AIAA SciTech
4-8 January 2016, San Diego, California, USA
54th AIAA Aerospace Sciences Meeting

SUAVE: An Open-Source Environment for Conceptual Vehicle Design and Optimization

Emilio Botero*, Andrew D. Wendorff†,
Timothy MacDonald‡, Anil Variyar‡, J. Michael Vegh‡,
Trent Lukaczyk‡, and Juan J. Alonso§
Stanford University, Stanford, CA 94305, USA

Tarik H. Orra¶, Carlos R. Ilario da Silva‖
EMBRAER, São José dos Campos, SP, 12277-901, Brazil.

SUAVE, a conceptual level aircraft design environment, incorporates multiple information sources to analyze unconventional configurations. Developing the capability to produce credible conceptual level design conclusions for futuristic aircraft with advanced technologies is a primary directive. This work builds upon previous work where SUAVE analyzed aircraft to show how SUAVE may be integrated into external packages to optimize aerospace vehicles.

In the context of optimization, SUAVE operates as a "black-box" function with multiple inputs and multiple outputs. Several convenient functions are provided to enable connecting the optimization packages to SUAVE more easily. Assuming an optimization algorithm is minimizing an objective subject to constraints by iteratively modifying input variables, SUAVE's code structure is general enough to be driven from a variety of optimization packages. To this point, connections to PyOpt and SciPy have been integrated into SUAVE.

We present results for a multi-mission regional aircraft, a family of UAVs and a tradeoff between noise and fuel burn on a large single-aisle aircraft. These designs show the immense amount of flexibility and diversity that SUAVE can handle. This includes various levels of fidelity. While SUAVE is setup from the beginning to handle multi-fidelity analysis, further study is necessary to integrate multiple fidelity levels into a single vehicle optimization.

I. Introduction

SUAVE, a conceptual level aircraft design environment, was created with the goal of providing an environment for both the analysis and the optimization of aerospace vehicles that would be compatible with exotic configurations. In prior work, a brief description of some of the analysis capabilities of the environment was presented.[1] In this current work, the vision, development, and application to optimization are detailed. We believe that SUAVE's setup will provide flexibility in aircraft study beyond traditional conceptual design limitations.

Examination of current aircraft design tools left room for increased generality and flexibility in optimization methodologies. SUAVE's initial code was written for use with multiple information sources and extreme flexibility. Any parameter that defines the vehicle (the configuration of the vehicle, the mission, or even the analysis) can be exposed to an optimizer. When combining the available optimization variables with the analysis flexibility, new paradigms in aircraft design can be explored. For example, a single aircraft can be

*Graduate Student, Department of Aeronautics and Astronautics, AIAA Student Member.
†Graduate Student, Department of Aeronautics and Astronautics, AIAA Student Member.
‡Graduate Students (authors in alphabetical order), Department of Aeronautics and Astronautics, AIAA Student Members.
§Professor, Department of Aeronautics and Astronautics, AIAA Associate Fellow.
¶Aircraft Conceptual Design Engineer - Embraer, AIAA Senior Member.
‖Technology Development Engineer - Embraer, AIAA Associate Member.

optimized under multiple missions and configurations concurrently, or a family or even a fleet of aircraft can be optimized simultaneously.

A core ideal of SUAVE's development is maintaining robustness during optimization. The analysis routines are coded in such a way that arbitrary inputs should always provide smoothly varying results. For robustness, an output should always be provided back to an optimizer even when the optimizer's attempted inputs are physically infeasible. This informs the optimization algorithm at every input and prevents failure of gradient-based optimizers. This requirement necessitates careful examination of the underlying code to ensure that analyses will always provide finite numerical results provided the inputs meet broad specifications.

Another key feature of SUAVE is the use of underlying physical principles to analyze exotic and unconventional vehicles. In these radical designs, the intuition of the classically trained designer starts to fail as the design space is shifted dramatically, and as a result handbook methods or rules of thumb become unreliable. Building intuition for this new design space requires both a tool such as SUAVE, and the capability to drive it in anyway the designer deems necessary. This could potentially include outside optimization routines or uncertainty quantification tools. This is not to say that this code is only applicable to yet undreamed of concepts. In fact, this environment is just as proficient in performing fundamental analysis on traditional aircraft. A designer can use these tools to examine and further improve these more traditional designs.

SUAVE can either be treated as a stand-alone analysis tool or combined with external tools for optimization. When combined with external tools, SUAVE acts as a "black-box" function. The external package could interrogate the function to provide gradients on a setup, optimize the setup, furnish a design space with constraint curves, perform uncertainty quantification, etc. The optimizer could be gradient based or even a population method such as a genetic algorithm. Further, several instances of these black-box functions can be instantiated to provide parallel finite difference gradient calculations. Most importantly, none of this requires advanced programming ability or extensive coding time to complete. All necessary templates are provided. This capability is important to the developers, as they hope aerospace designers will be able to devote their time to designing and studying new vehicles instead of writing code.

This paper is organized as follows: Section II provides the background of the code as well as the code structure. Section III contains the status of the analysis capabilities within the environment. Section IV describes the formulation of the optimization problems from the SUAVE perspective. Section V overviews the various optimizers that have been successfully connected to the environment. Section VI presents three optimization cases: a fuel burn optimization for regional transport with additional mission constraints, a feasibility study for a family of UAVs, and a comparison of optimal aircraft for two objectives of a 180 passenger single-aisle aircraft considering fuel burn and noise. Section VII contains a summary of current capabilities and outline of future development.

II. SUAVE Background

Robust and flexible aircraft optimization requires carefully built evaluation functions. SUAVE was constructed with this goal in mind. A more detailed discussion of the ideas presented here may be found in Reference 1.

A. Code Design Principles

To be capable of examining unconventional configurations at several levels of fidelity, we submit that a mission-level design code requires *flexibility*, *composability*, and *extensibility*. To implement these principles, Python was chosen for its combination of object oriented programming, duck typing, concise language, portability in open source, and the community standard as glueware.

We consider flexibility to be the ability for necessary changes to be locally confined. This means that a user will not need to dig through base classes in order to make changes necessary for a different implementation that utilizes existing modules. This flexibility was created by basing data structures on the Python dictionaries, allowing data to be added, removed, and modified without directly modifying code within the classes.

In order to design for high composibility, or the ability to create evaluation objects from a group of smaller objects, the code again relies on the data dictionary. This is structured to enable any number of analyses to add data to an evaluation. This means that any set of analyses can be created for a given vehicle and the results of those analyses can be made available for easy examination or further analysis.

We then consider extensibility. This refers to the ability of the code to grow. For the code to be extensible, the ability to add new functionality to the code without requiring significant changes to the underlying structure and data flow is necessary. SUAVE provides this by maintaining common interfaces for different types of analysis and by making use of traditional object-oriented principles like class inheritance.

One last design principle important for the multi-fidelity capability is attribute-method orthogonality. Attributes describe the characteristics for given components, such as wing geometry, while methods determine how those attributes are manipulated. This allows a given set of attributes to be evaluated at different levels of fidelity without requiring different configuration attributes.

B. Attributes

SUAVE relies on attributes to store data at every part of a vehicle analysis. Some important data structures in the current framework are Vehicle(), which stores all fixed parameters of a vehicle, and Conditions(), which stores anything that could fall under conditions in the broadest sense, such as freestream velocity or aircraft orientation. SUAVE missions are also built with attributes, allowing greater flexibility in analysis.

Another particularly important use of attributes is in the construction of energy networks, which are built in the Vehicle() data structure. These networks are composed of a series of components that determine how energy is generated, stored, and used in the aircraft. The flexibility of attributes here allows these networks to be expanded and condensed as desired for given levels of fidelity.

C. Methods

Methods perform actions on SUAVE attributes. By maintaining this separation, extensibility is permitted in the number of fidelity levels available in SUAVE. This hierarchy of fidelities needs to be organized in a useful way. Information sources are organized by discipline. They are given a descriptive name where possible. Additionally, they require common interfaces, which vary by discipline. However, they often must, at minimum, be able to accept the conditions data and manipulate it appropriately. In addition, multiple fidelities can be composed into higher level analyses. This structure allows low-fidelity and high-fidelity results to be combined in the same analysis as desired by the user. It also allows a small number of high-fidelity points to be calculated and applied to low-fidelity results with simple corrections. These compositions can be expressed in the top level run script, which enables rapid prototyping of functionality.

III. Analysis Capabilities

A. Base Functionality

SUAVE is currently a fully functioning aircraft conceptual design tool. Every major analysis discipline of aircraft design is represented in the code. A full break down of the analysis capabilities of SUAVE is described in Reference 1. For the sake of brevity, a partial list of the current capabilities is below:

- Aerodynamics for subsonic and supersonic flight
- Weight correlations for tube-and-wing, blended wing body, and human powered aircraft
- Segment based mission architecture
- Static and dynamic stability metrics
- Propulsion models for gas turbines and electric ducted fans
- Energy networks for battery, fuel-cell, and solar-panel based vehicles
- Noise correlations for tube-and-wing aircraft
- Basic performance estimation methods such as takeoff and landing field lengths and payload range diagrams

Some of these capabilities are also available in multiple levels of fidelity. For example, in the propulsion networks, different motor models exist from the most basic efficiency analysis to more complex analysis using manufacturer motor specifications. The open source nature of SUAVE lends the ability to contribute new analysis capabilities for future aircraft designs. Additionally, new capabilities are added as necessary for research requirements.

B. New Capabilities

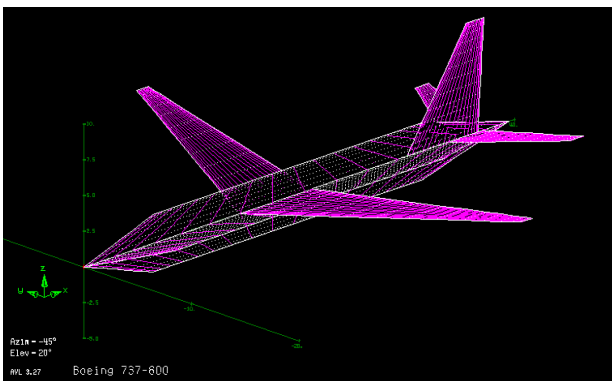
1. Athena Vortex Lattice

One of the main goals behind the development of SUAVE is the ability to perform design/optimization of unconventional aircraft configurations. However, the design of unconventional configurations requires being able to use higher fidelity analysis tools for the different disciplines when necessary. One of the first such advances in SUAVE is setting up an interface with Athena Vortex Lattice (AVL) which is a vortex lattice code developed for the "aerodynamic and flight-dynamic analysis of rigid aircraft".[2] Using AVL during the design process enables us to handle complex aircraft configurations and obtain accurate estimates of the lift, lift-induced drag, and stability derivatives for configurations in an automated fashion.

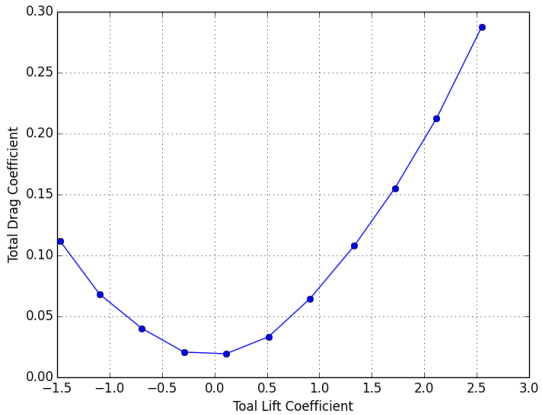
The AVL interface is implemented as a callable class similar to the other aerodynamics models implemented in SUAVE. This ensures that the AVL interface can easily be used as an analysis module.

The aircraft configuration stored in the vehicle data structure in SUAVE is passed in to the AVL interface along with the atmospheric conditions (Mach number, altitude) and the angle of attacks at which the model needs to be run. The interface first processes the geometry (fuselages and wings), and then generates the geometry input file for AVL. Figure 1(a) shows the AVL geometry generated automatically by the SUAVE-AVL interface visualized in AVL. The interface also processes the conditions and generates the input deck and a batch input file containing the cases that need to be run in AVL. AVL is then executed using system calls from the function and the commands in the input deck file are passed to the analysis program. Once AVL has finished running, the resulting file generated by AVL is parsed automatically and the results are stored into the SUAVE data structure.

Running AVL for each point in the mission might become extremely expensive. However, because the interface to AVL has been set up to resemble the other aerodynamics modules in SUAVE, the AVL model can be run as a pre-processing step for a set of angles of attack. The lift coefficient, induced drag coefficient, and moment coefficients obtained can be regressed to build surrogates without modifying the existing SUAVE structure. These surrogates can then be evaluated at the mission points to produce aerodynamic performance estimates inexpensively. The drag profile of the B737 computed using AVL is shown in Figure 1(b).



(a) Automatically Generated Boeing 737-800 Input Geometry for AVL



(b) Drag Polar for the Boeing 737-800 Obtained Using SUAVE-AVL

Figure 1: Results Obtained Using the SUAVE-AVL Framework

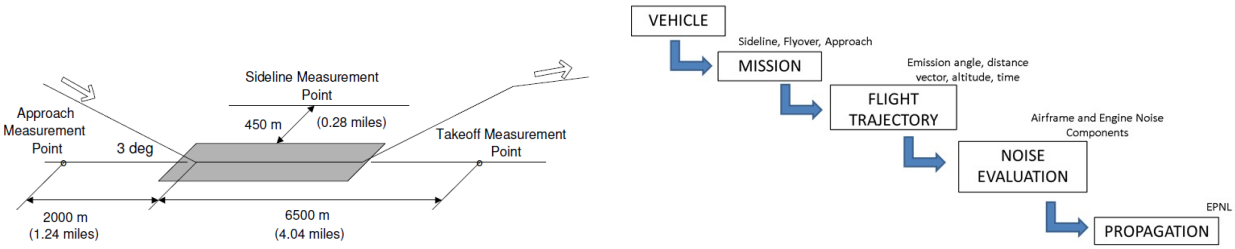
2. *Aircraft Noise*

External aircraft noise remains an important challenge not only for the next generation aircraft, but also for current airplanes in service. This is due to stricter noise regulations as well as airport restrictions, which are constantly under discussion. Therefore, the ability to predict the external noise level based on aircraft geometries, engine parameters, and mission performance is crucial for a conceptual design tool. In this sense, SUAVE now has the capability to predict the total Effective Perceived Noise Level (EPNL) for the three noise certification points (sideline, flyover, and approach), as presented in Figure 2(a). The method considers the total noise as a summation of the two main sources below:

- Airframe Noise: Fink's methodology[3] is applied to compute the sound spectra as a function of frequency, azimuth, and polar directivity angles. The airframe sources are divided into wing, tails, landing gears, flaps, and slats. The method calculates the one-third octave band sound pressure level for each component and then adds the acoustic pressure to obtain the sum of noise from all components.

- Engine Noise: The SAE ARP876D model[4] is used to predict the noise from the gas turbine jet exhaust system. In this model the jet is conceptually divided into three source regions: the primary (core) jet, the secondary (fan) jet, and the mixed jet. The total jet sound pressure level is a log base 10 of the sum of the time-mean-square sound pressures from the three aforementioned components. The engine model at this stage does not take into account the turbomachinery and combustion noise sources.

Once the near-field sound spectra is computed for each noise source, the noise module propagates them to the far-field desired position by applying corrections, such as atmospheric attenuation. Finally, the time-averaged EPNL values are computed. Figure 2(b) presents the calculation procedure for the external certification noise points.



(a) ICAO Certification Noise Measurement Points

(b) Procedure for Predictions with SUAVE Noise Module

Figure 2: External Noise Certification Predictions with SUAVE Noise Module

IV. Organization for Optimization

In the development of SUAVE, one of the major objectives was to make it flexible enough to interface with a multitude of different optimization packages. To adapt SUAVE to all the desired optimization programs, each optimization package must treat SUAVE as a "black-box" where the internal programs and processes remain unchanged. To formulate SUAVE as a black-box, the user must define all of the inputs, the connections between vehicles and missions, as well as the outputs to be returned. This allows the user to not only optimize a single vehicle for a single mission, but also optimize an aircraft over multiple missions, or optimize a fleet of vehicles for a set of missions. For instance, one can minimize fuel burn over one mission, while constraining it to be able to fly a given distance when loaded with more fuel, and be able to fly a given distance from a specified fuel. In addition, SUAVE allows design parameters, specified by the user, to map to their corresponding parameters inside the code. The general mathematical formulation can be written as

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & g_j(\mathbf{x}) = 0 \quad j \in \{1, ..., l\} \\ & h_k(\mathbf{x}) \leq 0 \quad k \in \{1, ..., m\} \\ & lb_i \leq x_i \leq ub_i, \ i \in \{1, ..., n\} \\ & \mathbf{x} \in \mathbb{R}^n \end{aligned} \tag{1}$$

where $\mathbf{x}$ is a vector containing n design variables x_i, which are each bounded by lower and upper bounds lb_i and ub_i. The objective of interest is $f(\mathbf{x})$. There are l equality constraints $g(\mathbf{x})$ and m inequality constraints $h(\mathbf{x})$. The design variables $\mathbf{x}$ would be some subset of the inputs to SUAVE and wrapping functions are provided to enable translation between data dictionaries and design vectors.

A. Code Inputs

When determining the inputs to SUAVE, the parts to which the inputs can be broken into are: vehicle inputs, mission inputs, analyses, vehicle-mission connections, procedure, and variable setup. By determining what inputs are specified and what missions are performed, the engineer will define what type of problem is being analyzed. Part of the code inputs would be the design variables of interest, but others are just the information required to setup SUAVE to run the desired analyses.

1. Vehicle

Within the vehicle inputs, the designer must first choose what type or types of configurations SUAVE will study. Does the designer want to optimize a single aisle aircraft for a 1,000 nmi mission or a family of transoceanic aircraft sharing a common wing where one carries 300 passengers, one carries 350 passengers, and a third aircraft carries 425 passengers? Depending on the type of optimization desired, SUAVE needs to be configured to generate those results. Part of the code inputs is determining what fidelity level or levels will be used to analyze the configurations. A CFD code could have different input requirements than a vortex lattice code or even handbook methods. Making sure the necessary data is provided to SUAVE for the desired analyses is the user's responsibility.

2. Mission

Beyond just looking at different vehicles over the same mission, we designed SUAVE to be able to run the same aircraft through different missions. Instead of optimizing the single aisle aircraft for a 1,000 nmi mission and not considering other missions, one could optimize over a 1,000 nmi mission, but add a constraint that the maximum range of the aircraft be 2,500 nmi. Just as one must specify the parameters that define each vehicle, one must also build the missions from the different segments available or add new segments.

3. Analyses

The level of analysis fidelity for each discipline is chosen based on the desired accuracy and computational resources available. Various levels may be chosen, depending on the problem the user wishes to run. For example, different fidelities of aerodynamics models or noise methods. With the vehicles and the missions specified, these analyses are linked with the corresponding vehicles and missions.

4. Vehicle-Mission Connections

Once the designer has defined the vehicles under consideration and the missions the vehicles needed to fly, the connection between vehicles and missions needs to be specified. This can be done by creating different configurations of the same vehicle. For instance, in takeoff and landing, specifying flap and slat deployment (and thereby modifying the geometry), or specifying that only the 300 passenger aircraft will fly 8,200 nmi outlines how the vehicles created are matched with the prescribed missions. This step tells SUAVE to have aircraft-1 run missions 1, 2, 3 while aircraft-2 only flies missions 1 and 3. It also specifies what results SUAVE will generate when the full analysis is completed.

5. *Procedure*

The analysis of the problem requires a set of sequential actions to be performed. This is the procedure. A great example of this would be resizing the horizontal tail of the aircraft after a new wing area is selected by the optimization algorithm to keep the horizontal tail volume constant. Additionally, the types of missions are then set here such as a long range mission and short field takeoff missions. Finally the constraints and objectives that require additional non-standard calculations can be performed as part of the procedure. An example of a non-standard constraint is fuel margin, that is, fuel volume available in the vehicle minus the fuel used to run the mission divided by the maximum zero fuel weight.

6. *Variable Setup*

The optimization interface provides a concise way to define several important features of the optimization problem. Including variable names (or tags), the initial guess of the variable, the lower and upper bounds, how it should be scaled to yield favorable numerics within the optimizer, and finally its units. Using the information provided in a tabular structure like the one shown below, accepting input vectors becomes much simpler, enabling SUAVE to pattern across multiple optimization packages. To switch between optimization packages or algorithms, only one line of the setup script must be modified.

```
# [ tag , initial, [lb,ub], scaling, units ]
problem.inputs = [
    [ 'aspect_ratio'  ,   10.    , (    5.    ,    20.   ) ,   10.  ,     Units.less],
    [ 'reference_area',  125.    , (   70.    ,   200.   ) ,  125.  , Units.meter**2],
    [ 'sweep'         ,   25.    , (    0.    ,    60.   ) ,   25.  ,  Units.degrees],
    [ 'design_thrust' , 24000.   , ( 10000.   , 35000.   ) , 24000. ,   Units.Newton],
    [ 'wing_thickness',     0.11 , (     0.07 ,     0.20) ,      .11,     Units.less],
    [ 'MTOW'          , 79000.   , ( 60000.   ,100000.   ) , 79000. ,       Units.kg],
    [ 'MZFW'          , 59250.   , ( 30000.   ,100000.   ) , 59250. ,     Units.less],
]
```

Furthermore, within SUAVE we allow each design variable to be defined in any user preferred name and then "alias" it to the internal data structure name. For example, `aspect_ratio` above would be an alias of `problem.vehicle.wings.main_wing.aspect_ratio`. SUAVE uses a very verbose methodology, but if the engineer would like to use a different set of variable names for optimization, the functionality is in place. Outputs to be used for the objective function, constraints, and output characteristics of interest can also be defined in the same manner. This flexible naming convention also allows multiple parameters inside of SUAVE to be varied as one design variable in the optimization process. This capability reduces the number of variables and constraints since there are no longer multiple variables with constraints requiring that they be equal.

B. Code Outputs

1. *Results Summary*

After all the code inputs, specifying vehicle characteristics, mission profiles, and vehicle-mission connections have been provided and the desired SUAVE analysis structure is generated, results are produced. Not all of the code outputs are relevant to a particular optimization problem. The results of interest are put into a summary data structure. This data structure allows a designer to quickly assess the results. The code outputs might need to be post-processed to generate the actual result we care about for our problem. For example, if we are trying to meet Stage 4 noise levels, we care only about generating a cumulative total, not matching certain levels at each condition. The objective function and constraints should be a subset of the final code outputs produced.

2. *Data Management*

This setup structure puts all of the data into a Python class. This class is called "Nexus". The Nexus object produced by the class is what is called by optimization packages. A variety of methods exists to return

computed values depending on the format the optimizer prefers. When an optimizer queries these methods, the method expects a scaled input vector that matches the inputs set prior. If setup by the user correctly, the inputs and outputs are scaled to order one. This setup allows SUAVE to accommodate any foreseeable optimization package.

V. Optimizers

With a general interface in place, SUAVE can be incorporated into a number of different optimization packages. SUAVE currently possesses wrappers that interface directly only with pyOpt and SciPy. However, the interface is designed to be easily extensible to allow additional optimizer support.

A. PyOpt

PyOpt is a Python package containing a variety of nonlinear optimizers.[5] Inputs to pyOpt are handled via the format shown in Section IV. Two of pyOpt's optimizers have been used and tested with SUAVE. The first is a Sparse Nonlinear Optimizer (SNOPT) module, which relies on a Sequential Linear Programming algorithm and quasi-Newton methods.[6] The second is the Sequential Least Squares Programming (SLSQP) algorithm, another quasi-Newton method.[7] A variety of other optimizers from pyOpt have also been integrated into this framework, including not only other quasi-Newton methods, but also population-based methods. Nonetheless, they have not been thoroughly tested at the time of this writing. Interested users are encouraged to explore the features for themselves and report any issues to the development team.

B. SciPy

SUAVE is also capable of interfacing with SciPy's optimization toolbox.[8] Like pyOpt, inputs to SciPy's optimization package are handled using the format in Section IV. The SciPy optimization package as of the time of writing includes a wide variety of optimization algorithms, including a Nelder-Mead simplex algorithm, SLSQP, and conjugate gradient methods, among others.[7,9] The simplex method has been used to solve a few optimization problems, an example of which can be seen in Reference 10. Currently, SUAVE's SciPy wrapper can only directly handle constraints from its SLSQP optimizer. To use other optimizers from SciPy's library, the user must define properly-scaled penalty functions within the interface's procedure file that alter and return the objective to ensure that the problem is well-formulated.

VI. Design Studies

Building upon the capabilities shown in our previous work,[1] three design studies are shown. We present an optimized regional transport, a feasibility study for a family of solar UAVs, and a trade study for commercial aircraft looking at the tradeoffs between noise and fuel burn.

A. Regional Transport

For our first case, we decided to use a standard tube-and-wing configuration as these methods inside SUAVE are most mature and most thoroughly developed. Our initial aircraft is closely based off of the Embraer E-190[11] and incorporates constraints necessary to design a reasonable vehicles while minimizing the fuel burn of the aircraft over a 1000 nautical mile mission. The vehicle must also fly 2300 nautical miles at maximum takeoff weight and takeoff from a short field and travel 500 nautical miles. This case uses 13 design variables incorporating geometric characteristics of the wing, takeoff weight for the three missions, a ratio of maximum zero fuel weight to maximum takeoff weight, and design thrust at top of climb. We currently resize the horizontal tail based on the horizontal tail area to wing area ratio of our baseline aircraft, baseline horizontal tail area is 26 m^2. The methods used to analyze the vehicle are first order methods outlined in a prior publication.[1] All design variables, their initial conditions, optimized results found using SNOPT in PyOpt, upper and lower bounds, and objective results are shown in Table 1.

Looking at these results, we see the vehicle pushes up to the bounds for the taper and sweep. Also, looking at the constraints, shown in Table 2, we see the span constraint is limiting the wing aspect ratio for the wing area necessary.

Table 1: Transport Design Variables with Initial Conditions, Optimal Results, and Bounds

		Initial	Optimized Result	Lower Bound	Upper Bound
Wing Aspect Ratio	[-]	8.4	12.4	5	20
Wing Area	$[m^2]$	92	104	70	200
Wing Tip Twist	[°]	-2	-4.52	-12	12
Wing Sweep	[°]	23	30	0	30
Wing Taper	[-]	0.28	0.1	0.1	1
Wing t/c	[-]	0.11	0.116	0.07	0.2
Design thrust	[N]	42500	36000	10000	70000
Maximum Takeoff Weight	[kg]	56200	56100	20000	100000
MZFW Ratio	[-]	0.765	0.799	0.7	0.99
Short Field Takeoff Weight	[kg]	48000	48600	20000	100000
Design Takeoff Weight	[kg]	50000	50600	20000	100000
Design Mission Fuel Burn	[kg]	4630	4070	-	-

Table 2: Constraints for Optimization Study

Constraints	Optimized Result	Bound		
MZFW Consistency	9.49e-7	$\geq$	0	[-]
Short Field Fuel Margin	0.0391	$\geq$	0	[-]
Design Range Fuel Margin	-1.01e-6	$\geq$	0	[-]
Maximum Range Fuel Margin	-1.01e-6	$\geq$	0	[-]
Wing Span	118	$\leq$	118	[ft]
Flyover Noise	91.5	$\leq$	92.5	[dB]
Sideline Noise	89.0	$\leq$	95	[dB]
Approach Noise	90.9	$\leq$	99	[dB]
Maximum Range Takeoff Length	1500	$\leq$	2056	[m]
Short Field Takeoff Length	1165	$\leq$	1165	[m]
Landing Field Length	1630	$\leq$	1700	[m]
Second Segment Max Range Climb Gradient	5.14	$\geq$	2.4	[%]
Second Segment Short Field Climb Gradient	7.35	$\geq$	2.4	[%]
Maximum Throttle	0.971	$\leq$	1	[-]

As one might expect, the takeoff weight margins approach zero since the optimal aircraft will not carry more fuel than necessary to complete the specified mission along with the necessary reserve requirements. The maximum zero fuel weight consistency constraint also approaches the bound since any excess weight would be payload not taken credit for in the objective. Additionally, we see the maximum throttle moves to close to 1, but not exactly 1 as would be expected as we use the top-of-climb point to design the vehicle, but the vehicle critical throttle condition is on takeoff which is not part of the mission. We have set the flap takeoff angle to 20 degrees and the landing flap angle to 30 degrees. If the takeoff flap angle were increased in a future optimization, then the design thrust could potentially be reduced. Comparing the drag breakdown of the baseline aircraft during the design mission shown in Figure 3 with the drag breakdown of the optimal aircraft profile shown in Figure 4, the compressibility drag and induced drag have been significantly reduced.

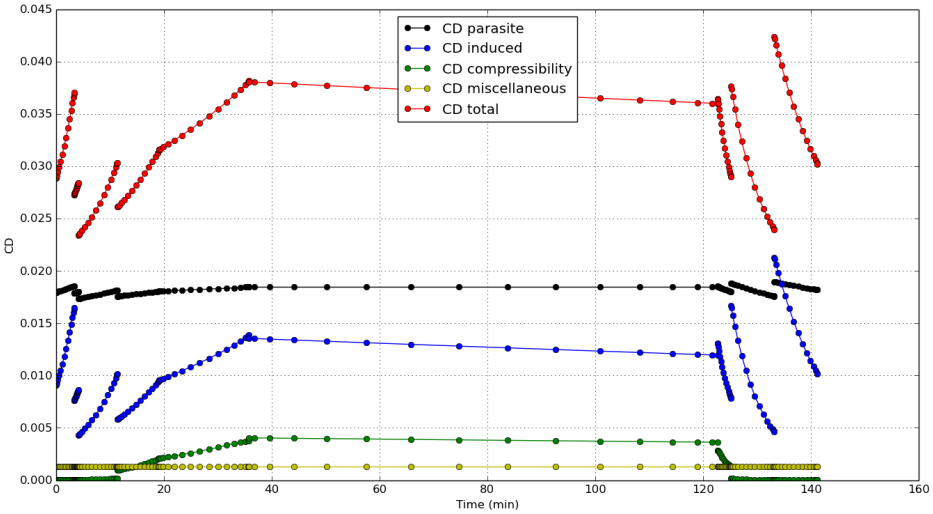


Figure 3: Baseline Aircraft Drag Breakdown

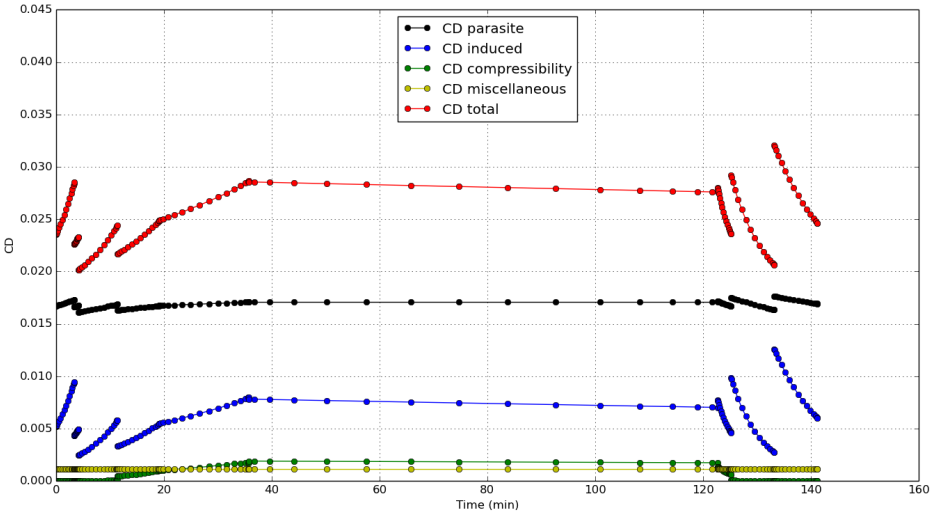


Figure 4: Optimal Aircraft Drag Breakdown

This is expected with the additional span and sweep added to the wing. With these changes, the maximum drag coefficient at any point is reduced by approximately 21%. Since the wing area has been increased, this drag value will not directly correspond to the fuel burn reduction, but having a 12% reduction in fuel burn seems reasonable.

Starting from an initially feasible design, the optimizer tries to obtain a more efficient aircraft with lower fuel burn while meeting the constraints including the gate constraint and the takeoff and landing field length constraints. This results in an increase in the main wing aspect ratio in order to reduce the induced drag. The increase in wing aspect ratio results in an increase in the main wing area. This causes the thickness-to-chord ratio of the wing to increase, ensuring that the wing weight does not increase significantly which causes the wing sweep to increase in order to reduce the compressibility drag of the aircraft. These design decisions made automatically by the optimizer match the basic design intuition which serves as a further validation of the results.

Overall, the results we see from this regional aircraft optimization match general trends one would expect.

We do realize, however, the need to add additional constraints for stability of the aircraft among others along with design variables for wing location, horizontal tail area, and flap and slat deflections. This case illustrates how SUAVE can be used in the optimization of a traditional tube-and-wing configuration flying multiple missions with a single aircraft.

B. Family of High Altitude Long Endurance Solar UAVs

Within SUAVE it is possible to instantiate different configurations of a single base vehicle. This ability is incredibly powerful for a user who would like to design a family of aircraft, like the Airbus A320 series. Further, one can use this even to instantiate completely different vehicles and analyze them together for a fleet.

To illustrate this we have chosen an atypical aircraft design, a solar UAV family. The analysis modules used in the optimization are outlined in a previous paper.[1] The family of vehicles has similar wing sections and motors, but differs in the number used. This concept was used on the NASA Helios as it evolved with added wing sections.[12] However, from the beginning, we will design two vehicles with differing requirements using the same hardware. This could allow an operator to meet the needs of different customers without purchasing an additional vehicle, or allow for more spare parts.

The vehicles had various requirements. First both vehicles must be capable of operating continuously throughout the tropical regions. Second, it must not conflict with normal commercial air traffic, therefore the loiter altitude was chosen to be 15 km. The smaller vehicle, "Vehicle A", carries a 25 kg payload that uses 50 watts of power. The larger vehicle, "Vehicle B", carries a 50 kg payload that requires 100 watts of power. Both vehicles also require 25 watts continuously to power avionics. Vehicle A has one motor and four wing sections, while Vehicle B has two motors and seven wing sections.

The mission is at constant dynamic pressure and uses a constant altitude loiter segment. Using dynamic pressure rather than velocity is convenient as it directly sizes the structure, and is a more useful quantity for designing a Solar UAV. The mission spans 24 hours, and is initialized at the Tropic of Cancer on the Winter Solstice. This is the most difficult day and location to fly a solar UAV within the Tropics.

Several assumptions to efficiencies are made. The motor is assumed to be 90% efficient and the propellers operate at 85% efficiency. These values were chosen after the designer had run many higher fidelity simulations to determine reasonable values for this type of vehicle. The solar panels are assumed to be 25% efficient and weigh 0.6 kg/m^2. The battery energy density is assumed to be 350 W-hr/kg. The design variables with bounds are shown in Table 3 with the optimized results.

Table 3: Design variables, optimized results, and bounds for the Solar UAV Family

Variable	Optimized Result	Lower Bound	Upper Bound
Section Chord Length [m]	2.04	0.01	10.0
Section Span [m]	10.1	1.0	12.0
Vehicle A Dynamic Pressure [Pascal]	45.0	1.0	1000
Vehicle B Dynamic Pressure [Pascal]	55.8	1.0	1000
Vehicle A Initial Charge [%]	26.0	1.0	100
Vehicle B Initial Charge [%]	27.7	1.0	100
Vehicle A Total Mass [kg]	363	50	1500
Vehicle B Total Mass [kg]	824	100	1500
Panel Area Ratio [%]	78.2	10.0	95.0

The vehicles are not optimized for an objective. Rather they are run through an optimizer to find a feasible vehicle using the constraints. In this case a designer may not have the intuition to select initial parameters that will produce a feasible vehicle based on prior experience. For example, with the prior Regional Transport aircraft the initial conditions were an existing regional jet that met constraints. A designer could then choose an characteristic he or she would want to optimize.

For these vehicles the constraints were mostly energy related. First, the energy in either vehicles battery cannot go to zero at any point during flight. Second, the energy at the end of the mission must be the same

or greater than the beginning of the mission. This ensures limitless endurance. The battery mass must also be greater than zero. This is applied as the battery mass is sized upon every iteration as the remaining mass after all other components. Finally the coefficient of lift of the wing must be less than 1.2.

The zero energy constraint and coefficient of lift constraints are formulated using a p-norm function. This was done to introduce smoother gradients than the using a maximum function for looking at the maximum value in a vector. The coefficient of lift, had the desired 1.2 subtracted off before, to scale it to zero. A feasible design will have both of these constraints at zero. The results of the constraints are shown in Table 4. The results are converged as far as the optimizer is concerned.

Table 4: Constraints for Optimization Study

Constraints	**Optimized Result**	**Bound**		
Vehicle A Zero Energy	0.0	$\leq$	0	[-]
Vehicle B Zero Energy	1.44e-07	$\leq$	0	[-]
Vehicle A Continuity	0.033	$\geq$	0	[-]
Vehicle B Continuity	-5.10e-07	$\geq$	0	[-]
Vehicle A Battery Mass	158	$\geq$	0	[kg]
Vehicle B Battery Mass	251	$\geq$	0	[kg]
Coefficient of Lift	0	$\leq$	0	[-]

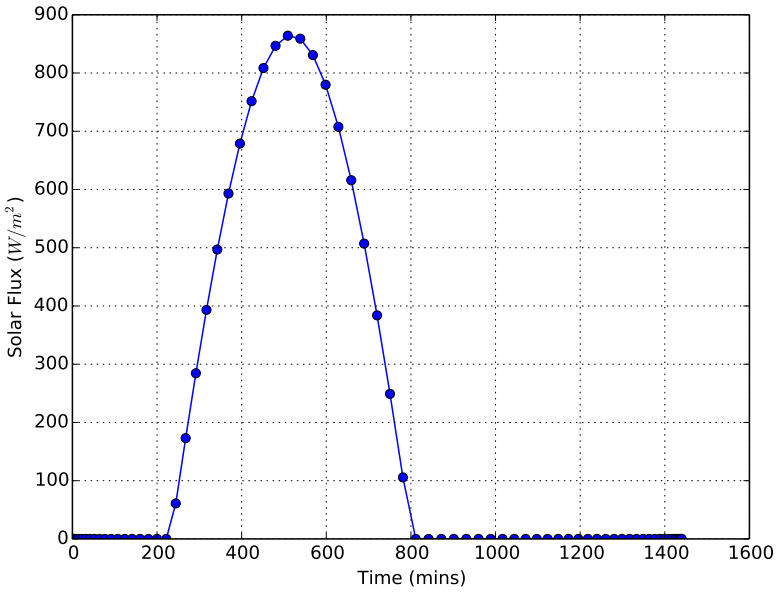


Figure 5: Solar Flux

The results showed that the battery mass constraints and the coefficient of lift constraints were not active at the feasible point. The solar flux available to the vehicle is rather the limited in magnitude and duration on the winter solstice as can be seen in Figure 5. The maximum value of 854 W/m^2, is far less than the standard AM0 value of 1367 W/m^2.[13] For the altitudes prescribed, the velocities that correspond to these dynamic pressures are 21.5 and 23.9 m/s, respectively. The battery charge states throughout the cycle can be seen in Figures 6 and 7. The energy required for the larger vehicle is about double that of the smaller one.

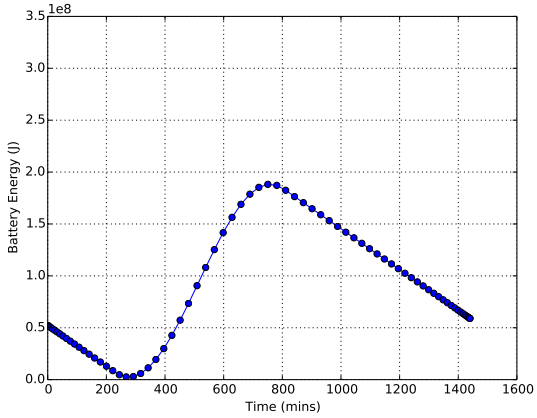


Figure 6: Vehicle A Battery Energy

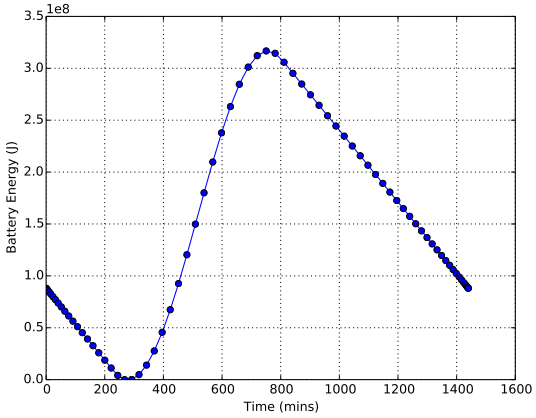


Figure 7: Vehicle B Battery Energy

C. Fuel Burn-Noise Single-Aisle Airliner

The last optimization case investigated in this work is aimed at including noise not only as a design constraint but also as an objective function by incorporating the high fidelity model, now available within SUAVE, for aircraft noise. The idea is to start from a reference aircraft and optimize it for two different cases: fuel burn and noise. Our baseline aircraft is based on a 180 passenger, single-aisle aircraft. The optimization procedure integrates disciplinary analyses including mission performance, aerodynamics, and component weights.

All design variables (their initial conditions from the reference airplane and optimized results for both the fuel burn and noise cases, found using SNOPT in PyOpt) and the objective results are shown in Table 5. Two specific design variables were included specially for the noise optimization study: takeoff speed increase and the thrust cutback altitude, which influences mainly the flyover noise point. The constraints for the noise optimization studies are presented in Table 6. In addition to traditional performance constraints such as field length performance and fuel margins, the maximum cumulative certification noise is included.

Table 5: Noise Evaluation Case: Design Variables with Initial Conditions, Optimal Results, and Bounds

		Reference airplane	Fuel Burn Case	Noise Case
Wing Area	$[m^2]$	124.8	133.0	159.1
Aspect Ratio	[-]	10.2	9.7	8.1
Wing Sweep	[°]	25.0	35.0	31.7
Wing t/c	[-]	0.105	0.111	0.115
Design Thrust	[N]	52700	48495	49545
MTOW	[kg]	79090	77465	77660
MZFW Ratio	[-]	0.770	0.786	0.777
Takeoff Flap Angle	[deg]	10	20	13
Landing Flap Angle	[deg]	40	40	3
Short Field Takeoff Weight	[kg]	64030	63674	63320
Design Takeoff Weight	[kg]	68520	67827	67644
Takeoff Speed Increase	[knots]	10	10	10
Noise Cutback Altitude	[ft]	305	305	382.2
Fuel Burn in Design Mission	[kg]	8820.3	8275.9	8611.0
Noise Cumulative Margin	[dB]	12.7	17.0	22.9

As can be seen in Table 5, when fuel burn is the objective function, the wing area and takeoff flaps are

Table 6: Constraints for Noise Optimization

		Bound		Initial	Fuel	Noise
MZFW Consistency	[kg]	$\geq$	0	7	0	0
Design Range Fuel Margin	[kg]	$\geq$	0	50	-11	9
Short Field Fuel Margin	[kg]	$\geq$	0	47	-11	10
Maximum Range Fuel Margin	[kg]	$\geq$	0	41	-13	0
Wing Span	[m]	$\leq$	35.97	35.97	35.97	35.97
Flyover Noise	[EPNdB]	$\geq$	0	3	4	7
Sideline Noise	[EPNdB]	$\geq$	0	5	7	7
Approach Noise	[EPNdB]	$\geq$	0	4	5	9
Cumulative Noise Margin	[EPNdB]	$\geq$	10	12	16	23
Maximum Range Takeoff Length	[m]	$\leq$	1985	2179	1929	1611
Landing Field Length	[m]	$\leq$	1385	1382	1298	1385
Second Segment Max Range Climb Gradient	[-]	$\geq$	0.024	0.054	0.037	0.025
Second Segment Short Field Climb Gradient	[-]	$\geq$	0.024	0.088	0.067	0.057
Maximum Throttle	[-]	$\leq$	1	1	1	1
Short Field Takeoff Length	[m]	$\leq$	1330	1441	1330	1118

increased compared to the initial aircraft to compensate for the reduced engine size. The wing thickness is also increased, to increase the maximum lift coefficient. This has required additional wing sweep in order to overcome the increase in the transonic drag. The aspect ratio is reduced to comply with the gate constraint, which limits the maximum allowed wing span.

When we look for the noise as the objective function, the wing area is further increased, in order to reduce the landing flap position, which plays an important role in the approach noise point. There is a trade-off between flap noise and wing area related noise. The wing sweep is reduced compared with the fuel burn case due to the need of higher maximum lift coefficient, so the landing speed is reduced and as consequence also the approach noise. Again, the aspect ratio is reduced to comply with the gate constraint. This reduction leads to a lower climb gradient, leading to higher engine thrust requirement. For the takeoff performance, it is noticed that the noise case presents better performance than the requirement. For noise purposes, the early takeoff allows the airplane to fly over the microphone at a much higher altitude, reducing the certified noise. This can also be verified by the takeoff flap position, which presents a trade-off between taking off soon and passing high over the microphone or taking off with a smaller flap deflection and a lower altitude, resulting in lower flap noise.

For completeness, Table 7, Table 8, and Table 9 show the influence of the airframe and engine noise sources for each certification point and for the optimization cases presented. In all cases, the airframe component is the major source for the approach condition whereas the engine component is the principal source for the sideline point. This results are only intended to corroborate the correctness of the physical aspects of the noise model. It is also possible to verify that due to the increased designed thrust from the fuel burn case, the sideline noise level has also increased compared to the noise case.

Table 7: Initial case: Source components breakdown

		Airframe	Engine	Total
Sideline	[EPNdB]	76.5	91.4	91.6
Flyover	[EPNdB]	87.8	81.3	88.7
Approach	[EPNdB]	96.1	66.2	96.1

Figure 8 depicts the airframe noise components breakdown (wing, horizontal tail, vertical tail, flap, slat,

Table 8: Fuel burn case: Source components breakdown

		Airframe	Engine	Total
Sideline	[EPNdB]	78.8	89.3	89.7
Flyover	[EPNdB]	86.7	78.8	87.4
Approach	[EPNdB]	94.7	63.6	94.7

Table 9: Noise case: Source components breakdown

		Airframe	Engine	Total
Sideline	[EPNdB]	77.5	89.7	89.9
Flyover	[EPNdB]	83.9	78.8	85.1
Approach	[EPNdB]	90.7	63.9	90.7

nose and main landing gears) for the approach condition for the three cases investigated. The landing flap configuration for the noise case is the major noise reduction achieved by the optimization, as expected. The main difference between the baseline design and the design optimized for fuel burn is the approach reference velocity (Vref).

The results shows a reduction of approximately 6.2% of fuel consumption for the fuel burn case and a 11 EPNdB quieter aircraft for the noise case. Although only the noise certification points were presented in this paper, SUAVE has the capability to investigate and optimize noise-based operations for airports with more noise restrictions, such as steep approach and dynamic cut back.

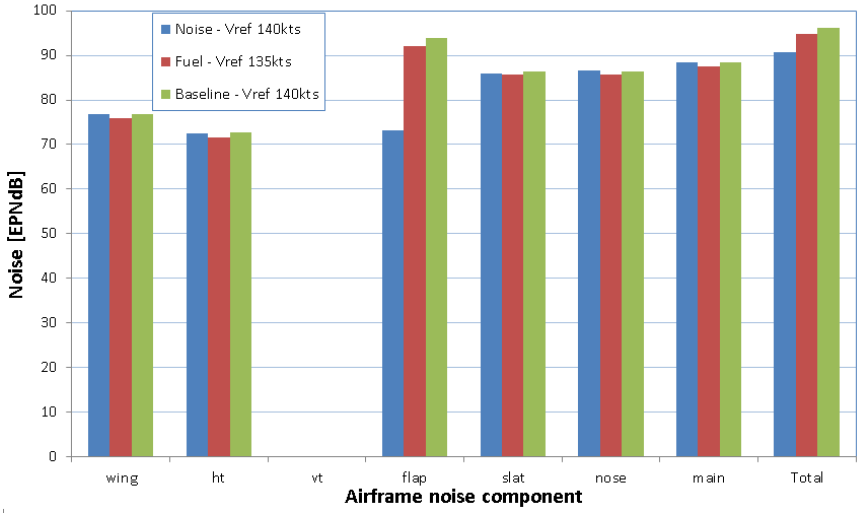


Figure 8: Airframe noise sources breakdown comparison

VII. Summary

Throughout this work, we have shown how SUAVE, a conceptual aircraft design environment, has been improved through new capabilities and integrated with external optimization packages. Building off of our previous work[1] where there is attribute-method orthogonality, we have added a higher fidelity aerodynamics model in AVL. In addition, we have improved the noise methods, critical in analyzing future designs due to more severe benchmarks being put in place for certification.

To take advantage of these new methods, we created an interface framework that is agnostic to the type of optimization package. Currently, links to both SciPy and PyOpt have been created allowing a multitude of different optimization algorithms to become available to users of SUAVE.

Three design studies were considered. In the first design study, we started with an existing feasible design of an existing regional transport similar to the Embraer E-190 and optimized it for reducing fuel burn. In the process we found a 12% reduction in fuel burn compared to the baseline vehicle. We observed that the optimizer made reasonable design trades from the perspective of an aerospace engineer using first order methods.

In the second study, we made an entry into an unexplored design space for a family of modular solar UAVs. Using the optimization framework, two configurations using similar motors and wing sections were driven to a feasible design. These two results are good starting points for a future in-depth study.

In the final study, starting from a Boeing 737, two optimizations with different objectives of noise and fuel burn were optimized while using the same set of constraints. It was observed that to reduce noise, flap angles for the main wing during takeoff and landing were decreased. However, to reduce fuel burn, the flap angles were increased. In both scenarios, vastly different, but feasible wing designs were chosen by the optimizer. This demonstrates SUAVE's ability to discover intuitive, but non-obvious results that support the designer in the design process.

The open-source nature of SUAVE has allowed the code to expand and fill needs the original code developers could not foresee. Thankfully, the coding style has allowed for greater expansion and we plan to continue the momentum of growing the code. Various tasks have yet to be tackled including geometry output and structural design. We invite others to attempt to dream up new designs and see what SUAVE can do for them to improve their vehicles.

Acknowledgments

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.

Anil Variyar would like to acknowledge the support of the Stanford Graduate Fellowship.

The authors would like to acknowledge Timothy Momose for his contributions in linking SUAVE with AVL as well as his numerous other additions throughout the code development.

References

[1]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," *AIAA Aviation*, Dallas, TX, June 2015.

[2]Drela, M. and Youngren, H., "AVL," http://web.mit.edu/drela/Public/web/avl/, February 2014.

[3]Fink, M., "Airframe Noise Prediction Method," Tech. Rep. FAA-RD-77-29, Federal Aviation Administration, 1977.

[4]SAE, "Gas Turbine Exhaust Noise Prediction," *Society of Automotive Engineers, Aerospace Recommended Practice ARP-876D*, 1994.

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

[6]Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Journal on Optimization*, Vol. 47, No. 1, 2002, pp. 99–131.

[7]Kraft, D., "A software package for sequential quadratic programming," *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988.

[8]Jones, E., Oliphant, T., Peterson, P., et al., "SciPy: Open source scientific tools for Python," 2001, [accessed 2015-06-01].

[9]Nelder, J. and Mead, R., "A simplex method for function minimization," *The Computer Journal*, Vol. 7, 1965, pp. 550–560.

[10]Vegh, J., Alonso, J., Orra, T., and Ilario da Silva, C., "Flight Path and Wing Optimization of Lithium-Air Battery Powered Passenger Aircraft," *AIAA 2015-1674 53rd AIAA Aerospace Sciences Meeting*, AIAA Scitech, Kissimee, FL, 2015.

[11]"Embraer 190," Tech. Rep. APM-1901, Embraer S. A., August 2005.

[12]Noll, T. E., Brown, J. M., Perez-Davis, M. E., Ishmael, S. D., Tiffany, G. C., and Gaier, M., "Investigation of the Helios prototype aircraft mishap," Tech. rep., 2004.

[13]für Sonnenenergie, D. G., *Planning and installing photovoltaic systems: a guide for installers, architects and engineers*, Earthscan, 2008.