ERDC MP-21-5

US Army Corps
of Engineers®
Engineer Research and
Development Center

Rotor Blade Design Framework for Airfoil Shape Optimization with Performance Considerations

Luke D. Allen, Joon W. Lim, Robert B. Haehnel,
and Ian D. Dettwiller

June 2021

Engineering Research and Development Center

The U.S. Army Engineer Research and Development Center (ERDC) solves the nation's toughest engineering and environmental challenges. ERDC develops innovative solutions in civil and military engineering, geospatial sciences, water resources, and environmental sciences for the Army, the Department of Defense, civilian agencies, and our nation's public good. Find out more at www.erdc.usace.army.mil.

To search for other technical reports published by ERDC, visit the ERDC online library at https://erdclibrary.on.worldcat.org/discovery.

ERDC MP-21-5
June 2021

Rotor Blade Design Framework for Airfoil Shape Optimization with Performance Considerations

Luke D. Allen and Robert B. Haehnel

Cold Regions Research and Engineering Laboratory
U.S. Army Engineer Research and Development Center
72 Lyme Road
Hanover, NH 03775

Joon W. Lim

U.S. Army Aviation and Missile Research
Development and Engineering Center
Moffett Field, CA 94035

Ian D. Dettwiller
Information Technology Laboratory
U.S. Army Engineer Research and Development Center
3090 Halls Ferry Road
Vicksburg, MS 39180

Final report

Prepared for U.S. Army Corps of Engineers
Washington, DC 20134

Under Program Element Number 0603465A, Project Number AL3

Preface

This study was conducted for the U.S. Army Corps of Engineers, in collaboration with the U.S. Army Combat Capabilities Development Command (CCDC) Aviation and Missile Center (AvMC), under Army Direct funding, Program Element 0603465, Project AL3. The technical monitor was Dr. Robert M. Wallace.

The work was performed by the Engineering Resources Branch (Caitlin Callaghan, Branch Chief) and the Terrestrial and Cryospheric Science Branch (John Weatherly, Branch Chief) of the Research and Engineering Division, (George Calfas, Division Chief), U.S. Army Engineer Research and Development Center, Cold Regions Research and Engineering Laboratory (ERDC-CRREL) and the Computational Analysis Branch (Jeffery Hensley, Branch Chief) of the Computational Science and Engineering Division (Devin Sham, Acting Division Chief), U.S. Army Engineer Research and Development Center, Information Technology Laboratory (ERDC-ITL). At the time of publication, the Deputy Director of ERDC-CRREL was Mr. David Ringelberg and the Director was Dr. Joseph Corriveau, the ITL Deputy Director was Ms. Patti S. Duett and the Director was Dr. David A. Horner.

This paper was originally presented at the *AIAA SciTech Forum* 11–15 and 19–21 January 2021, and published online on 4 January 2021.

The Commander of ERDC was COL Teresa A. Schlosser and the Director was Dr. David W. Pittman.

Rotor Blade Design Framework for Airfoil Shape Optimization with Performance Considerations

ABSTRACT

This work introduces a framework for automated rotor blade airfoil design optimization based on helicopter performance. The framework combines two computational workflows, each created using the Galaxy Simulation Builder (GSB) software package. First, the airfoil parameterization code ParFoil is used to generate a database of morphed airfoil geometries, with aerodynamic properties predicted by ARC2D (via C81Gen). The airfoil database is used to generate a surrogate model for airfoil performance coefficients based on ParFoil parameters. The second workflow utilizes the surrogate model to perform design optimization on a portion of a rotor blade. Optimization is carried out using GSB and the integrated Dakota numerical optimization library. This approach provides users with a variety of optimization algorithms and access to the Department of Defense Supercomputing Resource Center's (DSRC's) machines. The framework is demonstrated using Dakota's multiobjective genetic algorithm (MOGA) to perform a multiobjective, constrained optimization of the tip region of the standard UH-60A main rotor blade. The problem is formulated such that the power coefficient is minimized for forward flight and hover, simultaneously, while subject to a constraint on the rotor pitch link load. The airfoil thickness and thickness crest position of the outboard SC1095 airfoil are the only design parameters used in this example study. Analysis of select points from the Pareto-optimal set shows reductions in main rotor power requirements across a full range of forward flight speeds. The power coefficients for hover and forward flight with advance ratio $\mu = 0.3$ are reduced by up to 0.90% and 3.47%, respectively. Improvements of up to 7% are predicted for higher flight speeds approaching $\mu = 0.4$. Furthermore, it is shown that the predicted pitch link load can be reduced by as much as 19.2% without incurring a penalty on rotor performance.

I. Nomenclature

b	=	airfoil boat-tail angle
c	=	chord length
c_d	=	drag coefficient
c_l	=	lift coefficient
c_m	=	pitching moment coefficient
C_p	=	rotor power coefficient
d	=	airfoil leading edge droop
F_{pl}	=	pitch link load
k	=	airfoil leading edge radius
m	=	airfoil maximum camber
N_{cfd}	=	number of CFD evaluations
N_{opt}	=	number of optimizer evaluations
n	=	airfoil trailing edge camber
n	=	number of dimensions
p	=	airfoil camber crest position
q	=	trailing edge camber crest position
R	=	rotor radius
S	=	speedup ratio
t	=	airfoil maximum thickness
t_{cfd}	=	CFD evaluation time
t_{opt}	=	optimizer iteration time
w_s	=	vertex weight factor
x	=	airfoil thickness crest position
$\vec{x}$	=	design parameter vector

Subscripts

f	=	forward flight condition
h	=	hover condition

Acronyms

DSRC	=	DoD Supercomputing Resource Center
GSB	=	Galaxy Simulation Builder
HPC	=	High Performance Computer
LE	=	airfoil leading edge
TE	=	airfoil trailing edge
UTTAS	=	Utility Tactical Transport Aerial System

II. Introduction

The optimization of helicopter rotor systems is an important research topic. In particular, the development of advanced airfoils has been shown to greatly improve rotor efficiency [1,2]. Many methods for airfoil section optimization currently exist. In general, these methods pair a geometry generation or morphing tool with a numerical optimizer. Design parameterization tools construct new airfoil coordinates by blending several parameters using polynomial fitting and spline interpolation. ParFoil is one such tool [1]. This work extends the work done by Lim by incorporating the geometry creation capabilities of ParFoil with the Dakota numerical optimization library. Furthermore, the entirety of the optimization workflow is to be carried out using the capabilities of Galaxy Simulation Builder (GSB) [3]. The result is a self-contained optimization tool that can be implemented on a local workstation or in a High-Performance Computing (HPC) environment such as the Department of Defense (DoD) Supercomputing Resource Centers (DSRCs).

Optimization for rotorcraft poses difficult challenges. For a vehicle in forward flight, the angle of attack and relative flow velocity of the impinging air are constantly changing. The unsteady nature of the aerodynamic conditions of the rotor significantly widens the flight envelope one must consider when designing a new airfoil. It is advantageous, then, to optimize the rotor based on overall vehicle performance metrics rather than a metric more commonly used for isolated airfoil shape optimization (e.g., lift-to-drag ratio). Obtaining these metrics for final evaluation typically involves the use of high-fidelity computational fluid dynamics (CFD) codes such as Helios to form a time-accurate picture of performance characteristics. Unfortunately, the high cost associated with CFD simulations makes direct use for optimization intractable in most cases.

Lower-fidelity, preliminary design tools such as RCAS [4] and CAMRAD II [5] address some of the challenges of full CFD simulations. These comprehensive analysis codes invoke simplified physics models and lookup tables to calculate blade structural response (e.g., bending, twisting, vibration) in addition to aerodynamic loads. They are highly useful tools; however, they are not capable of matching the fidelity of CFD. Furthermore, comprehensive codes typically rely on lookup tables (C81 tables) to provide lift, drag, and moment data for airfoils. When not readily available, producing these C81 tables (named for the legacy code for which they were developed) necessitate the use of an additional two-dimensional (2D) CFD flow solver, offsetting some of the computational advantage held by the mid-fidelity comprehensive codes. A surrogate model approach is desirable to decrease overall analysis time in situations where large numbers of C81 tables will be needed to perform the optimization.

The objectives of this work are to 1) develop an automated, HPC-enabled workflow for building a surrogate model for the rapid generation of C81 airfoil performance tables, 2) develop an automated workflow for rotor optimization with vehicle performance considerations, and 3) demonstrate the capabilities of the framework with an example optimization problem.

III. Rotor Blade Optimization Framework

A. Galaxy Simulation Builder Overview

Complex digital workflows involving multiple simulations and disciplines can impose a tremendous burden on the analyst. Each component of the workflow likely requires pre- and post-processing work to produce and interpret results (e.g., input file setup, converting file formats, visualizing data). For simulations that will be repeated several or more times, such as optimization, it can be advantageous to automate the workflow. To handle the automation process, this work makes use of Galaxy Simulation Builder (GSB) [3]. Galaxy is a workflow management system developed by Stellar Science, Inc. and the Air Force Research Laboratory to handle the scheduled execution of programs in a simulation. Galaxy is flexible and can handle running small simulations on a local workstation as well as very large simulations with many concurrent jobs running on multiple HPC hosts, managed by the Galaxy coordinator.

One of the primary motivations for using Galaxy is its ability to run simulations on the DSRC's machines. Access to these machines provides researchers with vast computing resources. Galaxy handles the scripts and environment setup for proper queuing and execution on the target DSRC system. Individual tasks within a Galaxy simulation are referred to as modules and can range from data transfer events to high-fidelity simulation executions.

Analysis capabilities in Galaxy are largely provided through its integration with Dakota, an optimization and UQ software package developed by Sandia National Laboratories. See Dakota documentation in Ref. [6] for additional information and available capabilities. Galaxy provides customized GUI support for setting up Dakota simulations as part of the integration between the two tools. The combination of Galaxy's parallel job monitoring, management, and execution with Dakota's extensive analysis capabilities provides a powerful framework for high-fidelity and multidisciplinary analysis.

B. Surrogate Model for C81 Table Generation

Performance data for 2D airfoils required by comprehensive rotorcraft analysis codes, such as RCAS and CAMRAD II, are provided through C81 tables. Specifically, a C81 table provides coefficients of lift, drag and pitching moment (c_l, c_d, and c_m, respectively) for a 2D airfoil as functions of Mach number, M, and angle of attack, α. GSB was used to develop an automated workflow for the construction of a surrogate model capable of generating a C81 table from a set of parametric inputs. The traditional approach to C81 table generation is to use a CFD solver to populate each element in the table. While codes such as C81Gen exist to automate the table generation process [7,8], they are computationally expensive and consequently not well suited for optimization due to the large number of perturbations in geometry required to ensure adequate representation of the design space. The use of a surrogate model addresses this challenge directly by allowing computationally inexpensive C81 table generation for a large number of design variants.

The presented framework uses the ParFoil code to describe airfoil shape using a set of nine meaningful design parameters. ParFoil starts with a baseline airfoil geometry and morphs it into a new design by augmenting the design parameters. The design parameters are leading-edge radius (k), leading-edge droop (d), camber (m), camber crest position (p), thickness (t), thickness crest position (x), trailing-edge camber (n), trailing-edge camber crest position (q), and boat-tail angle (b). Before parameterization, the baseline airfoil coordinates are redistributed using a non-uniform radial basis spline (NURBS). The baseline values for the design parameters are then extracted from the spline curve. Parameter augmentation is performed by applying either a scaling factor or delta value to the baseline, and then updating the airfoil coordinates. The individual effects of varying the parameters are shown in Fig. 1. A summary of the t and x parameters used in this study, their formulations, and their feasible bounds for the SC1095 airfoil is given in Table 1.

Table 1 ParFoil parameter augmentation formulations with bounds for SC1095.

Parameter Description	Formulation	Feasible Bounds (SC1095)
Max Thickness	$t = t_0 f_t$	$0.5 \leq f_t \leq 1.3$
Thickness Crest Position	$x = x_0 + \Delta x$	$-0.1 \leq \Delta x \leq 0.25$

In this framework, the feasible upper and lower bounds for each parameter are provided to Dakota, and the space is sampled within these bounds. The design parameters for each sampled point are provided to ParFoil, and a new airfoil geometry file is generated. The output from ParFoil is passed to C81Gen for computation of airfoil performance

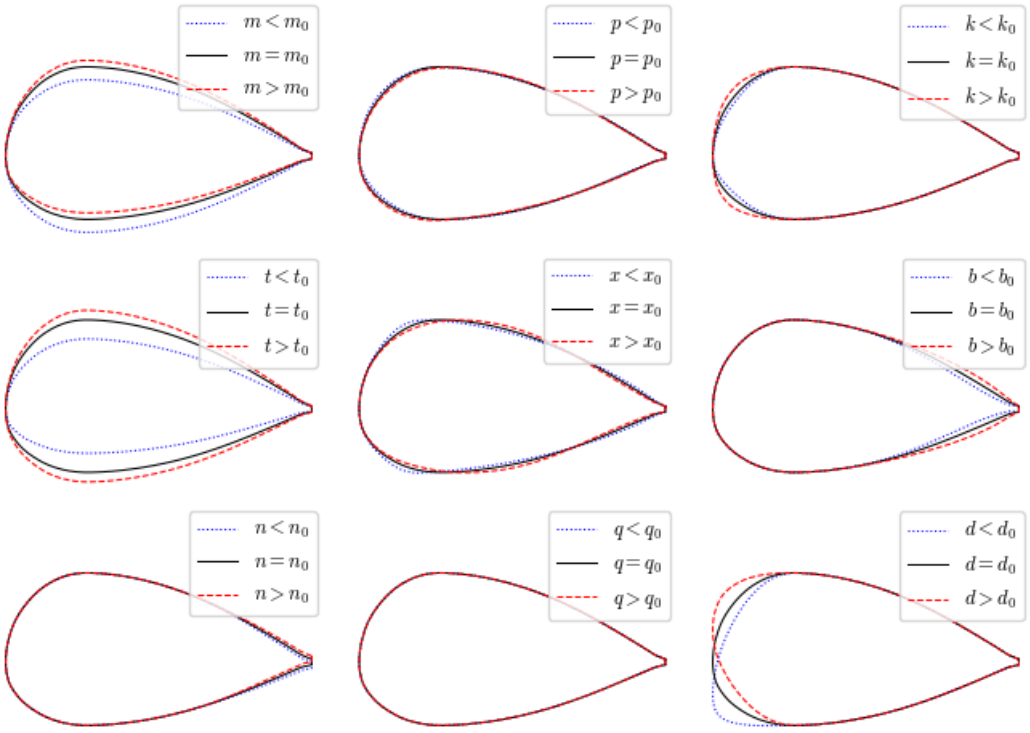

Fig. 1 Effects of parameter changes in ParFoil on airfoil geometry. The zero subscript denotes baseline value for the SC1095 airfoil. Note that changes in camber crest positions (p and q) have little visible effect for the SC1095 due to its near-symmetric design. Vertical scales are exaggerated.

coefficients. C81Gen is a wrapper for 2D airfoil mesh generation and the ARC2D Navier-Stokes CFD solver developed at NASA Ames Research Center [9]. Its purpose, as stated above, is to automate the generation and tabulation of performance data from simulations encompassing all practical operational ranges. A typical C81 table contains $\alpha - M$ pairings with $-180° \leq \alpha \leq 180°$ and $0.0 \leq M \leq 1.0$. To reduce computation time, the implemented framework limits C81Gen's computations to $-10° \leq \alpha \leq 20°$. This subset is then patched into a full C81 table computed from the baseline geometry for which experimental data is available. It is noted that despite the reduction in function calls required to populate the smaller table, total execution times for C81Gen still averaged approximately 2.6 hours running on 40-50 CPUs during testing, and that using greater numbers of CPU's (e.g., 80-100) did not improve computation times. While not excessive for a small number of simulations, the runtime makes it prohibitively expensive for direct use in optimization requiring evaluation of many hundreds or thousands of design points.

The feasible bounds are used by Dakota to populate the design space and the data is then collected and conditioned for model fitting. A Python interface was developed with functionality for C81 file manipulation and data storage. For the i^{th} table in the database, the set of ParFoil design parameters, $\vec{\boldsymbol{x}}_i$, and the table's location, y_i, are stored. A mapping function, λ, is then generated such that $\lambda(\vec{\boldsymbol{x}}_i) = y_i$. This mapping function is saved to disk and is available for future use. The mapping function serves as one component of the surrogate model. When combined with additional Python methods developed for this framework, the mapping function allows users to access the C81 database and perform linear interpolation between tables. Querying the model returns a list of coordinates forming the n-dimensional bounding hyper-box, or simplex, containing the query point. Weighting factors for each vertex of the simplex are determined using one of two methods: 1) normalized distances to each vertex in the case of data sampled on a regular grid, or 2) transformation to barycentric coordinates for the case of data sampled on an irregular or sparse grid.

The process described above is set up locally in GSB and launched on a specified host with a single command. The simulation flowchart as seen in GSB is shown in Fig. 2. Each module in the chart executes a specified program or script. Modules are executed in the order defined by arrows representing information flow. The workflow consists of two stages. The first stage is the database generation stage. This stage includes all of the modules between "Begin Parameter Sweep" and "End Parameter Sweep." Here, a parameter sweep can refer to whichever method is chosen to sample the design space, e.g., Latin hyper-cube sampling (LHS). Dakota handles variable substitution following the setup of separate working directories for each case. The "Preprocess" module is then used to handle any additional runtime configuration (e.g., export the ParFoil input file). ParFoil processes the input parameters and produces a

modified airfoil geometry file that is used by C81Gen to produce an airfoil table. The second stage is the surrogate model creation stage and is made up of the modules "Fetch Paths" and "Build Surrogate Model."

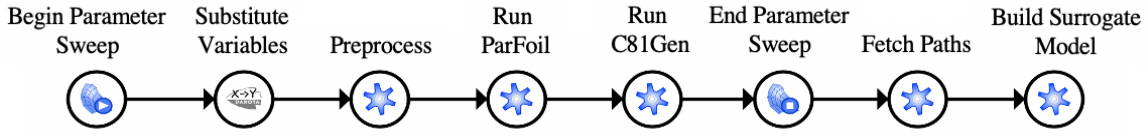

Fig. 2 Galaxy flowchart for preparing C81 table database and building the surrogate model.

C. Rotor Blade Optimization

A second workflow has been developed using GSB for rotor blade optimization. An example of the simulation flowchart as seen in GSB is shown in Fig. 3. The "Preprocess" module allows users to make any modifications to starting files that require information not known a priori (e.g., the path to working directory). This module also leverages the surrogate model described above to generate C81 tables using parameters defined by Dakota variable substitution. Next, one or more instances of RCAS are executed. The example in Fig. 3 shows two RCAS cases executed simultaneously. The RCAS scripts are provided by the user, allowing flexibility in the types of analyses being performed. Once the RCAS evaluations are complete, the objective and constraint function values are extracted in the "Pull Metrics" module and returned to Dakota. The metrics are defined by the user and can be a function of any quantity that is output from RCAS. Galaxy and Dakota take advantage of resources by concurrently evaluating as many asynchronous iterations as possible. The number of concurrent iterations depends on the optimization algorithm and allocated number of CPUs. A diagram of the entire two-step workflow, including the surrogate model builder and rotor blade optimizer are shown in Fig. 4 Details regarding the optimization techniques explored in this work are given in Sections V and VI.

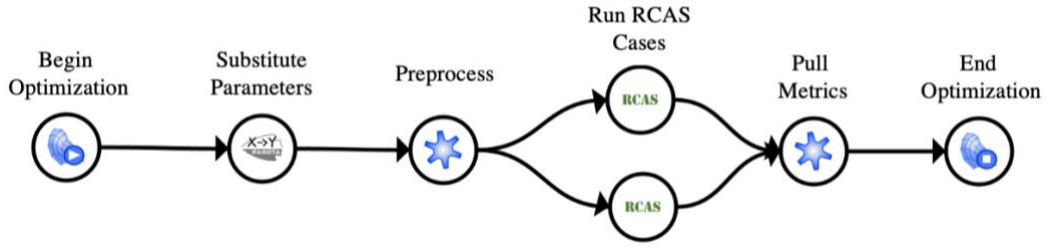

Fig. 3 Galaxy flowchart for rotor blade airfoil optimization.

IV. Surrogate Model Selection and Evaluation

A. Linear Interpolation of Tables

Interpolation in high dimensional space (taken here to mean five or more) poses several challenges. The number of grid points required to maintain array density on a regular grid becomes infeasibly large as the number of dimensions increases [10]. Additionally, coding practices dictate that the original dataset, or a copy, must be maintained, impacting model portability. Irregular grids present their own limitations due to dimensionality. Primarily, determining the location of a design point relative to the grid points becomes difficult as the number of dimensions increases. One reason for this is that the metrics used for determining the distance between points become less meaningful and less intuitive in high dimensions [11]. For example, it has been observed that the Euclidean distance between all points in a dataset approaches a constant value as the number of dimensions increases [12]. That fact also limits the usefulness of nearest-neighbors searches. Due to these practical considerations, the number of dimensions has been limited to be less than five during development and testing.

Regular grids have the advantage of being easily searchable. Searching for the location of an arbitrary design point defined by the parameter vector $\vec{x}^*$ in a n-dimensional regular grid is straightforward and involves determining the grid coordinates that bound $\vec{x}^*$ in each dimension. Thus, a bounding hyper-box is formed with $N = 2^n$ vertices. Weighting factors, w_s, for linear interpolation can then be calculated from the normalized distance from $\vec{x}^*$ to each

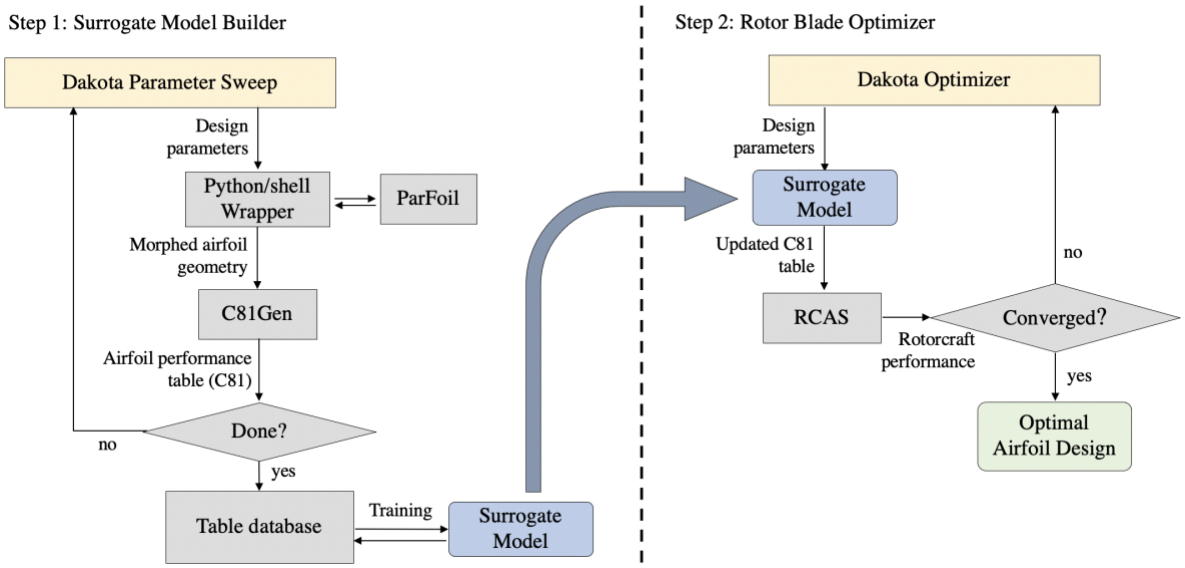

Fig. 4 Two-step optimization framework flowchart.

vertex. Irregular or sparse grids require a different approach. For this work, Delaunay triangulation via Python and SciPy's implementation of the Qhull library are used [13]. The algorithm computes the set of n-dimensional simplices (convex hulls of points) such that no point lies inside the circum-hypersphere of any simplex. Triangulation can be costly in high dimensions. The number of simplices required for the tessellation of P points in n-dimensions is $\mathcal{O}(P^{\lfloor n/2 \rfloor})$ [14]. Limitations are mitigated by selecting fewer dimensions for analysis. Querying the triangulation at point $\vec{\boldsymbol{x}}^*$ involves locating the simplex containing $\vec{\boldsymbol{x}}^*$and identifying the n+1 tables that form its vertices. Weighting factors for each vertex are then determined by transforming their cartesian coordinates into barycentric coordinates.

Regardless of the method used, once the bounding-box and weighting factors are determined, interpolation can then be performed via a weighted sum for each $\alpha - M$ pair across the tables. Thus, the coefficient value, c_* (representing c_l, c_d, or c_m), for a new table defined at point $\vec{\boldsymbol{x}}^*$, is

$$c_*(\vec{\boldsymbol{x}}^*, \alpha, M) = \sum_{s=1}^{N} w_s c_*(\vec{\boldsymbol{x}}_s, \alpha, M) \tag{1}$$

where s is a vertex of the bounding box, and w_s is the corresponding weighting factor.

A comparison of c_d trends at a range of Mach numbers for different C81 tables generated for the SC1095 airfoils is shown in Fig. 5. The simulated values from C81Gen (via ARC2D) differ somewhat from the experimental flight data at all Mach numbers. These discrepancies, along with similar ones for other Mach numbers and angles of attack, resulted in baseline C_p values that were inconsistent depending on which table was used. Therefore, a direct comparison between tables from flight data and modified tables from C81Gen was not representative of the true performance increase (the benefit is overpredicted). Instead, a fairer comparison is made by using the table from C81Gen as the baseline.

B. Machine Learning Models

Several machine learning algorithms were also investigated in addition to linear interpolation. The Scikit-Learn library of regressors provides a wide array of machine learning algorithms. The standardized interface proved to be useful in testing, as it allowed for quick transitions to different regressors with minimal adjustments to the code [15,16]. Several models stood out among those tested. Examples include the Random Forest Regressor, based on a method of aggregated decision trees, and Multi-Layer Perceptron Regressor, a neural network algorithm. Validation was performed by comparing the model predictions for c_l, c_d, and c_m against tables produced by C81Gen for parameter values unseen during training. Validation showed root-mean-squared-error (RMSE) values as low as 0.017 and 0.012, respectively, with corresponding R^2 values of 0.994 and 0.997, respectively. In addition to Scikit-Learn, neural networks using TensorFlow and Keras [17] were also investigated. Here, the RMSE and R^2 scores were 0.014 and 0.996, respectively.

While initially promising, a deeper investigation revealed that the error metrics reported above were insufficient or incomplete. Analyzing the coefficient trends for complete tables showed that the models did not perform equally well for all regions. The models had particular trouble accurately predicting drag for low angles of attack, where c_d is multiple orders of magnitude smaller than that of higher angles of attack. This behavior is due to the use of off-the-shelf error metrics when computing loss. The development of a suitable loss function for this problem is a topic of future research.

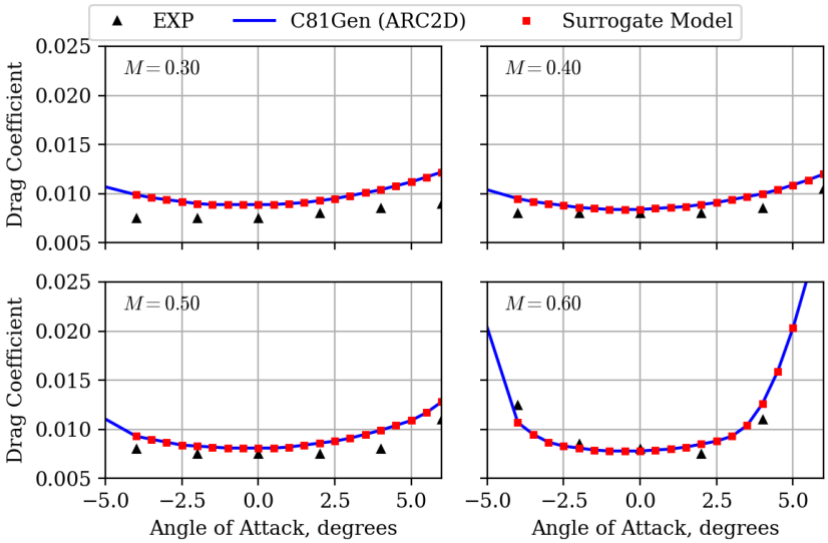

Fig. 5 Trends in c_d vs. angle of attack for the baseline SC1095 airfoil over a range of Mach numbers comparing C81 tables from multiple sources.

C. Cost Benefit of the Surrogate Model Framework

This section will discuss the advantages of using the surrogate model approach for C81 table production. Initially, many C81 tables must be generated using CFD simulations (via C81Gen). Each of these tables takes an average of approximately 100-120 CPU-hours to complete. By comparison, the surrogate model using linear interpolation on a regular grid can generate a new table in 0.5-0.8 seconds. Let the speedup, S, be defined as the ratio of CPU times for optimization performed with the surrogate model (including generation of the database) to that of optimization performed using only CFD. The ratio can be represented as

$$S = \frac{N_{opt}(t_{cfd} + t_{opt})}{N_{cfd}t_{cfd} + N_{opt}t_{opt}}, \tag{2}$$

where N_{opt} is the number of evaluations required by the optimizer, N_{cfd} is the number of tables generated via CFD for the surrogate model database, t_{opt} is the time for a single optimizer iteration, and t_{cfd} is the time to generate a table using CFD. The time for the surrogate model to generate a table is considered negligible.

Due to the relationship in Eq. (2), it is difficult to form a general statement regarding the advantage provided by the model. The answer depends on several factors, e.g., the optimizer used, level of accuracy needed, number of parameters considered. Additionally, the equation provides an estimate based on CPU time, not wall-clock time. Thus, the true speedup also depends on the levels of parallelization used throughout each step (e.g., optimizer concurrency, table generation concurrency, CPU allocation for CFD). The surrogate approach is particularly powerful for population-based optimization algorithms (e.g., MOGA) which rely on a relatively large number of function evaluations to determine the global minimum. Multiobjective, gradient-based optimizers are also good candidates since they generally require one to survey a variety of function weights and starting points. Section VI provides examples using multiple optimization algorithms and their associated estimated speedups.

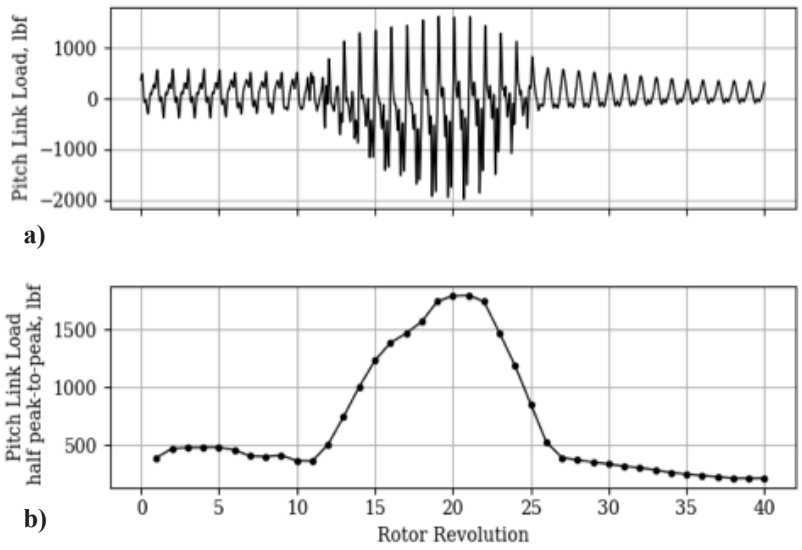

Fig. 6 Pitch link load plots for baseline UH-60A rotor during UTTAS pull-up maneuver. Plots a) and b) show the pitch link load time history (mean removed), and the half peak-to-peak pitch link load per rotor revolution, respectively.

V. Optimization Objectives and Constraints

To demonstrate the optimization framework, RCAS is used to predict rotor performance during multiple flight conditions, subject to changes in the main rotor tip airfoil geometry. The objective for this case is to simultaneously minimize the power coefficients of the rotor for forward flight and hover, $C_{p,f}$ and $C_{p,h}$, respectively. At the same time, a constraint function, F_{PL}, is calculated from the maximum load experienced by the rotor pitch link during an extreme maneuver.

The standard UH-60A main rotor has a radius $R = 26.83$ feet and a rotational velocity of 27.03 rad/s. For forward flight, the prescribed free stream velocity of 218 ft/s corresponds to an advance ratio of $\mu = 0.3$. Trim targets for this and the hover case are based on wind tunnel test conditions from Norman et al. [18]. The model consists of a single isolated rotor with 3-DOF hinge for blade lag, flap, and pitch. Pilot inputs for collective, lateral, and cyclic controls, as well as the nose-down pitching of the vehicle, are given initial values and then updated until equilibrium is reached. The hover condition uses the same model with updated initial conditions suitable for a zero free stream velocity.

The Utility Tactical Transport Aerial System (UTTAS) pull-up maneuver is used to investigate maximum design load on the pitch link. Flight conditions are based on test C11029 of the UH-60A Airloads Flight Test Program [19]. The rotorcraft enters the maneuver near its maximum forward flight speed and quickly pulls up, achieving load factors which greatly exceed the steady-state lift limit of the rotor. In the severe case of the C11029 test condition, the vehicle experiences a normal load factor of 2.1g. The maneuver lasts for approximately 40 rotor revolutions before returning to level flight. During the maneuver, the time history of the pitch link load is recorded as shown in Fig. 6a. For each rotor revolution F_{pl} is captured as the half peak-to-peak value of the pitch link load, as shown in Fig. 6b. The maximum value of F_{pl} experienced during the maneuver is used as the value of the constraint function. An airfoil design is considered viable only if the maximum F_{pl} less than or equal to the baseline value calculated for the unmodified rotor.

Let the objective function, f, be some combination of $C_{p,f}$ and $C_{p,h}$. The exact formulation of f depends on the particular optimizer used. The optimizer receives the set of parametric inputs, $\vec{\boldsymbol{x}}$, from Dakota such that $\vec{\boldsymbol{x}} = \{f_t, \Delta x\}$. The constrained optimization problem can thus be defined as

$$\begin{aligned} \underset{\vec{\boldsymbol{x}}}{\text{minimize}} \quad & f(C_{p,h}(\vec{\boldsymbol{x}}),\ C_{p,f}(\vec{\boldsymbol{x}})) \\ \text{subject to} \quad & \vec{\boldsymbol{x}} \in X, \\ & \max(F_{pl}(\vec{\boldsymbol{x}})) \leq \max(F_{pl}(\vec{\boldsymbol{x}}_0), \end{aligned} \tag{3}$$

where X is the feasible set of parameters satisfying the limits in Table 1, and $\vec{\boldsymbol{x}}_0$ is the parameter vector describing the baseline SC1095 airfoil.

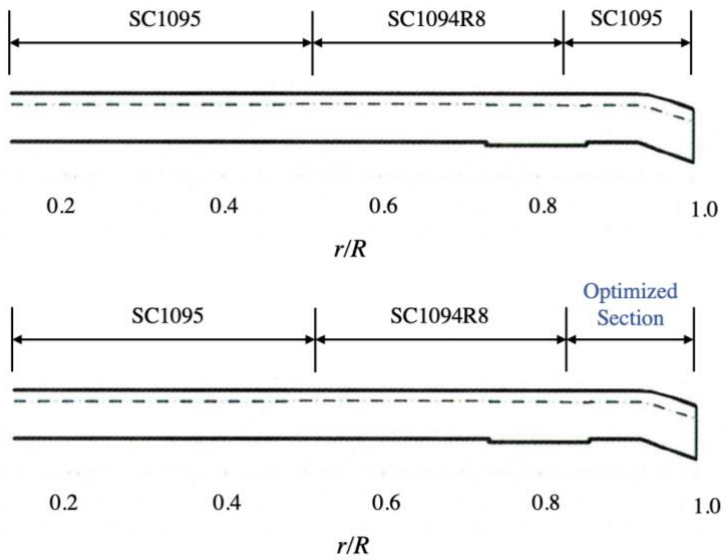

Fig. 7 Sketch of UH-60A standard rotor blade (top) and with the optimized section highlighted (bottom).

VI. Results and Discussion

The standard UH-60A main rotor was used as the baseline for this study. The blades are made up of the SC1095 airfoil from the blade root to a radial position of 0.48135R, the SC1094R8 from 0.48135R to 0.83851R, and back to the SC1095 from 0.83851R to the blade tip. This work morphs the outermost blade section, only as shown in Fig. 7. The tip section is morphed to minimize the power coefficient for two distinct flight conditions, forward flight with advance ratio $\mu = 0.3$, and hover, as discussed previously.

Optimization was carried out using two methodologies. The first used Dakota's implementation of a multiobjective genetic algorithm (MOGA) to perform an initial global optimization. This MOGA, based on the work of Eddy and Lewis [20], is an efficient method that uses distinct-point and clustering metrics to generate a distribution of points in the Pareto frontier that is as close to uniform as possible. The second algorithm was the gradient-based CONMIN Fletcher-Reeves conjugate gradient method [21]. This method was implemented with multiple starting points to achieve an optimal solution. In both cases, two degrees of freedom were considered for the initial studies. The parameters t and x (thickness and thickness crest position) were chosen because C_p was found to be most sensitive to changes in these parameters relative to the others. This sensitivity was reported previously by Lim [1] and is supported by subsequent observations during testing. The surrogate model for all cases presented below used linear interpolation on a regular grid with database size $N_{cfd} = 72$.

A. Unconstrained MOGA

The initial population used for the example case was 264. Optimization concluded after 58 generations and 3,000 function evaluations (2,429 unique) with a final population of 43 points representing the Pareto set. The results of this case are shown in Fig. 8. The horizontal and vertical axes represent the power coefficients for forward flight and hover, respectively, whereas the color of each point indicates the iteration. The baseline point refers to the C_p values for the unmodified rotor blade. The MOGA optimizer appeared well behaved, with a clear progression toward high-performing areas of the design space. The estimated speedup factor for this case provided by Eq. (2) is $S = 33.3$. This speedup represents a reduction in wall-clock time from 6 weeks to approximately 1.5 days based on the practices used for testing.

It can be seen in the Fig. 8 that C_p in hover is less sensitive to changes in airfoil shape than for forward flight. Furthermore, all Pareto points that reduce $C_{p,f}$ also reduce $C_{p,h}$. These observations inform the selection of two points from the Pareto set for further examination. The first point selected (c1940) provides the largest reduction in $C_{p,f}$, and the second (c1315) is a compromise in forward flight and hover performance. The relative response for these points is also indicated in Fig. 8. A summary of the design parameters and changes in C_p compared to the baseline SC1095

airfoil is shown in Table 2 with sketches of each geometry shown in Fig. 9. In both cases, the thickness of the airfoil is reduced. While the thickness crest position is moved toward the chord center and toward the leading edge for c1940 and c1315, respectively.

The points c1940 and c1315 were then evaluated for a range of forward flight speeds. The advance ratio was varied from $\mu = 0.0$ to $\mu = 0.4$ and the power coefficient was calculated by RCAS for each condition. The results in Fig. 10 show an incremental decrease in C_p for low flight speeds, with c1315 performing slightly better. Larger improvements are observed for higher flight speed, with c1940 yielding better results for $\mu \geq 0.24$. For very high flight speed ($\mu \geq 0.37$), power reductions of 6-7% are obtained.

Table 2 Optimized parameters and results for select points from unconstrained MOGA algorithm.

Case ID	t/c	x/c	$\Delta C_{p,f}$ ($\mu = 0.3$)	$\Delta C_{p,f}$ (hover)
Baseline	0.095	0.27	–	–
c1940	0.068	0.36	-3.38%	-0.62%
c1315	0.058	0.23	-2.51%	-0.83%

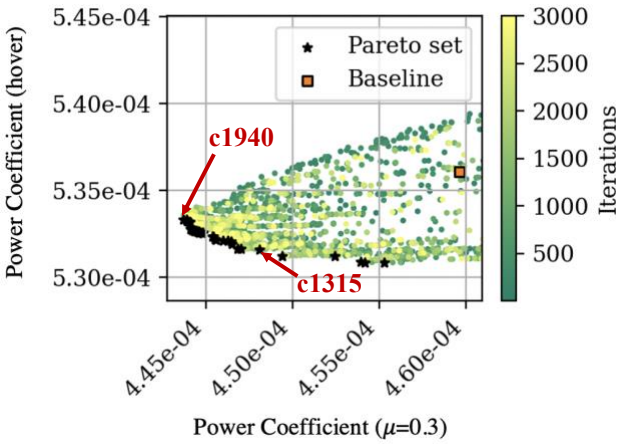

Fig. 8 MOGA optimization results for variation in airfoil thickness and thickness crest position.

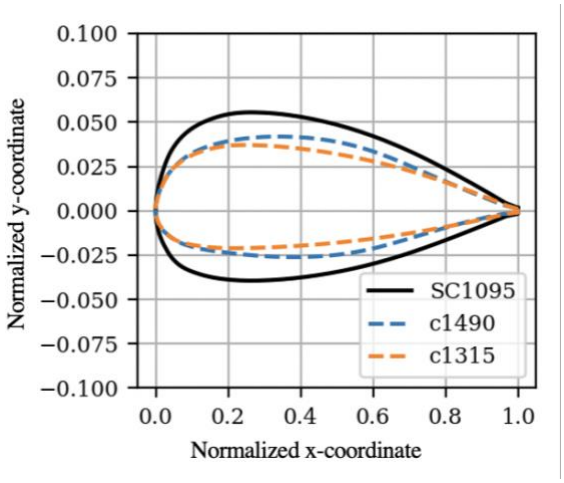

Fig. 9 Optimized airfoil geometries shown with baseline SC1095. Vertical scale is exaggerated.

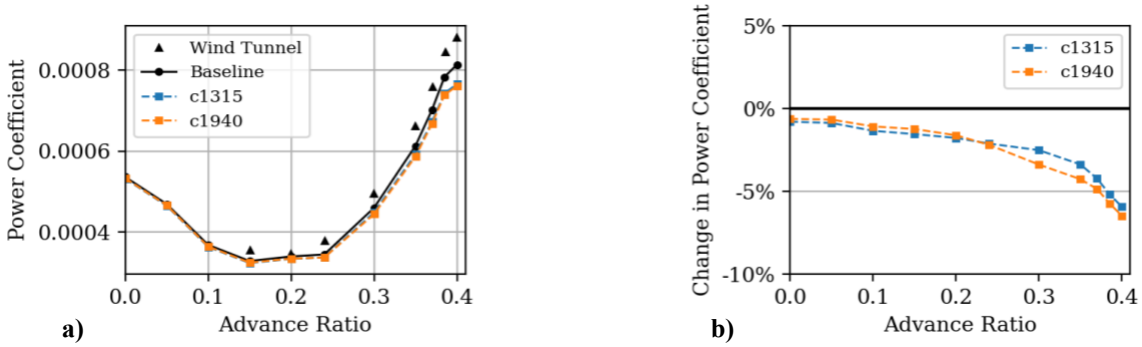

Fig. 10 Main rotor power coefficient (C_p) versus advance ratio for unconstrained MOGA airfoils. Plots show a) the values of C_p and b) percent change compared to baseline. Wind tunnel data from Ref. [18].

B. Constrained MOGA

The MOGA optimization described above was repeated with the constraint on F_{pl} active and all other conditions unchanged. The initial population was 264 and the optimization concluded after 79 generations and 3,000 function evaluations (2,338 unique). The final population was 23. The results for this case are shown in Fig. 11. The horizontal and vertical axes represent the power coefficients for forward flight and hover, respectively, while the color of each point indicates the change in the constraint function. Negative changes in F_{pl} represent improved characteristics. The estimated speedup ratio for this case provided by Eq. (2) is $S = 28.6$.

From the Pareto set, two points are selected for examination. The points are indicated in Fig. 11 and summarized in Table 3. The first (c2382) provides the maximum reduction in forward flight power, and the second (c1071) compromises forward flight and hover performance. The response surfaces for $C_{p,f}$, $C_{p,h}$, and F_{pl} are shown in Fig. 14a, b, and c, respectively, with the location of c2382. In addition to the largest improvement in $C_{p,f}$, point c2382 also resulted in a 19.2% reduction in F_{pl}; the largest decrease of any design in the Pareto set. This result showcases the ability of the framework and the MOGA optimizer to explore the design effectively and locate a point with favorable characteristics from the non-smooth F_{pl} response surface.

The $\Delta C_{p,f}$ values in Table 2 and Table 3 indicate that a better performance in forward flight was obtained when evaluating the constraint. Despite the difference being small, this result is counterintuitive. It is likely that subsequent MOGA generations, had they been allowed, would have slightly improved the results of the unconstrained optimization case.

Table 3 Optimized parameters and results for select points from constrained MOGA algorithm.

Case ID	t/c	x/c	$\Delta C_{p,f}$ ($\mu = 0.3$)	$\Delta C_{p,f}$ (hover)	ΔF_{pl} (UTTAS)
c2382	0.077	0.37	-3.43%	-0.43%	-19.2%
c1071	0.054	0.27	-2.22%	-0.90%	-0.6%

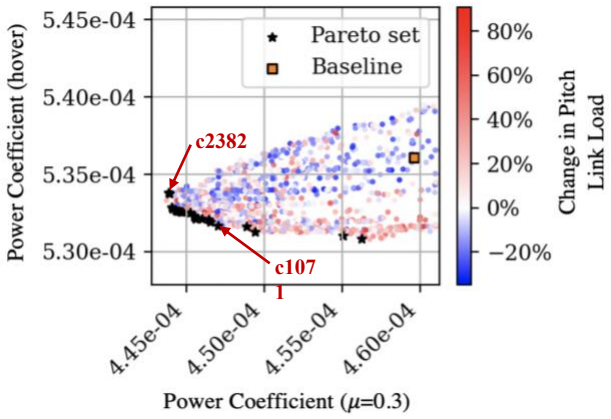

Fig. 11 Constrained MOGA optimization for variation in airfoil thickness and thickness crest position.

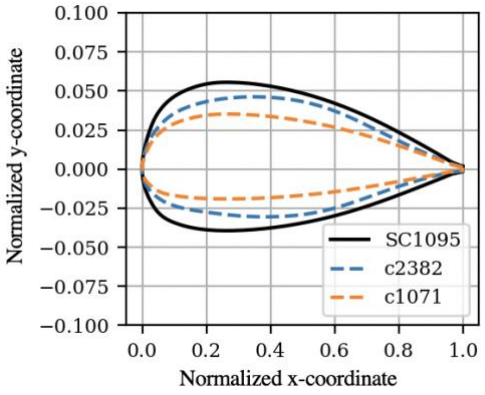

Fig. 12 Optimized airfoil geometries with F_{pl} constraint shown compared to baseline SC1095. Vertical scale is exaggerated.

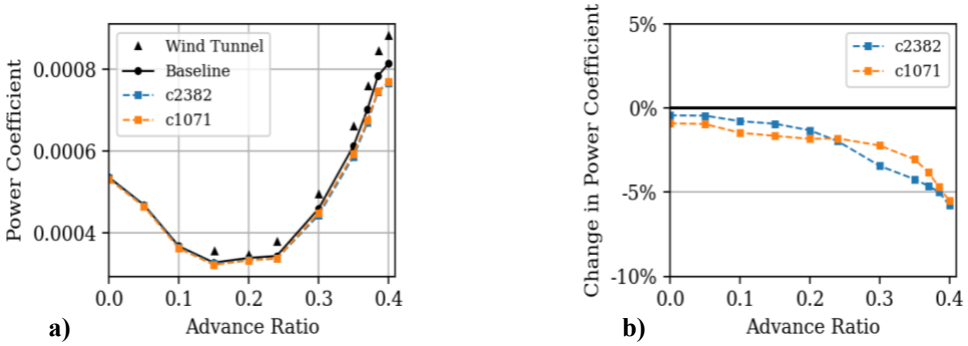

Fig. 13 Main rotor power coefficient (C_p) versus advance ratio for constrained MOGA airfoils. Plots show a) the values of C_p and b) the percent change compared to baseline. Wind tunnel data from Ref. [18].

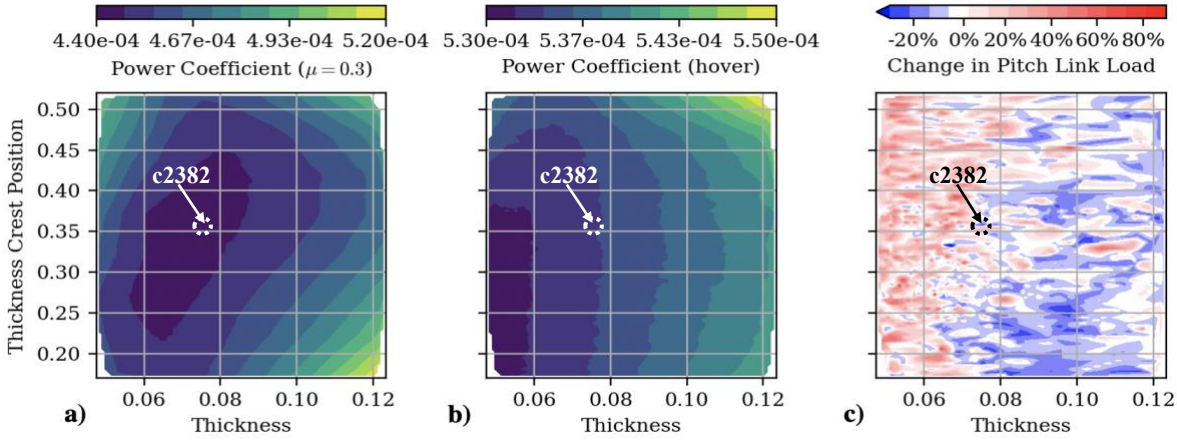

Fig. 14 Subplots a), b), and c) show the response surfaces for $C_{p,f}$, $C_{p,h}$, and ΔF_{pl}, respectively for changes in thickness (t/c) and its crest position (x/c).

C. Unconstrained Multiobjective CONMIN

A study was conducted to examine the feasibility of using a gradient-based method to perform the blade optimization. The goal of this study was the same as for the MOGA case, above. The objective function, f, is defined using a simple weighted-sum by the equation

$$f(\vec{x}) = w_f C_{p,f}(\vec{x}) + w_h C_{p,h}(\vec{x}), \tag{4}$$

where w_f and w_h are function weight coefficients for the forward flight and hover cases, respectively. Both C_p values are normalized such that $C_{p,f}(\vec{z}_0) = C_{p,h}(\vec{z}_0) = 1$.

Initial testing revealed some key shortcomings of the gradient-based approach. Starting from the baseline configuration consistently results in a solution which falls short of the global minimum. Furthermore, changing the weighting coefficients has an unpredictable effect on the solution, with an increased bias toward forward flight performance not necessarily improving $C_{p,f}$. In fact, in many instances, the opposite effect was observed. The reason for this difficulty is due to a large, flat area in the response surface. The challenge faced is not unlike the Rosenbrock function [22], often used for performance testing optimization algorithms. This flat region contains the true minimum but it is difficult for any gradient-based optimizer to traverse efficiently. The cube root of the response data is taken to increase curvature and mitigate the effect of the small gradient. Note that doing so does not change the solution, it only serves to increase gradient sensitivity along each axis. Thus, Eq. (4) is modified and the objective function becomes

$$f(\vec{x}) = w_f \sqrt[3]{C_{p,f}(\vec{x})} + w_h \sqrt[3]{C_{p,h}(\vec{x})}\,. \tag{5}$$

A sensitivity study was conducted for a range of starting points and weighting factors. The starting points are defined as A) the baseline airfoil ($f_t = 1.0, \Delta x = 0.0$), B) near forward flight optimum ($f_t = 0.7, \Delta x = 0.09$), and C) near hover optimum ($f_t = 0.53,\ \Delta x = 0.0$). Points B and C were determined from previous MOGA results. Additionally, due to the relative expected performance gains in each condition (i.e., expected $\Delta C_{p,f} > \Delta C_{p,h}$), the range of response weights are defined favoring performance in forward flight. For each starting point and weighting factor pair, the CONMIN Fletcher-Reeves conjugate gradient method [21] for unconstrained optimization is used to minimize Eq. (5). The full set of results from this survey are shown in Table 4. The total number of function evaluations performed in this survey was 930, giving this case an estimated speedup factor of $S = 9.9$. The weighted response surface contours and optimizer histories for a subset of the cases are shown in Fig. 15.

The results in Fig. 15 show the dependence of the result on the initial location. The optimizer failed to converge on the minimum value for all but starting point B, despite the measures taken above and having a relatively smooth, convex response surface. It is worth noting that cases with starting point B delivered marginally better forward flight performance than that of the MOGA cases. More work is needed to improve robustness and reliability of the gradient-based methodology for this problem.

Table 4 Optimized parameters and results for unconstrained CONMIN.

Case ID	$w_f : w_h$	t/c	x/c	$\Delta C_{p,f}$	$\Delta C_{p,h}$
A1	0.50 : 0.50	0.07864	0.3632	-2.95%	-0.34%
A2	0.80 : 0.20	0.08150	0.3790	-2.89%	-0.28%
A3	0.85 : 0.15	0.08160	0.3791	-2.89%	-0.28%
A4	0.90 : 0.10	0.08366	0.3817	-2.78%	-0.23%
A5	0.95 : 0.05	0.08304	0.3774	-2.81%	-0.28%
B1	0.50 : 0.50	0.06790	0.3658	-3.38%	-0.61%
B2	0.80 : 0.20	0.07249	0.3688	-3.47%	-0.51%
B3	0.85 : 0.15	0.07234	0.3693	-3.46%	-0.51%
B4	0.90 : 0.10	0.07230	0.3685	-3.47%	-0.51%
B5	0.95 : 0.05	0.07233	0.3689	-3.47%	-0.51%
C1	0.50 : 0.50	0.07114	0.3723	-3.43%	-0.52%
C2	0.80 : 0.20	0.06852	0.3219	-3.26%	-0.58%
C3	0.85 : 0.15	0.07223	0.3662	-3.44%	-0.51%
C4	0.90 : 0.10	0.07080	0.3460	-3.33%	-0.54%
C5	0.95 : 0.05	0.07188	0.3232	-3.21%	-0.47%

VII. Conclusion

This work represents significant initial progress in the development of a new, automated approach to rotor blade optimization. The work met each of the stated objectives by 1) developing a surrogate model workflow that was shown to substantially reduce the required computational resources of several example optimization cases, 2) creating a flexible workflow for rotor blade optimization that leverages the surrogate model, and 3) executing and evaluating a demonstration case on an HPC system. The demonstration successfully reduced the predicted UH-60A rotor power coefficient over a full range of forward flight speeds while simultaneously reducing the maximum pitch link load experienced during an extreme pull-up maneuver.

The challenges faced during this work help inform future development. The impact of the optimization framework on the UH-60A power coefficient is smaller than anticipated, based on prior results reported by Lim [1]. Several improvements are planned to address this and to add new capability. Future versions will allow for multiple airfoil sections to be modified independently. Furthermore, additional design parameters (e.g., blade twist, sweep, taper anhedral), constraints (e.g., acoustics), and surrogate model approaches (e.g., machine learning) will be considered.

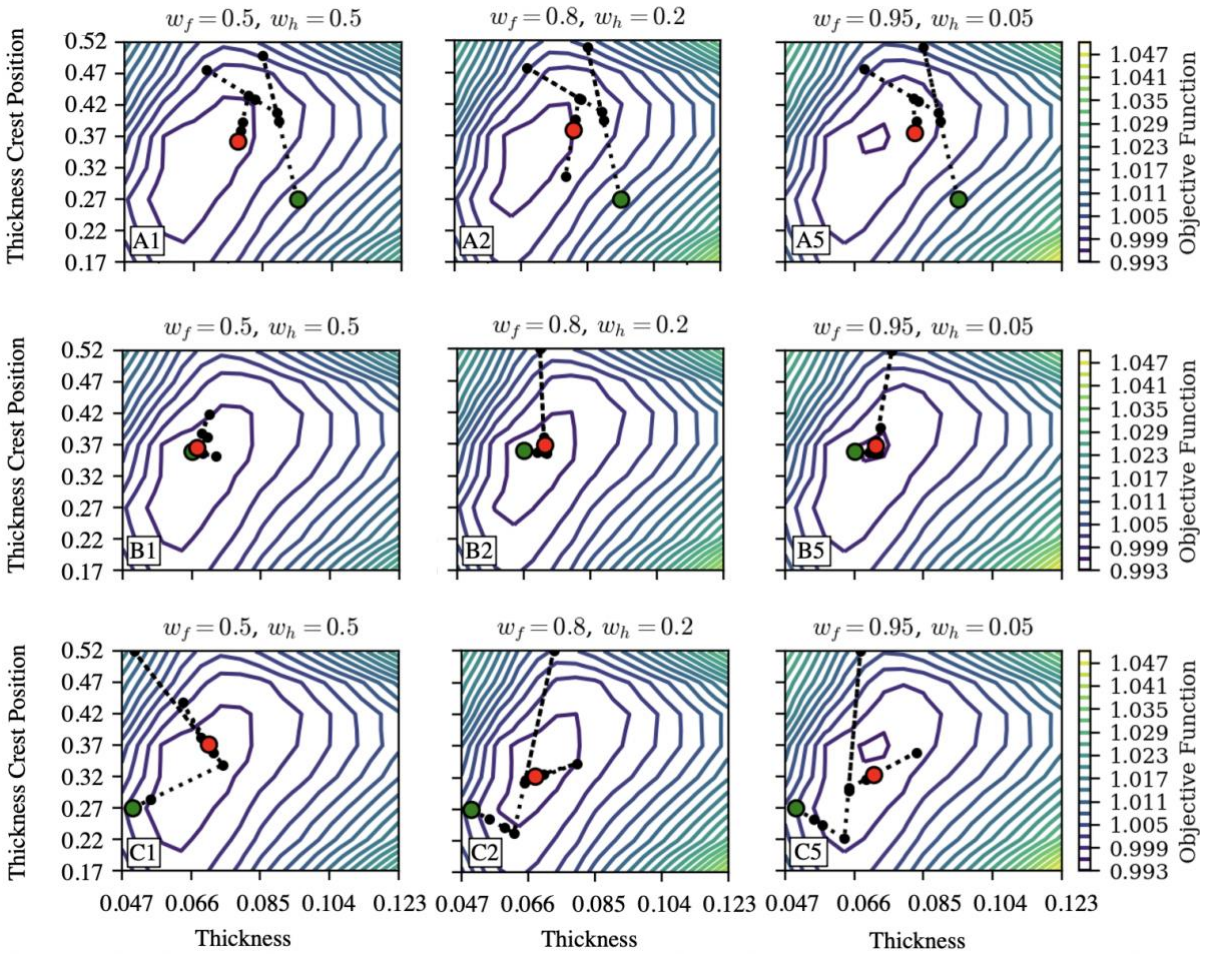

Fig. 15 Weighted response surfaces (colored contours) and optimizer histories (black dotted lines). The green and red circles are the start and end points for the optimizer, respectively.

Acknowledgements

Funding for this program was provided by the Deputy Assistant Secretary of the Army for Research and Technology and supported by the Future Vertical Lift Cross-Functional Team. This work was performed by the U.S. Army Engineer Research and Development Center under the Engineered Resilient Systems program and was made possible by the HPC Modernization Program's allocation of compute time on the ERDC DSRC's Cray XC40, Onyx.

References

[1] Lim, J. W. "Application of Parametric Airfoil Design for Rotor Performance Improvement," Proceedings of the 44th European Rotorcraft Forum, Delft, Netherlands, September 18-21, 2018.

[2] Lim, J. W. "Consideration of Structural Constraints in Passive Rotor Blade Design for Improved Performance," *The Aeronautical Journal*, Vol. 119, No. 1222, 2016, pp. 1513–1539. doi:10.1017/S0001924000011386.

[3] Stellar Science Ltd Co. "Galaxy Simulation Builder (GSB) User Guide, Version 8.1," High Power Electromagnetic Division, Air Force Research Lab, 2019.

[4] Saberi, H., Hasbun, M., Hong, J., Yeo, H., and Ormiston, R. A. "Overview of RCAS Capabilities, Validations, and Rotorcraft Applications," Proceedings of the American Helicopter Society 71st Annual Forum, Virginia Beach, Virginia, May 5-7, 2015.

[5] Johnson, W. "Rotorcraft Aerodynamic Models for a Comprehensive Analysis," Proceedings of the American Helicopter Society 54th Annual Forum, Washington, D.C., May 20-22, 1998.

[6] Adams, B. M., Bohnhoff, W. J., Dalbey, K. R., Ebeida, M. S., Eddy, J. P., Eldred, M. S., Gararci, G., Hooper, R. W., Hough, P. D., Hu, K. T., Jakeman, J. D., Khalil, M., Maupin, K. A., Monschke, J. A., Ridgway, E. M., Rushdi, A. A., Stephens, J. A., Swiler, L. P., Vigil, D. M., Wildey, T. M., and Winokur, J. G. "Dakota, a Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.7 Users Manual," Sandia National Laboratories SAND2014-4633, 2018.

[7] Rajagopalan, R. G., Baskaran, V., Hollingsworth, A., Lestari, A., Garrick, D., Solis, E., and Hagerty, B. "RotCFD - A Tool For Aerodynamic Interference of Rotors: Validation and Capabilities," Proceedings of the American Helicopter Society Future Vertical Lift Aircraft Design Conference, San Francisco, California, January 18-20, 2012.

[8] Koning, W. J. F., Johnson, W., and Allan, B. G. "Generation of Mars Helicopter Rotor Model for Comprehensive Analyses," Proceedings of the AHS International Technical Meeting on Aeromechanics Design for Transformative Vertical Flight, San Francisco, California, January 2018.

[9] Strawn, R., Mayda, E. A., and van Dam, C. P. "Automated CFD for Generation of Airfoil Performance Tables," NASA ARC-15649-1, 2009.

[10] Bellman, R. "Dynamic Programming," *Science*, Vol. 153, No. 3731, 1966, pp. 34–37. doi:10.1126/science.153.3731.34.

[11] Aggarwal, C. C., Hinneburg, A., and Keim, D. A. "On the Surprising Behavior of Distance Metrics in High Dimensional Space," Proceedings of the International Conference of Database Theory, 2001.

[12] Beyer, K. S., Goldstein, J., Ramakrishnan, R., and Shaft, U. "When Is 'Nearest Neighbor' Meaningful?," Proceedings of the International Conference on Database Theory, January 1999.

[13] Barber, C. B., Dobkin, D. P., and Huhdanpaa, H. "The Quickhull Algorithm for Complex Hulls," *ACM Transactions on Mathematical Software*, Vol. 22, No. 4, 1996. doi:10.1145/235815.235821.

[14] Seidel, R. "The Upper Bound Theorem for Polytopes: An Easy Proof of Its Asymptotic Version," *Computational Geometry*, Vol. 5, No. 2, 1995, pp. 115–116. doi:10.1016/0925-7721(95)00013-Y.

[15] Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G. "API Design for Machine Learning Software: Experiences from the Scikit-Learn Project," Proceedings of the European Conference on Machine Learning and Principles and Practices of Knowledge Discovery in Databases, 2013.

[16] 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, É. "Scikit-Learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.

[17] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. "TensorFlow: A System for Large-Scale Machine Learning," Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, Savannah, Georgia, November 2-4, 2016.

[18] Norman, T. R., Shinoda, P., Peterson, R. L., and Datta, A. "Full-Scale Wind Tunnel Test of the UH-60A Airloads Rotor," Proceedings of the American Helicopter Society 67th Annual Forum, Virginia Beach, Virginia, May 3-5, 2011.

[19] Bousman, W., and Kufeld, R. "UH-60A Airloads Catalog," NASA TM-2005-212827, August 2005.

[20] Eddy, J., and Lewis, K. "Effective Generation of Pareto Sets Using Genetic Programming," Proceedings of the ASME International Design Technical Conferences, Design Automation Conference, Pittsburgh, PA, 2001.

[21] Vanderplaats, G. N. "CONMIN – a FORTRAN Program for Constrained Function Minimization," NASA TM X-62282, 1973.

[22] Rosenbrock, H., H. "An Automatic Method for Finding the Greatest or Least Value of a Function," *The Computer Journal*, Vol. 3, No. 3, 1960, pp. 175–184. doi:10.1093/comjnl/3.3.175.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
June 2021	Final	

4. TITLE AND SUBTITLE

Rotor Blade Design Framework for Airfoil Shape Optimization with Performance Considerations

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

0603465

6. AUTHOR(S)

Luke D. Allen, Joon W. Lim, Robert B. Haehnel, and Ian D. Dettwiller

5d. PROJECT NUMBER

AL3

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

See next page.

8. PERFORMING ORGANIZATION REPORT NUMBER

ERDC MP-21-5

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Headquarters, U.S. Army Corps of Engineers
Washington, DC 20314

10. SPONSOR/MONITOR'S ACRONYM(S)

USACE

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

This paper was originally presented at the AIAA SciTech Forum 11–15 and 19–21 January 2021 and published online on 4 January 2021.

14. ABSTRACT

A framework for optimizing rotor blade airfoil shape is presented. The framework uses two digital workflows created within the Galaxy Simulation Builder (GSB) software package. The first is a workflow enabling the automated creation of a surrogate model for predicting airfoil performance coefficients. An accurate surrogate model for the rapid generation of airfoil coefficient tables has been developed using linear interpolation techniques that is based on C81Gen and ARC2D CFD codes. The second workflow defines the rotor blade optimization problem using GSB and the Dakota numerical optimization library. The presented example uses a quasi-Newton optimization algorithm to optimize the tip region of the UH-60A main rotor blade with respect to vehicle performance. This is accomplished by morphing the blade tip airfoil shape for optimum power, subject to a constraint on the maximum pitch link load.

15. SUBJECT TERMS

Helicopters—Design; Rotors (Helicopters)—Design; Helicopters--Computer simulation; Rotors (Helicopters)--Computer simulation; Genetic algorithms

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	SAR	21	

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Cold Regions Research and Engineering Laboratory
U.S. Army Engineer Research and Development Center
72 Lyme Road
Hanover, NH 03755

Development and Engineering Center
U.S. Army Aviation and Missile Research
75 B. S. Hood Road
Moffett Field, CA 94035

Information Technology Laboratory
U.S. Army Engineer Research and Development Center
3090 Halls Ferry Road
Vicksburg, MS 39180