CITATION

MLA: MacDonald, T., Clarke, M., Botero, E., Vegh, J. M. and Alonso, J. J., "SUAVE: An Open-Source Environment for Multi-Fidelity Vehicle Optimization" AIAA Aviation Forum, Denver, CO, June 2017. AIAA 2017-4437

BibTex

```
@proceedings{macdonald2017suave,
title={SUAVE: an open-source environment enabling multi-fidelity vehicle optimization},
author={MacDonald, Timothy and Clarke, Matthew and Botero, Emilio M and Vegh, Julius M and Alonso, Juan J},
booktitle={18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference},
pages={4437},
year={2017}
}
```

SUAVE: An Open-Source Environment Enabling Multi-Fidelity Vehicle Optimization

Timothy MacDonald*, Matthew Clarke1, Emilio Botero1, J. Michael Vegh1
Juan J. Alonso†
Stanford University, Stanford, CA 94305, USA

SUAVE is a conceptual level aerospace vehicle design environment that allows a user to incorporate new methods and information sources to analyze both conventional and unconventional configurations. This paper builds upon previous works where SUAVE analyzed and optimized several types of aircraft using low-fidelity methods, and also incorporated links with higher-fidelity tools. Here we demonstrate SUAVE's use in the multi-fidelity optimization of unconventional designs. The objective of this type of framework is to enable high performance while working with constrained computational resources. This capability will be demonstrated through the use of additive correction surrogates and trust region model management, with SUAVE managing the levels of fidelity according to these methods. Two different test cases are optimized here. We present results for a supersonic transport with varying wing area and aspect ratio, and a blended-wing-body aircraft with varying planform values subject to stability constraints. Both are analyzed at the cruise condition with two levels of analysis fidelity.

I. Introduction

SUAVE is an aerospace vehicle design environment established from the beginning to support multiple levels of fidelity. A designer can specify different analysis types by changing a single line on the input file. This enables significant time savings and flexibility over legacy tools that require extensive rewriting in order to change how an aerospace vehicle is evaluated, are not freely available, or are difficult to develop openly. [1–3] Previous work discussed analysis capabilities and single-fidelity optimization capabilities. [4–6] However, multiple levels of fidelity have not yet been incorporated in a single design process using optimization through SUAVE.

Designing unconventional aircraft is often challenging as tried-and-true correlations and handbook methods are not fitted to futuristic vehicles. To reduce uncertainty in the design, we must introduce physics-based high-fidelity simulations of the vehicle. These simulations tend to

be computationally expensive. If we are exploring a large design space, then these expensive tools can exceed the computational budget. However, if we rely solely on low-fidelity tools we may reach incorrect conclusions as these methods are tuned to conventional aircraft. Instead, this requires a hybrid approach. Multi-fidelity methods can be used to reduce optimization time by using higher-fidelity tools only when necessary. In general, the goal is to bring the right level of simulation at the right time to reduce computational expense and give more options to the designer.

This paper is organized as follows: Section II provides a background on the general and recently upgraded capabilities in SUAVE, with particular focus on those that are used in the optimization process. Section III details the methodology used to construct the multi-fidelity optimization cases. Section IV show the results of applying these schemes to two cases.

II. SUAVE Capabilities

A. General Capabilities

SUAVE is a fully featured conceptual design tool with an extensive list of low-fidelity capabilities for both conventional and unconventional aircraft, along with links to a growing set of higher-fidelity tools. These capabilities cover all of the major disciplines needed in the conceptual design stage. In its current form, these capabilities include:

- Aerodynamics for subsonic and supersonic flight
- Static and dynamic stability metrics
- Basic performance estimation methods such as takeoff and landing field lengths and payload range diagrams
- Energy networks for gas turbine, battery, fuel-cell, and solar-panel based vehicles
- Weight correlations for tube-and-wing aircraft, blended-wing-bodies, human powered aircraft, and small UAVs
- Noise correlations for tube-and-wing aircraft and components
- Segment-based mission architecture
- Geometry output for all supported aircraft configurations through OpenVSP [7]
- Euler CFD analysis for lift and inviscid drag calculation through SU2[8]
- Optimization with interfaces such as pyOpt and ability to add additional optimizers

The key capabilities discussed in this work are supersonic aerodynamics and static stability calculations. However, all necessary disciplines are used as part of the mission evaluations.

B. New Capabilities

1. OpenVSP Wave Drag

SUAVE has the general capability to produce geometry and run analyses through the OpenVSP[7] Python API. This has also been extended with other tools to allow the use of CFD[6]. In this work, we have implemented a link to the wave drag analysis that comes with OpenVSP. This analysis operates by taking the OpenVSP geometry and using an area rule method to find the volume wave drag for a given Mach number.

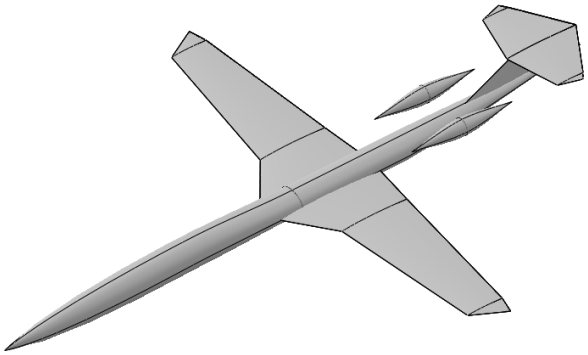

Fig. 1 OpenVSP Aircraft Geometry

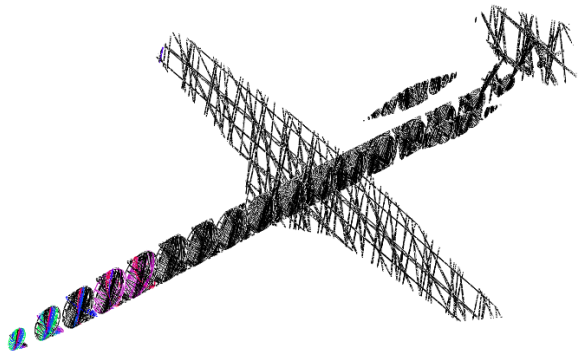

Fig. 2 Area Rule Slices

For this method, the number of slices and the number of rotations about the center are specified. The area at each slice is then found and an area distribution is fit along each rotation. This fit is created using the Eminton-Lord wave drag evaluation, which estimates the area distribution based on a Fourier series and the assumption that the fitted distribution will minimize the wave drag given the set points[9]. This assumption is important to note because it means that since increasing the number of slices will increase the number of non-ideal points in the distribution, it will also increase the drag, rather than lead to convergence. The OpenVSP developers chose this method under the assumption that the geometry created would not be exactly sculpted for the best wave drag properties, but that a final aircraft based on the given parameters would. The type of fits generated with this method can be seen in Figure 3, which shows a chart provided by the OpenVSP GUI.

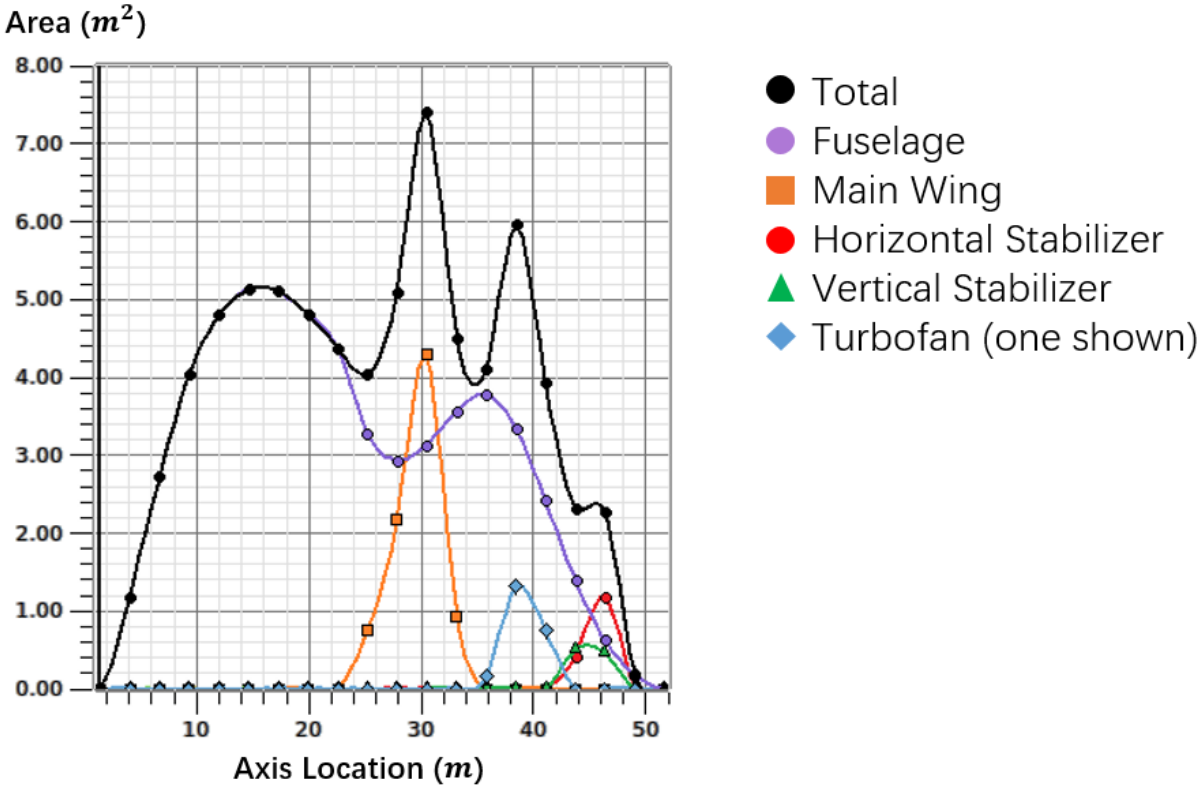

Fig. 3 OpenVSP Aircraft Area Fit

There are also other important notes about the limits in using this analysis. First, since this computes only the volume wave drag, lift wave drag must still be computed according to other methods. The ratio of these values, and therefore the importance, varies with flight conditions. For the cases investigated here, the volume wave drag dominates. Additionally, triangle intersections are used within OpenVSP to compute slice areas, and if these are not well aligned some areas may be miscalculated as zero. This is typically obvious in the drag results and can be identified numerically, but can cause difficulty for optimization if not handled well.

The OpenVSP default number of slices and rotations are 20 and 10 respectively. SUAVE also uses these values as defaults, but they are adjustable directly by the user if desired. The appropriate Mach number is fed into the analysis automatically and does not need to be specified. The analysis output is scaled according to the vehicle reference area. Since many evaluation points may be run at a single Mach number for a given configuration, results for each Mach number are saved and reused until the configuration is changed.

2. AVL Stability

Another important part of the conceptual design process is the study of an aircraft's stability in its foreseeable flight regimes. Though Athena Vortex Lattice[10] (AVL) has been previously used within SUAVE, this update adds stability capabilities and extends SUAVE's suite of tools for analyzing and optimizing aircraft configurations of all geometry types.

First, improvements were made in the translation of geometry from the SUAVE vehicle setup to the AVL interface. This was done in part to allow for the admissible comparison of optimization outcomes from higher order computational methods such as CFD using SU2[6]. These new changes now allow a more accurate model of control surfaces, complex fuselage geometry, and multi-segment wings. Figures 3 and 4 demonstrate this upgraded capability in the case of the Boeing 737-800's geometry.

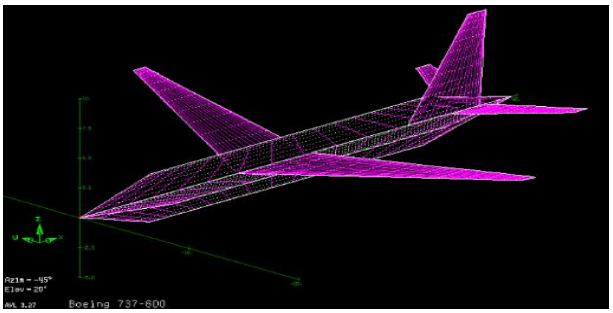

Fig. 4 Basic Boeing 737 Aircraft Geometry

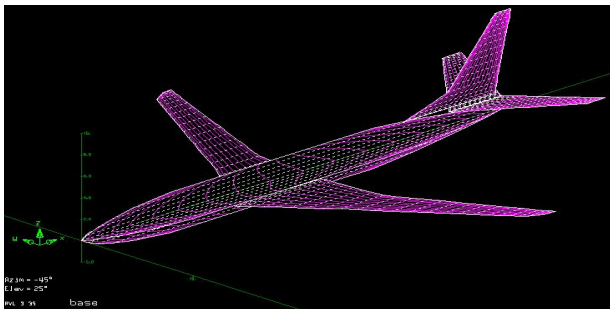

Fig. 5 Revised Boeing 737 Aircraft Geometry

These upgrades also allow the generation of aircraft such as box-wings, joined-wings, canard configurations, and blended-winged-bodies (BWB). See Figures 6 and 7 for illustrations.

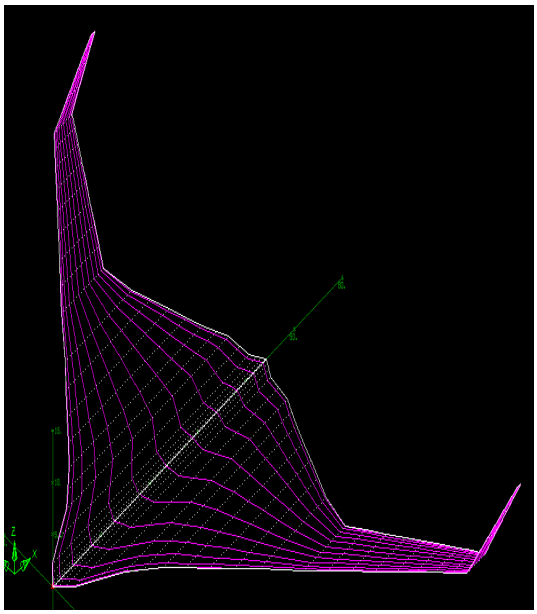

Fig. 6 BWB-450 Concept Aircraft

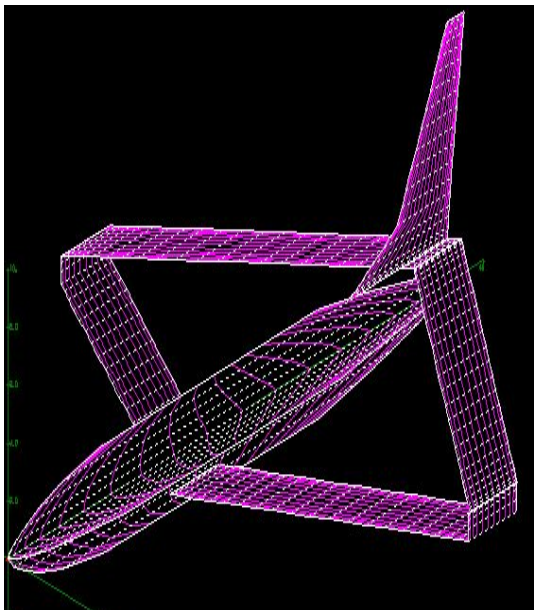

Fig. 7 Boeing 737 Derivative Joined-Wing Concept

In addition to the geometry upgrades, an AVL analysis routine has been built to compute the non-dimensional stability derivatives in both the longitudinal and lateral directions, the moment coefficients about the aircraft's center of gravity, and the neutral point. Since running a full stability analysis at each point in the mission can be extremely expensive, a surrogate model is built for key values by running through a set of representative angles of attack and Mach numbers. Scikit-learn's[11] Gaussian process function is used for this fit.

III. Multi-fidelity Optimization Methods

A. General Structure

In general, multi-fidelity optimization is meant to allow a reduction in cost required for a given level of accuracy or allow a higher level of accuracy to be achieved at a given cost. This is done by intelligently combining the results of analysis performed at multiple levels of fidelity. A common example is aerodynamic analysis types, which move from basic analytical formulations to expensive CFD solutions. SUAVE was designed to handle these types of analyses from the ground up by allowing the user to swap out analyses without modifying the vehicle or unrelated analysis types. This means that new analyses can be integrated and new optimizers can be tested without time spent significantly modifying the underlying code. The basic optimization problem is formulated mathematically in the equations below, with x as the design variables, f as the objective, and g and h as constraints. The key in the multi-fidelity case is to evaluate these equations with varying levels

of analysis and combine them in an intelligent way.

$$\begin{aligned} & \min f(x) \\ s.t. \quad & g(x) \leq 0 \\ & h(x) = 0 \end{aligned} \tag{1}$$

The multi-fidelity capabilities highlighted here were chosen to demonstrate how multiple levels of fidelity can be handled in SUAVE. We believe they may be useful for future users as they are, but the multi-fidelity capabilities are by no means limited to these methods. The general optimization structure in SUAVE is built on a class (called nexus below) that holds the optimization parameters, vehicles, missions, analysis types, optimization procedure, and any other data that will be needed as part of the optimization process. A top level optimize file allows the user to specify an objective, inputs, and constraints, along with their scaling and units. This setup also allows aliases so that a simple variable name can be linked with several parameters that are used in the SUAVE analysis or returned as outputs. A previous paper provides full details on how these methods function and demonstrates their use[5].

There are only two changes in the setup needed to perform multi-fidelity optimization. The first is a change to the procedure that will set analyses based on the fidelity level currently being evaluated. In our examples, only two fidelity levels are used, but more can be set if needed. The second is a change to the optimization method called. This is where a user-created function should be entered if the user requires an optimization algorithm that is not available in the standard distribution. To illustrate the simplicity of making this analysis change, we show the required code below in Figures 8 and 9. All the other optimization modules such as vehicle sizing can remain the same, though they could also be changed based on fidelity level if desired by the user.

```
if nexus.fidelity_level == 2:
    aerodynamics = SUAVE.Analyses.Aerodynamics.AVL()
    stability = SUAVE.Analyses.Stability.AVL()
elif nexus.fidelity_level == 1:
    aerodynamics = SUAVE.Analyses.Aerodynamics.Fidelity_Zero()
    stability = SUAVE.Analyses.Stability.Fidelity_Zero()
else:
    raise ValueError('Selected fidelity level not supported')
aerodynamics.geometry = copy.deepcopy(nexus.vehicle_configurations.base)
stability.geometry = copy.deepcopy(nexus.vehicle_configurations.base)
nexus.analyses.base.append(aerodynamics)
nexus.analyses.base.append(stability)

nexus.missions = mission_bwb.setup(nexus.analyses)
```

Fig. 8 Analysis Update Code

```
output = additive_setup.Additive_Solve(problem,max_iterations=100,num_samples=20)
```

Fig. 9 Optimizer Selection Code

B. Additive Surrogate Model

The first algorithm implemented here uses an additive correction function to map high and low fidelities together. First, a specified number of samples are collected for both the high and low fidelity models at given points. Since these are evaluated at the same points, the difference between the two sets of objectives and constraints can be quantified. This difference is then used to build an additive correction function, which is combined with the lower-fidelity analysis to create a corrected function that is used for optimization, providing an optimization subproblem as in (1).

In this implementation, the initial set of points is chosen with Latin Hypercube Sampling[12] of the design space. After the initial sampling is complete, a surrogate for the additive correction is formed using Scikit-learn's Gaussian process regression[11]. This surrogate provides us with both an estimated value and a probability distribution of that value at all points in the design space. We use this information to choose our next evaluation point by optimizing for expected improvement, a quantity described in detail in Reference EI and applied in a similar way in Reference krooEI. This method is essentially a way to improve the surrogate in less sampled areas that are most likely to provide improvement over the current best sample. This is often more effective than simpler approaches such as repeatedly optimizing the corrected function, and is therefore useful in situations with constrained resources. However, since the maximum expected improvement can be highly multi-modal, global optimization techniques become necessary. Here we use ALPSO[13], which is a particle swarm optimizer built into pyOpt[14]. Once a given number of expected improvement evaluations is completed, we optimize the corrected function once using SNOPT[15], which is a gradient based optimizer.

The full algorithm is outlined in Algorithm 1. In this outline, the inequality and equality constraints are combined in g for brevity, and correction fits for the objective and constraints are shown as α and β. EI is the expected improvement function, with Φ and ϕ being the standard normal cumulative distribution function and probability density function respectively, and σ as the standard deviation. We also provide the option to simply minimize the corrected function at each point. In that case, $f_{low}(x) + \alpha(x)$ from the given algorithm would be optimized using SNOPT for a new point instead, and a check for convergence would be done at each iteration with the condition being satisfied if the change in successive objective values is within a specified tolerance.

/* Initial sampling */
samples = Latin Hypercube (num dimensions, num samples, bounds);
/* Compute differences */
$f_{diff} = f_{high}(samples) - f_{low}(samples)$;
$g_{diff} = g_{high}(samples) - g_{low}(samples)$;
/* Fit functions for expected value and standard deviation */
α, σ_α = Gaussian Process Fit $(samples, f_{diff})$;
β, σ_β = Gaussian Process Fit $(samples, g_{diff})$;
$f_{corr}(x) = f_{low}(x) + \alpha(x)$;
$\hat{f}^* = min(f_{high}(samples))$;
$EI(x, \hat{f}^*) = (\hat{f}^* - f_{corr}(x))\Phi((\hat{f}^* - f_{corr}(x))/\sigma_\alpha(x)) + \sigma_\alpha \phi((\hat{f}^* - f_{corr}(x))/\sigma_\alpha(x))$;
num iterations = 0;
while *num iterations* < *max iterations* **do**
    increment num iterations;
    new point = optimize $EI(x, \hat{f}^*)$ $s.t.$ $g_{low}(x) + \beta(x)$ satisfied;
    update fits with additional point;
    get new $\hat{f}^*$;
end
pick initial x for below based on optimal sampled point;
f^*, x^* = optimize $f_{low}(x) + \alpha(x)$ $s.t.$ $g_{low}(x) + \beta(x)$ satisfied;

Algorithm 1: Additive Surrogate Method

C. Trust Region Model Management

The second algorithm we have chosen is a trust region model management (TRMM) algorithm. The basic idea of TRMM is also to solve a series of optimization subproblems using a cheaper low-fidelity model, but this time constrained inside of a trust region based around a previously sampled high-fidelity point. The trust region is a region where a corrected low-fidelity model is expected to be fairly accurate. To build the corrected model, both the high and low fidelity functions and gradients are evaluated at the region center and correction terms are calculated such that the value and gradients of the corrected function and high-fidelity function will match. Once an optimization of the low-fidelity corrected model is complete, the expansion, contraction, and movement of the trust region is updated according to a given set of rules.

In our implementation of TRMM we use additive correction terms for the corrected low-fidelity model. We accept a new trust region center if the low-fidelity corrected optimum within the previous region provides a decrease in the high-fidelity objective or constraint violation. We use a traditional trust region ratio to update the trust region size after the solution of each subproblem. Termination of the TRMM optimization is determined by a simple convergence criterion which is met if successive iterations show improvement of less than a set tolerance. The basic algorithm is outlined in Algorithm 2 below, with details omitted for brevity.

select initial trust region center (x_{center}) and size;
while *num iterations* < *max iterations* **do**
  /* Find parameters for correction and build correction functions */
  $b_f = f_{high}(x_{center}) - f_{low}(x_{center})$;
  $A_f = \nabla f_{high}(x_{center}) - \nabla f_{low}(x_{center})$;
  $f_{corr}(x) = f_{low} + A_f \bullet (x - x_{center}) + b_f$;
  repeat for constraints;
  /* Evaluate */
  f^*_{corr}, x^*_{corr} = optimize $f_{corr}(x)$ within trust region $s.t.$ constraints are satisfied;
  if *infeasible* **then**
    minimize norm of constraint violation
  end
  /* Check Convergence */
  if *objective change* < *tolerance* **then**
    return f^*_{corr}, x^*_{corr}
  end
  /* Determine size and center of next trust region */
  compute accuracy ratio ρ based on evaluations at center and optimum;
  if *objective or constraint violation improved* **then**
    set trust region center to x^*_{corr}
  end
  if *no improvement above* **then**
    shrink trust region
  else
    determine new size based on ρ
  end
end

Algorithm 2: Trust Region Model Management

IV. Cases

The cases here are meant to demonstrate the capability to evaluate unconventional designs. We have chosen a supersonic aircraft and a blended-wing body for these purposes. The new capabilities used with AVL and OpenVSP here have been validated in other sources, [10, 16] so we focus on the aircraft without presenting validation data here. Validation of other analysis methods used in SUAVE is available in previous publications[4–6]. These cases are limited in scope and are meant to show what is possible in a way that is easy for readers unfamiliar with the particular configurations

to examine, rather than make a claim that the final aircraft meet all necessary constraints to complete a full mission.

A. Supersonic Business Jet

1. Aircraft and Mission Definition

The first case we investigate is a supersonic business jet loosely based on the Aerion AS2[17]. This aircraft is meant to carry eight passengers a distance of 4750 nautical miles. A key feature is the unswept wing that allows the use of laminar flow to reduce drag. We use a Mach number of 1.4 at an altitude of 51,000 feet. The full mission is not modeled for this optimization case, and a point evaluation at the specified conditions is done instead.

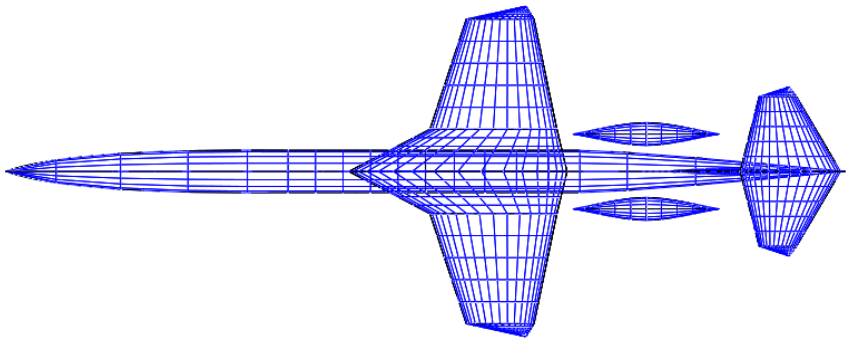

Fig. 10 SUAVE Generated OpenVSP Supersonic Business Jet Model

2. Optimization and Methodology

For the optimization case, we vary the main wing area and aspect ratio while attempting to minimize the rate of fuel burn. There are no constraints in this problem. The variables and their bounds are shown in Table 1.

What makes this case different from a typical subsonic aircraft is the large factor played by wave drag, which accounts for over half of the total drag on the aircraft at this condition. Unfortunately, low-fidelity correlations-based wave drag models do not do a particularly good job of capturing these effects. SUAVE's low-fidelity model is based on approximations for the wing and other bodies[4]. These methods also require that wave drag for each of the bodies is computed separately. To push the fidelity level a bit higher, we use the OpenVSP wave drag function discussed earlier to properly evaluate the full aircraft. Full aircraft wave drag due to volume is shown for a range of reference areas and aspect ratios in Figure 11 and 12, showing a significant difference. These coefficients have been normalized to a reference area of 125 m^2. The difference this causes in fuel burn is then shown

Table 1 Design Variables, Initial Values, and Bounds

Variable	Initial Value	Lower Bound	Upper Bound
Wing Area [m^2]	125	120	180
Aspect Ratio [-]	3.3	2.0	6.0

in Figures 13 and 14.

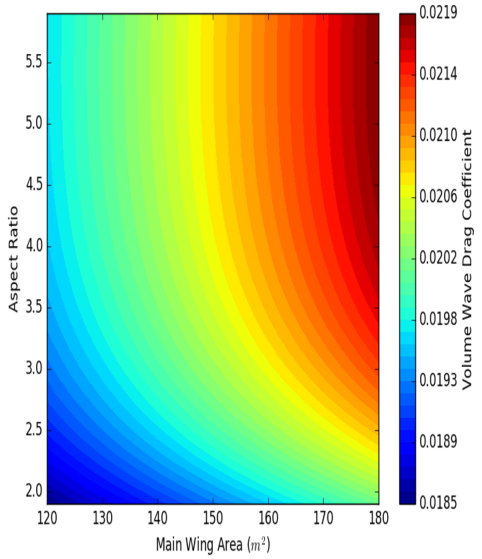

Fig. 11 Low-fidelity Volume Wave Drag

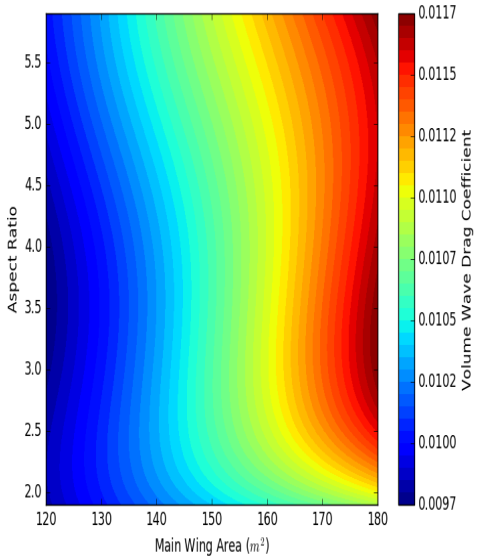

Fig. 12 OpenVSP Volume Wave Drag

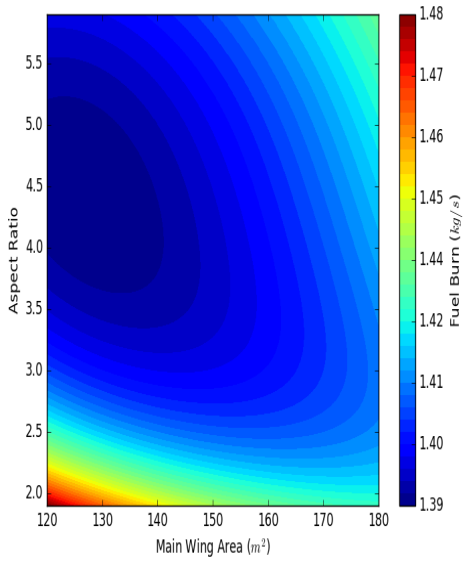

Fig. 13 Low-fidelity Fuel Burn

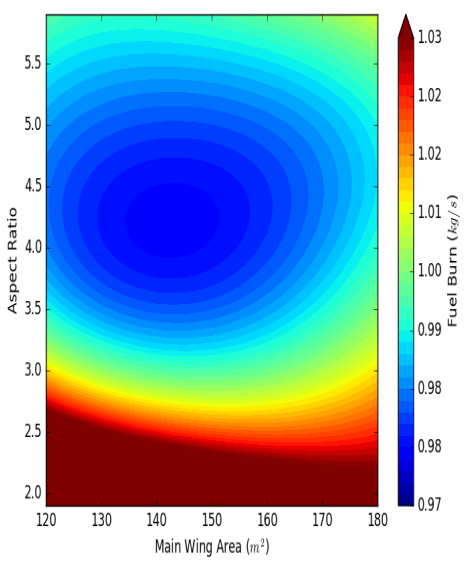

Fig. 14 OpenVSP Fuel Burn

3. Results

Both optimizations return about the same point. The difference is due to the lack of a convergence requirement in the additive correction method when maximum expected improvement is used. The fuel burn from the additive correction surrogate added to the low-fidelity results (instead of the VSP-based value) is shown in Figure 15 with points representing the OpenVSP evaluation points. The more scattered evaluation points in black show the initial sampling distribution, with the points chosen through maximum improvement estimation shown in orange. While these move towards the optimum in this case, this is not a general property of this method. The optimum is shown as a single point. The method allows 15 high-fidelity evaluations, with the first 10 being done according to Latin Hypercube Sampling. The optimum value is at wing area of 144.3 m^2 and aspect ratio of 4.204.

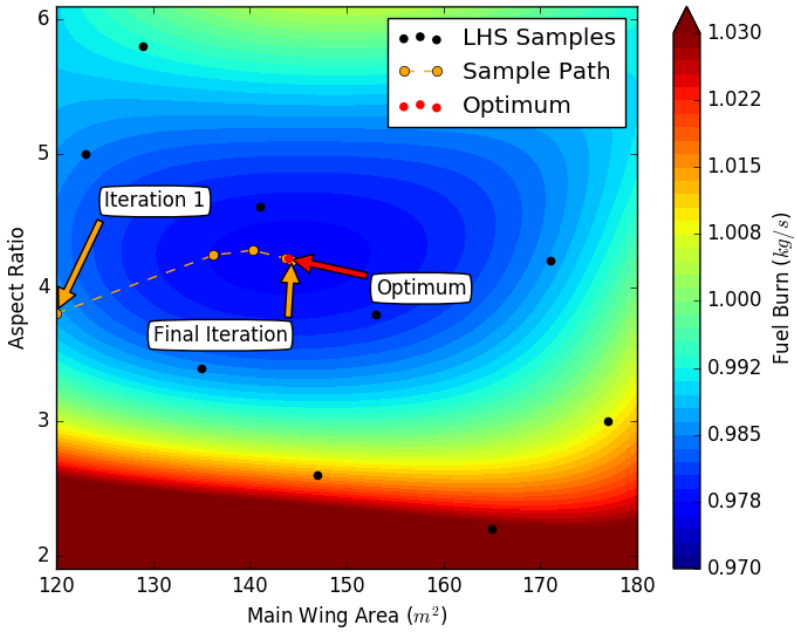

Fig. 15 Supersonic Jet Additive Correction Optimization Path

For the TRMM, we start at the initial value listed in Table 1. Figure 16 shows how the trust region optimal points move along design space. In this case the carpet plot shown is the VSP-based data. It required a total of 42 high-fidelity evaluations (14 iterations) before reaching convergence at the level of 1e-5. The optimum point here was at wing area 142.9 m^2 and aspect ratio 4.147.

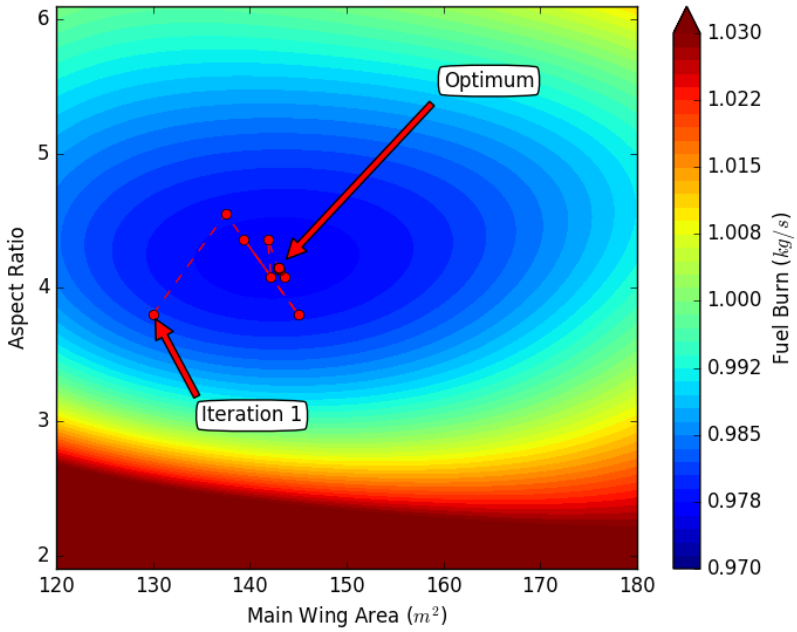

Fig. 16 Supersonic Jet TRMM Optimization Path

B. Blended-Wing-Body Transport

1. Aircraft Definition

The second configuration is a blended-wing-body design (BWB). This case illustrates SUAVE's flexibility in optimizing unconventional designs. This aircraft is geometrically quite different from conventional cases, and further demonstrates the need for higher fidelity methods. A more detailed analysis of the configuration can be seen in Reference Potsdam:1997. A BWB has also been previously analyzed with SUAVE[4].

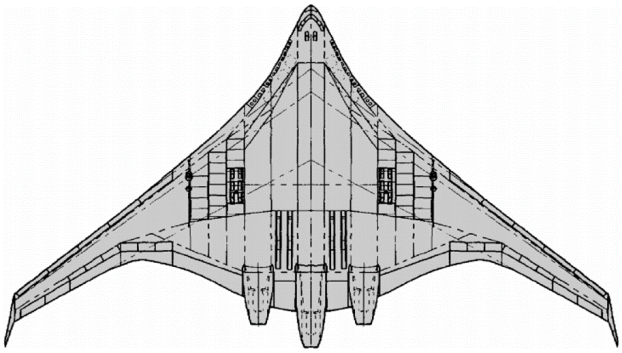

Fig. 17 BWB-450 Concept

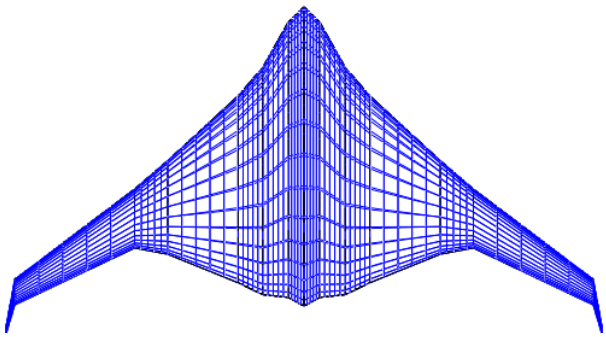

Fig. 18 SUAVE BWB

Table 2 Design Variables, Initial Values, and Bounds

Variable	Initial Value	Lower Bound	Upper Bound
Outboard Sweep [deg]	25	20	40
Outboard Twist [deg]	-1	-10	0

This BWB is set up to be similar to the BWB-450 concept[18]. It is a large commercial passenger aircraft meant for long haul missions. A reflex airfoil is used for the centerbody, and a symmetric airfoil is used on the outboard segments. Cruise conditions are set as similar to other commercial aircraft, with a speed of Mach 0.78 at the altitude of 35,000 feet. As with the previous case, only a point evaluation is done at the cruise condition.

2. Optimization and Methodology

In this case we are again optimizing for fuel burn, but this time the focus is on the constraints. The sweep and twist of the outermost horizontal section in Figure 18 are used as the design variables, with the coefficient of moment and the static margin as constraints. For the low-fidelity configuration, the changes in sweep and twist are averaged across the full vehicle. The coefficient of moment constraint is set to zero under the assumption that our design cruise point should not require trim from the control surfaces. We also require that the static margin be above 0 to avoid an unstable aircraft. The upper bound on sweep is chosen to avoid flutter and poorly captured weight effects of this type of wing.

The first level of fidelity here is a baseline vortex lattice model native to SUAVE. This model is limited in that it can only evaluate trapezoidal wings, and cannot return stability information. The stability information is instead calculated with a separate low-fidelity method[19]. The second level of fidelity is AVL, which can provide all of this information for a wing with multiple segments. Both fidelity levels use the same methods for drag, weight, etc. Plots of coefficient of moment for low and high fidelity are shown in Figures 19 and 20. Static margin is only computed with AVL here.

Table 3 Constraints for BWB Optimization

Constraint		Bound	
Coefficient of Moment	[-]	=	0
Static Margin	[-]	≥	0

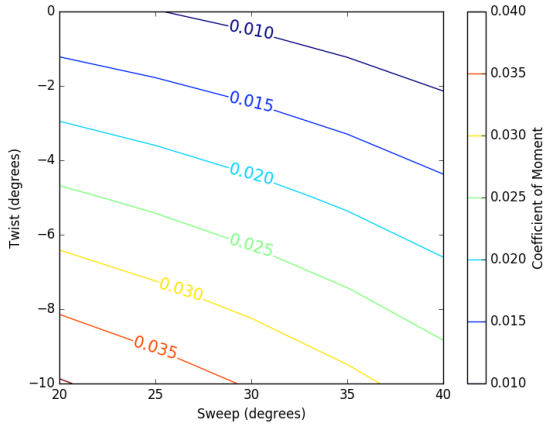

Fig. 19 Low-Fidelity Coefficient of Moment

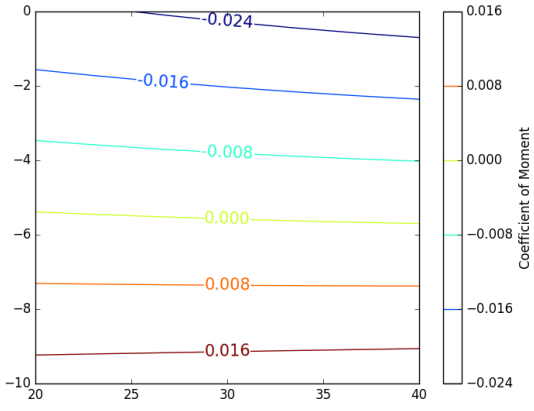

Fig. 20 AVL Coefficient of Moment

3. Results

In this case both optimizers come to the optimum values of sweep at 40 degrees and twist at -5.699 degrees. The fuel burn from the additive correction surrogate added to the low-fidelity results (rather than the direct AVL results) is shown in Figure 21, with points representing the evaluation points. In this case, a simple optimization of the corrected model was done for each iteration after initial sampling. The method required 13 higher-fidelity evaluations (3 after initial sampling) before reaching convergence at the level of 1e-6.

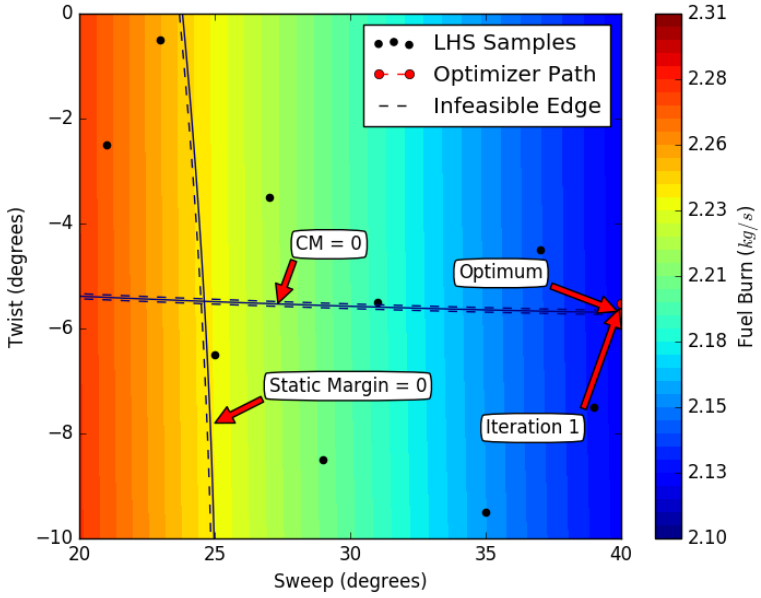

Fig. 21 BWB Additive Correction Optimization Path

For the TRMM, we start at the initial value listed in Table 2. The surrogate shown here is created directly from the AVL data. The trust region initially moves to minimize the coefficient of moment constraint. It changes direction when it crosses the stability margin constraint, and then moves along the coefficient of moment constraint to the optimum. This required 8 iterations with 24 high-fidelity function evaluations before reaching convergence at the level of 1e-5.

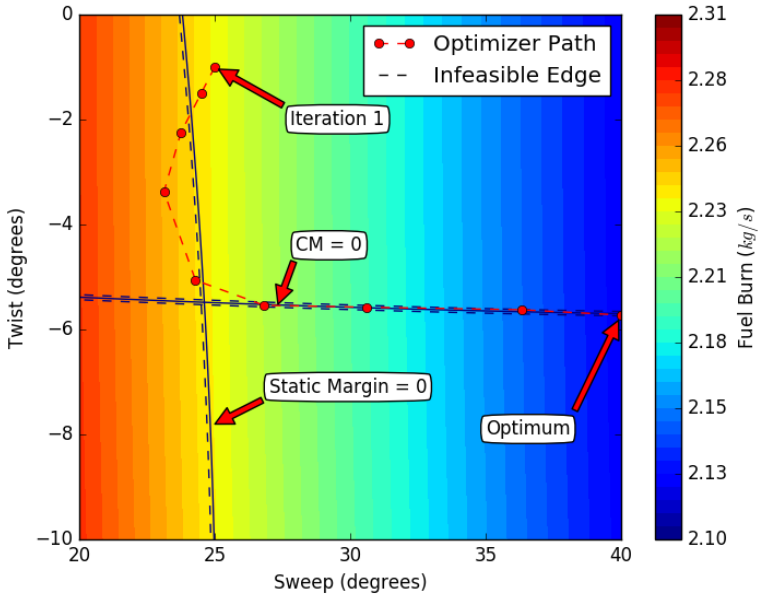

Fig. 22 BWB TRMM Optimization Path

V. Summary

This work demonstrates how the SUAVE framework is able to make use of multi-fidelity methods to optimize unconventional aircraft configurations. Focus is placed on using common multi-fidelity methodologies to build optimization schemes for these aircraft in a way that will be readily accessible by other users through this open-source design tool. In general, these methods and other multi-fidelity schemes are meant to allow the user to reduce computational cost for a given accuracy.

We have implemented two types of optimizers in this work. The first uses an additive correction surrogate to build a corrected low-fidelity model using the difference between a low-fidelity model and one of higher fidelity. The second uses a trust region model management approach to build corrected low-fidelity models and evaluate these models within regions where they are expected to be accurate. We demonstrate these methods, but also note that other methods can be added to SUAVE as needed without changing any of the analyses.

In order to show the ability to evaluate unconventional designs, we have evaluated two types of aircraft with these methods. The first was a supersonic business jet, which made use of two levels of fidelity in computing wave drag. The second was a blended-wing-body, which made use of two levels of fidelity to evaluate stability constraints. Different aircraft types and more levels of fidelity are also possible with minimal modification.

Overall, the multi-fidelity optimization setup requires selecting a optimizer and analysis for each level, but does not require significant changes to the underlying framework. This framework allows users to implement new analyses as needed while still being able to use all other SUAVE capabilities, and perform design studies based on these efforts.

Acknowledgments

The authors would like to acknowledge Rick Fenrich for providing TRMM Python code that was adapted to meet the requirements of this work, as well as assisting with the TRMM explanation.

Timothy MacDonald and Emilio Botero would like to acknowledge the support of the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program.

Michael Vegh would like to acknowledge the support of the DoD through the Science Mathematics and Research for Transformation (SMART) Scholarship Program.

References

[1] Drela, M., "Simultaneous Optimization of the Airframe, Powerplant, and Operation of Transport Aircraft," http://web.mit.edu/drela/Public/papers/RAeS/rt.pdf, May ????

[2] "CEASIOM: An Open Source Multi Module Conceptual Aircraft Design Tool," *International Journal of Engineering Research and Technology*, Vol. 2, No. 7, 2013.

[3] Boehnke, D., Nagel, B., and Gollnick, V., "An Approach to Multi-Fidelity in Distributed Design Environments," *IEEE Aerospace Conference*, 2011.

[4] Lukaczyk, T., Wendorff, A. D., Botero, E., MacDonald, T., Momose, T., Variyar, A., Vegh, J. M., Colonno, M., Economon, T. D., Alonso, J. J., Orra, T. H., and Ilario da Silva, C., "SUAVE: An Open-Source Environment for Multi-Fidelity Conceptual Vehicle Design," *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Dallas, TX, 2015.

[5] Botero, E., Wendorff, A. D., MacDonald, T., Variyar, A., Vegh, J. M., Alonso, J. J., Orra, T. H., and Ilario da Silva, C., "SUAVE: An Open-Source Environment for Conceptual Vehicle Design and Optimization," *54th AIAA Aerospace Sciences Meeting*, San Diego, CA, 2016.

[6] MacDonald, T., Botero, E., Vegh, J. M., Variyar, A., Alonso, J. J., Orra, T. H., and Ilario da Silva, C., "SUAVE: An Open-Source Environment Enabling Unconventional Vehicle Designs through Higher Fidelity," *55th AIAA Aerospace Sciences Meeting*, Grapevine, TX, 2017.

[7] Fredericks, W., "Aircraft Conceptual Design Using Vehicle Sketch Pad," *48th AIAA Aerospace Sciences Meeting*, Orlando, FL, 2010.

[8] Palacios, F., Colonno, M. R., Aranake, A. C., Campos, A., Copeland, S. R., Economon, T. D., Lonka, A. K., Lukaczyk, T. W., Taylor, T. W. R., and Alonso, J. J., "Stanford University Unstructured (SU2): An open-source integrated computational environment for multi-physics simulation and design," *51st AIAA Aerospace Sciences Meeting and Exhibit*, Grapevine, TX, 2013.

[9] McDonald, R., and Waddington, M., "Interactive Wave Drag in OpenVSP," , August 2015.

[10] Drela, M., and Youngren, H., "AVL," http://web.mit.edu/drela/Public/web/avl/, May 2017.

[11] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.

[12] McKay, M. D., Beckman, R. J., and Conover, W. J., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, No. 2, 1979, pp. 239–245. URL `http://www.jstor.org/stable/1268522`.

[13] Jansen, P., and Perez, R., "Constrained structural design optimization via a parallel augmented Lagrangian particle swarm optimization approach," *Computers and Structures*, Vol. 89, No. 1314, 2011, pp. 1352 – 1366.

[14] Perez, R., Jansen, P., and Martins, J., "pyOpt: A Python-Based Object-Oriented Framework for Nonlinear Constrained Optimization," *Structures and Multidisciplinary Optimization*, Vol. 45, No. 1, 2012, pp. 101 – 118.

[15] Gill, P., Murray, W., and Saunders, M., ""SNOPT: An SQP Algorithm for Large-Scale Constrainted Optimization," *Society for Industrial and Applied Mathematics*, Vol. 47, No. 1, 2005, pp. 99 – 131.

[16] Waddington, M. J., "Development of an Interactive Wave Drag Capability for the OpenVSP Parametric Geometry Tool," Master's thesis, California Polytechnic State University, San Luis Obispo, 2015.

[17] "AS2 Performance Objectives and Specifications," http://www.aerionsupersonic.com/technical-specifications, May 2017.

[18] Liebeck, R. H., "Design of the Blended Wing Body Subsonic Transport," *Journal of aircraft*, Vol. 41, No. 1, 2004, pp. 10–25.

[19] Raymer, D. P., *Aircraft Design: A Conceptual Approach*, 4th ed., American Institute of Aeronautics and Astronautics, 2006.