RCOTools Upgrades: Scalable Comprehensive Rotorcraft Design and Coupled Optimization

Carlos Pereyra

Aerospace Engineer
NASA Ames Research Center
Moffett Field, CA

Esther Kung

Undergraduate
University of California, Davis
One Shields Ave, CA

ABSTRACT

This paper presents updates to The Rotorcraft Optimization Tools (RCOTools) package to streamline iterative rotorcraft comprehensive design. The work is presented in three parts. Part I. a brief introduction to our simplified API is shown, in addition to a new mission profile dashboard. Part II. demonstrates high-throughput using the embarrassingly parallel paradigm to produce large-scale datasets structured by simple design of experiments (DOE) as shown by our discussion on urban air mobility (UAM) emission minimization. Such datasets provide a necessary component for rapid database generation and supervised machine learning. Part III. the API is used to couple rotor performance and sizing optimization. A simple technique for ultra-fast hover calibration is given, as well as possible applications for neural network modeling in comprehensive design. These enhancements accelerate design workflows and enable data-driven approaches for next-generation urban air mobility and planetary rotorcraft concepts.

NOTATION

AR	Aspect ratio (-)
FM	Figure of merit (-)
C_T/σ	Blade loading (-)
N_b	Number of blades (-)
σ	Solidity $N_b c/(\pi R)$ (-)
β_j	Twist rate at j'th radial station (deg / R)
κ	induced power factor (-)
cdo	mean drag coefficient (-)
μ_x	Advancing ratio in $\hat{x}$ direction, horizontal (-)
μ_z	Advancing ratio in $\hat{z}$ direction, vertical (-)
R	Blade radius (ft)
V_{tip}	Tip speed multiplied by 10^2 (ft/s)
h_{alt}	Altitude multiplied by 10^3(ft)
P_{eng}	Engine power (hp)
$W_{\text{fuel-miss}}$	Mission fuel (burned fuel and reserve) (lb)
W_{E}	Weight empty (lb)
Q_{D}	Disk loading (lb / ft^2)
Q_{W}	Wing loading (lb / ft^2)

INTRODUCTION

Optimization of Vertical Take-off and Landing vehicles (VTOL) present myriad of design variables to be explored in the conceptual design stage. Capturing the sensitivity of a design to i.) aeromechanical performance, ii.) sizing, iii.) battery-propulsion, iv.) controls, v.) aero-elastic (ie. structural) aspects (and many more) compound the number of design cases to consider computationally. Examples include D. Sarojini et. al. demonstrating up to 108 parameters in geometry optimization of an urban air mobility concept demonstrator for the Comprehensive Aircraft high-Dimensional DEsign Environment (CADEE) package (Ref. 1).

Similar to the CADEE effort, RCOTools seeks to couple low-fidelity aerodynamics solvers, aeroelastic, acoustic, and controls applications for rotorcraft design (Ref. 2). While RCOTools is continuously adding new features (as will be discussed in section PART I. PACKAGE UPDATES), its interface mitigates setup and configuration limitations for simple optimization studies using Verified and Validated (V&V) software.

This framework also serves as a systematic approach for supervised machine learning (ML) and database development. It is possible to develop accurate math-model representations of concept vehicles from comprehensive design analysis software. Primary support at current time includes

- NASA's Design and Analysis of Rotorcraft (NDARC) (Ref. 3),
- Comprehensive Analytical Model of Rotorcraft Aerodynamics and Dynamics (CAMRADII) (Ref. 4),
- Comprehensive Hierarchical Aeromechanics Rotorcraft Model (CHARM) (Refs. 5, 6),

Presented at the Vertical Flight Society's Vertical Lift Aircraft Design & Aeromechanics Specialist's Conference, San Jose, CA, USA, Jan. 27-29, 2026. This is a work of the US Government and is not subject to copyright protection in the US

as well as IXGEN and NPSS.

The first development of Rotorcraft Optimization Tools (RCOTools), a rotary-wing optimization framework at the comprehensive design level (Ref. 2) includes aspects of single discipline optimization (e.g. confined to NDARC, or CAMRADII only). The same functionality is performed in a simplified, and scalable manner, amenable to surrogate modeling as demonstrated in Sec. PART II. USAGE DEMONSTRATION.

RCOTools's capability to couple NDARC and CAMRADII is demonstrated on the 90 passenger tandem compound (TC90) design (Refs. 7, 8). The objective of coupling is to provide a systematic approach to simultaneously determining optimal i.) rotor performance and ii.) sizing. A case study of this is demonstrated in Sec. PART III. TC90 CASE STUDY. The tandem compound has historical significance as a design which showed impressive performance for civil transport (Ref. 9). The choice of studying the TC90 serves as a classic benchmark example for single discipline optimization explored throughout this document, and may be formulated as multi-objective optimization problem. The work demonstrated here outlines the necessary steps required to complete an automated NDARC to CAMRADII design loop with the TC90.

Software development on the RCOTools package is continuous and suggestions are welcomed. The heritage of RCOTools can be traced back to initial NDARC automation software (Ref. 10), now current versions (2.4.5+) of RCOTools support more applications and include examples of parallelized computing using *dask* (Refs. 11, 12). Asynchronous operation allow users to complete large numbers of cases. Such examples can be adapted to quickly generate large-scale datasets for any conceptual model - a necessary step in composing surrogates to be used in single and multi-objective optimization. To learn this package, there is a plethora of tutorials provided that demonstrate best practices for operational efficiency.

NDARC is a commonly used tool in conceptual design for rotor and fixed wing sizing. For the lift plus cruise (LPC) configuration recent NDARC usage may be seen in these articles (Refs. 13, 14). It is asserted here that any NDARC design may be enhanced by accurately considering drag and induced power surrogates, as this may have a non-negligible effect on sizing (and component requirements such as engine power). A simple solution for rotor-calibration is given in Sec. TC90 Rotor Calibration; while, results of direct power coefficient updates will be presented in Sec. TC90 Size and Comprehensive Analysis Coupling Results.

A final and brief note on leveraging these physics-based modeling software applications is offered for composing neural network representations. Such math-models may be easily utilized for optimization as discussed in Sec. Initial Steps Toward Neural Surrogate Modeling.

PART I. PACKAGE UPDATES

RCOTools is a Python package that wraps around command line interface (CLI) applications such as NDARC, CAMRADII via file input / output (IO). These codes output custom format that RCOTools parses. Upgrades were performed to improve parsing efficiency and to simplify workflow automation.

Architecture

Input and output data are modeled by first importing the application of interest's modules,

```
import rcotools.ndarc as ndarc
import rcotools.camrad as camrad
```

which give access to class structures per file type (see the next sections).

The Python language offers portability across various computing systems (e.g. Macintosh, Windows, Linux, etc.) and mitigates effort to install dependencies (via *setuptools*). The wrapper's performance is not bandwidth limited nor does it introduce much overhead, thus it is not necessary to use lower-level languages. Additionally, Python's strong data-structuring flexibility makes it an obvious choice for data transformations.

This wrapper is designed as an object oriented framework ensuring *extensibility*, and *flexibility* for additional features. To perform any optimization case involving the aforementioned design executables, it is an essential step to read / modify input parameters. Legacy features (i.e. `getByAccessText()`) are preserved, while dictionary-based data-structures are preferred in later versions +Version 2.4.5. Namelist elements are mapped exactly as dictionary elements for more intuitive interfacing.

NDARC File Model

NDARC file types, such as input namelist conventionally given `.list`, `.njob` extensions and output ASCII text files (e.g. `.perf`) are represented by Python classes. For example, a namelist file may be read as,

```
nml = ndarc.ListFile()
nml.read(filename=f'tc90.list')
```

then input FORTRAN namelist contents are represented in Python as dictionary objects.

```
nml.params['Rotor 1']['diskload'] = d
nml.params['Rotor 1']['Vtip_ref'] = vtip
nml.update() # updates file contents
```

NDARC input file contents are easily modified by updating `params` dictionary elements held in memory, where these changes are piped directly to NDARC via standard input [1].

Initial namelists are parsed by generalized regex expressions, matching each section based on reusable patterns. The section identification allows for inline replacement of modified parameters, thereby minimizing repeated code and conserving initial namelist structure, which makes it easier to debug file changes and identify potential input errors.

CAMRADII IO Module

CAMRADII file types, such as input namelists are conventionally given `.list` extensions.

```
c_nml = camrad.ListFile()
c_nml.read('tc90_hov.list')
```

Similar to the NDARC parameter adjustment, CAMRADII parameters may be defined like so.

```
c_nml.params['TRIM']['CTTRIM'] = cts
...
c_nml.params['ROTOR']['STRUCTURE'] \
            ['ROTOR 1']['RADIUS'] = r
c_nml.params['ROTOR']['STRUCTURE'] \
            ['ROTOR 1']['SIGMA'] = sigma
c_nml.update()
```

The CAMRADII IO Module reads and modifies files in the same manner as the NDARC IO module. RCOTools wraps the CAMRADII module in three functions to generate the necessary supplementary files to execute CAMRAD cases: `run_airfoil()` produces the airfoil deck, `run_rotor()` makes rotor geometry input, and `run_case()` sets up the rotor conditions; respective parameters may be read and modified in each of these functions in the same way as the NDARC IO module.

Ideal NDARC-CAMRADII Coupling Workflow

With the IO modules, RCOTools enables a intuitive and hands-off approach to comprehensively sizing rotorcraft. Each design variable found in the input files maps directly to python dictionary elements.

Sharing Python scripts for iterative conceptual design stage is standardized with this framework and enables more difficult design goals to be investigated. For instance the sizing-rotor performance procedure shown by the complete iterative design cycle follows the flow of data transfer shown in Fig. 1. For clarity the NDARC discipline represents sizing factors, and CAMRADII contributes aeroelastic performance.

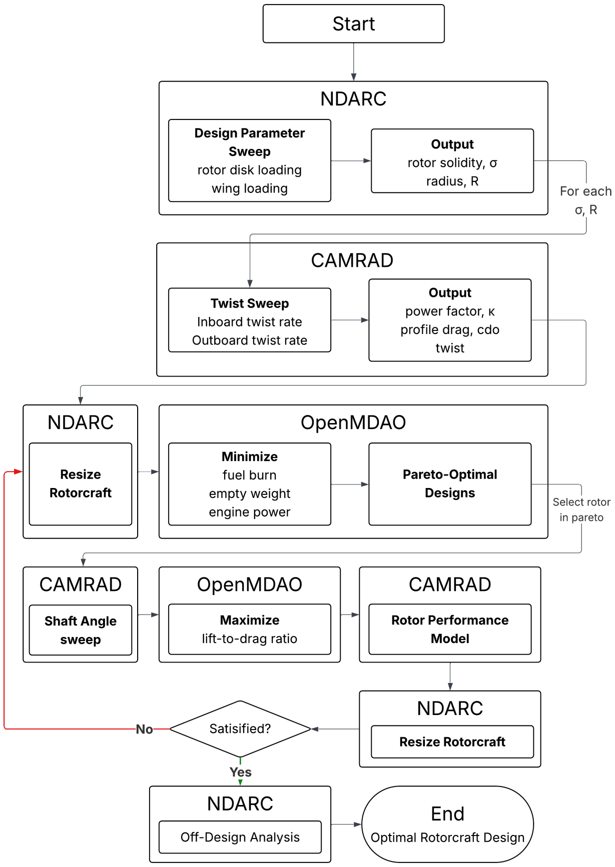

Fig. 1. Vehicle sizing flow in large civil compound helicopter by Johnson and Russell (Ref. 7).

Parallelizing RCOTools is no longer restricted to usage on Linux operating systems. Since these two underlying applications are singly threaded, RCOTools may compactly used in a embarrassingly parallelized fashion via a `compute()` function. The Dask framework for distributed computation is instrumental in generating large-scale datasets to represent the aircraft design space. The design space is either generated from a design of experiments or a Monte Carlo sampling based on metrics in a previous sizing step. In the former case, each parameter is swept in equal spacing to produce a regular grid of theoretical designs; in the latter case, the data may not present a structure. Depending on the regularity of the dataset, RCOTools has three methods to train a surrogate model for the design space: Spline interpolations for semi-structured data, a Kriging model for sparse irregular data, and neural network for various irregular data.

Mission Dashboard

The NDARC solution namelist file can be parsed into nested dictionaries to access each data entry. Since NDARC has particular classes of solutions that are mission-based, the nested dictionaries can be expanded into Pandas DataFrames, which is the basis of the NDARC mission visualization dashboard.

By calling the command `rco_ndarc_dash.exe` fol-

[1] NDARC input namelists files do not need to be written to disk, however it is a good practice to store them for later inspection

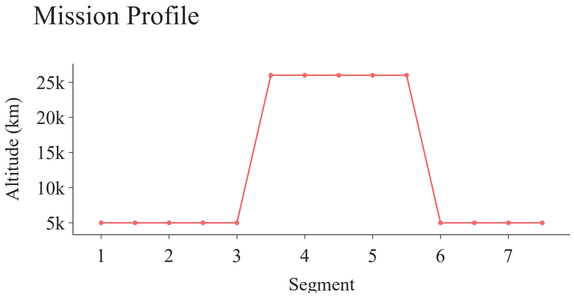

Fig. 2. Altitude as a function of mission segment dashboard visual for TC90 mission profile demonstrated in case study section.

lowed by the file path of the solution file, the mission visualization dashboard will be launched as a local server and viewed in a browser. The dashboard is ribbon-based and tabs are categorized based on:

- Mission Segments
- Mission Parameters
- Flight Conditions
- and Flight State.

Each category may be selected so users can quickly access and glance at data, such as in Figure 2. These categories are derived from the NDARC dictionary structure. To gain an intuitive sense of the mission path (and other data) per segment drop down options (in the lower part of the interface) allow users to select views based on: segment, flight time, air distance, or ground distance. This enables users and others to get a better sense of the NDARC setup before running a case.

PART II. USAGE DEMONSTRATION

UAM Rotor Sizing

RCOTools may be readily applied to size optimization problems of urban air mobility (UAM) concept vehicles. Inspired by G. Willink's thesis on the UAM concept vehicles for the reduction of emissions and cost using NDARC (Ref. 15), a similar objective may be explicitly written as,

$$\begin{aligned} \min_{x} \quad & \hat{E}(x) \\ \text{s.t.} \quad & 1.5 \leq Q_{\text{D}} \leq 7 \\ & 4.5 \leq V_{\text{tip}} \leq 7 \\ & 6.5 \leq h_{\text{alt}} \leq 12.5 \end{aligned} \tag{1}$$

for the design variables $x = [Q_{\text{D}}, V_{\text{tip}}, h_{\text{alt}}]^T$. A surrogate math-model $\hat{E}$ for predicting Emissions Trading Scheme (ETS) may be fit and used in the objective in equation (1). In addition to this single objective optimization, another functional $\hat{F} \rightarrow [\text{ETS}, \text{Cost}]$ that predicts emissions and cost, is then used for efficient multiobjective optimization, according to equation (2).

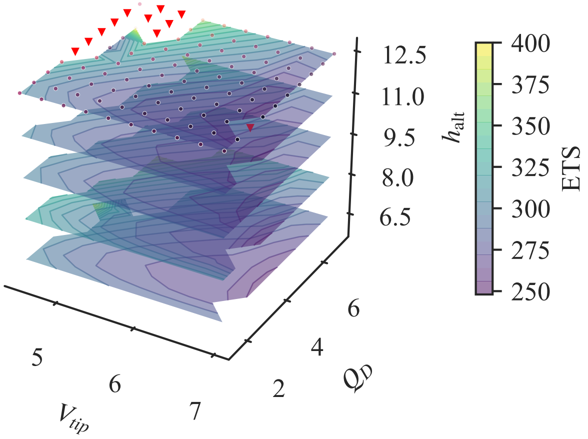

Fig. 3. UAM quadrotor reference vehicle as a function of three design variables (V_{tip}, Q_{D}, h_{alt}).

$$\min_{x} \quad \hat{F}(x) \tag{2}$$

The Emissions Trading Scheme is a built-in NDARC metric that evaluates emissions based on the total weight of CO_2, W_{CO_2}, produced by the fuel combustion and production, along with any other energy generation (Ref. 16), formulated as such

$$W_{CO_2} = K_{fuel} W_{burn} + K_{energy} E_{burn}$$

where default K_{fuel} (lb/lb) and K_{energy} (kg/MJ) were used (and may be referred to in (Ref. 17). Given that aircraft missions spend a substantial time in cruise, the emissions generated in the cruise segment can be swept for operating parameters $x = [Q_{\text{D}}, V_{\text{tip}}, h_{\text{alt}}]^T$ to determine the minimal ETS output E, for the ranges specified in equation (3)

$$\begin{aligned} Q_D &\sim [1.5, 7] \in \mathcal{R}^{20} \\ V_{\text{tip}} &\sim [4.5, 7] \in \mathcal{R}^{20} \\ h_{alt} &\sim [6.5, 12.5] \in \mathcal{R}^{20}. \end{aligned} \tag{3}$$

This sweep composes a dataset of $N = 8 \times 10^3$ samples, as visually represented in Figure 3. Each plane demonstrates emission contour maps, where the surfaces here appear to be smooth and convex which indicated that simple math-models may be fit to the dataset. The red points indicate nonconverged solutions, which can be explained by physically infeasible design input combinations. These occasionally nonconverged points on a regular grid define the data as a "semi-structured" dataset. Spline and Kriging models were trained on the dataset to create continuous models for gradient-based optimization.

For spline models, OpenMDAO provides a semi-structured interpolation module with options to train a linear spline, Lagrange second and third order spline, or an Akima spline on the data (Ref. 18). For Kriging models provided by Surrogate Modeling Toolbox (SMT) (Ref. 19), exponential (Ornstein-Uhlenbeck), squared (Guassian), exponential with variable

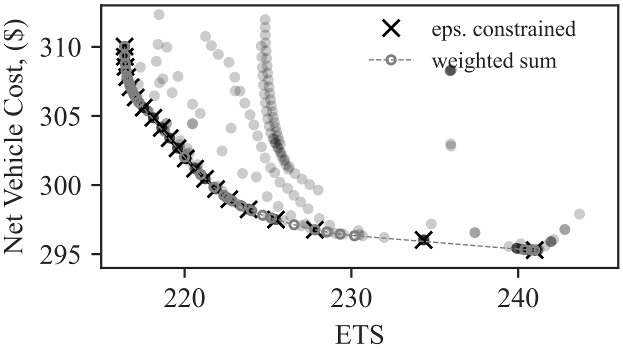

Fig. 4. UAM concept vehicle quadrotor (turboshaft) Pareto front layered over dataset (gray circles).

power (0.0, 2.0], Matérn 5/2, Matérn 3/2, and exponential squared sine correlation functions can be chosen to construct the surrogate model. The Kriging models were significantly more time-consuming while the spline models were fast to train and versatile in prediction array shape.

A test dataset 30% of the trained data is randomly generated in between the grid to validate the accuracy of the models trained. It was observed that spline models produced lower mean squared errors, corresponding to higher accuracy. The second order Lagrange interpolation produced the lowest mean square error, hence was used for determining optimized parameters x of surrogate model $\hat{E}$. These results are seen in Table 1.

Table 1. UAM emissions optimized operating parameters, turboshaft propulsion model.

$Q_{\mathbf{D}}$	$V_{\mathbf{tip}}$	$h_{\mathbf{alt}}$
2.659	7	6.82

To facilitate multi-objective optimization, RCOTools includes a demonstration of trade-off between ETS and Cost with a Pareto front. The weighted sum method is first implemented to provide context for the array of trade-off options in multi-objective optimization, then the epsilon constraint method is used to create a more robust Pareto front (Ref. 20). The set of optimal values is visualized in Figure 4.

This example entailing surrogate model to optimization routine is packaged as a jupyter notebook walkthrough in RCOTools, enabling easy adaptation to other NDARC studies. In the early design stages, surrogate models may be advantageous for design space visualization and optimization studies, instead of using conventional discrete derivatives that do not thoroughly explore the design space.

PART III. TC90 CASE STUDY

A major focus of RCOTools is to enable coupling between sizing and rotor comprehensive performance as predicted by lifting line vortex models (e.g. CAMRADII, CHARM, etc.).

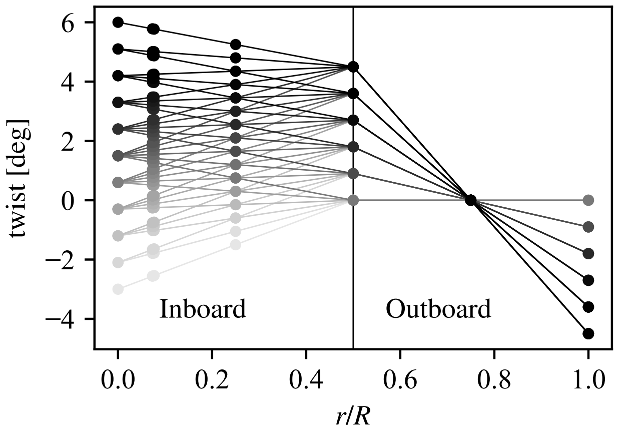

Fig. 5. Candidate pitch angles swept over in large scale hover runs.

Earlier work by C. Russell and W. Johnson outlined comprehensive analysis of the tandem compound 90 passenger rotorcraft for civil transport (Ref. 7). The TC90 serves as a fundamental benchmark case for sizing and aeromechanical coupling of a LPC type configuration, where the objective is to optimize size and rotor geometry for hover and forward flight by determining optimal radial twist distribution and blade-planform (under the constraint $10 \leq AR \leq 20$).

For clarity, twist values determined by the linear piecewise function

$$\alpha(r_k) = \begin{cases} \beta_0(r_k - 1/2) - \frac{1}{4}\beta_1 & \text{for} \quad k \in \{0,1,2,3\} \\ \beta_1(r_k - 3/4) & \text{for} \quad k \in \{4,5,6\} \end{cases} \tag{4}$$

dependent on inboard / outboard twist rates β_0 and β_1 visually represented in Figure 5. The original investigation defines the radial stations at $r_k = [0, 11/150, 23/300, 0.25, 0.5, 0.75, 1]$ and are also used here for consistency.

The following sections investigate setup and automation for enhancing the initial compound helicopter optimization by iteratively updating rotor size based on corresponding induced / profile power data. The analysis demonstrated here is not restricted to the TC90 and may in general be applied to planetary rotorcraft as well as future urban air mobility use concept vehicles.

TC90 Rotor Optimum Sizing Problem

The prerequisite for any rotor optimization is a baseline rotor radius and solidity choice. Here, blades are considered constant chord and $N_b = 7$. An initial choice of rotor configuration may be determined by minimizing the disk loading. The initial TC90 analysis in (Ref. 7) found $Q_D \approx 15$ (lb/ ft^2) worked reasonably well for a blade aspect ratio of $AR \approx 17.3$ (thus satisfying the desired constraint $AR \leq 20$); thereafter, Q_D was fixed and the wing loading Q_W was swept in search of lower vehicle weight and fuel burn, thus constituting fixed point iteration. Then finally the optimal blade twist distribution (and trim) is found using the Q_D, Q_W pair. In short, the sizing portion of this early design optimization is formulated as,

$$\begin{aligned} \min_{x} \quad & \|(W_E(x), W_{\text{fuel-miss}}(x), P_{\text{eng}}(x))\|_2 \\ \text{s.t.} \quad & 10 \leq AR \leq 20 \end{aligned} \tag{5}$$

for the inputs $x = [Q_D, Q_W, r, \sigma, C_T/\sigma]$). The seminal RCOTools article took steps to go beyond fixed-point iteration, and explored this design space both using DOE and a discrete derivative approach (Ref. 2); however, for such changes to be made to the rotor geometry the rotor must be re-trimmed in order to determine an accurate power model.

Coupling this minimization with trimmed rotor solutions may have non-negligible impacts. Design variables such as empty weight W_{E}, fuel usage $W_{\text{fuel-miss}}$, and engine power P_{eng} are directly linked to rotor profile power, and thus the correct usage of NDARC requires an accurate description of the induced / profile power surrogate. This requires either a user input an accurate *rotor-calibration*, or insert κ and *cdo* per $(Q_D, Q_W, r, \sigma, C_T/\sigma)$ configuration. RCOTools now includes a (hover) calibration routine, which is used on the published TC90 optimal rotor, and then a fully multidisciplinary optimization is performed. Interestingly, the early TC90 sizing used the CH90 rotor calibration, which differed significantly from the calibration shown later, so those results are also presented for comparison.

TC90 Rotor Calibration

Since NDARC's inception, rotor-calibration has been a subject of investigation. In fact there exists a tool to aid in the process of obtaining calibrations for both hover and forward-flight (Ref. 21). For hover, a simpler approach based on nonlinear least-squares projection exists which yields accurate parameter calibration.

Rotor calibration may be reformulated as a nonlinear least-squares (NLS) problem for determining surrogate coefficients. NLS methods are widely available via Python (e.g. Scipy) and are designed to minimize the residual error. In hover, W. Johnson's rotor calibration manual gives the relation,

$$\begin{aligned} \kappa_h(C_T/\sigma, \mu, \mu_x) =& \kappa_{\text{hover}} + \kappa_{\text{h1}}\Delta_h + \kappa_{\text{h2}}\left|\Delta_h\right|^{X_{h2}} \\ & + (\kappa_{\text{climb}} - \kappa_{\text{hover}})\, T \end{aligned} \tag{6}$$

used as a scaling factor for induced power $p_i = \kappa p_{\text{ideal}}$ (Ref. 22) and for hover $\kappa = \kappa_h$ for $\mu = 0$ according to (Ref. 23, page 23). Here $|\Delta_h| = C_T/\sigma - (C_T/\sigma)_{\text{ind}}$ and the transitionary variable is given by,

$$T = \begin{cases} 0 & \text{if } \mu_z = 0 \\ 1 & \text{else if } \mu_z \neq 0,\, \frac{C_T}{\sigma} = 0 \\ \frac{2}{\pi}\arctan\left(\left(\frac{|\mu_z|/\lambda_h}{M_{\text{axial}}}\right)^{X_{\text{axial}}}\right) & \text{else if } \mu_z \neq 0,\, \frac{C_T}{\sigma} > 0 \end{cases}$$

where only hover is considered thus $T = 0$.

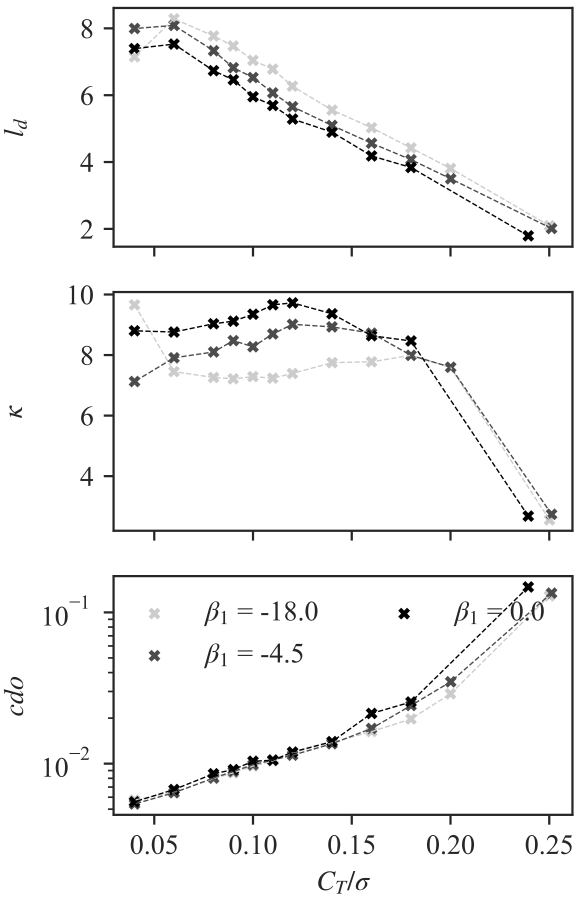

Fig. 6. TC90 Forward flight CAMRADII modeling predictions for $R = 46.2$, $\sigma = 0.123$ and $V_{kts} = 225$ at various outboard twist, rotor-geometry matches C. Russell optimal design.

By eliminating the absolute value in (6) and rewriting the relation with a smoothing constant $\varepsilon = 10^{-10}$ so that $|\Delta_h|$ instead becomes,

$$|\Delta_h| = \left[(C_T/\sigma - (C_T/\sigma)_{\text{ind}})^2 + \varepsilon^2\right]^{1/2} \tag{7}$$

yields a differentiable function. This means that a residual $\delta = y - \kappa_h(x)$ may be defined and easily deployed in NLS, where $x = [C_T/\sigma, \mu, \mu_x]^T$ and y is the computed induced power via comprehensive analysis (e.g. CAMRADII, CHARM, etc.).

Similarly, the surrogate model for mean profile power is given by,

$$cdo^{\text{hov}}(C_T/\sigma, \mu, \mu_x) = d_0^{\text{hel}} + d_1^{\text{hel}}\Delta + d_2^{\text{hel}}\Delta^2 + d^{\text{sep}}\Delta_{\text{sep}}^{X_{\text{sep}}} \tag{8}$$

where,

$$\begin{aligned} \Delta &= |C_T/\sigma - (C_T/\sigma)_{\text{Dmin}}| \\ \Delta_{\text{sep}} &= |C_T/\sigma| - (C_T/\sigma)_{\text{sep}} \end{aligned}$$

and the coefficients that compose this regression are explicitly seen in Tables 2 and 3. Once again a smoothing parameter may be applied so that,

$$\begin{aligned} \Delta &= ((C_T/\sigma - \alpha_2)^2 + \varepsilon^2)^{1/2} \quad \text{and} \quad \alpha_2 = (C_T/\sigma)_{\text{Dmin}} \\ \Delta_{\text{sep}} &= ((C_T/\sigma - \alpha_5)^2 + \varepsilon^2)^{1/2} \quad \text{and} \quad \alpha_5 = (C_T/\sigma)_{\text{sep}} \end{aligned} \tag{9}$$

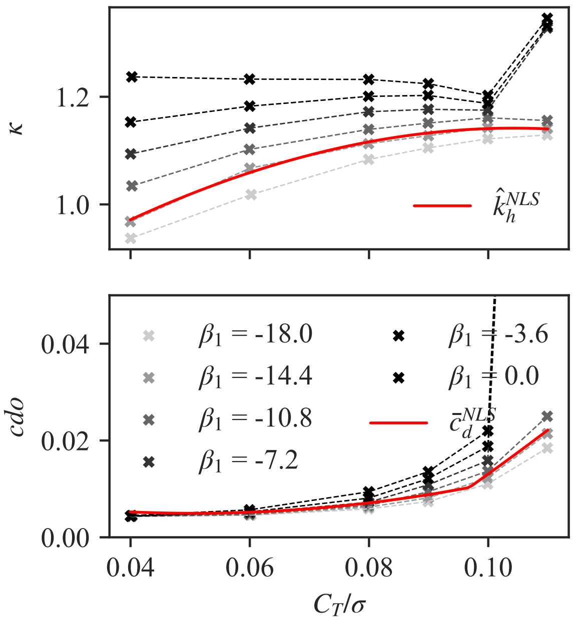

Fig. 7. TC90 hover CAMRADII run data (gray-scale) and calibration (red) for a single selected inboard / outboard twist rate $\beta_0 = 0$ and $\beta_1 = -14.4$ pair.

ensure differentiability. A thorough detailing of each variable's physical connection is found in (Ref. 22). By inserting the relations (7) and (9) into κ_h and *cdo* expressions respectively one may approximate original forms that include absolute values.

The optimal rotor found in (Ref. 7) ($r = 46.2$, $\sigma = 0.123$, $\beta_0 = 0$, and $\beta_1 \approx -14.4$) is used to demonstrate that these smoothed functions yield accurate predictions for a range of C_T/σ values. The resulting coefficients from the smoothed functions may be seen in Tables 2 and 3. Using the fitted model for predictions may be seen in Figure 7 and shown by the red curve.

In general, nonlinear least squares scheme should be a rapid computation (on the order of a millisecond) on a modern personal computer, and should yield accurate results. However, as a cautionary note NDARC's *constant* model may treat the κ functional-surrogate as $\kappa \rightarrow \kappa_{\text{hover}}$ when $\mu = \mu_z = 0$. This will trigger negative κ errors. Thus when using the first order residual $\delta = y - \kappa_h$ (where y is κ hover data) the resulting calibrations (for hover) may pose problems for NDARC. It is suggested that values in Table 2 be refined using the modification mentioned next or constrained NLS in the future.

A modified NLS residual is used to determine κ_h coefficients acceptable for NDARC sensitivity studies (see Appendix Sec. TC90 CALIBRATION REVISITED). Details of nonlinear least squares are found (Ref. 24) where it is shown that it involves the computation of Jacobian terms. This is the reason for introducing a smoothing factor. In fitting *cdo* values, the residual may be tuned to higher polynomial order to improve nonlinear solution process (for Figure 7 as shown here a regularization penalty factor is also added to avoid overfitting).

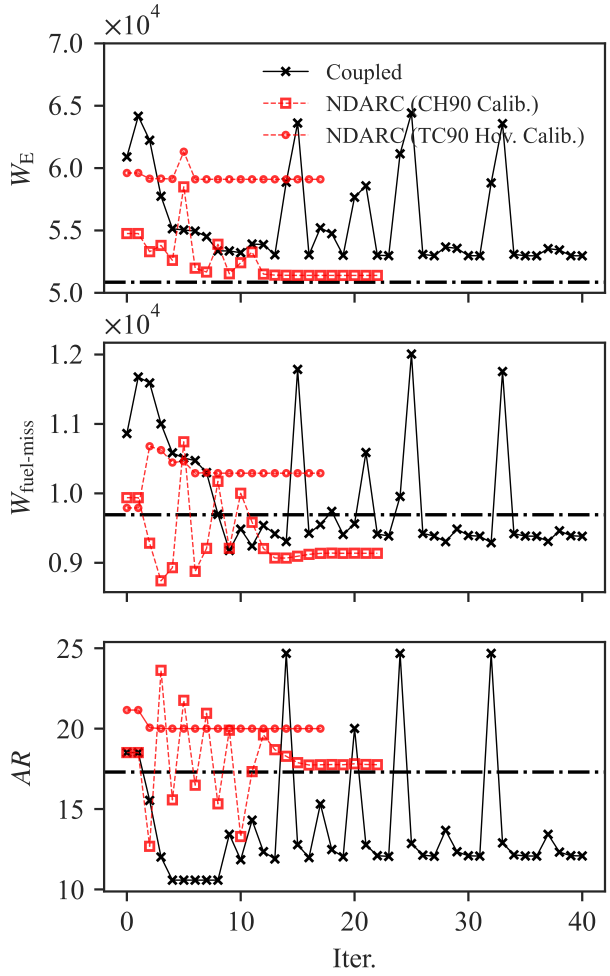

Fig. 8. TC90 coupled (solid-black), non-coupled optimization (red), and results from literature (dash-dotted-black) (Ref. 7).

A coupled NDARC-CAMRADII workflow would require a calibration for each planform geometry (i.e. radius, β pair) for a given sweep of C_T/σ. Thus for NDARC-CAMRADII coupling, calibration is not used. Instead, the nominal value of κ itself may be given per mission segment which in general supersedes any regression coefficient settings. This is true of the *cdo* setting as well. Our discussion of finding the optimal TC90 calibration for hover is given to demonstrate the sizing process is sensitive to the choice of calibration parameters, and that least-squares may be applied with some tuning.

TC90 Size and Comprehensive Analysis Coupling Results

The linkage of NDARC and CAMRADII fundamentally poses a semi-implicit problem. The two disciplines must use consistent blade planform and power coefficients.

Often in rotor sizing k and *cdo* induced / profile power coefficients are assumed to be fixed. However, to meet certain design goals, for instance some target FM or minimal empty weight, these coefficients may vary as a function of geometric variables (e.g. r,σ,C_T/σ and twist rates β_0, β_1) as seen in Fig-

Table 2. κ_h **regression coefficients** ($\beta_0 = 0$, $\beta_1 = -14.4$)

$k_{\textbf{hover}}$	$k_{\textbf{h1}}$	$k_{\textbf{h2}}$	$X_{\textbf{h2}}$	$(C_T/\sigma)_{ind}$
-26.33	67.74	0.49	85.41	0.49

Table 3. *cdo* **regression coefficients** ($\beta_0 = 0$, $\beta_1 = -14.4$)

$d_0^{\textbf{hel}}$	$d_1^{\textbf{hel}}$	$(C_T/\sigma)_{\textbf{Dmin}}$	$d_2^{\textbf{hel}}$	$d^{\textbf{sep}}$	$(C_T/\sigma)_{\textbf{sep}}$	$X_{\textbf{sep}}$
-0.73	0.32	0.097	2.15	1.78	1.98	-1.40

ure 6 and 7. In order to ensure both sizing and comprehensive analysis use consistent settings, one may define an iterative design loop as seen by the XDSM diagram in Figure 10 illustrating the coupling process. This figure is a detailed version of Figure 1. After enough iterations, an optimal rotor geometry will be determined; while also converging to consistent power coefficients shared between NDARC and CAMRADII.

How much of an impact does ensuring consistent power coefficients have on the overall vehicle design? To answer three scenarios are looked at,

a. NDARC runs with the initial CH90 calibration (default calibration),

b. NDARC runs with the TC90 hover calibration,

c. and NDARC-CAMRADII coupled runs.

To implement the optimization RCOTools is used as a conduit for the ν'th geometry design iteration and transferred to updated $\nu + 1$ design iteration. In one cycle, NDARC rotor geometry changes are passed to CAMRADII and may result in shifts in FM, k, and cdo, which are then passed back to NDARC for the rotor / vehicle to be resized ($\nu + 1$ iteration). A simple Python Notebook resource has been written to perform the aforementioned coupling procedure. Modeling this semi-implicit problem entails creation of explicit wrapper components. RCOTools is utilized in defining derived classes from OpenMDAO's `om.ExplicitComponent()` where this case for the TC90 may be adapted for other studies. Essentially, it is left up to the user to define the necessary dictionary keys to retrieve one of the many NDARC / CAMRADII outputs to be used as objective design variables. It is suggested here that users either use `rco_gui.exe`, `ndarcview.exe`, or `camradview.exe` to determine dictionary keys.

Coupling NDARC and CAMRADII (each iteration) produces non-negligible design differences. A comparison[2] between manual fixed-point iteration and coupling is seen in Table 4 where weight metrics (e.g. W_E, $W_{\text{fuel-miss}}$) and aspect ratio AR differ by ≈ 20%. While a significantly lower AR is found via coupling, it is difficult to say which method is closer to reality. It is interesting to note the twist rates converged towards similar values.

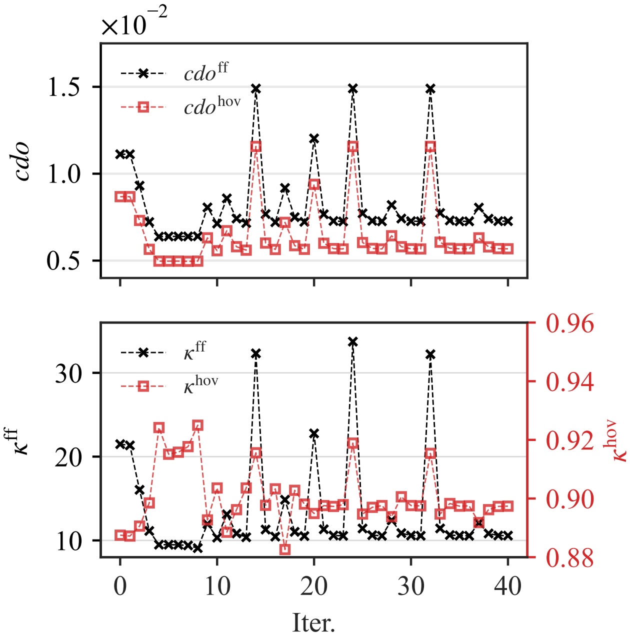

Fig. 9. TC90 power coefficients per iteration.

Converged coupling results were determined by the sequential least squares programming (SLSQP) algorithm. Average step sizing is initialized with $\delta = 5 \times 10^{-3}$. Each minimization procedure begin with the same wing load $Q_W^0 = 80$ (lb / ft^2) and disk load $Q_D^0 = 7$ (lb / ft^2), and are constrained so that the blade aspect ratio are within the range $10 \leq AR \leq 20$. Additionally, the norm in equation (5) did not apply a normalization since each quantity was roughly on the same scale. Variation in the objective function seen via the weight metrics may be seen per iteration in Figure 8. Larger spikes may be due to the relatively coarse step size. Refinements to the step size could be made; however, the coupling process had a wall time of nearly five hours.

As mentioned already, each design iteration uses updated induced power coefficient κ and mean drag coefficient cdo. The hover power coefficients were updated in the first, second and last mission segments, while the forward-flight coefficients are updated in the fifth and sixth segments shown in Figure 2. In addition to the adjustment of disk load Q_D and wing load Q_W the inboard / outboard twist are varied dur-

[2] The asterisk in Table 4 is used to indicate the fuel-burn instead of mission fuel.

Table 4. TC90 coupled and non-coupled results.

Variable	NDARC-CII (Coupled)	NDARC-CII (Manual Iter. (Ref. 7))	NDARC (CH90 Calib.)
W_{E}	51899.64	64789	50422.42
$W_{\text{fuel-miss}}$	9382.18	14545*	9136.81
P_{eng}	4891.76	6996	3937.38
AR	12.08	17.3	17.76
Q_D	12.26	15	8.34
Q_W	79.91	90	101.47
β_0	0.1	0.	-
β_1	-14.01	-15.	-
W.-time (s)	19561.09	-	2746.27

ing the optimization procedure in order to calculate the partial derivatives with respect to the design variables. After 40 iterations the coupling routine converged towards a local minimum $\|(W_E(x), W_{\text{fuel-miss}}(x), P_{\text{eng}}(x))\|_2$ and the power coefficients settled towards a plateau shown in Figure 9. Note, two scales are used in the lower pane of Figure 9, since the scale of the induced power coefficient in forward-flight is a decade higher than that of the hover power coefficient.

A very simple sensitivity test on the effects of calibration κ_h is performed using the NLS method mentioned earlier. A special tuning is used to obtain values in Sec. TC90 CALIBRATION REVISITED, which were subsequently used in the red circular points of Figure 8. The specially tuned coefficients match the data for the optimal rotor of (Ref. 7) (e.g.$r = 46.2$, $\sigma = 0.123$, $\beta_0 = 0.$, $\beta_1 = -14.4$) which in general seems to have over predicted many of the design values. Regardless, the hover calibration has a significant impact and must be considered during NDARC sizing. Using defaults may lead to misleading results. A direct comparison between the results for the two different calibrations (TC90 vs CH90) may be seen in Table 6. The CH90 forward flight coefficients were still being used. It was found in this study that the easier option to ensure κ, *cdo* consistency is to use the coupling procedure.

SUMMARY AND FUTURE WORK

Towards Pareto Optimal Rotor and Wing Designs

While the previous section focuses on a single objective formulation, expanding problem formulations may yield additional computation and convergence challenges. In practice, optimization still requires some experience and knowledge of the design limits by the engineer. To gain an intuitive sense of reasonable and optimal results given by NDARC, CAMRADII, etc. depending on the intent of analysis, a genetic algorithm multi-objective formulation, may aid in automation of this type of discovery. An example objective may be explored in the future to maximize FM, rotor lift share l_d as,

$$\begin{aligned} \max_{x} \quad & \left(\hat{H}_{NN}(x; w_j),\ \hat{F}_{NN}(x; w_k)\right) \\ \text{s.t.} \quad & 0.04 \le \sigma \le \frac{N_b}{20\pi} \\ & 20 \le r \le 40 \end{aligned} \tag{10}$$

where the given design variables are $x = [\beta_k, r, \sigma, C_T/\sigma]^T$ and the functionals $\hat{H}(x) = [FM, \kappa, cdo]$, and $\hat{F}(x) = [L_d, \kappa, cdo]$ are evaluations by CAMRADII (or CHARM, or a Neural Surrogate model as will be discussed in the next section). This formulation is non-trivial and generally leads to Pareto-optimal solutions, which require thousands of trimmed rotor cases to be performed, as is done in the low Reynolds number airfoil optimization tool (Ref. 25). As can be seen in Table 4 the wall times for 40 coupled iterations were 19,561 seconds ($\approx$ 5 hours on a 5.2 GHz CPU), which is largely due to the compute time of finding trim solutions.

Additional computational complexity may be introduced by exploring other initial conditions in search of a global minima. While the underlying applications work reasonably well for small numbers of cases, their solvers must be amenable to hyper threading and recasting solution methods for linear algebraic methods for faster solving times. Faster NDARC, and comprehensive analysis is a necessary step towards Monte-Carlo scale sampling of the Pareto front.

Initial Steps Toward Neural Surrogate Modeling

It is common to compute discrete derivatives of CLI applications based on some finite forward differencing steps when access to underlying nonlinear functions are not accessible directly (e.g. CAMRADII, CHARM, etc.). Efforts to make direct access to derivatives available entails building code that exposes function calls directly; such examples have been shown by the CADEE program which creates a symbolic translator (Ref. 1). However, in this work we avoid such issues by determining approximate math-models or surrogates which may accurately represent physical behavior, albeit with great amounts of training data.

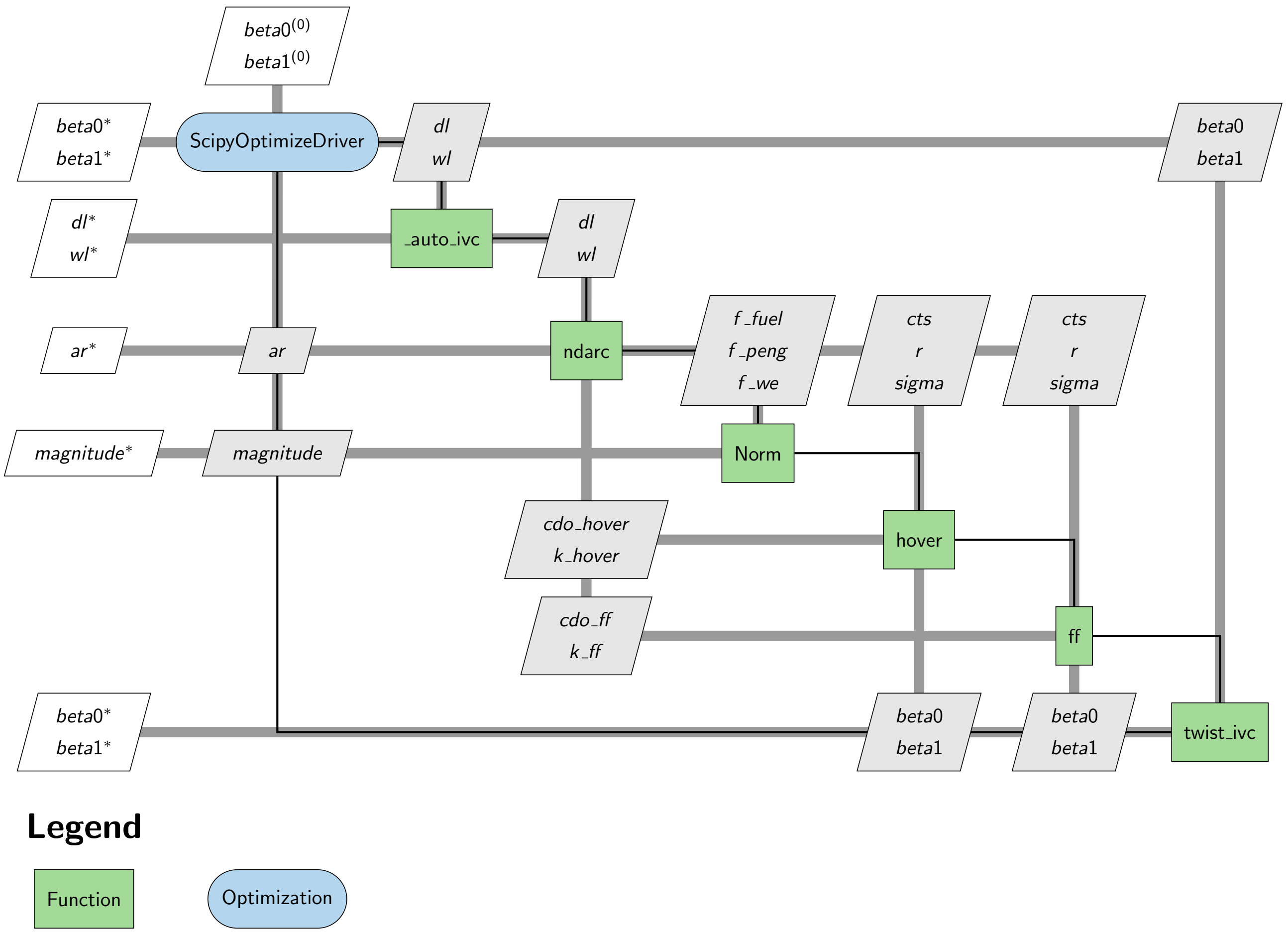

Fig. 10. TC90 sizing and rotor performance coupling XDSM diagram.

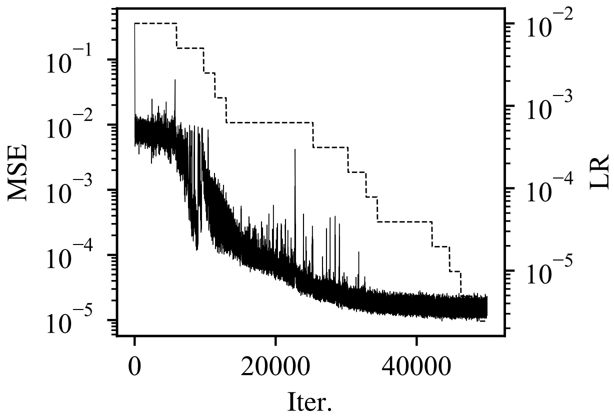

Fig. 11. Hover FM model training convergence (solid) and learning rate (dotted).

Approximate model forms of $\hat{H} \rightarrow \hat{H}_{NN}$ may be represented by simple multi-layer perceptrons or neural networks (NN). In our numerical training for accurate NN models, the sigmoidal activation function is applied to weight-coefficients, and optimally trained with random sampling permutations each iteration.

Back-propagation is performed on a simple loss function $\|y - \hat{y}\|_2$ with the Adam optimizer (Ref. 26) available through Optax (Ref. 27). Here, y is the desired output data and $\hat{y}$ is the approximate model (e.g. $\hat{H}_{NN}$ or $\hat{F}_{NN}$). To adequately capture the Hover figure of merit response data a three-layer NN (each with 50 nodes) was trained on a permuted subset of data. See residuals in Figure 11 where a dynamic plateau learning rate adjustment routine is leveraged. Convergence may be achieved in relatively short amounts of time on a personal computer. This is especially true using the JAX ecosystem of Python packages. Used here is the equinox package which exhibited noticeable speed up on CPU and is GPU compatible (Ref. 28).

Again, the benefits of leveraging NN models are numerous. First, they are easily auto-differentiable, and evaluate predictions rapidly, which could accelerate the coupling procedure. Second they are portable and easily distributable as look-up tables for when software is not available for collaborators. The example shown here only scratches the surface of using such models; however, we demonstrate that such data can be used to form surrogate models. In future work, model validation will be performed, as well as epistemic uncertainty quantification. One of the pitfalls of using such models is ensuring input is within reasonable design space. RCOtools seeks to include best practices on this in future releases.

Summary

We have demonstrated new features of a Python API that is currently distributed by the NASA software center. The package has been updated in order to:

- interface with commonly used comprehensive design codes NDARC, CAMRADII, Neural Surrogate,
- provide mission profile dashboard / visualization,
- perform inline file IO insertions,
- require minimal effort to parallelize (ie. Dask).

We have also demonstrated common use cases where RCOTools may be used to perform:

- sizing / performance surrogate modeling,
- constrained multiobjective optimization,
- and rotor calibration.

The last point is an essential component in the iterative design process of optimizing a rotor. Power must be considered in sizing, as demonstrated by optimal results presented in the coupling results of the TC90 section.

In a secondary paper, large-scale V&V coupling will be applied to UAM concept vehicles.

Author contact carlos.z.pereyra@nasa.gov.

ACKNOWLEDGEMENTS

The authors thank Larry Meyn for developing RCOTools and for his support. Gratitude is extended to Carl Russell for communication and for providing the original TC90 CAMRADII and NDARC input files. Appreciation is also given to Dr. Wayne Johnson for guidance and private correspondence, and to Ramesh Kolar for testing. Funding supporting this work were provided by Convergent Aeronautics Solutions (CAS), Revolutionary Vertical Lift Technology (RVLT), and Subsonic Vehicle Technologies and Tools (SVTT) programs.

APPENDIX

ROTOR CALIBRATION VALIDATION

Traditionally, rotorcraft calibration coefficients are determined manually through the use of spreadsheet methods. This process can be tedious, thus we demonstrate a straight forward way of accurately determining hover power coefficients (e.g. κ_{hover}, κ_{h1}, ..., $(C_T/\sigma)_{\text{ind}}$,) and mean drag coefficients (e.g. d_0^{hel}, d_1^{hel}, d_2^{hel}, ..., X_{sep}).

Typically, solutions found via their nonlinear least squares implementation are obtained in milliseconds. The benchmark cases shown here were quickly computed based on the formulation seen in Sec. TC90 Rotor Calibration. The resulting benchmark rotor-calibration fit is compared with spreadsheet values from (Ref. 22). Residuals may be seen in Figures 12 and 13. Programming solutions for κ_h rotor-calibration may be done in a few lines of code from the Scipy package (Ref. 29). See Python code snippet below.

```
import numpy as np
from scipy.optimize import least_squares

def f(p, x, eps = 1e-10):
    dh = x - p[2]
    return (p[0] + p[1]*dh +
            p[3]*((dh**2 + eps**2)
            **(p[4]/2.)))

def res(p, x, y):
    return y - f(p, x)

p0 = np.ones((5,))
x = data.cts.to_numpy()
y = data.kappa.to_numpy()
k_lsq = least_squares(res, p0, args=(x, y))
```

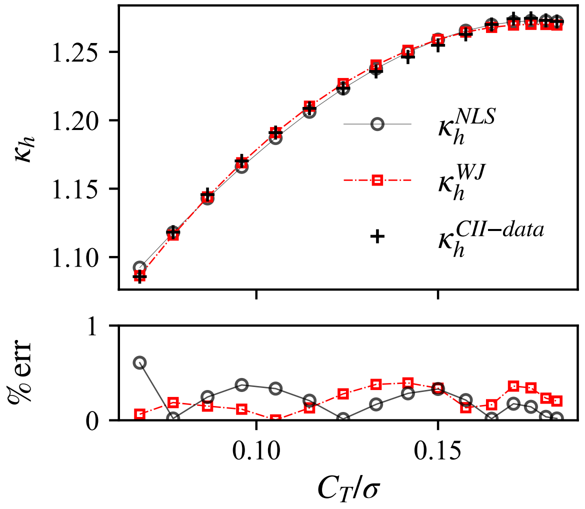

Fig. 12. Comparison of induced power coefficient κ_h. Predictions via least-squares (squares) and manual methods.

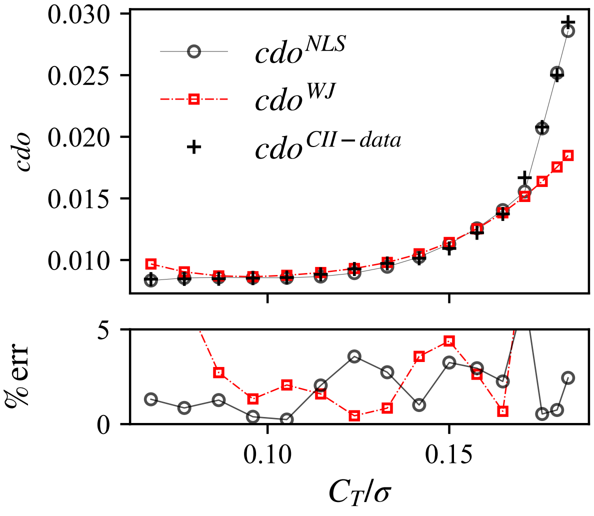

Fig. 13. Comparison of mean drag coefficient cdo. Predictions via least-squares (squares) and manual methods.

TC90 CALIBRATION REVISITED

NDARC returns an error due to input values for the coefficients in Tables 2 and 3. The residuals had to be fine tuned to values that yielded coefficients closer together. The following coefficients shown in Table 5 were used in the NDARC TC90 optimization and were determined using the residual $\delta = (y - \hat{k}_h)^2$. Results of running NDARC with these refinements are shown in the last column of Table 6 and compared with the default CH90 calibration.

Table 5. κ_h regression coefficients ($\beta_0 = 0$, $\beta_1 = -14.4$)

k_{hover}	k_{h1}	k_{h2}	X_{h2}	$(C_T/\sigma)_{ind}$
0.066	7.43	2.49	0.186	0.13

Table 6. TC90 coupled and non-coupled results.

Variable	NDARC	NDARC
	(CH90 Calib.)	(TC90 Hov. Calib.)
W_{E}	50422.42	57934.66
$W_{\text{fuel-miss}}$	9136.81	10290.60
P_{eng}	3937.38	5529.20
AR	17.76	20.00
Q_D	8.34	7.40
Q_W	101.47	107.42
β_0	-	-
β_1	-	-
W.-time (s)	2746.27	1119.01

REFERENCES

[1] Sarojini, D., Ruh, M. L., Joshy, A. J., Yan, J., Ivanov, A. K., Scotzniovsky, L., Fletcher, A. H., Orndorff, N. C., Sperry, M., Gandarillas, V. E., *et al.*, "Large-scale multidisciplinary design optimization of an evtol aircraft using comprehensive analysis," AIAA SciTech 2023 forum, 2023.

[2] Meyn, L. A., "Rotorcraft Optimization Tools: Incorporating Rotorcraft Design Codes into Multi-Disciplinary Design, Analysis, and Optimization," Rotorcraft Optimization Tools: Incorporating Rotorcraft Design Codes into Multi-Disciplinary Design, Analysis, and Optimization, 2018.

[3] Johnson, W., *NDARC - NASA Design and Analysis of Rotorcraft*, Ames Research Center, 2015.

[4] Johnson, W., "Technology Drivers in the Development of CAMRAD II," American helicopter society aeromechanics specialists conference, San Francisco, California, 1994.

[5] Wachspress, D. A., Quackenbush, T. R., and Boschitsch, A. H., "Rotorcraft interactional aerodynamics calculations with fast vortex/fast panel methods," ANNUAL FORUM PROCEEDINGS-AMERICAN HELICOPTER SOCIETY, Vol. 56, 2000.

[6] Wachspress, D. A., Quackenbush, T. R., and Boschitsch, A. H., "First-principles free-vortex wake analysis for helicopters and tiltrotors," ANNUAL FORUM PROCEEDINGS-AMERICAN HELICOPTER SOCIETY, Vol. 59, 2003.

[7] Russell, C. and Johnson, W., "Exploration of configuration options for a large civil compound helicopter," Paper ARC-E-DAA-TN6212, 69th American Helicopter Society Annual Forum, 2013.

[8] Russell, C. and Basset, P.-M., "Conceptual design of environmentally friendly rotorcraft-a comparison of nasa and onera approaches," Paper ARC-E-DAA-TN21833, AHS Annual Forum and Technology Display, 2015.

[9] Ormiston, R., "Revitalising advanced rotorcraft research–and the compound helicopter," *The Aeronautical Journal*, Vol. 120, (1223), 2016, pp. 83–129.

[10] Sinsay, J., Hadka, D., and Lego, S., "An integrated design environment for NDARC," AHS International Technical Meeting on Aeromechanics for Vertical Lift, 2016.

[11] Rocklin, M., "Dask: Parallel Computation with Blocked algorithms and Task Scheduling," Proceedings of the 14th Python in Science Conference, edited by K. Huff and J. Bergstra, 2015.

[12] Dask Development Team, *Dask: Library for dynamic task scheduling*, 2016.

[13] Vegh, J. M., Basset, P.-M., Beals, N., Scott, R., Perret, R., and Singh, R., "A comparison of three conceptual design approaches applied to an electric distributed lift aircraft," *CEAS Aeronautical Journal*, 2024, pp. 1–15.

[14] Gladfelter, M., He, C., Saberi, H., Caudle, D., Singh, R., Malpica, C., and Silva, C., "Enhancements, Verification, and VMS Integration of VTOL Concept Vehicle Simulation Models," 80th Vertical Flight Society's (VFS) Annual Forum & Technology Display, 2024.

[15] Willink, G. C., *Utilizing Regression to Optimize Key Variables for Minimizing CO2 Emissions and Fuel Consumption of Urban Air Mobility Vehicles*, Ph.D. thesis, The George Washington University, 2025.

[16] Johnson, W., *NASA Design and Analysis of Rotorcraft - Input and Data Structures*, NASA Ames Research Center, Moffett Field, CA, Version 1.18, 2025.

[17] Johnson, W., *NASA Design and Analysis of Rotorcraft - Theory*, NASA Ames Research Center, Moffett Field, CA, Version 1.16, 2022.

[18] Gray, J. S., Hwang, J. T., Martins, J. R. R. A., Moore, K. T., and Naylor, B. A., "OpenMDAO: An Open-Source Framework for Multidisciplinary Design, Analysis, and Optimization," *Structural and Multidisciplinary Optimization*, Vol. 59, 2019, pp. 1075–1104.
doi: 10.1007/s00158-019-02211-z

[19] Saves, P., Lafage, R., Bartoli, N., Diouane, Y., Bussemaker, J., Lefebvre, T., Hwang, J. T., Morlier, J., and Martins, J. R. R. A., "SMT 2.0: A Surrogate Modeling Toolbox with a focus on Hierarchical and Mixed Variables Gaussian Processes," *Advances in Engineering Sofware*, Vol. 188, 2024, pp. 103571.
doi: https://doi.org/10.1016/j.advengsoft.2023.103571

[20] Jasa, J., "Practical MDO," , 2021.

[21] Zappek, V. and Yavrucuk, I., "Fuel Cell Sizing for a UAV with Intermeshing Rotors using a Genetic Algorithm for NDARC Rotor Performance Calibration," FORUM 2023 - Vertical Flight Society 79th Annual Forum and Technology Display, 2023.

[22] NASA Ames Research Center, Moffett Field, CA, *Model Calibration - Rotor Performance*, NASA Ames Research Center, Moffett Field, CA, 2018.

[23] NASA Ames Research Center, Moffett Field, CA, *NDARC — NASA Design and Analysis of Rotorcraft Theoretical Basis and Architecture*, NASA Ames Research Center, Moffett Field, CA.

[24] Wright, S., Nocedal, J., *et al.*, "Numerical optimization," *Springer Science*, Vol. 35, (67-68), 1999, pp. 7.

[25] Koning, W. J., Perez, B. N. P., Cummings, H. V., Romander, E. A., and Johnson, W., "ELISA: A Tool for Optimization of Rotor Hover Performance at Low Reynolds Number in the Mars Atmosphere," 2024 Transformative Vertical Flight, 2024.

[26] Kingma, D. P. and Ba, J., "Adam: A Method for Stochastic Optimization," , 2017.

[27] DeepMind, Babuschkin, I., Baumli, K., Bell, A., Bhupatiraju, S., Bruce, J., Buchlovsky, P., Budden, D., Cai, T., Clark, A., Danihelka, I., Dedieu, A., Fantacci, C., Godwin, J., Jones, C., Hemsley, R., Hennigan, T., Hessel, M., Hou, S., Kapturowski, S., Keck, T., Kemaev, I., King, M., Kunesch, M., Martens, L., Merzic, H., Mikulik, V., Norman, T., Papamakarios, G., Quan, J., Ring, R., Ruiz, F., Sanchez, A., Sartran, L., Schneider, R., Sezener, E., Spencer, S., Srinivasan, S., Stanojević, M., Stokowiec, W., Wang, L., Zhou, G., and Viola, F., "The DeepMind JAX Ecosystem," , 2020.

[28] Kidger, P. and Garcia, C., "Equinox: neural networks in JAX via callable PyTrees and filtered transformations," *Differentiable Programming workshop at Neural Information Processing Systems 2021*, 2021.

[29] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, Vol. 17, 2020, pp. 261–272.
doi: 10.1038/s41592-019-0686-2