Final Report:
A Basic Optimization of the DJI II Propeller

Jacob Child
Student, FLOW Lab
Mechanical Engineering
Brigham Young University
Email: childjacob2@gmail.com

Adam Cardoza
Graduate Student Mentor, FLOW Lab
Mechanical Engineering
Brigham Young University
Email: adam.cardoza.online@gmail.com

ABSTRACT

In order to gain a sound understanding of basic aerodynamic principles, become familiar working with Julia, and begin to grasp the theories of propeller design and optimization this two semester long introductory project was completed. This project included creating an aerodynamic terms dictionary, an airfoil plotter and analyzer using XFoil, a propeller analyzer using a blade element momentum (BEM) solver made by the Flow Lab, completing propeller design space trade study, and completing a basic propeller optimization. Each of these individual assignments led up to and contributed to the basic propeller optimization. The optimization will be the focus of this paper, and the other projects will only be discussed in relation to that. The optimization was done by using a Genetic Learning Algorithm within the Evolutionary.jl package to vary the chord and twist of each propeller blade section. The objective function called functions within CCBlade.jl that were used to analyze the propeller geometry and help find an "optimal" blade geometry whose goal was to increase thrust and constrain (or keep below a threshold) torque. Following a numerous amount of algorithm runs, it was found that, for the given conditions and restraints, it was *possible to decrease torque, but very difficult (ie the optimal solution was hard to find) to increase thrust. The findings highlight the general relationship between torque and thrust, the general relationships between chord length and twist being varied and the output thrust and required torque. The findings also demonstrate the strengths and shortcomings of a genetic learning algorithm, although that will not be a focus.*

INTRODUCTION

To optimize is to try and find the set of factors that create the best performing solution. In the case of a propeller optimization this translates to changing the geometry of the propeller to increase or decrease certain factors of propeller performance. This project was meant to be basic and an introduction to optimization, as such there were many factors that were simplified or ignored in order to make the optimization quicker and smoother. The objective of the optimization was to increase thrust while operating under a single constraint- keeping required torque below a certain threshold. The operating conditions of the optimization were to have the propeller spinning at 5000 rpm and an Advance Ratio of 0 (hovering).

This paper will discuss the methods by which the optimization was done and constraints found and chosen, the results that were found by the algorithm, and an in depth interpretation and explanation of the found results.

METHODS

To obtain the given results several 3rd party packages were utilized to drastically reduce the amount of coding and troubleshooting necessary. At the highest level Evolutionary.jl was used to setup and run a genetic learning algorithm (GA). A GA is an algorithm that attempts to mimic the genetic evolutionary process by encoding the given parameters for an optimization problem into different genes that are then varied and distributed among individuals. The most fit individuals, or set of parameters that perform the best, are breeded (their genes are crossed over and mutated) with other individuals to create the next generation. Using this type of algorithm made the implementation and set up a bit easier than a gradient based optimization algorithm that would be beyond the scope of the project. A downside of GA's are their heavy resource usage, as was found when trying to utilize larger population sizes. Beneath using the Evolutionary.jl package, the CCBlade.jl package was extensively used. This package enabled the extension and smoothing of the needed airfoil data that was plugged into the blade analysis. CCBlade.jl also did nearly every step of the blade analysis, from defining the parameters to computing the results at each section. In order to fully to successfully run the connections between all of the packages and how to utilize the given data was coded. To start with the original geometry of the DJI II propeller was imported (from publicly available data given by the Flow Lab) along with its already calculated airfoil performance data. That data was smoothed and extended and then the baseline performance analysis was done to compare with later. An initial population for the GA was then generated by randomly selecting a data point between the appropriate bounds. That initial population was then fed into the GA which in turn varied the chord and twist at each section in an attempt to find a better performing propeller.

RESULTS AND DISCUSSION

Overview

As Genetic algorithm's are fundamentally stochastic in nature the results of the optimization varied greatly even when constraints were kept the same, and when constraints were changed the results varied even more. Not only did the problems input constraints change the optimization performance, settings within the algorithm, such as mutation program and rates and similar settings, also caused the optimization to have varied results. All of these factors combined led to an unfortunate but informative end; thrust was never verifiably improved by varying the chord and twist of each section, however, torque was improved (ie it required less torque) under certain circumstances. In preparation for the full optimization a more basic algorithm was created that varied the pitch of the blade as a whole, and changed the chord of the blade as a whole (by adding or subtracting a % of Rtip). This algorithm did improve thrust, but only by 0.7%, and as it was just used for testing the results and settings, the output values weren't verified. It does show that the number of design variables (2 versus 54) may have had a large part to play in the ability of the algorithm to optimize.

Thrust and Torque Trends

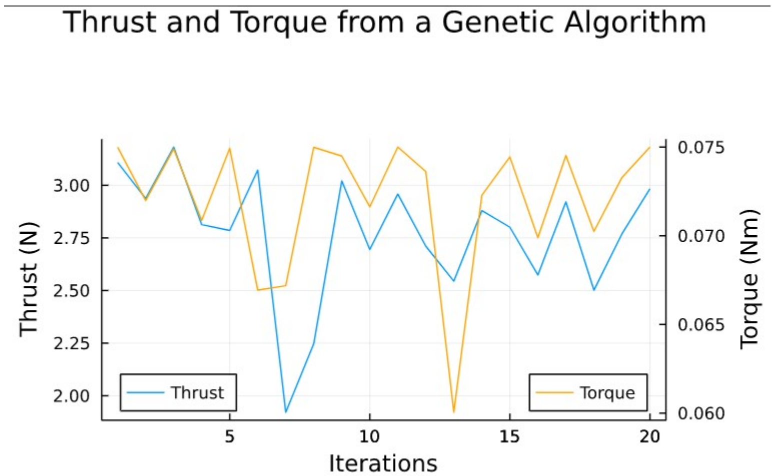

Fig. 1. A plot of 20 different optimization runs (labelled as "iterations") showing the Thrust and torque of the final result of each.

It is readily apparent in Figure 1 that in general when thrust goes up, the torque required also goes up. This is because in the optimization thrust can be increased in only two ways (because RPM is fixed at 5000), either increasing the chord or the twist. Increasing the chord of the blade increases lift (see figure 2) as the lifting surface grows larger and the blade is in turn able to generate more lift. The downside of this is that increasing the chord adds material, and thus adds

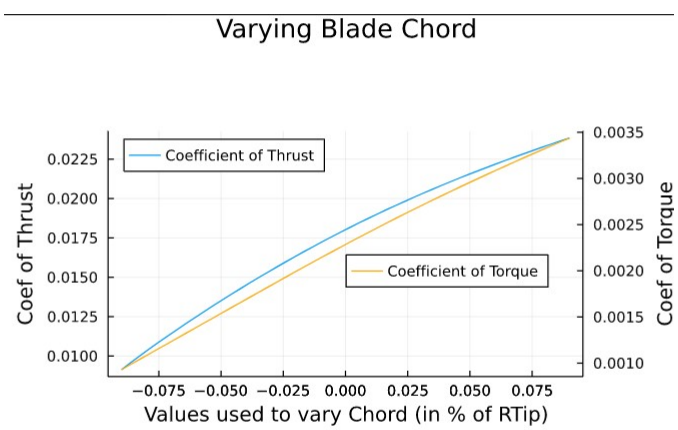

Fig. 2. A plot of the coefficients of thrust and torque as the chord of the whole blade was increased.

drag on the larger surface area, in turn increasing torque. Increasing (or pitching up) the twist of a blade section creates more lift as it ”grabs” more air and ”throws” it down than a more horizontal section. This increases drag due to the larger frontal surface area, and that increase in drag increases torque.

The above relationship between chord and twist with thrust and torque is why it is important to note runs (or iterations) in the figure where torque seems to break the torque/thrust relationship. This is special because it shows where the algorithm was beginning to optimize, or find a collection of chord and twist parameters that manage to increase thrust without increasing torque as much. This is a more ”optimized” design.

Parameter and Geometry differences

The DJI II blade imported into the optimization was split into 27 sections, each of these sections had its chord and twist varied. Several runs (ie letting the algorithm run through) were done limiting the torque to .06Nm (the original value), while the algorithm decreased the torque, it was often quite far below the original thrust value. The allowed torque was then increased, and in turn the thrust increased. When the torque was limited to 125% of the original torque the thrust was still 21% below the original value, but when the torque was allowed to increase 300% (it ended up increasing by 200%) the thrust increased by 47%. These geometry differences of these three cases compared to the original will be discussed in order to find the cause of the differences. For the sake of discussion the first third of the blade will be the first 10 sections, the middle will be around sections 11-20, and the end or tip will be sections 21-27. All comparisons are done with respect to the original blade. This is done so as not to lose reality to theory, and also because no blade outperformed the original in terms of thrust. Comparing the geometry differences of each case will enable discussion as to the effect each section and the associated parameters have on the flow characteristics compared to the original.

Case 1

Figure 3 shows the percent increases or decrease at each section of the chord and twist under the first case (torque limited to .06Nm). The beginning of the blade seems to alternate between a high pitch and a large chord. While the inconsistency between values and alternating between

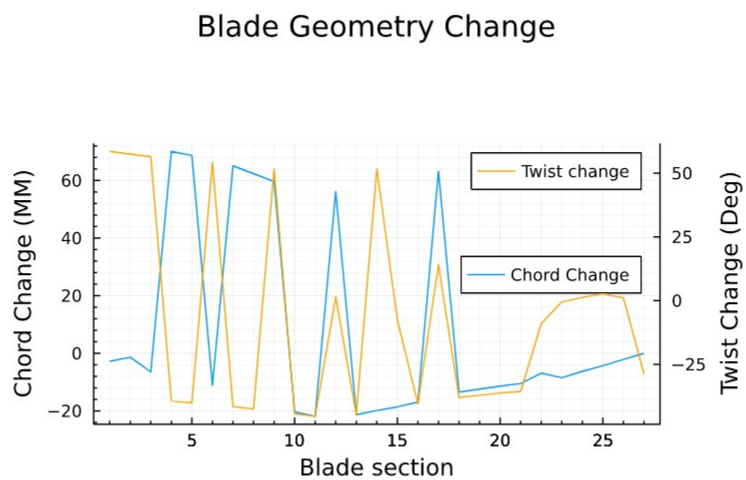

Fig. 3. A plot of the change at each section in chord and twist as compared to the original DJI II propeller. The old values were subtracted from the new values to obtain the plot

large chord and high twist is odd, it logically makes sense as both can produce larger amounts of lift. Towards the root of the blade there are higher twists help to generate more lift when there is a slower inflow (due to the blade rotating), this is a pattern in real propellers and in general can be seen in this blade, although when there is a large chord, the twist drops. This is probably due to having both a high twist and large chord would increase the torque considerably.The middle of the blade becomes confusing with portions having both a negative twist and a decreased chord. This is likely a contributor to why this blade produces 36% less torque than the original layout. There are sections that have higher twist or larger chord and those would help to generate needed lift. The smaller chord sections are likely a reason that torque decreases by 1.56% in this case. The end of the blade is twisted down and has a chord smaller than the original, this would decrease torque, but also decrease thrust, yet another reason this case under performs.

Case 2

Figure 4 shows the percent increases or decrease at each section of the chord and pitch under the second case (torque limited to 125% of the original value). It is interesting to note that, compared to case 1 this blade produces more thrust (still 20% less than the original) but also requires more torque (25% more than the original). The very root of this blade is twisted down and has a slightly smaller chord than the original, this likely does not affect too much as the root spins the slowest. Just after the root the blade quickly jumps higher in both twist and chord length. The increase

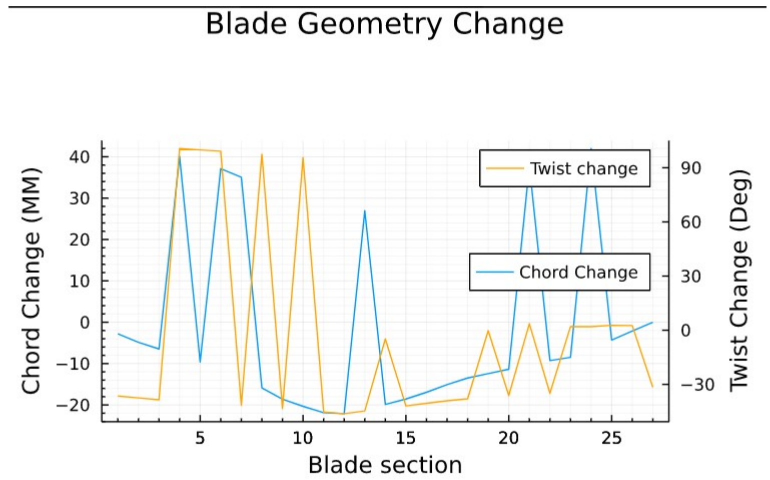

Fig. 4. A plot of the change at each section in chord and twist as compared to the original DJI II propeller. The old values were subtracted from the new values to obtain the plot

will generate more thrust, however, just as before, generally if the chord is high, the twist is lower. It should also be noted that there are several sections whose twist is changed greater than 80 degrees. This means that 80 degrees was added to the original twist, so it is likely these portions of the blade are actually upside down. This negatively affects thrust, thus the decrease of thrust by 20% The middle to the end of the blade keeps a more decreased twist than the original as well as smaller chords than the original (until the end). At the end of this blade the chord increases drastically and is likely a large contributor to the 25% more required torque. The fact that this blade is quite convoluted but is still able to perform within 25% of the original is extremely interesting and shows that the algorithm is possibly taking advantage of things that are not actually feasible in the real world, as well as relying on one part of the blade (the beginning) to generate most of the lift for the whole blade.

Case 3

In case 3 the algorithm was allowed to triple the amount of required torque if necessary in order to increase thrust. This case is especially interesting as it is the only case that outperforms the original blade with respect to thrust (a 47% thrust increase), however it does so at the cost of nearly a 200% increase in the required torque. A dig into the geometry changes (shown in figure 5) shows that the beginning portion of the blade is likely a larger thrust contributor as it has several portions

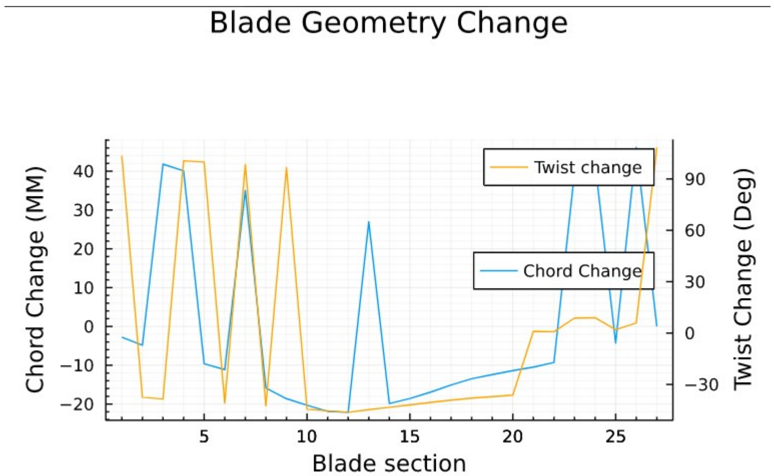

Fig. 5. A plot of the change at each section in chord and twist as compared to the original DJI II propeller. The old values were subtracted from the new values to obtain the plot

that are either twisted up or have larger chords as we have seen in both the previous cases. It is interesting though that in this case there are two sections (4 and 7) that have both a large chord and a high twist at the same time. While this would provide thrust, it would also greatly increase required torque. It should also be noted that some of these sections are likely tilted backwards as was found above as well. The middle section of this blade is generally twisted down and has a much smaller, this likely increases torque more, but does not create much lift. The tip/end of the blade has some more aerodynamically sound parameters. The twist is very close to the original, which towards the tip is getting close to horizontal, and the chord is increasing. The end of the blade spins the fastest, and a large chord provides more surface/lifting area, so the tip likely produces a large amount of thrust, however also a large amount of torque due to extra material far from the hub.

Geometry Conclusion It is readily apparent that all 3 cases would likely not be a feasible blade in the real world, and that the algorithm is likely also exploiting a weakness in the model. More research needs to be done into the behavior of CCBlade.jl under parameters that are not realistic (ie large chords twisted past vertical) as it is possible the outcome may be something that incorrectly ”benefits” the results. It was interesting to see that the algorithm struggled to have the chord and twist work in tandem and typically relied on one or the other to perform in a section. This is supported by the previously done trade study, as it was shown that varying chord or pitch, independently of all other factors could be an effective way of increasing thrust. Despite that, it is impossible to ignore the absurdity of these blades.

CONCLUSION

This exploratory introduction to a basic optimization was successful in teaching about optimization requirements and setup as well as helped to highlight factors that directly affect the performance of a propeller blade. It was found that both increased chord size and higher twist can help to generate thrust, but they can also greatly increase required torque. The beginning of the blade (closer to the root) can be an effective place to have much larger chords and higher twists as the torque required close to the hub is less than it would be further from the hub. This was seen in all cases. It was also found that using a largely stochastic algorithm with many different variable settings leads to a large variability of results, and a large overhead time to tune settings to a point where consistent good results are obtained. Future work will need to be done in large part focused on the algorithm problem. The model will need to be better defined so unrealistic results are less likely. The objective function will need to be analyzed to make sure that no weaknesses can be exploited and to understand in what realm its results are feasible. Work also needs to be done into what selection, crossover, and mutation programs and rates are most successful (in the case of a genetic algorithm). It is highly likely that a GA is not the best algorithm for this case and research should be done into other optimization options. It should also be noted that a successful optimization was performed when varying the chord and pitch of the blade as a whole (not by section) so further research should be down into that as it was more successful.

APPENDIX A: GITHUB EXPLANATION

All code for this project and all of my other projects can be found at Github Link. Each project has its own folder. This project is contained in the Basic Optimization folder and can be entirely run and controlled from the BasicOptimizer.jl file. The top portion of the file imports necessary data and the bottom is the setup for the GA. To be frank, this is the least polished code of all projects. For a better understanding of the design space go to the Trade Study folder and run the TradestudyTopFile.jl there. It will help to paint a picture of how varying various parameters (on the blade as a whole, not by section) affects performance.

APPENDIX B: CLASS REVIEW

I consider this class a success. Comparing where I was at the very beginning (before spring semester) to now is crazy. I basically knew nothing about aerodynamic principles and feel like now I have a much deeper grasp and understanding of them. I wrote thousands of lines of Julia code and am much more comfortable with it. I successfully completed a trade study this semester and blade and airfoil analyses last semester. I began to sink my teeth into optimization but did not have enough time or guidance (no fault of Adam's he went above and beyond at attempting to help, it was just a new package for both of us with only a few days to figure it out) to have a good outcome. Overall it was a lot of work but I learned a lot. All of the reports for each were helpful in helping me to better put forth my thoughts, but each of them had the feel of a final report even though they were supposed to be more simple like a memo, Jupyter NB, or markdown document. Writing each of them took a lot of time, with a decent amount of time spent on ”polishing” or figuring out how to do things like nice figures- while important I feel like those did not necessarily need to be a factor in every report, but would have been easier to have less polished reports turned in throughout. Each project definitely built up to the next one which was really cool to see how it all fits together (I probably wrote the least amount of code for the optimization as it was just adapting other stuff), however the write ups did not feel like that. Each write up felt like a stand alone document and going back and trying to use things from past documents in others did not really work. Somehow tying them all together would make the final report much less daunting, or having a rough draft earlier etc. Overall I absolutely loved what I learned and want to finish the optimization on my own time and then maybe run some of my own and 3d print and compare propellers on my drone!