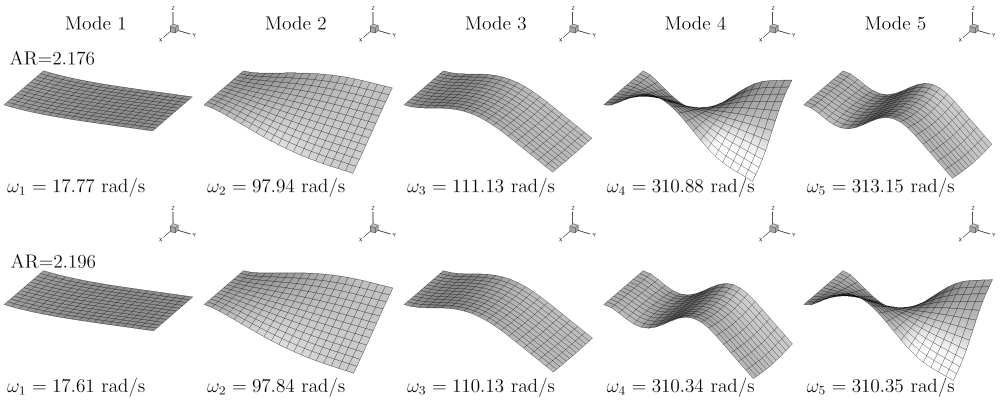


Figure 17. First five mode shapes for designs at aspect ratio 2.169 and 2.196. Mode 4 and 5 swap between the two designs.

V. Conclusions

In this work an in-house efficient and robust flutter analysis method was developed, implemented and verified using commercial software. Formulation of a continuous flutter and divergence constraint suitable for gradient based optimization is developed and implemented. Sensitivities of the proposed flutter analysis and constraint is computed accurately and efficiently using a combination of analytic methods (adjoint) and automatic differentiation in reverse, which scale with the number of outputs, in contrast to methods such as finite-difference, complex-step or forward mode AD which all scale with the number of inputs. This enables us to perform an optimization of problems with tens or hundreds of design variables. Sensitivities of the flutter constraint, with respect to all design variables, is thus evaluated using only one evaluation. Furthermore, modes shapes are considered to be a function of the design variables and are differentiated accordingly. This is in contrast to other works which consider fixed mode shapes resulting in potentially incorrect sensitivities, in particular when considering planform optimization. The proposed flutter constraint is demonstrated on an idealized wing using a truly multidisciplinary objective, the Breguet range equation, by maximizing range. Range is increased by 60% while maintaining a divergence and flutter free design.

Acknowledgements

The material is based upon work supported by Airbus in the frame of the Airbus / Michigan Center for Aero-Servo-Elasticity of Very Flexible Aircraft. Special thanks to Patrick Teufel for his expert advice and review of this paper.

References

[1] Kenway, G. K. W., Kennedy, G. J., and Martins, J. R. R. A., "A Scalable Parallel Approach for High-Fidelity Aerostructural Analysis and Optimization," *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Honolulu, HI, April 2012, AIAA 2012-1922.

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

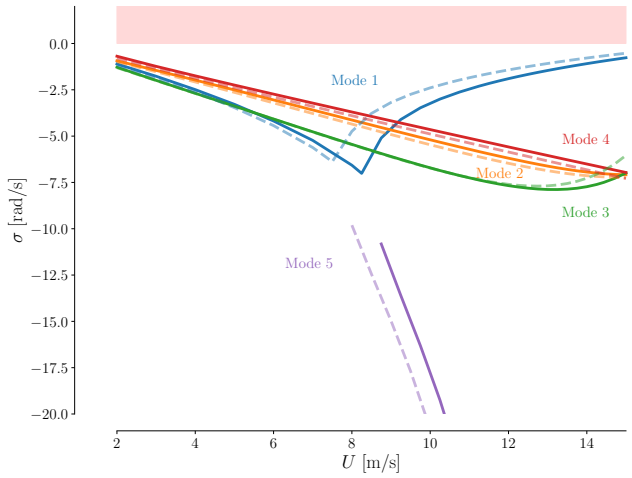
[3] Kenway, G. W. K. and Martins, J. R. R. A., "High-fidelity aerostructural optimization considering buffet onset," *Proceedings of the 16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Dallas, TX, June 2015, AIAA 2015-2790.

[4] 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.

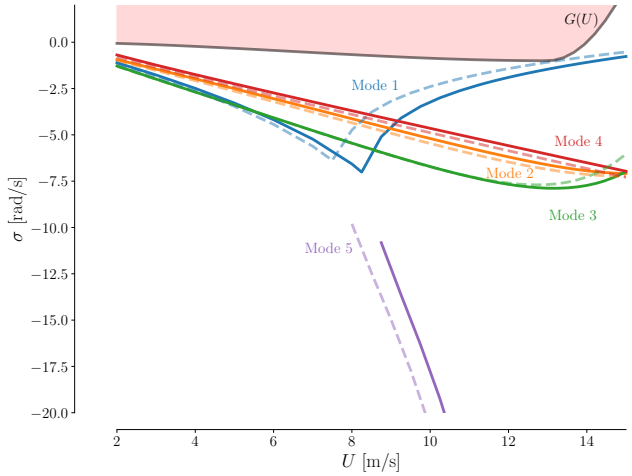
[5] Jonsson, E., Kenway, G. K. W., Kennedy, G. J., and Martins, J. R. R. A., "Development of Flutter Constraints for High-fidelity Aerostructural Optimization," *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Denver, CO, June 2017, AIAA 2017-4455.

[6] Stanford, B. K. and Dunning, P. D., "Optimal Topology of Aircraft Rib and Spar Structures Under Aeroelastic Loads," *Journal of Aircraft*, Vol. 52, No. 4, Sept. 2014, pp. 1298–1311. doi:10.2514/1.C032913.

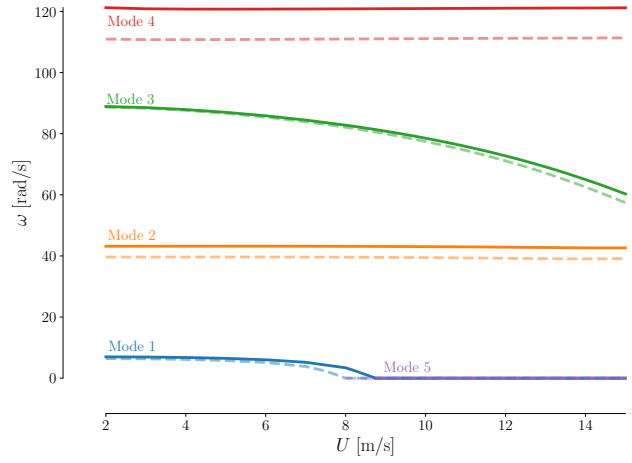
[7] Stanford, B. K., Jutte, C. V., and Wieseman, C. D., "Trim and Structural Optimization of Subsonic Transport Wings Using Nonconventional Aeroelastic Tailoring," *AIAA Journal*, Vol. 54, No. 1, Oct. 2015, pp. 293–309. doi:10.2514/1.J054244.



a) Damping

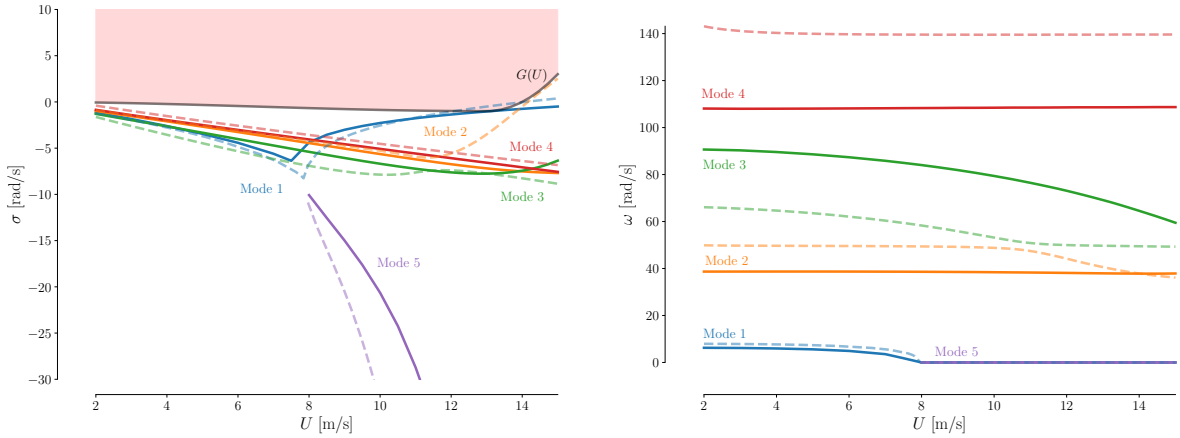


b) Damping shown with constraint boundary, expanding the unstable region into the negative damping region.



c) Frequency

Figure 18. Damping and frequency plots for designs with aspect ratio 5.235 (solid) and 5.706 (dashed) for a fixed thickness. Increasing the aspect ratio shifts the bifurcation of mode 1 to a lower velocity. Frequency changes minimally as expected.



a) Damping **b) Frequency**

Figure 19. Baseline (dashed) and optimized (solid) frequency and damping characteristics.

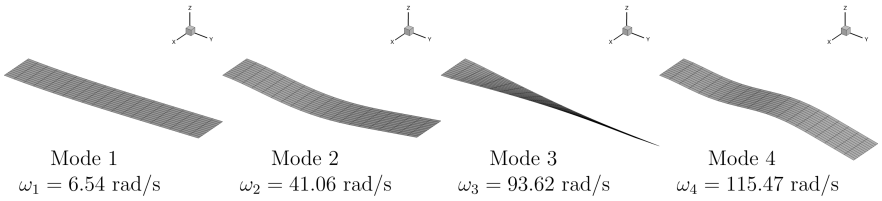


Figure 20. First 4 modes shapes for the optimized geometry.

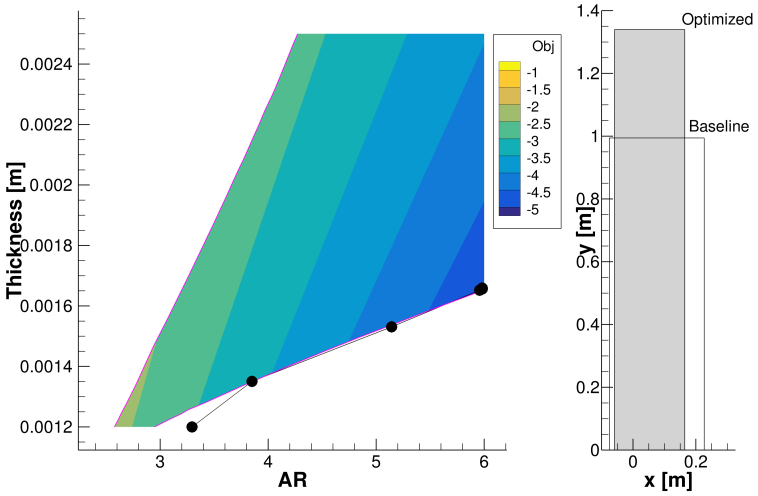


Figure 21. Optimization results showing the major iterations (left) taken by the optimizer and the initial and optimized wing planform (right). Optimum aspect ratio and thickness are $AR^* = 6, t^* = 0.00166$ m, respectively

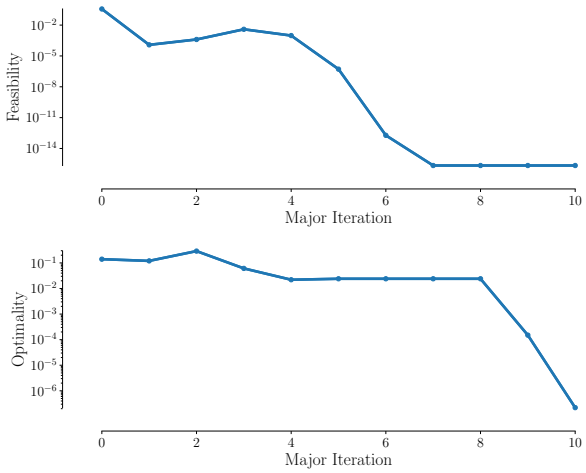


Figure 22. Optimization convergence history.

[8] Bartels, R. E. and Stanford, B. K., "Aeroelastic Optimization with an Economical Transonic Flutter Constraint Using Navier-Stokes Aerodynamics," *Journal of Aircraft*, Vol. 55, No. 4, 2018, pp. 1522–1530. doi:10.2514/1.C034675.

[9] Lambe, A. B. and Martins, J. R. R. A., "Extensions to the Design Structure Matrix for the Description of Multidisciplinary Design, Analysis, and Optimization Processes," *Structural and Multidisciplinary Optimization*, Vol. 46, August 2012, pp. 273–284. doi:10.1007/s00158-012-0763-y.

[10] Kenway, G. K., Kennedy, G. J., and Martins, J. R. R. A., "A CAD-Free Approach to High-Fidelity Aerostructural Optimization," *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, No. AIAA 2010-9231, Fort Worth, TX, Sept. 2010. doi:10.2514/6.2010-9231.

[11] Sederberg, T. W. and Parry, S. R., "Free-form Deformation of Solid Geometric Models," *SIGGRAPH Comput. Graph.*, Vol. 20, No. 4, Aug. 1986, pp. 151–160. doi:10.1145/15886.15903.

[12] Kennedy, G. J. and Martins, J. R. R. A., "Parallel Solution Methods for Aerostructural Analysis and Design Optimization," *Proceedings of the 13^{t}h AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Forth Worth, TX, September 2010.

[13] Kennedy, G. J. and Martins, J. R. R. A., "A parallel aerostructural optimization framework for aircraft design studies," *Structural and Multidisciplinary Optimization*, Vol. 50, No. 6, December 2014, pp. 1079–1101. doi:10.1007/s00158-014-1108-9.

[14] Brown, S. A., "Displacement Extrapolation for CFD+CSM Aeroelastic Analysis," *Proceedings of the 35th AIAA Aerospace Sciences Meeting*, Reno, NV, 1997, AIAA 1997-1090.

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

[16] Blair, M., "A compilation of the mathematics leading to the doublet lattice method," Tech. rep., DTIC Document, 1992.

[17] MSC Software Corp, *MSC Nastran 2012 Release Guide*.

[18] Rodden, W. P., Taylor, P. F., and McIntosh, S. C., "Further Refinement of the Subsonic Doublet-Lattice Method," *Journal of Aircraft*, Vol. 35, No. 5, Sept. 1998, pp. 720–727. doi:10.2514/2.2382.

[19] Hassig, H. J., "An approximate true damping solution of the flutter equation by determinant iteration." *Journal of Aircraft*, Vol. 8, No. 11, 1971, pp. 885–889.

[20] Rodden, W. P., Bellinger, E. D., Harder, R. L., and Center., L. R., *Aeroelastic addition to NASTRAN*, National Aeronautics and Space Administration, Scientific and Technical Information Branch, 1979.

[21] Bathe, K.-J., *Finite element procedures*, Klaus-Jurgen Bathe, 2006.

[22] Anderson, E., Bai, Z., Dongarra, J., Greenbaum, A., McKenney, A., Du Croz, J., Hammarling, S., Demmel, J., Bischof, C., and Sorensen, D., "LAPACK: A Portable Linear Algebra Library for High-performance Computers," *Proceedings of the 1990 ACM/IEEE Conference on Supercomputing*, Supercomputing '90, IEEE Computer Society Press, Los Alamitos, CA, USA, 1990, pp. 2–11.

[23] van Zyl, L. H., "Aeroelastic Divergence and Aerodynamic Lag Roots," *Journal of Aircraft*, Vol. 38, No. 3, May 2001, pp. 586–588. doi:10.2514/2.2806.

[24] Chen, P., "Damping perturbation method for flutter solution: the g-method," *AIAA journal*, Vol. 38, No. 9, 2000, pp. 1519–1524.

[25] Rodden, W. P. and Bellinger, E. D., "Aerodynamic lag functions, divergence, and the British flutter method," *Journal of Aircraft*, Vol. 19, No. 7, July 1982, pp. 596–598. doi:10.2514/3.44772.

[26] van Zyl, L. H., "Use of eigenvectors in the solution of the flutter equation," *Journal of Aircraft*, Vol. 30, No. 4, July 1993, pp. 553–554. doi:10.2514/3.46380.

[27] 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).

[28] Bisplinghoff, R. L., Ashley, H., and Halfman, R. L., *Aeroelasticity*, Courier Corporation, 2013.

[29] Ringertz, U., "On structural optimization with aeroelasticity constraints," *Structural optimization*, Vol. 8, No. 1, 1994, pp. 16–23.

[30] Stanford, B., Beran, P., and Bhatia, M., "Aeroelastic Topology Optimization of Blade-Stiffened Panels," *Journal of Aircraft*, Vol. 51, No. 3, March 2014, pp. 938–944. doi:10.2514/1.C032500.

[31] Langthjem, M. A. and Sugiyama, Y., "Optimum Shape Design Against Flutter Of A Cantilevered Column With An End-mass Of Finite Size Subjected To A Non-conservative Load," *Journal of Sound and Vibration*, Vol. 226, No. 1, 1999, pp. 1–23. doi:https://doi.org/10.1006/jsvi.1999.2211.

[32] Odaka, Y. and Furuya, H., "Robust structural optimization of plate wing corresponding to bifurcation in higher mode flutter," *Structural and Multidisciplinary Optimization*, Vol. 30, No. 6, dec 2005, pp. 437–446. doi:10.1007/s00158-005-0538-9.

[33] Hajela, P., "A root locus-based flutter synthesis procedure," *Journal of Aircraft*, Vol. 20, No. 12, 1983, pp. 1021–1027. doi:10.2514/3.48206.

[34] Stanford, B. K., "Role of Unsteady Aerodynamics During Aeroelastic Optimization," *AIAA Journal*, Vol. 53, No. 12, Sept. 2015, pp. 3826–3831. doi:10.2514/1.J054314.

[35] Haftka, R. T., "Parametric Constraints with Application to Optimization for Flutter Using a Continuous Flutter Constraint," *AIAA Journal*, Vol. 13, No. 4, April 1975, pp. 471–475. doi:10.2514/3.49733.

[36] 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. doi:10.1016/S1474-6670(17)65584-8.

[37] Poon, N. M. K. and Martins, J. R. R. A., "An Adaptive Approach to Constraint Aggregation Using Adjoint Sensitivity Analysis," *Structural and Multidisciplinary Optimization*, Vol. 34, No. 1, July 2007, pp. 61–73. doi:10.1007/s00158-006-0061-7.

[38] Wrenn, G. A., "An Indirect Method for Numerical Optimization Using the Kreisselmeier–Steinhauser Function," Tech. Rep. CR-4220, NASA Langley Research Center, Hampton, VA, 1989.

[39] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Journal of Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006. doi:10.1137/S1052623499350013.

[40] 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.

[41] Griewank, A. and Walther, A., *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd ed., 2008.

[42] 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.

[43] Hascoët, L. and Pascual, V., "TAPENADE 2.1 User's Guide," Technical report 300, INRIA, 2004.

[44] Pascual, V. and Hascoët, L., "Extension of TAPENADE Towards Fortran 95," *Automatic Differentiation: Applications, Theory, and Tools*, edited by H. M. Bücker, G. Corliss, P. Hovland, U. Naumann, and B. Norris, Lecture Notes in Computational Science and Engineering, Springer, Berlin, Germany, 2005.

[45] 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.

[46] Hascoet, L. and Pascual, V., "The Tapenade automatic differentiation tool: Principles, model, and specification," *ACM Trans. Math. Softw.*, Vol. 39, No. 3, May 2013, pp. 20:1–20:43. doi:http://dx.doi.org/10.1145/2450153.2450158.

[47] 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.