Parallel Aircraft Trajectory Optimization with Analytic Derivatives

Robert D. Falck* and Justin S. Gray†
NASA Glenn Research Center, Cleveland, OH, 44135, U.S.A.
and Bret Naylor‡
DB Consulting Group, Inc., Cleveland, OH, 44135, U.S.A.

I. Nomenclature

α	angle of attack
C_D	drag coefficient
C_L	lift coefficient
C_T	thrust coefficient
d	static control variable
D	drag
Δ	collocation defect
η	elevator deflection angle
g	gravitational acceleration
γ	flight path angle
γ_a	atmospheric ratio of specific heats
h	altitude
L	lift
LGL	Legendre-Gauss-Lobatto
M	Mach number
P	atmospheric pressure
q	dynamic pressure
r	range
R	atmospheric gas constant
ρ	atmospheric density
RHS	right hand side
S	reference aerodynamic area
SFC	thrust specific fuel consumption
t	time
τ	throttle setting
$Temp$	atmospheric temperature
T	thrust
τ	throttle parameter
u	dynamic control variable
v	velocity
W_f	fuel weight
W_{total}	total aircraft weight
x	state variable
$XDSM$	extended design structure matrix

*Aerospace Engineer, Mission Analysis and Architecture Branch, rfalck@nasa.gov, AIAA Member.
†Aerospace Engineer, Propulsion System Analysis Branch, justin.s.gray@nasa.gov, AIAA Member.
‡Senior Software Engineer, Propulsion System Analysis Branch, bret.a.naylor@nasa.gov.

II. Abstract

Trajectory optimization is an integral component for the design of aerospace vehicles, but emerging aircraft technologies have introduced new demands on trajectory analysis that current tools are not well suited to address. Designing aircraft with technologies such as hybrid electric propulsion and morphing wings requires consideration of the operational behavior as well as the physical design characteristics of the aircraft. The addition of operational variables can dramatically increase the number of design variables which motivates the use of gradient based optimization with analytic derivatives to solve the larger optimization problems. In this work we develop an aircraft trajectory analysis tool using a Legendre-Gauss-Lobatto based collocation scheme, providing analytic derivatives via the OpenMDAO multidisciplinary optimization framework. This collocation method uses an implicit time integration scheme that provides a high degree of sparsity and thus several potential options for parallelization. The performance of the new implementation was investigated via a series of single and multi-trajectory optimizations using a combination of parallel computing and constraint aggregation. The computational performance results show that in order to take full advantage of the sparsity in the problem it is vital to parallelize both the non-linear analysis evaluations and the derivative computations themselves. The constraint aggregation results showed a significant numerical challenge due to difficulty in achieving tight convergence tolerances. Overall, the results demonstrate the value of applying analytic derivatives to trajectory optimization problems and lay the foundation for future application of this collocation based method to the design of aircraft with where operational scheduling of technologies is key to achieving good performance.

III. Introduction

Trajectory analysis performs a key role in the design of aircraft by modeling performance over the course of a mission. Often, for aircraft, fuel burn or range are the primary quantities of interest, but trajectory analysis results are also important for aircraft noise prediction, propulsion sizing, and operational performance prediction. Trajectory analysis tools model aircraft performance over time via numerical integration of governing equations of motion. There are many implementations of mission analysis, using different formulations and methods. Flight Optimization System (FLOPS)[1] is a NASA developed code built with the well developed energy state approximation[2] that simplifies trajectory analysis to an exchange between potential and kinetic energy. The simplification allows for an efficient implementation, but effectively uses a 2 degree of freedom model that lacks information about angle of attack, roll, and yaw. Two newer tools, SUAVE[3] and PyMission,[4] abandon the energy state approximation in favor of a collocation based approach that does account for aircraft angle of attack. The OTIS[5] tool also uses collocation based approach, but combined with an implicit time integration scheme that offers potential computational savings. OTIS has more commonly been applied to spacecraft trajectory optimization and optimal control, but its approach is equally applicable to aircraft trajectory optimization.[6]

The advent of fuel saving operational concepts like continuous decent trajectories[7] and formation flying,[8,9] have made trajectory analysis a more central part of aircraft design. Similarly, new aircraft technologies such as active load alleviation,[10] hybrid electric propulsion,[11,12] and morphing wings[13,14] couple the physical design of the aircraft with the time-varying characteristics of its operation. While trajectory analysis has always been an important analysis in the aircraft design process, these new operational concepts and technologies have elevated its importance and motivates a much tighter integration of mission analysis with aerodynamics, weights, and propulsion models into large multidisciplinary models.

The development of large multidisciplinary aircraft models, including trajectory analysis, leads to complex problems with large numbers of disciplines and many design variables. These models can have 100's or 1000's of design variables because of the need to specify schedules with reasonable resolution across long time scales. This naturally encourages the application of multidisciplinary design optimization (MDO) methods. When the numbers of design variables grow large, gradient based optimization with analytic derivatives is the most effective optimization technique. However, older tools such as FLOPS and OTIS are not well suited to this approach because neither provides analytic derivatives and both present implementation challenges with integration into a larger multi-disciplinary model. Addressing these implementation challenges was one of the major motivations behind SUAVE, which has demonstrated tight propulsion design coupling[3] and a modular design that can easily handle a broad range of aircraft types and missions.[15] PyMission was designed to provide adjoint analytic derivatives and to integrate tightly with high fideility analyses.[16] The SUAVE and

PyMission work clearly demonstrated the value of integrating trajectory analysis into a multidisciplinary design optimization to integrate mission performance as part of objective function. The PyMission work also shows how adjoint analytic derivatives, applied to aircraft mission analysis, enable design optimization on a large number of design variables. The capability to handle larger design spaces is a key feature that enables the method to efficiently handle problems that include time-varying control schedules.

As stated earlier, both SUAVE and PyMission implement a collocation method with an explicit time integration scheme. This paper discusses the development of a collocation method that employs an implicit, Legendre-Gauss-Lobatto (LGL) collocation scheme, with adjoint derivatives, built within the OpenMDAO framework.[17] Prior work has shown a significant computational advantage for the implicit approach,[6] which motivates this investigation. The intent of this work is to test the applicability of the implicit LGL collocation scheme to aircraft mission analysis within a multidisciplinary design optimization context. We explore use of various techniques to reduce the time required to find optimal aircraft trajectories, including solving derivatives analytically in both forward and adjoint modes, parallelization of derivative computation, and parallelization of multiple trajectories solved simultaneously.

IV. Approach

This work consists of two primary efforts. First we develop a modular, general optimal control tool within the OpenMDAO framework. The tool is general in that it is left to the user provide the "right-hand-side" which provide both the equations of motion for the dynamics involved, as well as any auxiliary calculations which may be useful as outputs, nonlinear constraints, or objective variables. The second effort is the development of the equations-of-motion for aircraft trajectory optimization, based on the approach originally proposed by Kao et. al,[4] which provide a good balance of efficiency and fidelity.

A. Overview of the LGL Collocation Scheme

The optimal control techniques employed in this work were heavily influenced by those in OTIS.[5] OTIS allows different optimal control techniques to be employed depending on the problem at hand, including collocation and multiple shooting techniques. In a similar fashion, this effort focused on a tool that would be expandable to a variety of techniques in the future, although we concentrated on a collocation-based technique for the problem at hand. Collocation techniques are computationally efficient in that they require relatively few evaluations of the equations of motion compared to shooting methods. The biggest drawback of collocation is that it is common to need to know the order of events which cause discontinuities in the state variables w.r.t. time *a priori*. Events which cause discontinuities, such as mass jettisons or engine startup/shutdown, are not a factor in the commercial aircraft trajectory analyzed here.

Implicit collocation and pseudospectral methods come in a variety of forms.[18] Paris and Hargraves employed a Hermite-Simpson based approach in the original version of OTIS.[6] Solutions using a Legendre-Gauss-Lobatto based technique have been developed by Herman and Conway[19] and Fahroo and Ross,[20] and were incorporated into later version of OTIS. Approaches based on the Legendre-Gauss-Radau quadrature, such as the LGL scheme, have become increasingly popular and are used on a variety of applications.[21]

In the LGL scheme, dynamics are modeled by an interpolating piecewise polynomial describing the state variable as a function of time. Each physical state variable is associated with its own piecewise polynomial. A notional cartoon of a collocating polynomial is shown in Figure 1. The *cardinal nodes*, red circles, of the polynomial are varied such that the shape of the polynomial matches the physics prescribed a given set of equations of motion. The errors between the polynomial and the equations of motion, called *defects*, are measured at the *interior nodes*, blue squares, by comparing the slope of the polynomial with the predicted time derivative from the equations of motion at that point.

For a problem with multiple states, there is one piecewise polynomial and associated set of defects measured at the interior nodes for each state. The collection of all the equations of motion for all the states is referred to as the *right-hand-side* of the problem. The *right-hand-side* relies on several parameters, including the current value of the state variables (x) and time (t). In addition, other parameters may be necessary. These are generally divided into two groups; dynamic controls (u) whose values change in time, and static controls (d) whose values are fixed throughout the phase. Practically that means that static controls have a single scalar value used throughout a phase and the dynamic controls are array variables with the same number of values as the number of cardinal+interior nodes in the phase.

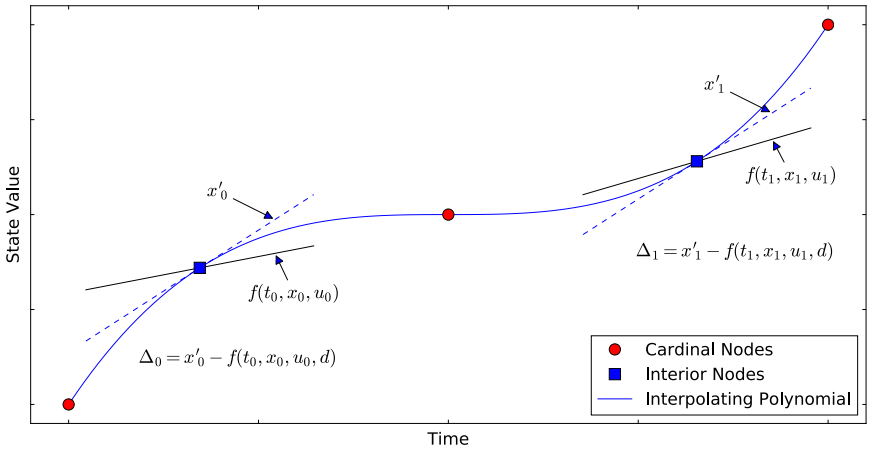

Figure 1. A representation of a collocation segment with three cardinal nodes and two interior nodes. Collocation defects are the difference between the slope of the interpolating polynomial ($\dot{x}$) and the time-derivative given by the equations of motion (f). The equations of motion are a function of time, states, dynamic controls, and static controls.

For any given problem, the trajectory is broken down one or more *phases*. Within a phase, the equations of motion are held constant, but they can change between phases. Using the same equations of motion within two adjacent phases allows for discontinuous changes such as stage separation for a launch vehicle. Each phase has one or more *segments*. Each segment is consists of a set of nodes on which polynomials representing the state and control time-histories is based. It is often beneficial to vary the length of different segments to better refine the model in areas of high dynamics. Similarly, each segment can be modeled as a polynomial of a different order. The spacing and order of the various segments define the computational *grid* for the collocation problem. Grid refinement methods can be used automatically adjust the duration of segments to provide the highest accuracy at the lowest overall cost.[22–24]

The collocation technique transforms an optimal control problem into a nonlinear programming problem. Various software packages exists which can solve this nonlinear programming problem. In collocation techniques, nonlinear optimizers are used to minimize some objective function (e.g. minimize fuel-burn for an aircraft trajectory) by varying the initial and final times of the phases, the values of the static controls, the values of the state variables at the cardinal nodes, and the values of the dynamic controls at the cardinal and interior nodes. The defects are posed as equality constraints which must be equal to zero for the solution to be feasible. Treating the state variables as design variables and the defect constraints as equality constraints results in the Simultaneous Analysis and Design (SAND) architecture,[25,26] which allows the whole problem to be solved as a single NLP problem. In this work, the SNOPT optimizer[27] was employed to solve the NLP problem, which uses is an SQP based method.

In addition to constraining the defect values, the user can impose additional nonlinear constraints on the problem. These come in two forms; boundary constraints and path constraints. Boundary constraints can be imposed on the initial or final value of a variable in a phase. Path constraints are generally evaluated at each node, both cardinal and interior, within the phase. For a phase with n_c cardinal nodes and n_i interior nodes, a single path constraint therefore adds $n_c + n_i$ constraints to the problem. In later sections we explore the use of constraint aggregation techniques to reduce the number of NLP constraints due to a path constraint to two.

B. Optimal Control within OpenMDAO

OpenMDAO provides two key constructs to build models with. *Components* represent the smallest block of computational work, where the actual equations defining parts of the model are implemented. Sets of components are then assembled with a *Group*, which acts as a container and manages the data passing between them. Groups can contain both individual components, as well as other groups which allows for the construction of a model hierarchy.

To implement an LGL collocation scheme within the OpenMDAO framework, each *phase* is defined as a

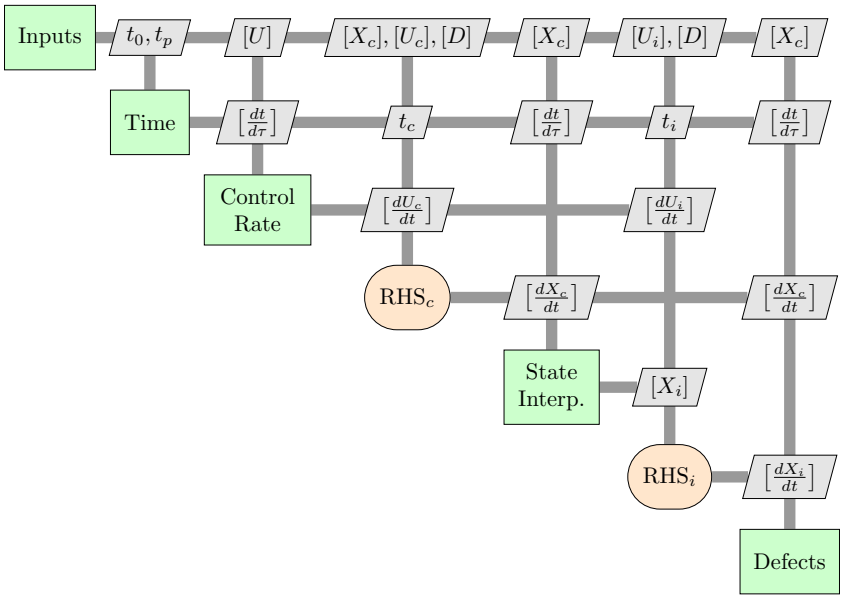

Figure 2. The XDSM for a single phase of an LGL collocation problem as implemented in OpenMDAO.

group, showin in Figure 2. Within the phase there are a number of components, represented as the rectangles, that handle the mathematics of evaluating the interpolating polynomials for the segments, and computing state values and their the derivatives at the interior nodes. To compute the *defects*, the *right-hand-side* must also be evaluated twice, once at the cardinal nodes (RHS_c) and once at the interior nodes (RHS_i). The *right-hand-side* is implemented as a its own sub-group within the the phase. A single RHS Group class is defined by the user, and then two instances of it are created, one for the cardinal nodes and one for the interior nodes. In Figure 2, the RHS_c and RHS_i groups are represented by the ovals. The RHS Group class is the responsibility of the user to define, in order to implement the specific equations of motion for their problem. The assembly of the phase group, for performing the analysis is handled automatically by a provided implementation in the tool.

C. Equations of Motion for Commercial Aircraft

The equations of motion governing aircraft dyamics must be defined, so they can be computed in the RHS_c and RHS_i blocks in Figure 2. Various formulations for the equations of motion have been proposed in previous studies on the optimization of commercial aircraft trajectories. Betts and Cramer applied a direct collocation technique to the problem of a planar aircraft trajectory by collocating five states of a planar aircraft trajectory (h, r, v γ, W_{total}) and using C_L and T as control variables.[28] Lukaczyk, Wendorff, et. al. used a Chebyshev pseudospectral technique to optimize trajectories with assumptions on distinct phases of flight (climb, level cruise, descent, etc.)[3] Kao, Hwang, Martins, Gray, and Moore developed a hybrid approach based on the collocation of cubic B-splines using range as the independent variable.[4] In that approach, altitude and Mach number are discretized at various B-spline nodes with respect to range, and the differential equation governing aircraft weight is integrated w.r.t. time.

The approach in this work builds on the work of Kao et. al. For a given aircraft empty weight and a specified range, we seek to minimize the aircraft initial takeoff weight (and thus fuel burn, W_f) by finding the optimal flight profile between two points on a flat, two-dimensional Earth. Figure 3 shows simple free-body-diagram for a commercial aircraft flying in a planar (2D) flight path. The equations of motion for range and fuel weight are:

$$\dot{r} = v \cos\gamma \tag{1}$$

$$\dot{W}_f = -\text{SFC}\, q\, S\, C_T \tag{2}$$

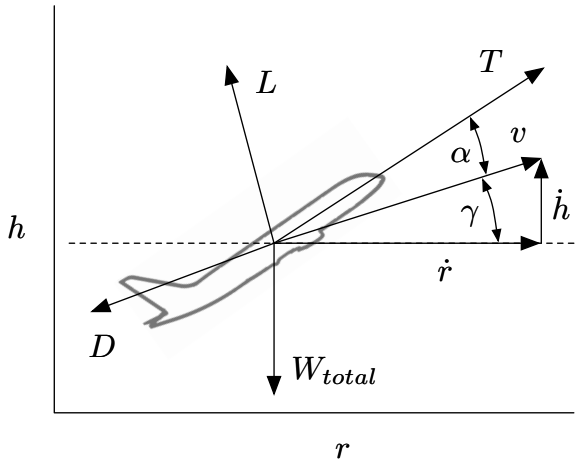

Figure 3. Free-body diagram for a commercial aircraft on a planar flight path.

The thrust specific fuel consumption (SFC) is based on a sea-level value scaled based on altitude.

$$SFC = SFC_{sl} - 1.471 \times 10^{-6} \frac{N}{km\, Ns} h \tag{3}$$

The altitude time-history and Mach number (M) time-history are treated as dynamic controls in this approach. The differentiation matrix associated with the collocating LGL polynomials is used to extract the altitude rate of change, $\dot{h}$, from the altitude time history.

$$\dot{h} = \frac{2[D_u] \cdot h}{t_{seg}} \tag{4}$$

where $[D_u]$ is the LGL differentiation matrix associated with the collocating LGL polynomial and t_{seg} is the duration of a given segment.

Atmospheric properties are interpolated from data in the same way as done in Reference 4. Once atmospheric pressure (P), density (ρ), and temperature ($Temp$) are known, velocity can be computed as:

$$v = M\sqrt{\gamma_a\, R\, Temp} \tag{5}$$

where γ_a, R, and $Temp$ are the ratio of specific heats, gas constant, and temperature for the atmosphere, respectively. The flight path angle then computed as:

$$\gamma = \arcsin \frac{\dot{h}}{v} \tag{6}$$

Aerodynamic properties of the vehicle are interpolated from quad-variate data tables:

$$C_L, C_D, C_M = f(M, h, \alpha, \eta) \tag{7}$$

where α is the angle of attack and η is the elevator deflection angle of the aircraft. Aerodynamic data used is based on the NASA Common Research Model (CRM).[29] To enforce steady flight and trim the aircraft, α and η are solved via OpenMDAO's Newton solver, such that the residuals in equations (10) and (11) are converged.

$$C_T = \frac{W_{total} \sin\gamma}{\cos\alpha\, q\, S} + \frac{C_D}{\cos\alpha} \tag{8}$$

$$\tilde{C}_L = \frac{W_{total} \cos\gamma}{q\, S} - C_T \sin\alpha \tag{9}$$

$$R(\alpha) = C_L - \tilde{C}_L = 0 \tag{10}$$

$$R(\eta) = C_M = 0 \tag{11}$$

Coefficient of thrust, C_T, is taken as an independent parameter to provide decent numerical scaling for equations (8) through (11). The actual thrust is then:

$$T = C_T\, q\, S \tag{12}$$

We define a throttle value, τ, as the current thrust divided by the maximum available thrust, which itself is a function of the current Mach and altitude, as well as sea-level values pressure (P_{sl}), and maximum thrust (T_{max_sl}). To ensure that the optimizer keeps thrust within physically valid limits τ is constrained to be between 0.01 and 1 throughout the mission. However, it should be noted that this formulation only requires τ to be within the bounds at the final converged solution and that during convergence it could exceed the physically valid range.

$$T_{max} = T_{max\_sl} \left(\frac{P}{P_{sl}} \right) \tag{13}$$

$$\tau = \frac{T}{T_{max}} \tag{14}$$

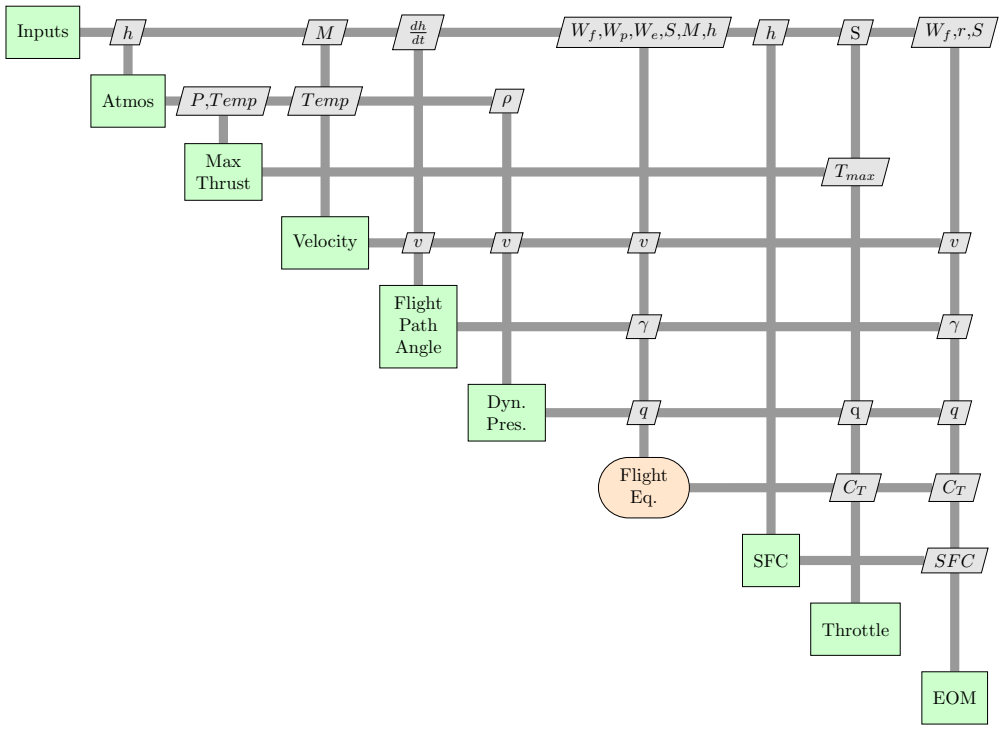

Figure 4. The extended design structure matrix the right-hand-side of the commercial aircraft trajectory optimization problem.[31]

The extended design structure matrix for the *right-hand-side* equations is shown in Figure 4. The Flight Equilibrium group in this figure represents the implicit equations (10) and (11), which are solved with a Newton solver. The presence of these implicit equations is one of the major motivations for using the OpenMDAO framework, which provides automatic coupled derivatives that allow the analytic derivatives for the optimizer to be computed very efficiently.[30]

V. Cases and results

Table 1 summarizes the problem formulation for the planar commercial aircraft trajectory optimization problem with standard path constraints. Note in this formulation the initial values of range, altitude, and fuel weight, as well as the final values of range and altitude, are bound-constrained variables.

	Variable/function	Description	Lower	Upper	Dimension
minimize	$W_f[0]$	initial fuel weight			
parameters	t_0	initial time	0	0	1
	t_p	flight duration	100	7200	1
	$r_c[1:-1]$	range at cardinal nodes	0	∞	11
	$W_f[:-1]$	fuel load at cardinal nodes	0	10^5	12
	$h[1:-1]$	altitude at nodes	0	20	23
					48
subject to	$\dot{h}_{(i,-1)} - \dot{h}_{(i+1,0)}$	control rate segment$_i$ continuity	0	0	11
	$\Delta r_i[:]$	range defects	0	0	12
	$\Delta W_f[:]$	fuel-weight defects	0	0	12
	$\tau[:]$	throttle limit	0.01	1.0	36
					71

Table 1. The commercial aircraft optimal control problem with 12 segments of 2 cardinal nodes each.

We examine two ways of constraining the throttle parameter τ. If left unconstrained, the optimizer is free to find a trajectory in which the thrust necessary to balance (8) through (11) is outside the aircraft's capability. The throttle parameter τ is evaluated at each node (both cardinal and interior) in the trajectory, and treated as a path constraint using one of two methods. When treated as a traditional path constraint, the value of τ at each node is subject to constraint, resulting in an additional scalar constraint for each node in the trajectory. The other method tested uses the Kreisselmeier-Steinhsauser (K-S) function to aggregate the extremal values of τ into two scalar constraints; one for the upper bound and one for the lower bound.[32] The use of constraint aggregation reduces the number of rows in the Jacobian by combining a set of constraints to a single scalar value, which is computationally beneficial when using the adjoint method to solve for total derivatives for the optimizer.

A. Comparison of Trajectories of Varying Range

To check the general functionality of the tool, it was run on missions of varying range. Figure 5 shows the result of a sweep on range. Note the range in the plot is normalized to 1.0 to give an indication as to how quickly the aircraft reaches cruise altitude. These results match up well with those reported in the original work on PyMission, which serves as a verification of this implementation of the equations of motion.[4]

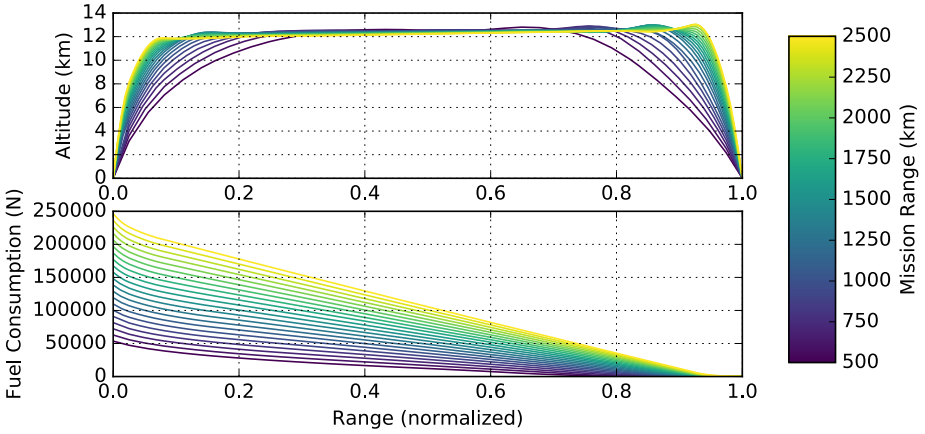

Figure 5. Flight profile and fuel consumption vs normalized range for constant Mach mission ranges from 500-2500km.

Of note is the oscillation in the altitude history at certain ranges. These oscillations are not physical, but rather artifacts due to non-optimal segment spacing. In the top graph of Figure 6 we show that increasing the number of segments, for an optimization on a fixed 1500 km mission, removes these artifacts and converges to a smoother result. However, adding segments naively across the entire mission is computationally wasteful. Ultimately this will be addressed by a grid refinement algorithm that allows us to achieve a good balance of accuracy in the solution and run time.

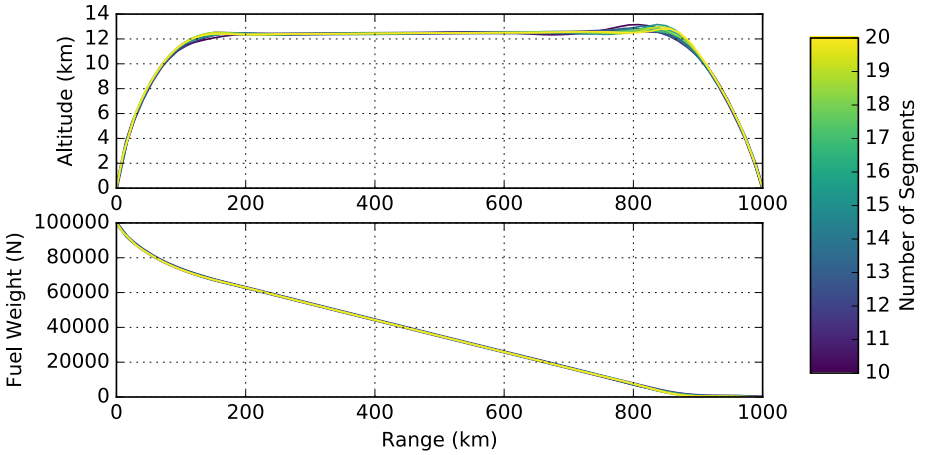

Figure 6. Flight profile and fuel consumption vs range for constant Mach mission with varying number of collocation segments.

The data in the bottom graph of Figure 6 shows that regardless of the number of segments the overall fuel burn does not vary by more than 0.025%. Hence we conclude that the oscillations don't adversely affect the fuel burn predictions and that it is reasonable to save computational cost in this case by reducing the segment count.

B. The effect of derivative calculation on performance and results

For the single commercial aircraft trajectory optimization problem defined in Table 1, we used four techniques to compute the objective gradient and constraint Jacobian:

1. Serial finite-difference
2. Parallel finite-difference (4 CPU's)
3. Serial forward analytic derivatives
4. Serial adjoint analytic derivatives

We examine the performance of computing finite-difference derivatives in both serial and parallel, where the design variables to be perturbed are divided up among the available CPUs. Given the availability of multi-core CPUs on practically all desktops and laptops, its reasonable to assume that parallel finite-difference (FD) is an option for practically any user. Note that in this implementation we do not employ sparse finite differencing techniques,[33] where multiple finite difference steps can be taken simultaneously across the non-linear analysis. This technique can provide a substantial speed increase for derivative computation with finite-differences, but can only be employed in special cases where many of the design variables do not directly affect the objective function and only affect specific constraints. This kind of sparsity if a fundamental property of the collocation forumaltion used here, where the objective function is only directly affected by design variable values in the last segment of the simulation. Future work will compare sparse-finite differencing to analytic derivatives methods.

We also examine both forward and adjoint modes for computing total derivatives. Forward mode is generally faster when there are fewer design variables than constraints, while adjoint mode is generally faster when there are fewer constraints than design variables. Both forward and reverse methods offered promise for

this problem, depending on whether the full set of path constraints on τ were used of if they were aggregated via K-S function.

For general collocation problems, the best analytic derivative method will be problem specific. For example, an initial value problem with no static or dynamic controls (a pure simulation without any optimal control features) will have a square Jacobian matrix and should favor neither method, though in practice the forward method is usually faster for this case. Adding dynamic controls substantially increases the number of design variables (one per-cardinal node, per-segment, per-dynamic control) and would thus tend to favor adjoint mode. Conversely, each additional path constraint adds a constraint at each node and would thus tend to favor forward mode. In the constant Mach problem posed in Table 1, we have one dynamic control (h) and two path constraints (upper and lower bounds on τ). We also add constraints at each segment boundary to ensure that the dynamic controls are C^1 continuous (continuous in their first time-derivative). That yields a total of 48 design variables and 71 constraints. Therefore, the forward mode should provide better performance.

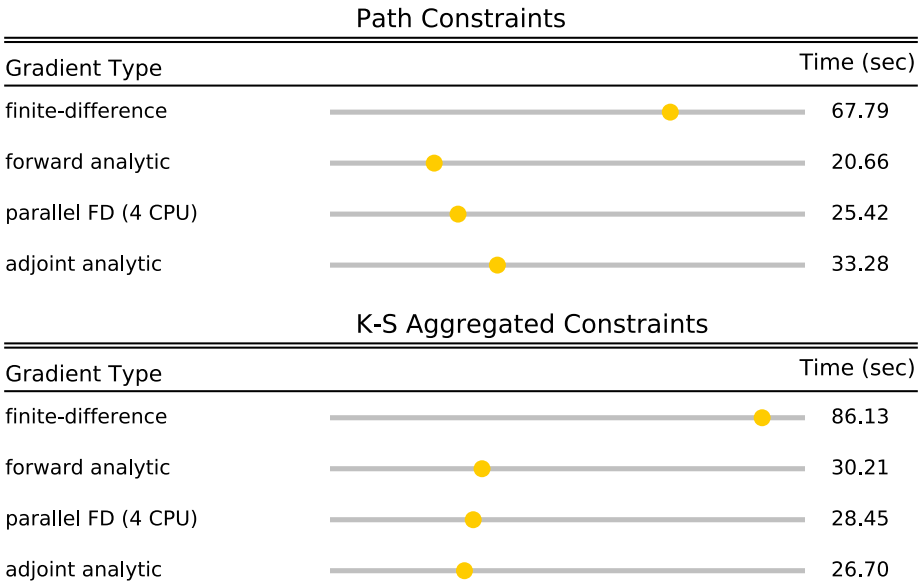

Table 2. Performance results for various derivative methods. Note the relative change in performance of the adjoint analytic method when changing from standard path constraints to K-S aggregated path constraints.

Constraint type	Fuel Burn (kg)	Iterations
Path	20063.1	22
K-S	20073.4	25

Table 3. Performance results for single mission with forward analytic derivatives.

Table 2 shows the run time results for the various derivatives computation modes with the aggregated and non-aggregated path constraints. As expected, the forward analytic method of computing derivatives performs the fastest when the path constraints are not aggregated, with the parallel FD coming in a close second. The adjoint analytic method is about 30% slower, which is also expected given the relative number of design variables and constraints. The serial finite-difference method is 3 times slower than the forward analytic derivatives. These speed results clearly demonstrate the benefit of analytic derivatives in terms of overall compute cost. The serial finite-difference case is both slow and computationally expensive. While the wall time for the parallel finite-difference case is nearly equal to that of the forward analytic case, that speed is achieved by spreading the compute cost across 4 CPUs and thus the overall compute cost is the same for both serial and parallel finite-difference.

When aggregating the path constraint via the K-S function the number of constraints is reduced by 34 because the 36 τ constraints (one at each of the 36 cardinal and interior nodes in the phase) are aggregated

into 2 scalar constraints. In that case the adjoint solution shows better performance compared to the forward method, which is expected. The K-S constrained versions sometimes exhibit more difficulty in convergence. This is a well known weakness of the K-S aggregation approach, where multiple active (or nearly active) constraints can cause convergence trouble.[34,35] This difficulty in convergence is not present in the non-aggregated form. Also note that the K-S function provides a slightly conservative value for constraint satisfaction, and thus the objective is slightly worse than in the non-aggregated case. Some of these issues could possibly be alleviated by using an altenate aggregation formulation, such as the induced aggregation method devloped by Kennedy and Hicken.[34] Investigations into alternate aggregation methods will be the subject of future work.

C. Solving Multiple Simultaneous Trajectories in Parallel

In addition to using parallelization to improve performance of the finite difference scheme, OpenMDAO gives us the capability to parallelize all computations. In this work we considered three approaches. In the first, we distribute the RHS evaluations across the available CPUs. For particularly large problems this may be beneficial. But with only 12 segments and a total of 36 nodes, the computational burden is not particularly high to begin with. Having particularly expensive calculations in the RHS would potentially change this but, as is, we saw zero benefit to parallelizing calculations within a trajectory.

The second approach we examined, one that is effective even for fairly simple trajectories such as those shown here, is to use parallelization to solve multiple trajectories simultaneously. Using the same basic problem formulation, we define four individual trajectories, each with a different range target, and solve them simultaneously. The objective function for the overall problem is defined by summing up the initial fuel weights of each of the individual trajectories.

By composing the objective as a linear combination of the four mission fuel burns, the objective of course depends on all four trajectories. However the various constraints on each mission are independent of calculations in all the other missions and depend only on the design variables for that mission. Because of this independence, it is possible to solve for some of the derivatives in parallel. This situation is analagous to the sparse finite-differencing method mentioned earlier, but it applies to analytic derivatives instead. This kind of parallelism of the constraints is common in multi-point optimization problems, and OpenDMAO provides an additional level of parallelism to help take advantage of it. Results for both the serial and parallel derivatives are presented, for comparison purposes and give an indication of the additional increase in parallel scaling that is offered by OpenMDAO. Currently our implementation is limited to parallelizing analytic derivative computations across multiple trajectories However, given the sparse nature of segments, the parallel derivatives approach also shows promise for use within a phase.

Table 5 shows the results of these various approaches in terms of run time. The objective and iteration counts are shown in Table 4. All standard path constraint cases showed similar objectives and iteration counts regardless of the derivative type used, and the same is true for the aggregated constraint cases. The data shows a 60% reduction in compute time for the parallel-derivatives approach vs the serial-derivatives approach, which highlights the importance of solving the derivatives in parallel when the appropriate sparsity is available in the problem. With out the parallel derivatives, using only parallelization for the non-linear function evaluations, only a modest compute savings was achieved compared to the serial case. It's only with the parallel adjoints that appropriate parallel scaling is achieved.

Constraint type	Fuel Burn (kg)	Iterations
Path	50865.7	30
K-S	50943.0	59

Table 4. Performance results for multiple missions with forward analytic derivatives.

As before, the aggregated constraint cases resulting in larger (less optimal) fuel consumption because of the conservative nature of the K-S constraint. They also showed a significant increase in the number of optimization major iterations needed to reach the same convergence, compared to the non-aggregated cases. As in the single mission case, adjoint analytic derivatives provide the fastest performance among the K-S constrained cases. For the cases using a standard path constraint, the forward analytic derivative is fastest, and that case is fast overall.

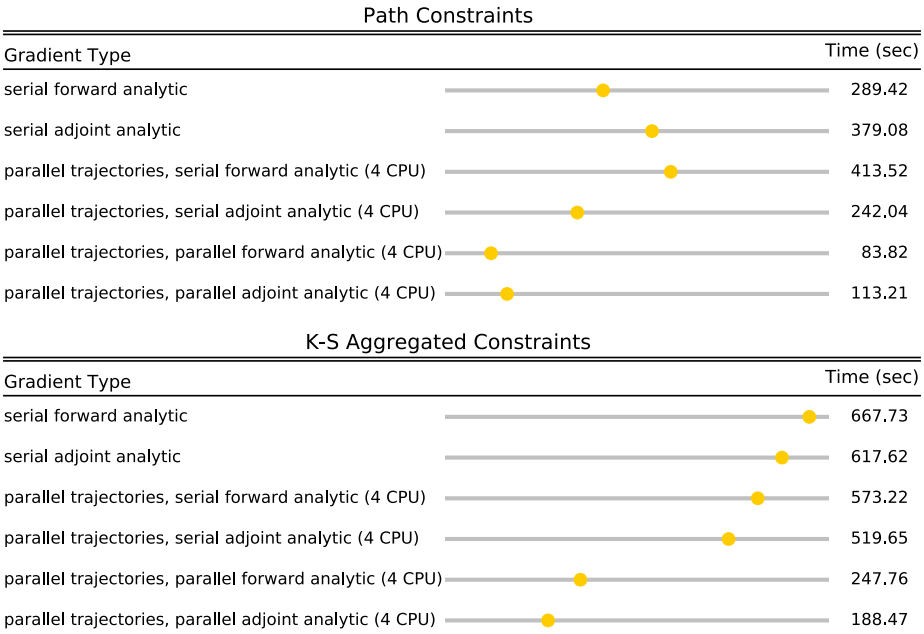

Table 5. Performance results for various parallelization schemes. All cases with aggregated τ constraints took 60 iterations to converge. Cases using a standard path constraint converged in 26 iterations.

VI. Conclusions

We have demonstrated the applicability of a segmented Legendre-Gauss-Lobatto collocation scheme to the problem of commercial aircraft trajectory optimization. We use the approach of Reference 4 in which altitude and Mach number are treated as dynamic control variables whose values are specified at nodes throughout the problem, while steady flight assumptions are applied throughout the given flight profile.

We examined various techniques for computing derivatives for the problem. The use of standard path constraints in the problem, in which a constraint is added at every node within the phase, results in a Jacobian matrix with far more rows than columns. In that case, the use of forward analytic derivatives resulted in faster run times than those cases which used adjoint analytic derivatives. We addressed this issue through the use of constraint aggregation via the Kreisselmeier-Steinhsauser (K-S) function from Reference 32. The adjoint derivative approach was indeed fastest when the path constraint on throttle was aggregated, but poor convergence properties resulting from the use of the K-S function meant that the use of standard path constraints with the forward analytic method was fastest overall.

The segmented nature of the collocation approach gives us the opportunity to parallelize the computations involved in a number of ways. Three such ways are demonstrated here. We can parallelize the simultaneous optimization of multiple trajectories, or the computation of both finite-difference derivatives or analytic derivatives. Further improvements can be made by leveraging the sparsity of the resulting nonlinear programming problem to compute more derivatives simultaneously and in parallel; a parallel and analytic extension of the sparse finite difference approach used in state of the art collocation-based trajectory optimization software. The results demonstrate that taking advantage of the parallelism for solving analytic derivatives is key to achiving efficient parallel scaling. Future work will remove some of the simplifying assumptions in the equations of motion and will leverage the optimal control formulation to study aircraft design problems with time varying control schemes as part of the design space.

References

[1]McCullers, L. A., "Aircraft configuration optimization including optimized flight profiles," NASA CR CP-2327, National Aeronautics and Space Administration, 1984.

[2]Rutowski, E. S., "Energy Approach to the General Aircraft Performance Problem," *Journal of the Aeronautical Sciences*, Vol. 21, No. 3, pp. 187–195.

[3]Lukaczyk, T. W., Wendorff, A. D., Colonno, M., Economon, T. D., Alonso, J. J., Orra, T. H., and Ilario, C., *SUAVE: An Open-Source Environment for Multi-Fidelity Conceptual Vehicle Design*, American Institute of Aeronautics and Astronautics, 2015.

[4]Kao, J., Hwang, J., Martins, J., Gray, J. S., and Moore, K. T., *A modular adjoint approach to aircraft mission analysis and optimization*, American Institute of Aeronautics and Astronautics, 2016/04/13 2015.

[5]Paris, S. W., Riehl, J. P., and Sjauw, W. K., "Enhanced Procedures for Direct Trajectory Optimization Using Nonlinear Programming and Implicit Integration," *Proceedings of the AIAA/AAS Astrodynamics Specialists Conference and Exhibit*, No. 2006–6309, 2006.

[6]Hargraves, C. R. and Paris, S. W., "Direct trajectory optimization using nonlinear programming and collocation," *Journal of Guidance, Control, and Dynamics*, Vol. 10, Aug. 1987, pp. 338–342.

[7]Clarke, J.-P. B., Ho, N. T., Ren, L., Brown, J. A., Elmer, K. R., Tong, K.-O., and Wat, J. K., "Continuous descent approach: Design and flight test for Louisville International Airport," *Journal of Aircraft*, Vol. 41, No. 5, 2004, pp. 1054–1066.

[8]Maskew, B., "Formation Flying Benefits Based on Vortex Latic Calculations," NASA CR CR-151974, National Aeronautics and Space Administration, 1977.

[9]Blake, W. and Multhopp, D., "Design, Performance and Modeling Considerations for Close Formation Flight," *23rd Atmospheric Flight Mechanics Conference, Guidance, Navigation, and Control*, AIAA, 1998.

[10]Reckzeh, D., "Multifunctional Wing Moveables: Design Of The A350XWB And The Way To Future Concepts," *29th Congress of the International Council of the Aeronautical Sciences*, St. Petersburg, Russia, September 2014.

[11]Felder, J. L., Kim, H. D., and Brown, G. V., "Turboelectric Distributed Propulsion Engine Cycle Analysis for Hybrid-Wing-Body Aircraft," *47th AIAA Aerospace Sciences Meeting, Orlando, FL, January*, 2009.

[12]Welstead, J. R. and Felder, J. L., "Conceptual Design of a Single-Aisle Turboelectric Commercial Transport with Fuselage Boundary Layer Ingestion," *54th AIAA Aerospace Sciences Meeting*, 2016.

[13]DECAMP, R. and Hardy, R., "Mission adaptive wing advanced research concepts," *11th Atmospheric Flight Mechanics Conference*, 1984, p. 2088.

[14]Burdette, D. A., Kenway, G. K., and Martins, J., "Performance Evaluation of a Morphing Trailing Edge Using Multipoint Aerostructural Design Optimization," *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2016, p. 0159.

[15]Botero, E. M., Wendorff, A., MacDonald, T., Variyar, A., Vegh, J. M., Lukaczyk, T. W., Alonso, J. J., Orra, T. H., and da Silva, C. I., "A Multidisciplinary Design Optimization Framework for Design Studies of an Efficient Supersonic Air Vehicle," *54th AIAA Aerospace Sciences Meeting*, 2016, AIAA-2016-1275.

[16]Hwang, J. T. and Martins, J. R. R. A., "Parallel allocation-mission optimization of a 128-route network," *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Dallas, TX, 2015.

[17]Gray, J., Moore, K. T., and Naylor, B. A., "OpenMDAO: An open source framework for multidisciplinary analysis and optimization," *13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Fort Worth, TX, AIAA, AIAA-2010-9101*, 2010, pp. 5–7.

[18]Betts, J. T., "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 2, 1998, pp. 193–207.

[19]Herman, A. L. and Conway, B. A., "Direct optimization using collocation based on high-order Gauss-Lobatto quadrature rules," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 3, 2016/04/13 1996, pp. 592–599.

[20]Ross, I. M. and Fahroo, F., "Pseudospectral Knotting Methods for Solving Nonsmooth Optimal Control Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 3, 2016/04/13 2004, pp. 397–405.

[21]Garg, D., Patterson, M., Darby, C., Francolin, C., Huntington, G., Hager, W., and Rao, A., *Direct Trajectory Optimization and Costate Estimation of General Optimal Control Problems Using a Radau Pseudospectral Method*, American Institute of Aeronautics and Astronautics, 2009.

[22]Betts, J. T. and Huffman, W. P., "Path-constrained trajectory optimization using sparse sequential quadratic programming," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 1, 1993, pp. 59–68.

[23]Patterson, M. A., Hager, W. W., and Rao, A. V., "A ph mesh refinement method for optimal control," *Optimal Control Applications and Methods*, Vol. 36, No. 4, 2015, pp. 398–421.

[24]Darby, C. L., Hager, W. W., and Rao, A. V., "An hp-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Applications and Methods*, Vol. 32, No. 4, 2011, pp. 476–502.

[25]Haftka, R. T., "Simultaneous Analysis and Design," *AIAA Journal*, Vol. 23, No. 7, 1985, pp. 1099–1103.

[26]Martins, J. R. R. A. and Lambe, A. B., "Multidisciplinary Design Optimization: A Survey of Architectures," *AIAA Journal*, Vol. 51, 2013, pp. 2049–2075.

[27]Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM review*, Vol. 47, No. 1, 2005, pp. 99–131.

[28]Betts, J. T. and Cramer, E. J., "Application of direct transcription to commercial aircraft trajectory optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 1, 1995, pp. 151–159.

[29]Vassberg, J., Dehaan, M., Rivers, M., and Wahls, R., *Development of a Common Research Model for Applied CFD Validation Studies*, American Institute of Aeronautics and Astronautics, 2016/04/13 2008.

[30]Gray, J. S., Hearn, T. A., Moore, K. T., Hwang, J., Martins, J., and Ning, A., "Automatic Evaluation of Multidisciplinary Derivatives Using a Graph-Based Problem Formulation in OpenMDAO," *15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, American Institute of Aeronautics and Astronautics, 2014/07/08 2014.

[31]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.

[32]Martins, J. and Poon, N. M., "On structural optimization using constraint aggregation," *VI World Congress on Structural and Multidisciplinary Optimization WCSMO6, Rio de Janeiro, Brasil*, Citeseer, 2005.

[33]Betts, J. T. and Frank, P. D., "A sparse nonlinear optimization algorithm," *Journal of Optimization Theory and Applications*, Vol. 82, No. 3, 1994, pp. 519–541.

[34]Kennedy, G. J. and Hicken, J. E., "Improved Constraint-Aggregation Methods," *Computer Methods in Applied Mechanics and Engineering*, Vol. 289, 2015, pp. 332–354.

[35]Kennedy, G. J., "Strategies for adaptive optimization with aggregation constraints using interior-point methods," *Computers & Structures*, Vol. 153, 2015, pp. 217–229.