

Multidisciplinary Design Optimization of a Transonic Truss-Braced Wing using Physics-Based Models

Timothy R. Brooks*, Kenneth T. Moore[†], and Eliot Aretskin-Hariton[‡]
NASA Glenn Research Center, Cleveland, OH, 44135, USA

Irian Ordaz[§] and Jason Kirk[¶]
NASA Langley Research Center, Hampton, VA, 23681, USA

Economic and environmental forces have placed pressure on the aviation industry to produce future aircraft designs with substantial improvements over those flying today. In order to meet these ambitious goals, new concepts such as the Transonic Truss-Braced Wing (TTBW) are being considered which diverge from the traditional tube-and-wing design. While this concept offers potential performance benefits over its traditional counterpart, it also features more coupling between various physics disciplines that must be considered during the design of the concept. This unique challenge provides a ripe opportunity for the application of multidisciplinary design optimization tool sets. The final work will couple together the mission analysis capability of Aviary with OpenAeroStruct for aerodynamic analysis and TACS for structural analysis. Using these tools we will preform a design optimization on the TTBW concept minimizing design fuel burn. The optimization will include design variables for mission trajectory, wing geometry, and structural sizing variables.

I. Nomenclature

C_{D_0}	=	aircraft skin friction drag
C_{f_e}	=	skin friction regression constant
d_j	=	distance state
D	=	aircraft total drag
h_j	=	altitude state
L	=	aircraft total lift
R	=	residual constraint
S_{ref}	=	aircraft reference area
S_{wet}	=	aircraft wetted area
t_0	=	phase initial time
W_j	=	aircraft weight state
W_{fuel}	=	aircraft fuel weight
W_{gross}	=	aircraft gross weight
α_j	=	angle of attack state
Δt	=	phase duration
θ_i^{wing}	=	wing spanwise twist vector

II. Introduction

IN order to keep pace with increasing environmental and economic incentives, the U.S. is working with the global community to achieve net-zero greenhouse gas emissions by 2050. To accomplish this goal, NASA has invested in

*Systems Engineer, Banner Quality Management Inc., and AIAA Member.

[†]Senior Analyst, Banner Quality Management Inc., and AIAA Member.

[‡]Aerospace Technologist, Propulsion Systems and Technologies Branch, and AIAA Member.

[§]Aerospace Engineer, Aeronautics Systems Analysis Branch, AIAA Member

[¶]Aerospace Engineer, Aeronautics Systems Analysis Branch, AIAA Member

various initiatives such as the Sustainable Flight National Partnership (SFNP) [1] and the Advanced Aircraft Concepts for Environmental Sustainability (AACES) 2050 [2] effort to mature and integrate key technologies for next-generation subsonic transports, as well as explore new aircraft concepts that can continue to reduce aviation environmental impact.

In support of the SFNP, NASA's Model-Based Systems Analysis and Engineering (MBSA&E) effort is working to develop a systems-level, digital integration across SFNP projects. This architecture will support the assessment, advancement, and adoption of sustainable technologies for subsonic transport aircraft concepts with 2030 entry into service and beyond. The main objectives of this effort include the development of an open, cross-project, cross-program, external-capable MBSA&E framework building off NASA's investments and capabilities across various NASA projects. The effort will also support coordinated, integrated systems analysis studies including the development of common, open, reference and vision vehicle models and technology benefit assessments and sensitivity studies informed by the SFNP demos.

The thrust toward increased sustainability has led to increased interest in novel and advanced concepts which often produce new challenges associated with tighter coupling between design disciplines in ways not seen on their traditional counterparts. The Transonic Truss-Braced Wing (TTBW) is one such concept, offering greater aerodynamic performance due to its large span, but offering more structural complexity due to the design of the wing and truss members [3]. This makes the concept an ideal candidate for Multidisciplinary Design Analysis and Optimization (MDAO).

To date, there has already been great interest in applying MDAO to the TTBW concept. In work by Aretskin-Hariton et al. [4] a fuel burn optimization for the concept was presented in which the design was optimized for: mission trajectory, electrified turbine size, electric motors size for climb assist, and battery capacity. This work coupled a number of physics-based disciplinary analyses together including an engine analysis using pyCycle [5] and a drag table built using aerodynamic analyses from VSPAERO [6]. While this study did include a physics-based analyses, the consideration for aircraft design variables during the optimization were limited to the sizing of the battery for climb. This limited the benefits that could be captured through optimization. Leader et al. [7] expanded on this work by increasing the fidelity of the electrical analysis used to size the battery model for a hybrid-electric version of the TTBW model. Work by Fazal et al. [8] showed the capability for integrating uncertainty quantification into the TTBW design framework and demonstrated the importance of these considerations within the design process.

Past works on other concepts have shown the benefit that comes with considering aircraft and mission design under the same optimization. Hwang et al. [9], using the NASA Common Research Model [10], conducted a coupled aircraft design, airline flight route allocation, and mission trajectory optimization problem to maximize airline profit. The study found an addition 2.3% improvement in airline profit relative to a design in which the aircraft design and mission trajectory were optimized separately. Kaneko and Martins [11] performed a simultaneous design and trajectory optimization of two unmanned aerial vehicle concepts considering several optimization coupling architectures. In their work they found an additional reduction of 4.2% in energy required to climb when compared to a design in which vehicle design and trajectory were optimized sequentially.

Our planned work will expand upon the work presented by Aretskin-Hariton et al. [4] in several ways. First, we will include physics-based aerodynamic and structural analyses in the optimization problem. This will allow us to better capture system-level metrics within the optimization and arrive at better optimized designs. Second, in addition to the original mission optimization problem, we plan to add aircraft geometric and structural design variables. This will allow us to simultaneously optimize both the aircraft design and mission for greater synergistic benefits.

III. Aircraft Model

The aircraft model used in this work is a non-proprietary concept referred to as the Transonic Truss-Braced Wing Tech Collector. The concept was developed by NASA to allow future vehicle technology studies, open publication of results, and easier collaboration with parties outside of NASA. The model is based off of the Boeing SUGAR aircraft concept [3], a next-generation aircraft concept with an target for entry into service of 2035. The vehicle concept is shown in Fig. 1 and top-level design parameters for the aircraft can be found in Table 1.

IV. Tool Background

The tools used in the current work are: Aviary [12], OpenAeroStruct [13], and pyGeo [14]. Aviary serves as the top-level backbone aircraft design framework into which all other analysis tools are wrapped. OpenAeroStruct serves as a medium-fidelity aerodynamic model to be used during mission analysis. pyGeo serves as a geometry engine that ties

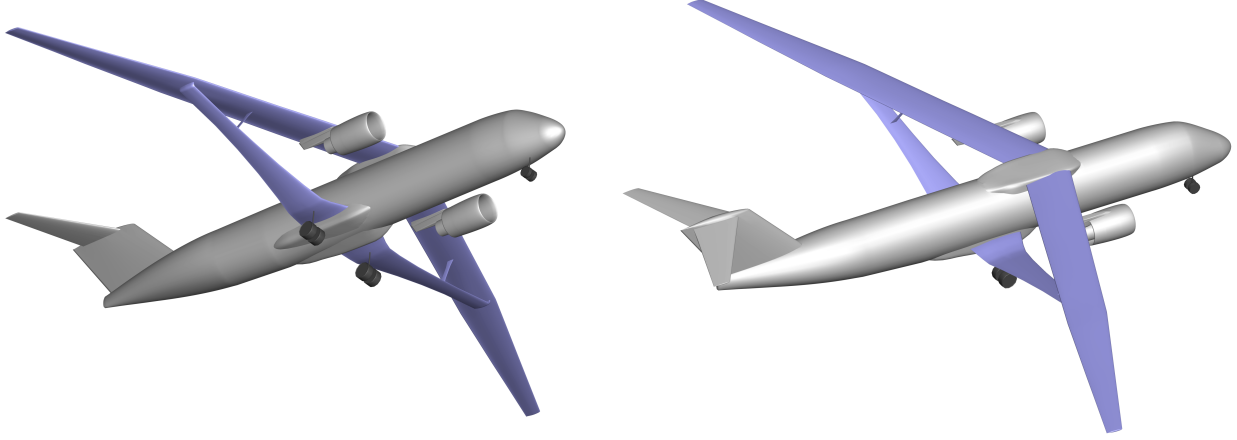


Fig. 1 Transonic Truss-Based Wing geometry model used for this work [8].

Table 1 TTBW specifications.

Parameter	Value	Units
TOGW	145 000	lb
OEW	88 000	lb
Passengers	154	
Design range	3 200	nm
Cruise Mach	0.8	
Aspect ratio	19.7	
Reference Area	1 475	ft ²
Span	170	ft
Sweep	21.8	deg

top-level geometric design parameters to the computational mesh used by OpenAeroStruct. In addition to these current suite of tools, we plan to extend the current work by considering aerostructural sizing using TACS [15]. The details of each tool will be elaborated below.

A. Aviary

Aviary^{*} is an open-source framework for performing aircraft conceptual design in an MDAO context [12]. It includes the fundamental methods from two legacy NASA tools FLOPS (Flight Optimization System) [16] and GASP (General Aviation Synthesis Program) [17], both of which use low-fidelity correlations and empirical relationships that are relevant for existing traditional aircraft designs. The TTBW aircraft described here represents a deviation from the existing data, and Aviary provides a way to replace any existing calculation or subsystem with a user-provided tool that is more accurate or even higher fidelity. For certain key disciplines, the internal calculations are replaced with the tools described in the subsequent sections; for the remainder, which is primarily the subsystem mass buildup apart from the wing, the FLOPS correlations in Aviary are used.

Aviary is written entirely in Python and built on top of the OpenMDAO framework [18]. This gives it access to the optimization capabilities of OpenMDAO, in particular, the support for efficient total derivative calculation across a large heterogeneous model for use in gradient-based optimization. Some of the efficiency comes from the use of graph-coloring algorithms to exploit model sparsity, so that independent linear systems can be simultaneously solved.[19]

To simulate the trajectory, Aviary uses Dymos, which is another tool built in OpenMDAO that can integrate Ordinary

^{*}<https://www.github.com/openmdao/aviary>

Differential Equations (ODE) using a variety of implicit and explicit methods [20]. While some methods (i.e, shooting methods) start at an initial point in time and integrate to a final time, other methods (collocation methods) solve for the entire time history of the integration variables, utilizing the optimizer to drive all of the defect residuals to zero. Aviary provides the most support for the latter, and it is used in this study. OpenMDAO’s gradient support and graph-based coloring enable the collocation problem to be solved efficiently.

Aviary includes two levels of fidelity for the aircraft equations of motion – energy (or height-energy equations) and the two-degree of freedom equations. We chose to use the slightly-lower fidelity energy method because we already had robustly-converging TTBW models with the simpler (drag table) aerodynamics [4]. The height-energy method uses a single differential equation that describes the aircraft motion using energy (potential and kinetic) as the time integration variable [21]. The trajectory to be simulated consists of three simple flight phases: climb, cruise, and descent, with boundary conditions that come from the TTBW’s design mission. Each is implemented as a Phase in Dymos, all of which use the energy equation in the ODE, which contains the equation that Dymos solves. Other disciplines such as aerodynamics and propulsion are also included in the phase to compute other quantities such as thrust or drag that contribute to the equations at the integration points. None of the disciplines used in this study are providing any additional differential equations at present, though the capability is supported.

B. OpenAeroStruct

OpenAeroStruct[†] is an open-source library for aerostructural analysis and optimization of flexible-winged aircraft [13]. The library is written using the OpenMDAO framework [18]. OpenAeroStruct features a Vortex Lattice Method (VLM) aerodynamic representation of the wing that can be optionally coupled to a 6 degree of freedom finite-element beam model of the wingbox structure. OpenAeroStruct features adjoints for both its aerodynamic and structural solvers which allows the models to be integrated in larger gradient-based MDO problems through OpenMDAO. For this work, we utilize only OpenAeroStruct’s aerodynamic model. This is due to the fact that it is not currently possible to model multiple connected lifting surfaces (i.e. the wing and strut) through the library’s structural model.

Minor modifications were made to the library to improve memory performance when calling multiple OpenAeroStruct instances on the same machine. These modifications removed the nominal Aerodynamic Influence Coefficient (AIC) matrix assembly routine required to solve the linear aerodynamic potential equation. This routine was instead replaced with a memory-efficient matrix-free solver that utilizes Scipy’s [22] Generalized Minimum Residual (GMRES) solver [23]. Additional derivatives necessary for these changes were computed using the JAX Automatic Differentiation library [24].

The VLM mesh used to model the TTBW aircraft in this work is shown in Fig 2. The model includes the aerodynamic influence of the wing, strut, and horizontal tail. Due to the symmetrical nature of the analyses considered in this work, only half the aircraft is modeled for computational efficiency. The mesh features 259 vortex lattice panels in total. Since VLM only captures inviscid drag, in addition to the drag generated by OpenAeroStruct we add a viscous skin friction drag computed using an empirical model based on aircraft wetted area [25],

$$C_{D_0} = C_{f_e} \frac{S_{\text{wet}}}{S_{\text{ref}}}, \quad (1)$$

where C_{f_e} is chosen to be 0.0030, a value historically representative of civil transport type aircraft.

C. pyGeo

pyGeo[‡] is an open-source library for handling geometric parameterization in MDAO [14]. The library has multiple options for geometry parameterization, including: free-form deformation volume [26], OpenVSP [6], and Engineering Sketch Pad [27]. In this work we utilize an OpenVSP model (seen in Fig. 1) to define the geometry of the aircraft. pyGeo is responsible for updating the spatial locations of the nodes on the VLM mesh as the OpenVSP geometry design variables (span, wing twist, etc.) are perturbed. Since OpenVSP does not currently have support geometric design sensitivities necessary for gradient-based optimization, pyGeo must finite-difference across the geometry model to estimate the sensitivities of the VLM nodal locations with respect to OpenVSP geometry parameters. pyGeo is connected to OpenAeroStruct’s aerodynamic solver using MPhys, a library that standardizes connecting multi-disciplinary physics solvers in OpenMDAO [28].

[†]<https://www.github.com/mdolab/openaerostruct>

[‡]<https://www.github.com/mdolab/pygeo>

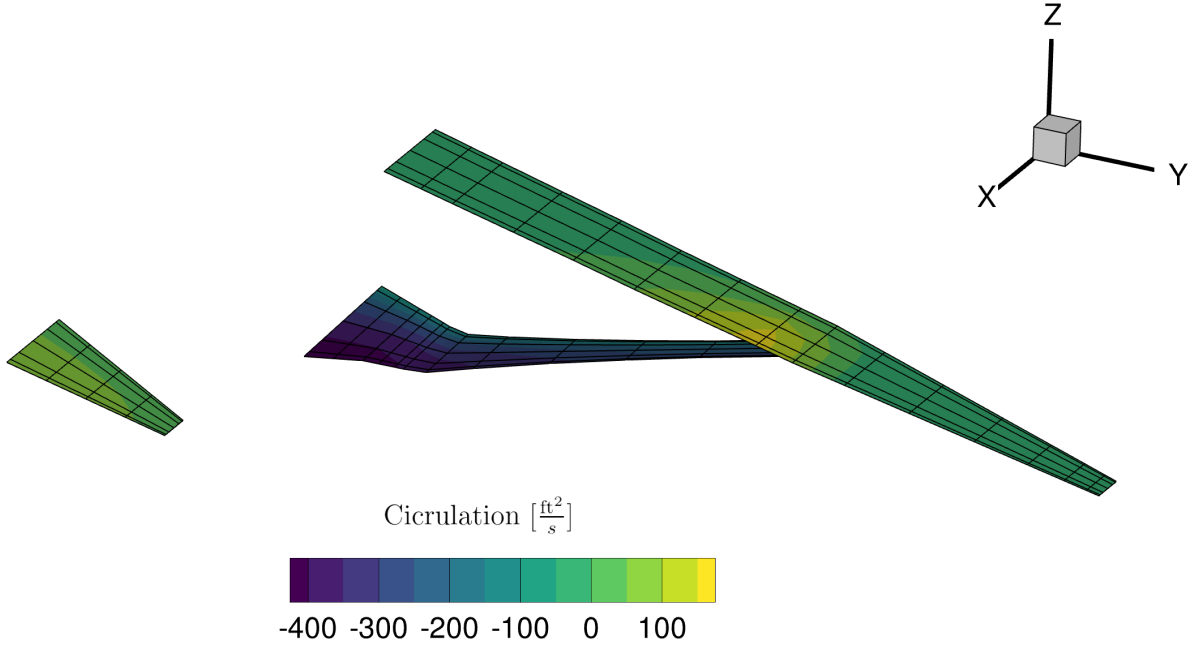


Fig. 2 TTBW half-mesh used in OpenAeroStruct aerodynamic VLM model.

D. TACS

TACS[§] is an open-source parallel finite-element analysis library for structural design and optimization [15]. The library is capable multiple structural analysis modes, including static, dynamic, and modal and buckling eigenvalue problems. Each of these capabilities includes a relevant adjoint implementation allowing for design sensitivities to be readily computed. The core of the code is written in C++ for speed and features a Python user interface for ease of use. We will again utilize the MPhys library to wrap this model into OpenMDAO and tie it with OpenAeroStruct's aerodynamic solver and pyGeo's geometry engine to facilitate aerostructural optimization in the future work.

The structural model that would be used for this work is shown in Fig 3. Similar to the VLM model, the FEM model is a half model of the vehicle. The model includes the main wingbox structure and main and jury strut. The main wingbox structure (skins, ribs, and spars) is modeled using shell elements, the main strut using a series of beam elements, and the jury strut a rod element. The joints of the at the intersection of each member are modeled as pinned using rigid body elements. The model contains 9,196 elements and 53,766 structural degrees of freedom in total. Design variables will be added for each colored panel on the wingbox as well as spanwise segments of the main strut, allowing each to be sized by the optimizer.

V. Tool Integration

An eXtended Design Structure Matrix (XDSDM) [29] is shown in Fig. 4 which illustrates how the optimization problem is connected across the various disciplines in our current work. We utilize SNOPT [30] as the top-level optimizer for this problem. SNOPT is an optimizer that has been designed to handle large sparse optimizations problems, as considered here, efficiently. The top level objective is to minimize the fuel weight by allowing the optimizer to size the aircraft while choosing an optimal trajectory and designing the wing twist. In the aircraft sizing problem, the design gross weight of the aircraft is declared as a design variable, and a constraint is added to balance the fuel computed in the pre-mission detailed mass subsystem against the fuel consumed in the mission and the computed or calculated reserve fuel.

The only subsystem that is outside of the optimization loop is the pre-mission propulsion analysis, which computes the engine mass, and sets a few configuration parameters. Since there are no engine design variables in this problem, it only needs to be run once before the optimizer iteration begins.

[§]<https://www.github.com/smdogroup/tacs>

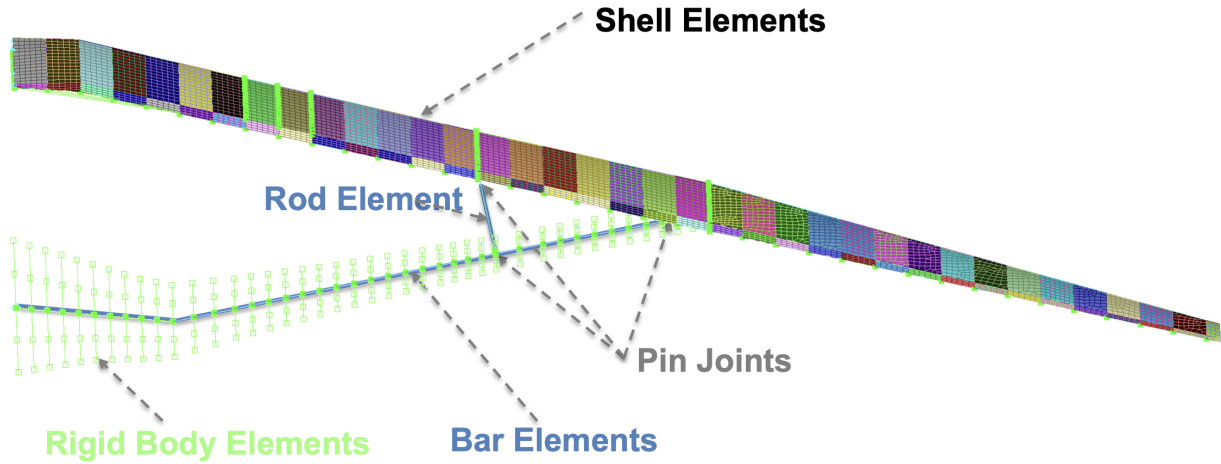


Fig. 3 TTBW half-mesh used in TACS structural FEM model.

The remaining pre-mission components include geometry and aircraft subsystem weights. OpenVSP provides the aircraft wing geometry including the effects of twist angle. A surrogate model developed for the TTBW concept is used to predict structural weight of the wing based on small geometric changes. This wing weight gets fed into the Aviary weight subsystem, which uses empirical-based relations to predict all of the remaining weights of the aircraft, culminating in the fuel weight as the final output.

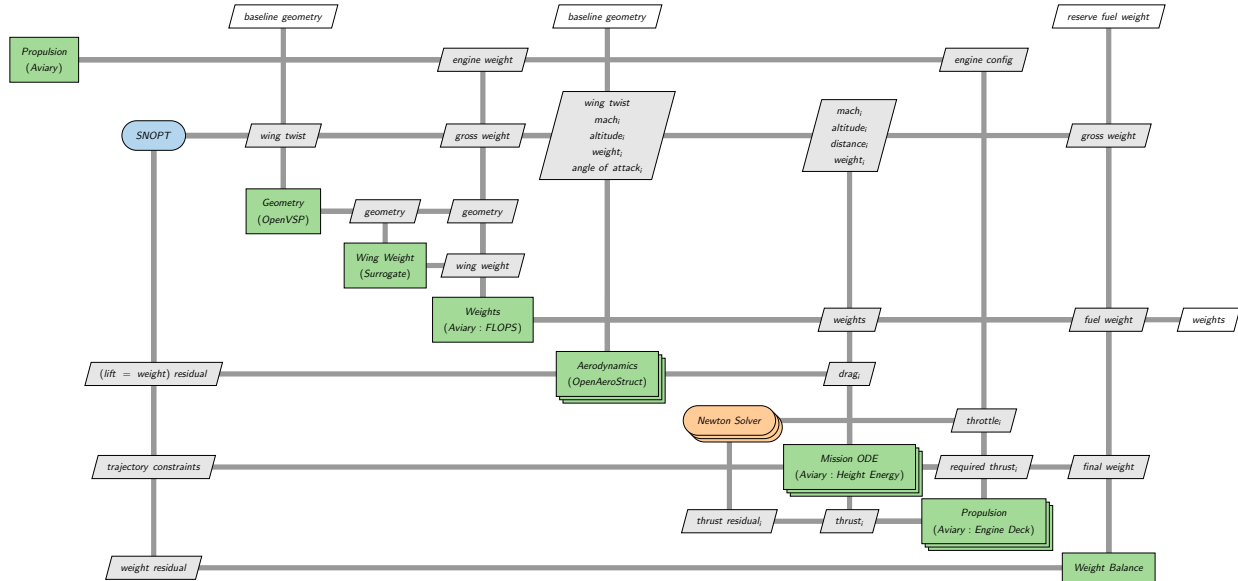
The mission ODE analysis consists of three Dymos phases: a climb, cruise, and descent. The climb and cruise phases are discretized using 3 third-order segments yielding 12 integration points, while the descent phase has 1 third-order segment, giving 4 points, for a total of 28 integration points. The distance and mass are states in the equations of motion, so there are constraints on the defects between the computed and approximated time derivatives, which the optimizer drives to zero. There are additional boundary and compatibility constraints between each pair of phases, and at the start and the end of the trajectory.

In order to couple OpenAeroStruct's aerodynamic solver to the mission analysis, we utilize a direct coupling approach [11]. In this approach, multiple OpenAeroStruct solver instances are connected to each mission phase, one for each discrete time integration point. Each solver instance is passed a weight state from its corresponding mission integration point that dictates the amount of lift the model must generate. In addition, the current geometric design variables can be passed to the an internal pyGeo component that then generates an updated mesh for analysis. OpenAeroStruct then computes the total drag at each point, which is then used to complete equations of motion for the mission analysis. Due to the fact that each OpenAeroStruct instance can be solved independently of each other, we parallelize the process by running the cases on multiple processors using an OpenMDAO *ParallelGroup*. This approach differs from the coupling strategy used by Aretskin-Hariton et al. [4], in which VSPAERO was coupled to the mission analysis using a precomputed drag table outside of the optimization loop. This allows the effects aerodynamic/geometric design within the optimization to be considered and gives access to a larger design space.

Finally, for the mission propulsion, an engine deck provides the values for thrust and fuelburn that are needed during integration of the mission equations of motion. Aviary includes the capability to read a static engine deck and create a metamodel that computes thrust and fuelburn for the altitudes, Mach numbers, and throttle settings that are needed at the evaluation points in the mission. Note that the throttle setting is computed by balancing thrust versus the required thrust computed from the energy method. A Newton solver is used to drive this balance to zero, rather than letting the optimizer control the throttle.

VI. Preliminary Results

The optimization problem considered in the current work is summarized in Table 2. The objective of the optimization is to minimize the aircraft fuel burn by simultaneously sizing the aircraft gross weight while optimizing the trajectory and designing the wing spanwise twist. The trajectory is split into 3 phases: climb, cruise, and descent. In each of



these phases, the altitude, distance, and aircraft mass are design variables that are chosen by the optimizer in a way that satisfies the collocation defects for the mass and distance states, the intra-phase linkage constraints, and any initial and final constraints. In addition, the climb and descent phases include path constraints that prevent the trajectory from descending during climb or climbing during descent; this is done to prevent any wiggles that might occur in the trajectory profile. To close the aircraft, a final weight residual assures that the fuel weight consumed in the mission plus the reserve fuel weight balances the total take-off weight. The aerodynamics subsystem introduces an angle of attack design variable for each OpenAeroStruct model instance, because angle of attack is not computed in the ODE. The optimizer chooses an angle of attack that constrain the lift output from each OpenAeroStruct model to match the target lift for its associated mission point. The spanwise twist of the wing is included as a design variable, with 9 in total.

In order to provide a baseline for comparison, we run two versions of the optimization problem: one where all optimization design variables described in Table 2 (twist-optimized) and another in which we drop the wing twist design variables from the problem (baseline). The twist-optimized case takes roughly 35 hours of wall time for optimization, while the baseline case takes only 4 hours. The large discrepancy in run time between the two cases is largely due to the finite-differencing over the OpenVSP model required for the twist-optimized design. This is due to the fact that updating the model in between finite difference perturbations is significantly expensive. In an attempt to speed up the process, we are looking into reducing the number of geometry objects included in the OpenVSP model to only those strictly necessary for the aerodynamic analysis. The number of processors used in both runs are currently limited to 3 max due to a bug in one of the underlying libraries that we are also investigating at time of writing.

The altitude and Mach flight profile over time is shown in Fig. 5. We see only minor differences in the optimal mission for both designs, mainly in the duration of the climb and descent phases. This result is likely a sign that wing twist does not play a large role in the optimal aircraft trajectory. However, we believe that by including geometric design variables that effect the wing planform (span, sweep, area, etc.) in future work, more pronounced differences in optimal trajectory are likely to be revealed. Fig 6 shows the normalized lift distribution across the aerodynamic model (wing, strut, and tail) at the start of cruise for both designs. The twist-optimized design achieves a more elliptical lift distribution to reduce induced drag and reduce fuel burn. Fig 7 shows the twist-optimized wing geometry. We see that it is mostly smooth with exception of small kink at wing root due to twist not being fixed there by design. A final fuel burn savings of 8% (19,900 vs 21,500 lb) relative to unoptimized baseline geometry.

VII. Future Work

For the final paper, we plan to add several improvements relative to what is presented today. The first aspect that we plan to add to our modeling for the final paper is a trim constraint for OpenAeroStruct. This can be implemented in a

	Variable/function	Description	#
Minimize	W_{fuel}	Mission fuel weight	
with respect to	W_{gross}	Aircraft gross weight	1
	$t_0^{\text{climb}}, t_0^{\text{cruise}}, t_0^{\text{descent}}$	Climb, cruise, descent phase initial time	3
	$\Delta t^{\text{climb}}, \Delta t^{\text{cruise}}, \Delta t^{\text{descent}}$	Climb, cruise, descent phase duration	3
	$W_j^{\text{climb}}, W_j^{\text{cruise}}, W_j^{\text{descent}}$	Climb, cruise, descent weight state	23
	$d_j^{\text{climb}}, d_j^{\text{cruise}}, d_j^{\text{descent}}$	Climb, cruise, descent distance state	23
	$h_j^{\text{climb}}, h_j^{\text{cruise}}, h_j^{\text{descent}}$	Climb, cruise, descent altitude polynomial controls	12
	$\alpha_j^{\text{climb}}, \alpha_j^{\text{cruise}}, \alpha_j^{\text{descent}}$	Climb, cruise, descent angle of attack state	28
	θ_i^{wing}	Wing twist	9
	Total design variables		102
Subject to	$L_j^{\text{climb}} - W_j^{\text{climb}} = 0.0$	Climb lift equals weight	12
	$L_j^{\text{cruise}} - W_j^{\text{cruise}} = 0.0$	Cruise lift equals weight	12
	$L_j^{\text{descent}} - W_j^{\text{descent}} = 0.0$	Descent lift equals weight	4
	$0.0 < \text{throttle}^{\text{traj}} < 1.0$	Throttle constraints	17
	$\frac{dh_j}{dt}^{\text{climb}} > 0.0$	No descent during climb	12
	$\frac{dh_j}{dt}^{\text{descent}} < 0.0$	No climb during descent	4
	$\frac{dW_j}{dt}^{\text{traj}} - \frac{dW_j}{dt}^{\text{approx}} = 0.0$	Weight collocation defects	21
	$\frac{dd_j}{dt}^{\text{traj}} - \frac{dd_j}{dt}^{\text{approx}} = 0.0$	Distance collocation defects	21
	$g_{\text{boundary}}^{\text{traj}} = 0.0$	Phase linkage constraints	9
	$R_{\text{weight}} = 0.0$	Weight residual	1
	$R_{\text{range}} = 0.0$	Range residual	1
	Total constraints		114

Table 2 Optimization problem formulations

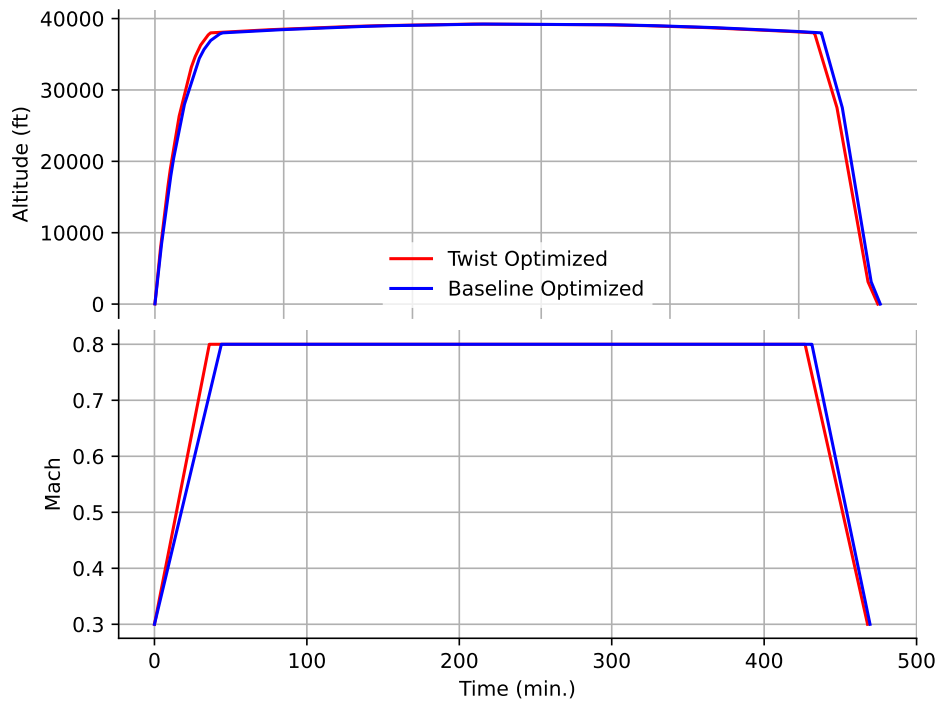


Fig. 5 Comparison of optimized altitude/mach mission profiles for baseline and twist-optimized designs.

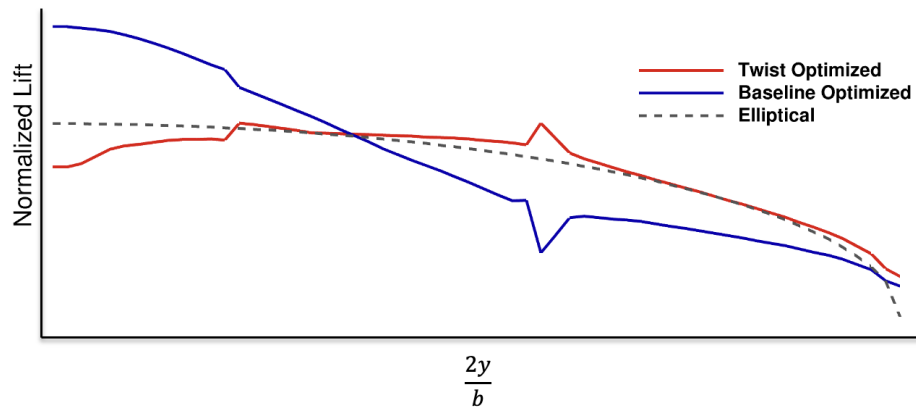


Fig. 6 Comparison of normalized lift distribution between baseline and twist-optimized designs.

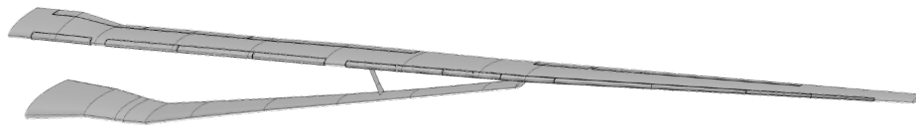


Fig. 7 Twist-optimized wing geometry.

similar fashion to the lift constraints that we already have imposed by constraining pitching moment about the center of gravity to be zero ($M_y = 0.0$). Furthermore, this would capture effect of additional drag due to trim. Additionally, we plan to add TACS to our design framework which will enable joint aerostructural optimization. The higher fidelity structural model that TACS will provide to the work will be key as it will allow us to accurately capture weight tradeoffs when expanding the geometric design space by including planform design variables (span, wing area, sweep, etc.) in addition to the wing twist variables in this work. This could result in a larger system level impact and change how the vehicle flies the mission.

Acknowledgments

This work was completed with support from the Systems Analysis and Integration sub-project under the Advanced Air Transport Technology (AATT) project in the Advanced Air Vehicles Program.

References

- [1] Wahls, R. A., “NASA Sustainable Flight National Partnership Overview,” *8th International Workshop on Aviation and Climate Change*, 2023. URL https://ntrs.nasa.gov/api/citations/20230007095/downloads/UTIAS8th-ACC-SFNPoverview-2023may31_final-Wahls.pdf.
- [2] Heidmann, J. D., “An Overview of NASA Sustainable Aviation,” *8th International Aerospace Innovation Forum*, 2022. URL https://ntrs.nasa.gov/api/citations/20220011765/downloads/Presentation_An%20Overview%20of%20NASA%20Sustainable%20Aviation_Heidmann.pdf.
- [3] Harrison, N. A., Hoffman, K., Lazzara, D. S., Reichenbach, E. Y., Sclafani, A. J., and Droney, C. K., “Subsonic Ultra Green Aircraft Research: Phase IV Final Report–Volume I Mach 0.80 Transonic Truss-Braced Wing High-Speed Design Report,” 2023.
- [4] Aretskin-Hariton, E., Gratz, J., Kirk, J., Lyons, K., Jasa, J., Moore, K., Falck, R., Caldwell, D., Kuhnle, C., Recine, C., Hendricks, E., and Olson, E., *Multidisciplinary Optimization of a Transonic Truss-Braced Wing Aircraft using the Aviary Framework*, 2024. <https://doi.org/10.2514/6.2024-1084>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2024-1084>.
- [5] Hendricks, E. S., and Gray, J. S., “pyCycle: A Tool for Efficient Optimization of Gas Turbine Engine Cycles,” *Aerospace*, Vol. 6, No. 87, 2019. <https://doi.org/10.3390/aerospace6080087>.
- [6] McDonald, R. A., and GlouDEMANS, J. R., *Open Vehicle Sketch Pad: An Open Source Parametric Geometry and Analysis Tool for Conceptual Aircraft Design*, 2022. <https://doi.org/10.2514/6.2022-0004>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2022-0004>.
- [7] Leader, M. K., Aretskin-Hariton, E., and Moore, K., *Multidisciplinary Optimization of a Transonic Truss Braced Wing Aircraft With Hybrid-Electric Propulsion*, 2024. <https://doi.org/10.2514/6.2024-4294>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2024-4294>.
- [8] Fazal, B., Schmidt, J., Phillips, B. D., Ordaz, I., and Moore, K., *Integration of Uncertainty Quantification in a Model-Based Systems Analysis and Engineering Framework*, 2024. <https://doi.org/10.2514/6.2024-4559>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2024-4559>.
- [9] Hwang, J. T., Jasa, J. P., and Martins, J. R. R. A., “High-Fidelity Design-Allocation Optimization of a Commercial Aircraft Maximizing Airline Profit,” *Journal of Aircraft*, Vol. 56, No. 3, 2019, pp. 1164–1178. <https://doi.org/10.2514/1.C035082>, URL <https://doi.org/10.2514/1.C035082>.
- [10] Vassberg, J., Dehaan, M., Rivers, M., and Wahls, R., *Development of a Common Research Model for Applied CFD Validation Studies*, 2008. <https://doi.org/10.2514/6.2008-6919>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2008-6919>.
- [11] Kaneko, S., and Martins, J. R., *MDO Formulations for Simultaneous Design and Trajectory Optimization*, 2024. <https://doi.org/10.2514/6.2024-2231>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2024-2231>.
- [12] Gratz, J., Kirk, J., Recine, C., Jasa, J., Aretskin-Hariton, E., Moore, K., and Marfatia, K., *Aviary: An Open-Source Multidisciplinary Design, Analysis, and Optimization Tool for Modeling Aircraft With Analytic Gradients*, 2024. <https://doi.org/10.2514/6.2024-4219>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2024-4219>.
- [13] Jasa, J. P., Hwang, J. T., and Martins, J. R. R. A., “Open-source coupled aerostructural optimization using Python,” *Structural and Multidisciplinary Optimization*, Vol. 57, No. 4, 2018, pp. 1815–1827. <https://doi.org/10.1007/s00158-018-1912-8>.

- [14] Hajdik, H. M., Yildirim, A., Wu, E., Brelje, B. J., Seraj, S., Mangano, M., Anibal, J. L., Jonsson, E., Adler, E. J., Mader, C. A., Kenway, G. K. W., and Martins, J. R. R. A., “pyGeo: A geometry package for multidisciplinary design optimization,” *Journal of Open Source Software*, Vol. 8, No. 87, 2023, p. 5319. <https://doi.org/10.21105/joss.05319>.
- [15] Kennedy, G. J., and Martins, J. R., “A parallel finite-element framework for large-scale gradient-based design optimization of high-performance structures,” *Finite Elements in Analysis and Design*, Vol. 87, 2014, pp. 56–73. <https://doi.org/https://doi.org/10.1016/j.finel.2014.04.011>, URL <https://www.sciencedirect.com/science/article/pii/S0168874X14000730>.
- [16] Wells, D. P., Horvath, B. L., and Mccullers, L. A., “The Flight Optimization System Weights Estimation Method,” 2017. URL <https://ntrs.nasa.gov/api/citations/20170005851/downloads/20170005851.pdf>.
- [17] Hague, D., “GASP - General Aviation Synthesis Program,” Vol. 5, 1978. URL <https://ntrs.nasa.gov/api/citations/>.
- [18] 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. <https://doi.org/10.1007/s00158-019-02211-z>.
- [19] Gray, J. S., Hearn, T. A., and Naylor, B. A., *Using Graph Coloring To Compute Total Derivatives More Efficiently In OpenMDAO*, 2019. <https://doi.org/10.2514/6.2019-3108>.
- [20] Falck, R., Gray, J. S., Ponnappalli, K., and Wright, T., “dymos: A Python package for optimal control of multidisciplinary systems,” *Journal of Open Source Software*, Vol. 6, No. 59, 2021, p. 2809. <https://doi.org/10.21105/joss.02809>, URL <https://doi.org/10.21105/joss.02809>.
- [21] Capristan, F. M., and Welstead, J. R., “An Energy-Based Low-Order Approach for Mission Analysis of Air Vehicles in LEAPS,” 2018. <https://doi.org/10.2514/6.2018-1755>.
- [22] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nature Methods*, Vol. 17, 2020, pp. 261–272. <https://doi.org/10.1038/s41592-019-0686-2>.
- [23] Saad, Y., and Schultz, M. H., “GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems,” *SIAM Journal on Scientific and Statistical Computing*, Vol. 7, No. 3, 1986, pp. 856–869. <https://doi.org/10.1137/0907058>, URL <https://doi.org/10.1137/0907058>.
- [24] Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q., “JAX: composable transformations of Python+NumPy programs,” 2018. URL <http://github.com/google/jax>.
- [25] Raymer, D. P., *Aircraft design: A conceptual approach*, 4th ed., AIAA education series, American Institute of Aeronautics and Astronautics, Reston, Va., 2006. URL <http://www.loc.gov/catdir/toc/ecip068/2006004706.html>.
- [26] Sederberg, T. W., and Parry, S. R., “Free-form deformation of solid geometric models,” *SIGGRAPH Comput. Graph.*, Vol. 20, No. 4, 1986, p. 151–160. <https://doi.org/10.1145/15886.15903>, URL <https://doi.org/10.1145/15886.15903>.
- [27] Haimes, R., and Dannenhoffer, J., *The Engineering Sketch Pad: A Solid-Modeling, Feature-Based, Web-Enabled System for Building Parametric Geometry*, 2013. <https://doi.org/10.2514/6.2013-3073>, URL <https://arc.aiaa.org/doi/abs/10.2514/6.2013-3073>.
- [28] Yildirim, A., Jacobson, K. E., Anibal, J. L., Stanford, B. K., Gray, J. S., Mader, C. A., Martins, J. R. R. A., and Kennedy, G. J., “MPhys: A Modular Multiphysics Library for Coupled Simulation and Adjoint Derivative Computation,” *Structural and Multidisciplinary Optimization*, 2024. (In press).
- [29] 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, 2012, pp. 273–284. <https://doi.org/10.1007/s00158-012-0763-y>.
- [30] Gill, P. E., Murray, W., and Saunders, M. A., “SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization,” *SIAM Rev.*, Vol. 47, No. 1, 2005, p. 99–131. <https://doi.org/10.1137/S0036144504446096>, URL <https://doi.org/10.1137/S0036144504446096>.