Optimization of Multiple Rotors

J. Spencer *
Brigham Young University, Provo, Utah, 84601

This report describes a project completed as part of my research for the Brigham Young University FLOW Lab to discover ideal conditions for airfoil performance. Computational stimulations of propellors allow researchers to perform low-risk studies and develop designs more quickly. This investigation found the optimal chord length magnification and the optimal rotation angle for a rotor's efficiency. This was done with a computational simulation in the Julia programming language. This paper defines the operating conditions used in this optimization, presents rotors that were optimized at several blade counts, and discusses areas of possible future development in this research. This paper found that for the operating conditions used, rotors with the cord reduced to the optimization's lower limit, 50 percent, with slight negative twist angles were the most efficient. These results could be applied to rotors designed to produce the maximum output power for a given input power requirement, without regard to other conditions like the thrust or torque coefficients. Future work could combine these findings with other conditions to find the most efficient rotor that can also provide a required amount of thrust or operate at a variety of different rotational velocities.

Nomenclature

J	Advance Ratio
α	Angle of Attack (radians)
ϕ	Angle of Rotation (radians)
M_b	Bending Moment (N·m)
c	Chord length (m)
D	Diameter (m)
c_d	Drag Coefficient
η	Efficiency
ρ	Fluid Density (kg/m^3)
v_∞	Freestream Velocity (m/s)

*Undergraduate Researcher, Brigham Young University FLOW Lab.

c_l	Lift Coefficient
C_P	Power Coefficient
RPM	Revolutions Per Minute
σ	Rotor Solidity
C_T	Thrust Coefficient
C_Q	Torque Coefficient

I. Introduction

Propellors come in a variety of different shapes and sizes. This paper describes how one propellor, a NACA 4412 airfoil attached to an APC 10x7 rotor, was optimized to perform more efficiently at a specific advance ratio. Because different blade counts could be more appropriate for different applications, blade counts of one, two, and three were each optimized and are compared in this report. This report produced useful code that could be applied to different rotors in the future.

A. Context

Computer simulations are very useful for modeling propellors. It would require significant investment of time, space, and money to perform an optimization like this, in which hundreds of slightly different propellors are compared. The first potential-flow codes, programs for modeling the pressure distribution of rotors, were developed in the 1960s and 1970s. Xfoil, the program used in this project, was first developed by MIT in the 1980s, with its current version dating to 2013. Xfoil.jl is the Julia wrapper used in this project, developed by Taylor McDonnell.*

B. Paper Outline

This paper first reviews the procedure used in the rotor optimization, including the equation used in the optimization and the initial conditions and constraints applied. Next, it reviews the results and compares them with each other. Finally, it draws conclusions based on the results of this experiment and discusses their meaning. All code used for each section of this report can also be accessed through a GitHub repository.†

II. Procedure

This project was performed using Julia programming language.‡ Jula is available for free and is useful for a variety of reasons. Like other languages, it can easily store data in vectors and perform rapid calculations with these objects. It is compiled just-in-time at runtime, which makes it run faster than interpreted languages like python without needing to

*More information about both Xfoil and Xfoil.jl is available at `https://flow.byu.edu/Xfoil.jl/stable/` and `https://en.wikipedia.org/wiki/XFOIL`

†This repository can be accessed in the Rotor-Design branch of `https://github.com/JoeSpencer1/497R-Projects/tree/Final-Report`

‡Julia is available at `https://julialang.org`.

be totally compiled before running like C. Being compiled just-in-time makes Julia faster than interpreted languages like python. Julia packages, including those used during this semester, are also easily available from a package manager.[§]

A. Computer Code

Some Julia packages used for this project include Xfoil.jl,[¶] CCBlade,jl,[‖] and SNOW.jl.[**] Xfoil.jl analyzes the lift, drag, and moment coefficients of an airfoil from its geometry and angle of attack, and CCBlade.jl calculates a rotor's thrust, power, and efficiency[**?**]. This investigation used SNOW.jl to repeat itself until it converged on an optimal solution.

B. Design Constraints

In the rotor design, the rotor is created and evaluated using Xfoil.jl and CCBlade.jl, as described previously. Data about the rotor, including its loads in the normal and tangential directions and its torque, were recorded. This data was then multiplied by a safety factor of $n = 1.1$ to determine the maximum allowable loads before the rotor would break or a different material would be required. $n = 1.1$ was the safety factor defined in the project prompt, but it was included in the code as an optional argument that could be adjusted for other circumstances. After constraints were determined, rotor variable types called *rotortest* were created with variable chord thickness magnification and rotation angle and the program used SNOW.jl to find the rotor with the optimal twist angle and chord thickness magnification for the objective function.

The objective function used in this optimization is very simple, as shown in equation (**??**).

$$f(x_1, x_2, x_3...) = \eta \tag{1}$$

.

In addition to restrictions on the moment and torque mentioned previously, the cord thickness was kept within a factor of two of the original and the twist angle was kept between $-90°$ and $90°$. These constraints are listed in table (**??**). They ensured that the optimization found a reasonable solution confined to real limits, shown in table (**??**).

[§]This article describes some of the benefits of Julia. `https://towardsdatascience.com/5-ways-julia-is-better-than-python-334cc66d64ae`

[¶]Xfoil.jl is available at `https://github.com/byuflowlab/Xfoil.jl`

[‖]CCBlade.jl is available at `https://github.com/byuflowlab/CCBlade.jl`

[**]SNOW.jl is available at `https://github.com/byuflowlab/SNOWl.jl`

Table I: Optimization objective, parameters, and constraints	
Maximize:	η
By varying:	scale of the chord, c
	twist angle, ϕ
Subject to	total torque $\leq$ 110% original
	normal moment $\leq$ 110% original
	tangential moment $\leq$ 110% original
	$-90° <$ twist angle $< 90°$
	50% < chord magnification < 200%

The optimized rotors each had other features in common between them, shown in the last entry of table (**??**). Ning (page 200) stated that the advance ratio is available from other variables already listed in the table by equation (**??**), in which v_∞ is the free stream velocity, n is the rotational velocity in revolutions per second, and D is the outer diameter [**?**].

$$J = \frac{v_\infty}{nD} \tag{2}$$

Table II: Input value limits in rotor design

Parameter	**Default Value**	**Minimum Value**	**Maximum Value**
Chord Length,	100% length	50%	200%
Twist Angle,	0°	−90°	90°
Rotational Velocity, RPM	6000 RPM		
Blade Count	2 blades	1 to 3)	
Hub-to-tip ratio	10%		
Air Density	1.225 kg/m^3		
Diameter	0.254 m (10 in.)		
Velocity	12 m/s (26.84 mph)		
Advance Ratio	0.472		

III. Optimization

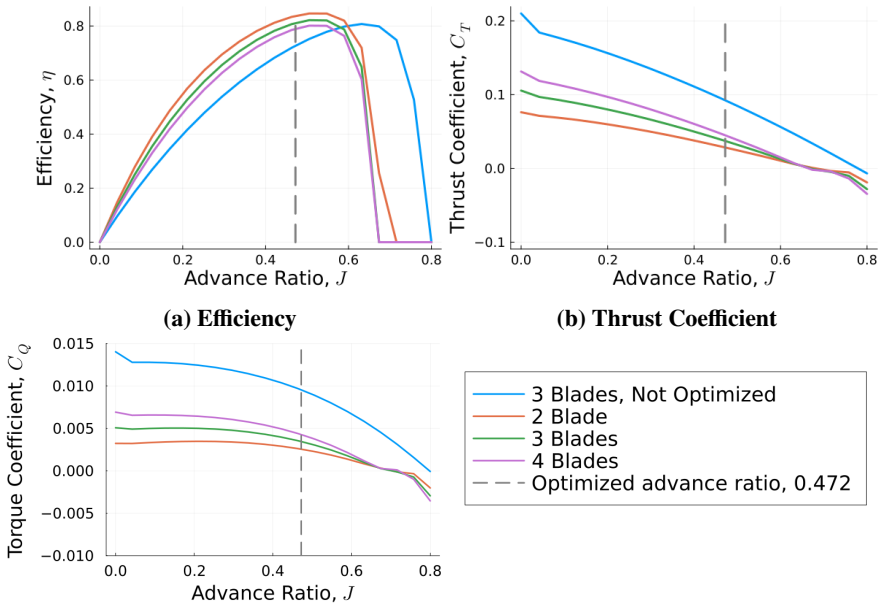
The objective function, equation (**??**), was optimized using SNOW.jl. The code was designed so that SNOW.jl found the maximum by finding the minimum of its negative. This is similar to the technique and sign convention described by Martins and Ning in page 10 of Engineering Design Optimization [**?**].

IV. Results

Although the chord thickness changed during the optimization for all blade counts, both the rotor identification and the NACA number stayed the same throughout the optimization. Changing the pitching angle only rotated the rotor blade, and changing the chord length magnified the size of its entire profile uniformly.

A. Results Plot

Figure (**??**) shows that while the optimized design increased the efficiency of even the rotor with a higher blade count above the rotor pre-optimization, it dramatically decreases the thrust and torque coefficients. The efficiency at higher advance ratios also decreased. These decreases in other rotor properties may not be desired. This illustrates one problem with optimizers: if created correctly, they will do exactly what they are programmed to do and may sacrifice some desirable properties to further optimize the objective function.



(a) Efficiency

(b) Thrust Coefficient

(c) Torque Coefficient

Fig. 1 Efficiency, Thrust Coefficients, and Torque Coefficients Compared at Different Advance Ratios

If necessary, minimum thrust and torque coefficients could be provided as parameters along with the maximums shown below the objective function in table (**??**). These could find the angle of rotation and chord thickness that would

provide the maximum efficiency while still maintaining some required thrust. The optimizer might not reduce the rotor to its minimum possible chord thickness in every case as it did in this optimization.

B. Results Table

Table (**??**) shows the optimal twist angles and chord magnifications for each different rotor blade count. At the conditions listed in table (**??**), the optimal chord length magnification was constant, and the optimal angle of rotation was less than $-3°$ for each rotor. It gradually became more negative as the blade count increased.

Table III (Table IV.B): Optimized chord magnification and rotation angles for different blade counts

Blade Count	**Chord Thickness Multiplication**	**Twist Angle**
3 (Default)	1.0	0°
2	0.50	−2.70°
3	0.50	−2.88°
4	0.50	−2.94°

V. Discussion

The propellor was checked at blade counts of 1, 2, and 3. These results found that for all three blade counts, the optimal propellor was as thin as possible, with a small negative angles of attack. The finding about thinner rotor blades being more efficient agreed with Saraf, Nouli, Ravalet, and Bakir's findings, but the negative rotation angle was surprising at first [**?**].

One lesson learned from this optimization is that the user should be careful what is optimize for, because the computer will optimize exactly what it is told, even if that is not what the user actually wants. An optimization that is written incorrectly or has unseen loopholes can not only give a misleading answer but also waste a lot of time. In the case of this optimization, I think other constraints could have been added to the optimization

A. Efficiency Equation

One problem with this optimization is that a rotor's efficiency is available as a function of the advance ratio, the thrust coefficient, and the power coefficient. The equation for efficiency is described by Andrew Ning in *Computational Aerodynamics*, page 200 [**?**] using the relation below, equation (**??**).

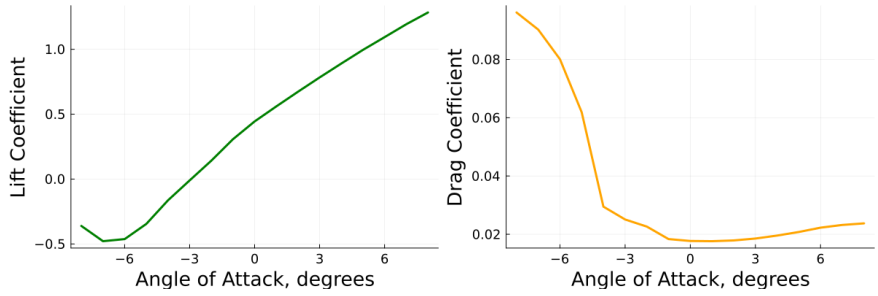
$$\eta = J\frac{C_T}{C_P} \tag{3}$$

With J kept constant, as it was in this research, this equation can be maximized in either by maximizing C_T or by

minimizing C_P, which is equal to C_Q multiplied by 2π. Inspection of figure (**??**) reveals that the optimizer did the latter. The torque coefficient C_Q is significantly lower for each rotor. While the thrust coefficient C_T did not decrease by as much as C_Q, it is still much lower than before. Although more efficient, the newly optimized propellor has a much lower solidity and is better suited for a different environment.

B. Angle of Rotation

The optimal angles of rotation were between $-2.70°$ and $-2.94°$, as shown in table (**??**). These negative angles of rotation were surprising at first, because angles near $0°$ were expected to be the most efficient. Figure (**??**), from airfoil analysis conducted earlier in the semester, shows that $0°$ angles of attack do not necessarily correspond to the lowest lift and drag coefficients for the NACA 4412 airfoil used in this design. Although that finding was for more simple analysis of an airfoil, it was concluded that a negative angle of rotation was possible.



(a) Lift Coefficient **(b) Drag Coefficient**

Fig. 2 Lift and Drag Experienced by NACA 4412 Airfoils
These plots show how lift and drag change for an airfoil at different angles of attack.

VI. Conclusion

This optimization combined several different Julia packages to find the optimal angle of rotation and rotor chord length for a rotor's efficiency. It found that rotors with very small chord lengths at negative angles of rotation were the most efficient. Other optimizations could be performed by simple editing of the code used in this research to find which rotors perform best in other applications or environments.

VII. Acknowledgements

The author would like to thank Adam Cardoza for mentoring him in learning the codes used in the FLOW Lab, directing his research, helping him through questions and challenges that he discovered during his work.