# Propeller Optimization at Different Blade Counts

Joseph Spencer [*]
*Brigham Young University, Provo, Utah, 84601*

**Propellers can be modified to perform better under a variety of conditions. One project completed as part of my research for the Brigham Young University FLOW Lab explores ideal design variables for propeller performance. This investigation used computer simulations to find the optimal chord length magnification and the optimal pitching angle for a propeller's efficiency. This report defines the operating conditions used in this optimization, presents propellers that were optimized at several blade counts, and discusses areas of possible future development in this research. I found that for the operating conditions used, propellers with the chord reduced to the optimization's lower limit, 50 percent, with negative pitching angles were the most efficient. These results could be applied to propellers designed to produce the maximum output power for a given input power requirement, without regard to other desired outputs. Future work could combine these findings with other conditions to find the most efficient propeller that can also provide a required amount of thrust or operate at a variety of different rotational velocities.**

## I. Introduction

PROPELLERS come in a variety of different shapes and sizes. This paper describes how I used one propeller, an APC 10x7 rotor[*] with a NACA 4412 airfoil[†], as a baseline to optimize a propeller's efficiency at a specific advance ratio. Because different blade counts could be more appropriate for different applications[‡], I optimize and compare blade counts of 2, 3, and 4 in this report. I then compares the optimized propellers of each different blade count.

Computer simulations are very useful for modeling propellers. It would require significant investment of time, space, and money to perform an optimization like this by physically manufacturing propellers. In the potential-flow code (PFC) simulation used in this report, I instead computationally compared the properties of hundreds of slightly different propellers to obtain a final result. A PFC is a program for modeling the pressure distribution of a rotor. PFCs were first programmed in the 1960s and 1970s. Xfoil, a program used to create the airfoil model used in this project, was first developed by MIT in the 1980s, with its current version dating to 2013. Xfoil.jl, the Julia wrapper for Xfoil,

[*]Undergraduate Researcher, Brigham Young University FLOW Lab.

[*]Rotor geometry obtained from the UIUC database at `https://m-selig.ae.illinois.edu/props/volume-1/propDB-volume-1.html`

[†]propeller geometry obtained from supporting information for CCBlade.jl at `https://github.com/byuflowlab/CCBlade.jl/tree/master/data`

[‡]Differently sized propellers can produce different efficiencies and noise levels, among other things. `https://hartzellprop.com/are-more-propeller-blades-better/`

was developed by Taylor McDonnell[§].

In this paper I first explain the procedure used in the rotor optimization, including the objective function used in the optimization and the initial conditions and constraints applied. Next, I review the results and compare them with each other. Finally, I draw conclusions based on the results of this optimization and discusses their meaning. All code used for each section of this report can also be accessed through a GitHub repository[¶].

## II. Methods

Computational stimulations of propellers allow researchers to perform low-risk studies and develop designs more quickly. I performed this project 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 with precompiled functions, which makes it run faster than interpreted languages like python without needing to be totally compiled before running after any change is made like C[**]. Julia packages, including those used during this semester, are also readily available from a package manager.

### A. Models

Some Julia packages used for this project include Xfoil.jl[††], CCBlade,jl[‡‡], QuadGK.jl[§§], and SNOW.jl[¶¶]. Xfoil.jl, discussed previously, analyzes the lift, drag, and moment coefficients of an airfoil from its geometry and angle of attack using a potential-flow code with a panel method. It divides the airfoil into many small surfaces and finds the lift, drag, and moment forces at each of the smaller surfaces.

The rotor in this design was created beforehand and included in the documentation for CCBlade.jl[***]. Airfoil data could be created using XFoil.jl, but I selected this provided data because its domain had already been widened using Viterna extrapolation to include a complete range of angles of attack[†††]. The airfoil polars used to analyze this are presented in figure (1).

[§]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 through https://github.com/JoeSpencer1/497R-Projects/tree/Final-Report

[‖]Julia is available at https://julialang.org.

[**]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

[§§]QuadGK.jl can be found at https://github.com/JuliaMath/QuadGK.jl

[¶¶]SNOW.jl is available at https://github.com/byuflowlab/SNOW1.jl

[***]Same address listed in introduction note, https://github.com/byuflowlab/CCBlade.jl/tree/master/data

[†††]Further explanation of this method is available at https://flow.byu.edu/CCBlade.jl/stable/howto/

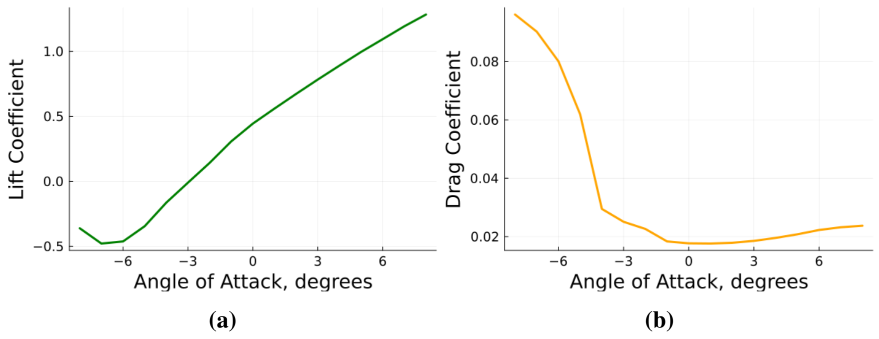


**Fig. 1 Lift (a) and drag (b) coefficients of NACA 4412 airfoils**

After the airfoil was created, I lofted it over the APC 10x7 rotor used in the project, and evaluated the propeller's torque and power coefficients across a range of advance ratios with CCBlade.jl. Propellers in the optimization were compared at a specific advance ratio, $J = 0.472$, which is explained later in this report and listed in the bottom entry of table (1).

CCBlade.jl uses blade element momentum (BEM) theory[‡‡‡] to calculate a propeller or turbine's behavior. BEM theory, which combines blade element theory and momentum theory, calculates the axial and tangential induction factors and the pitching angle, the angle between the free stream velocity and the tangential velocity of the propeller. After these three quantities are found, they are combined to find forces acting in all directions on the propeller[1].

I found the propeller's total torque and total moment from the torque output of the propeller's CCBlade.jl evaluation and from integrating the normal and tangential force data output by CCBlade.jl across the rotor blade using an integration function from the QuadGK.jl package. With the rotor blade's dimensions known, the forces found by integrating with QuadGK.jl provided reasonable limits for use in the optimization.

## B. Optimization

In this optimization, I modified the chord length and pitching angle for propellers at the same advance ratio to find the most efficient one. The optimization objective function, design variables, and constraints are displayed in equation (1), with the advance ratio $J = 0.475$ calculated from equation (2).

$$\begin{aligned} \text{maximize}: \quad & \eta \text{ at } J = 0.475 \\ \text{with respect to}: \quad & c \qquad\qquad\qquad \theta \\ \text{subject to}: \quad & 50\% \leq c \leq 200\% \quad -90^\circ \leq \theta \leq 90^\circ \\ & M_n \leq 1.1 \times M_{n0} \quad M_t \leq 1.1 \times M_{t0} \\ & T \leq 1.1 \times T_0 \end{aligned} \tag{1}$$

[‡‡‡]More information about BEM theory can be found at `https://en.wikipedia.org/wiki/Blade_element_momentum_theory`

In this optimization, rotor thickness was not allowed to increase or decrease by a factor of more than 2, and the propeller could not rotate more than 90° in either direction. The propeller's torque and normal and tangental moments did not exceed 110% of the values calculated from the CCBlade.jl output when the original propeller was created.

$$\begin{aligned} J &= \frac{v_\infty}{nD} \\ &= \frac{12}{(\frac{6000}{60}) \times 0.254} \end{aligned} \tag{2}$$

Table (1) shows shared features and design variables of the propellers optimized. The advance ratio, $J$, is available from other variables already listed in the table by equation (2), in which $v_\infty$ is the free stream velocity, $n$ is the rotational velocity in revolutions per second, and $D$ is the outer diameter.

| **parameter** | **default value** | **minimum value** | **maximum value** |
|---|---|---|---|
| chord length, | 100% length | 50% | 200% |
| pitching angle, | 0° | −90° | 90° |
| rotational velocity | 6000 RPM | | |
| blade count | 3 blades | 2, 3, or 4 | |
| hub-to-tip ratio | 10% | | |
| air density | 1.225 kg/$m^3$ | | |
| outer diameter | 0.254 m (10 in.) | | |
| velocity | 12 m/s (26.84 mph) | | |
| advance ratio | 0.472 | | |

**Table 1 Initial values and limits in propeller design**

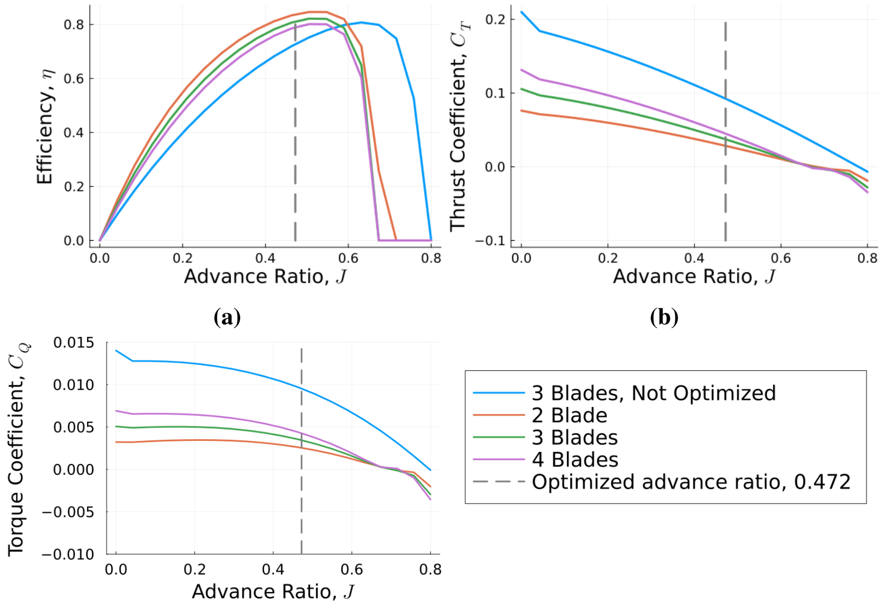
SNOW.jl, developed by the FLOW Lab, finds either the maximum or minimum of a function. The code used to design the propeller was prepared so that SNOW.jl found the maximum of the objective function by finding the minimum of its negative. This is similar to the technique and sign convention described by Martins and Ning in *Engineering Design Optimization*[2]. SNOW.jl can also use fixed environmental variables and set limits or constraints on design variables to contain the domain of possible solutions that can be optimized. The program written for this project used both of these inputs to obtain the desired solutions, so SNOW.jl was a very useful tool.

## III. Results and Discussion

Although the propeller's profile remained the same throughout the optimization, the chord length decreased for all blade counts. Changing the pitching angle only rotated the propeller blade, and changing the chord magnified the size of its entire profile uniformly along its entire length. Rotor and airfoil themselves retained the same NACA and APC identification numbers throughout the optimization.

Figure (2) demonstrates that while the optimized design increased the efficiency of even the propeller with the higher blade count above the propeller pre-optimization, it dramatically decreased the thrust and torque coefficients. The efficiency at higher advance ratios also went to zero more quickly, so the range of advance ratios over which the propeller maintained high efficiency was smaller. Maximizing the propeller efficiency at a single advance ratio by only changing the chord length and pitching angle may require other propeller performance measures to be sacrificed.

The plots in figure (2) demonstrate this finding visually. The optimized efficiency was higher than the original efficiency at every blade count. This was the desired result, as stated by the optimization function in equation (1). The figure also shows how much smaller the thrust and torque coefficients became at this increased efficiency.



(c)

**Fig. 2 Efficiency (a), thrust coefficient (b), and torque coefficient (c) compared at different advance ratios**

Table (2) shows the optimal pitching angles and chord magnifications found by SNOW.jl for each different propeller blade count. At the conditions listed in table (1), the same optimal chord length magnification was found, and the optimal pitching angle was less than $-3°$ for each propeller. These results prove a few things about the propeller designed.

| **blade count** | **chord length magnification** | **pitching angle** |
|---|---|---|
| 3 (Default) | 1.0 | 0° |
| 2 | 0.50 | −2.70° |
| 3 | 0.50 | −2.88° |
| 4 | 0.50 | −2.94° |

**Table 2 Optimized chord magnification and pitching angles for different blade counts**

First off, propellers with lower solidity appear to be more efficient. All of the propellers optimized had the lowest chord length allowed by the program used. Rotor solidity can be calculated with equation (3), which shows that reducing the chord length, $c$, or the blade count, $n_b$, reduces the solidity, $\sigma$. The finding that lower solidity leads to greater efficiency seems reasonable, because wind turbines with long, thin blades, designed to efficiently harvest wind from the environment, have very low solidity according to this equation. This finding about propeller blades with less solidity being more efficient also agrees with Saraf, Nouli, Ravalet, and Bakir's findings[3].

$$\begin{aligned}\sigma &= \frac{c}{s} \\ &= \frac{cn_b}{2\pi\sqrt{\frac{r_h^2+r_t^2}{2}}}\end{aligned} \tag{3}$$

Table (2) shows that although the optimal pitching angles were very similar for all three different blade counts, they became minutely more negative as the blade count increased. This further decrease in the already negative pitching angle could be at least partially explained by the airfoil polars shown in figure (1). Increasing the pitching angle slightly above $-3°$ would make its lift coefficient higher and its drag coefficient slightly lower. Figure (1), from airfoil analysis conducted earlier in the semester, shows that angles of attack near $-3°$ for the NACA 4412 airfoil used in this design correspond to a very low drag coefficient.

Changing the pitching angle of this propeller would change the forces and efficiency produced by a propeller with a blade count of 2 differently than propellers with blade counts of 3 or 4, because the wake instability created by each propeller blade affects the other blades[4]. This change in blade-to-blade interactions is why the curves in figure (2) are shaped differently from each other, even though they are optimizations of propellers with the same airfoil and rotor profiles.

One problem with this optimization is that a propeller's efficiency can be obtained as a function of its advance ratio, thrust coefficient, and power coefficient. The equation for efficiency is described by Andrew Ning in *Computational Aerodynamics*[5], using the relations in equation (4).

$$\begin{aligned}\eta &= J\frac{C_T}{C_P} \\ &= \frac{TV_\infty}{TV_\infty + \frac{1}{2}\dot{m}(V_w - V_\infty)^2}\end{aligned} \tag{4}$$

Ning also wrote that when thrust is increased for a propeller, it comes at the expense of decreased efficiency[5]. Conversely, a propeller optimization that maximizes efficiency will decrease its thrust coefficient. The torque and moments output by the optimized propellers did not even come near the limits put in place by the original propeller. For example, the original propeller blades experienced a normal moment of $3.753 \times 10^6 N \times m$, but the optimized propeller with the same blade count only experienced $3.808 \times 10^5 N \times m$.

When I kept $J$ constant in this research, inspection of figure (2) reveals that the optimizer maximized equation (1) by minimizing $C_P$, which is equal to $C_Q$ multiplied by $2\pi$. The torque coefficient $C_Q$ is significantly lower for each propeller. 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 propeller produces much less power and may be better suited for an entirely different function than the one it started with.

To prevent the optimizer from sacrificing thrust for efficiency, I could provide a minimum thrust or minimum torque as parameters along with the maximums shown below the objective function in equation (1). I could use this additional constraint to find the pitching angle and chord magnification that would provide the maximum efficiency while still maintaining some required thrust or being powered by the same motor. If this function is used, the optimizer might not reduce the propeller to its minimum possible chord length in every case as it did in this optimization.

## IV. Conclusion

In this optimization, I used several Julia libraries to find the optimal combination of chord length and pitching angle for the maximum efficiency of a propeller at a known advance ratio. I discovered that for the conditions listed in table (1), the optimal propeller has 50% its original chord length, which is the lower limit of the optimization, and a pitching angle between $-2.70°$ and $-2.94°$, depending on the blade count.

One discovery gained from these findings is that thinner propeller blades with smaller chord length make more efficient propellers. This finding is related to the rotor solidity, described in equation (3). The optimization using SNOW.jl chose the minimum chord length as the most efficient each time. Another evidence that increasing solidity decreases efficiency can be observed in figure (2). The rotor solidity increased proportionally to the blade count. As it increased, the efficiency at each advance ratio decreased.

With this information, I decided to find what the optimal chord length actually was. I removed the original 50% lower limit of the optimizer and instead set it to 1% in the function argument. SNOW.jl found a propeller that was much more efficient, and had exactly 1% of the original chord length. This is a totally unrealistic propeller to actually use, though, because it would not produce thrust. One APC 10x7 propeller found on the internet had a 1.02 cm hub thickness to start with[§§§], and lowering the chord to 1% would decrease it to 102µm. It appears that even in extreme cases, decreasing chord length always increases propeller efficiency even when it is most likely no longer desired.

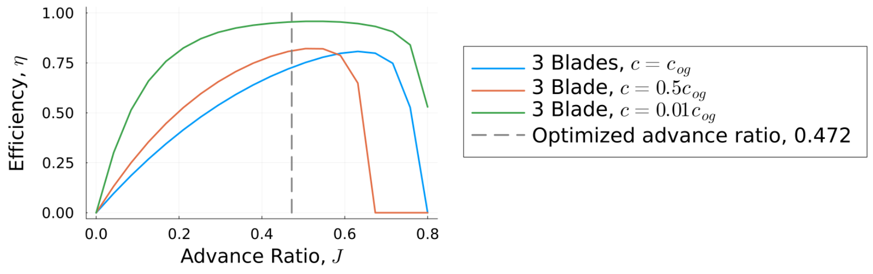


**Fig. 3 Comparison of propellers with different cord lengths**

Another discovery made in this research was that this propeller was most efficient with negative blade pitching angle. To investigate this result further, I ran another simulation with the Julia code I had written to see what happened to this blade pitching angle when the propeller was increase back to its original chord length. The plot in figure (4) shows the original propeller compared to the optimized propeller and a propeller with the optimized pitching angle but the same chord length as the original propeller.

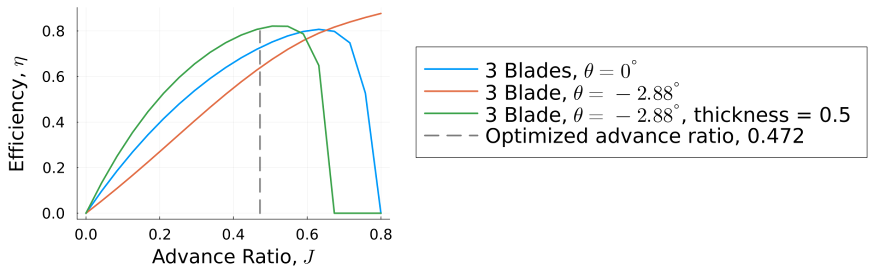


**Fig. 4 Comparison of identical propellers at different pitching angles to optimized propeller**

The different curves plotted in figure (4) show that the propeller only became more efficient at the $-2.88°$ pitching angle when its chord length was also reduced significantly. Otherwise, it became less efficient at the advance ratio measured but more efficient at higher advance ratios. This showed that thicker rotors and thinner rotors are better suited for different rotational velocities and free stream velocities.

With the added information from plot (3) and plot (4), I programmed one final optimization that eliminated the problem of SNOW.jl converging to the propeller with the minimum thickness. I added another constraint to the objective function used for the initial optimization, described by equation (1). I added a minimum constraint to the propeller's torque, $Q \geq 0.9Q_0$. The torque coefficient $C_T$ and power coefficient $C_P$ are related by a factor of $2\pi$, so keeping the propeller torque within 90% of the original would also maintain 90% of the original power output, or $P \geq 0.9P_0$.

[§§§]This propeller was found at `https://www.apcprop.com/product/10x7e/`.

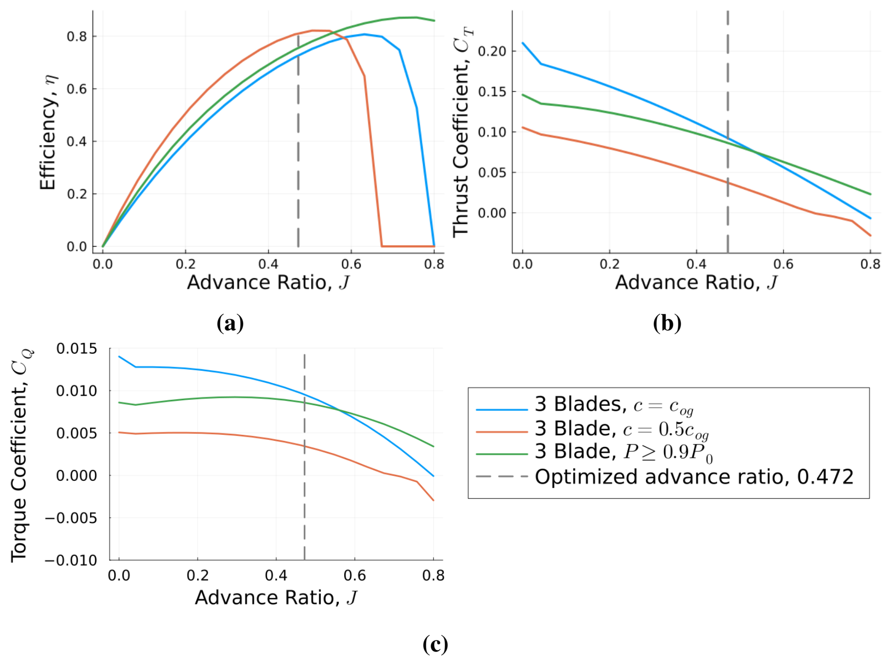


**Fig. 5 Efficiency (a), thrust coefficient (b), and torque coefficient (c) compared between different optimizations**

The plots in figure (5) show two results of this new design. First, the propeller efficiency increases at the measured advance ratio, but not by quite as much as the propeller without the minimum power constraint. Second, it is even more efficient at higher advance ratios. This optimized propeller's final chord length was still decreased to 53.2% of its original chord length, which was near although not quite at the 50% lower limit given to SNOW.jl. Unlike the propellers optimized previously, though, this optimized propeller's pitching angle was positive, 2.96°.

Future work could investigate other possible constraints to place on this optimization, or change the constraints already present. Other optimizations could vary other aspects of the propeller. For example, the rotor length could be optimized while maintaining a maximum power output, or the shape of the airfoil could be changed entirely or even lofted through different shapes. The rotor could also be modified to an optimal shape. As long as the user understands well the constraints that must be met, optimizations have immense possibilities.

Performing an optimization aided by computers allowed me to explore a broader design space as new ideas were rapidly generated that had not been considered before.

## V. Acknowledgements

Adam, thank you for mentoring me in learning the codes used in the FLOW Lab, directing my research, and helping me through questions and challenges that I encountered during this work.

## References

[1] Ning, A., “Using Blade Element Momentum Methods with Gradient-Based Design Optimization,” *Structural and Multidisciplinary Optimization*, Vol. 64, No. 2, 2021, pp. 994–1014. https://doi.org/http://dx.doi.org/10.1002/we.1636.

[2] Martins, J. R. R. A., and Ning, A., *Engineering Design Optimization*, Cambridge University Press, 2021.

[3] Sarraf, C., Nouri, H., Ravelet, F., and Bakir, F., “Experimental study of blade thickness effects on the overall and local performances of a Controlled Vortex Designed axial-flow fan,” *Experimental Thermal and Fluid Science*, Vol. 35, No. 4, 2011, pp. 684–693. https://doi.org/https://doi.org/10.1016/j.expthermflusci.2011.01.002.

[4] Felli, M., Guj, G., and Camussi, R., “Effect of the number of blades on propeller wake evolution,” *Experiments in fluids*, Vol. 44, No. 3, 2008, pp. 409–418.

[5] Ning, A., *Computational Aerodynamics*, Brigham Young University, 2022.