2018

SEMESTER 2

363 Rotor Design Report

Connor McDowall
530913386
cmcd398

September 30, 2018

Contents

Listings

List of Figures

1 Summary

1.1 Rotor Purpose

The rotor is a commercial product. The target market is rural communities in the higher windy latitudes around the world. The product will consist of a kitset used to build a general assembly, a fixture on a locally constructed tower. See 1. for rotor application and kitset components. The kitset will include instructions for assembly.

1.2 Achievement of Purpose

Our chosen design of six blades with fifteen sd7003 aerofoils per blade will: meet the design criteria (below), be easy to both distribute to and assemble by rural communities using the compact kitset, and contain key components and samples of easily assessible surfacing and adhesive materials. The surfacing materials are duct tape, sellotape and hot glue.

1.3 Design Criteria

The rotor must operate in wind conditions between eight to twelve knots, rotating at 140 rpm at the target windspeed of ten knots. The rotor must have between three to eight blades. The general assembly must be easy to assemble from the kitset using the instruction manual. Blade surfacing and adhesives must be easy for rural communities to source for repairs and maintenance. The rotors components and materials must have the strength and durability to withstand the operating environment and are of low cost for maintenance and repair.

1.4 Analysis

The analysis was broken up into a series of discrete steps: Understanding the fundamental rotor theory. Writing an Iterative scheme in MATLAB. Conducting complete enumeration with every aerofoil design. Ruling out a series of aerofoil designs in a stage one selection process. Iterating on selected aerofoil designs, changing the number of blades and sections. Selecting the number of blades and aerofoils in a stage two selection process. Making manufacturing considerations in the design. Making the final selection based on the all major factors. Considering surfacing materials.

The main factors considered in the design are: The lift to drag (CL/CD) ratios for the aerofoil design across a range of angles of attack (alpha) and different windspeeds expressed by different Reynolds numbers. The margin of error in CL/CD ratios with changes in the angle of attack and windspeed, assessed in the CL/CD vs alpha plots for every aerofoil. The thickness of the aerofoil. The camber of the aerofoil. The shape of the aerofoil. The surface area of the aerofoils. The cost, manufacturability and durability of surfacing materials.

1000 different aerofoil designs were investigated in complete enumeration. After, 37 aerofoil designs were considered for stage two. In stage two, 333 different combinations were considered, the 37 selected aerofoils from stage one with 4,5 or 6 blades and 14,15 or 16 sections. 37 from stage two were considered to make the final selection. 1259 unique designs were considered in the process.

1.5 Conclusion

Our final design is a rotor with six blades with fifteen sections per blade. Each section is a sd7003 aerofoil design with varying chord lengths between 28.3 and 19.5cm, and a total blade length of 69.0cm from the centre of the hub. Each aerofoil has a camber of 1.2% of chord length. Each aerofoil has a different blade setting angle in the range of 28.6 to 13.0 degrees. The aerofoils are equally spaced along the blades, starting at 26cm from the centre of the hub and ending at 69.0cm. The aerofoils are surfaced using sellotape, duct tape and hot glue. See 2 for exact parameters and dimensions.

The design was the best of the last 37 in the final selection stage, maximising the Cl/CD ratios whilst having the most reasonable margin of error with changing angles of attack and windspeeds. The aerofoil had a smooth, continuous shape with slight camber, easy to laser cut and surface. Each aerofoil was thick enough to insert the blade rods. The weight per blade is low considering the number of sections, aerofoil surface area and thickness of the acrylic. Six blades decreased the chord lengths and increased our margin of error for manufacturing faults. Fifteen sections maximised the CL/CD ratios compared to other combinations. Sellotape and duct tape are durable, easy to apply and inexpensive compared to other materials. The hot glue is easy to apply and inexpensive compared to other adhesives.

1.6 Cost

The total rotor, two blade setting rigs and one extra blade will cost $241.79. See 3 for a full cost breakdown.

1.7 Performance

Our turbine will achieve 47.6W with a torque of 3.25m across all three windspeeds with 140rpm. The turbine will generate a power coefficients of 0.3978, 0.3632 and 0.3064 at 8,10 and 12 knots respectively.

2 Development of Design

2.1 Assumptions

We made the following assumptions to conduct our analysis: Fundamental rotor theory is applicable and does not break down. Assume losses due to wake rotation, aerodynamic drag, tip losses and losses attributable to having a finite number of blades. The mechanic system the rotor spins on is 100% efficient. Wind direction is perpendicular to the rotor and flows in one dimension.

2.2 Drawings

See 4.4 for drawings on the aerofoil design and the blade distance setting tool used in construction.

2.3 Method of Analysis

See 4 for our nine-stage method of analysis.

(1) Fundamental Rotor Theory. We learnt extracting power from the wind using a rotor. We learnt the following concepts: Lift and drag around an aerofoil design and deriving their subsequent coefficients. Learning how to extract power from the wind considering the Betz Limit, Bernoulli Analysis, axial induction factors, pressures, velocities and forces. The limitations of rotor design. The blade element momentum method considering both axial and angular induction factors. Incorporating axial induction factors and taking into consideration wake rotation and Prandtl tip loss. See 5 for lectures slide three through to eight where the aforementioned theory was covered.

(2) An iterative scheme. XfoilIteration.m was devised using the fundamental theory in MATLAB. The names of the aerofoil designs were extracted from an aerofoil bank. The aerofoil names were passed into evaluatePlot.m to generate CL/CD plots across a range of alphas and Reynolds numbers. The CL, CD and alpha values were found by using the callXfoil.m and Xfoil.m scripts, written by Kevin Jia. The plots were saved in a directory. The fundamental theory calculations were written in evaluateTurbine.m, returning the following parameters: Power coefficient of rotor power over system power, number of blades and sections, windspeed, angle of attack, Reynolds number, radii for aerofoil locations, chord lengths and the blade setting angles. These parameters were stored in an excel spreadsheet. See 5.1 for aforementioned MATLAB scripts.

(3) Complete Enumeration. The iterative scheme was applied to approximately 1000 aerofoil types, focusing on the CL/CD vs Alpha plots for a range of Reynolds numbers. These plots were stored in a directory. See 5 for a subset of plots.

(4) Stage One Selection. Each plot was accessed on the CL/CD magnitude and margin of error for changing alphas and Reynolds numbers. An ideal plot was one with a high CL/CD ratio with flat curves and tight bands, a similar shape to a rainbow. This stage eliminated 963 designs, leaving 37 as listed in 6.

(5) Blade and Section Iteration. The number of blades and sections varied in another implementation of the iterative scheme. After investigating combinations of blades (four, five or six) and sections (fourteen, fifteen or sixteen). These results were stored in an excel spreadsheet. See 7 for one third of total iterations.

(6) Stage Two Selection. After reviewing the spreadsheet, six blades with fifteen sections gave the best CL/CD ratios with adequate chord lengths for the 37 aerofoils considering manufacturing errors and blade redundancy.

(7) Manufacturing Considerations. The 37 aerofoil designs were assessed on four factors: their plot from stage one selection, their shape looking at the profile on airfoiltools.com, their camber, and their thickness calculated from a percentage of chord length. The plots were ranked amongst themselves between 0 (low) to 3 (high). The shape had to be smooth and continuous for easy laser cutting. The foil design must have some camber and have a low surface area. The minimum thickness of the aerofoil must be as greater than 14mm to allow a 10mm diameter rod insert with 2mm spacing to the edge either side. See 8 for the evaluation.

(8) Final Selection. After considering the aforementioned factors in the manufacturing considerations, sd7003 with 6 blades and 15 aerofoils was the chosen design. See 9 for a profile and plot of this design.

(9) The materials used must be strong enough to withstand the applied operating loads to meet the target of 140 rpm. The acrylic (PMMA) aerofoils and aluminium hubs/rods/hub-rod inserts are strong enough to withstand those loads and easy to manufacture. The blade surfacing material must be lightweight, smooth and easy to apply to create a smooth surface on the aerofoil design. Achieving these criteria leads to maximising the lift to drag ratios, maximising the power extracted from the wind, and therefore meeting the objectives outlined in the design criteria. After considering materials, sellotape on the blade surfaces with duct tape on the sides for strength were chosen. These will minimise weight, be strong enough to withstand the operating environment, and are easy to apply and source. Other materials such as shrink wrap are difficult to apply and source, not suitable for the repair and maintenance in rural communities. Hot glue is a suitable yet cost effective adhesive with adequate strength to fix the aerofoils to the rods.

3 Design Calculations

3.1 Final Input Parameters

$V_u(ms^{-1})$	5.14	RPM ($\frac{r}{min}$)	140	Hub Radius (m)	0.26	Cp	0.3632	C_L	0.7202
$\rho, air(kgm^{-3})$	1.29	Torque (Nm)	3.25	# of Blades	6	P_s (W)	47.6475	C_D	0.0218
η_{system}	1	# of sections	15	Re	60,000	α	0.0873		

3.2 Blade Chords and Angles

Calculate the blade radius (R). R needs to be divided into the number of sections starting from the hub radius (0.26m) until the end (0.6896m). I will use one aerofoil section (The hub radius, r = 0.26m) to demonstrate. The calculations account for Prandtl tip loss and wake rotation. When starting the iterative scheme, begin with the betz limit (C_p = 0.593). C_p will update on subsequent iterations until the tolerance criteria is met.

$$\text{Rotor Radius: } R = \sqrt{\frac{2P_s}{C_p\eta\rho\pi V_u^3}} = \sqrt{\frac{2 \times 47.6475}{0.3632 \times 1 \times 1.29\pi \times 5.1444^3}} = 0.6896m$$

$$\text{Tip speed ratio: } \lambda_r = \frac{\Omega r}{V_u} = \frac{2 \times 140 \times \pi \times 60^{-1} \times 0.26}{5.1444} = 0.7410$$

$$\text{Local wind angle: } \phi = \frac{2}{3}tan^{-1}(\frac{1}{\lambda_r}) = \frac{2}{3}tan^{-1}(\frac{1}{0.7410}) = 0.6221rad$$

$$\text{Chord length with Wake Rotation: } c = \frac{8\pi r}{BC_L}(1 - cos\Phi) = \frac{8\pi 0.26}{6 \times 0.7202}(1 - cos(0.6221) = 0.2833m$$

$$\text{Blade Setting Angle: } \beta = \frac{180}{\pi} \times (0.6221 - 0.0873) = 30.6420 \text{ degrees}$$

Repeating this process for all other radii in the final design, the blade chord and angles are as follows.

Radii (m)	0.26	0.2907	0.3214	0.3520	0.3827	0.4134	0.4441	0.4748
Chord length (m)	0.2832	0.2821	0.2785	0.2732	0.2667	0.2598	0.2524	0.2449
Blade Setting Angle (Degrees)	30.64	28.57	26.68	24.94	23.34	21.88	20.54	19.31

Radii (m)	0.5055	0.5361	0.5668	0.5975	0.6282	0.6589	0.6896	
Chord length (m)	0.2373	0.2298	0.2226	0.2155	0.2088	0.2021	0.1959	
Blade Setting Angle (Degrees)	18.17	17.14	16.17	15.28	14.46	13.69	12.98	

3.3 Rotor Torque and Co-efficient of Power

Coefficients (r=0.26m): $C_n = C_Lcos(\phi) + C_Dsin(\phi) = 0.5980$ and $C_t = C_Lsin(\phi) - C_Dcos(\phi) = 0.4019$

$$\text{Factors: } F = \frac{2}{\pi}cos^{-1}(e^{-f}) \text{ where } f = \frac{B(R-r)}{2rsin(\phi)} = \frac{6 \times (0.6896 - 0.26)}{2 \times 0.26sin(0.6221)} = 8.5059$$

$$\text{Factors: } F = \frac{2}{\pi}cos^{-1}(e^{-8.5059}) = 0.9999$$

$$\text{Blade Solidarity: } \sigma' = \frac{Bc}{2\pi r} = \frac{6 \times 0.2832}{2\pi 0.26} = 1.0404$$

$$\text{Axial Induction Factor: } a = \frac{\sigma' C_n}{4Fsin^2(\phi) + \sigma' C_n} = \frac{1.0404 \times 0.5980}{4 \times 0.9999sin^2(0.6221) + 1.0404 \times 0.5980} = 0.3142$$

$$\text{Tangential load :}\frac{1}{2}\rho\frac{V_u^2(1-a)^2}{sin^2(\phi)}C_tc = \frac{1}{2}1.29\frac{5.1444^2(1-0.3142)^2}{sin^2(0.6221)} \times 0.4019 \times 0.2832 = 2.6920N$$

$$\text{Incremental Torque :}\Delta Q_{i,i+1} = \int_{r_i}^{r_{i+1}} P_T r dr = \int_{r_i}^{r_{i+1}} P_T r dr = \int_{0.26}^{0.2907} 2.6920 r dr = 0.0228Nm$$

Need the torque from all sections. Repeat with the remaining sections and sum to get $Q = 3.250$ as required.

$$\text{Power extracted: } P_E = Q \times \Omega = 3.250 \times 2 \times 140 \times \pi \times 60^{-1} = 47.6475$$

$$\text{Power available: } P_T = \frac{1}{2}\rho\pi R^2 V_u^3 = \frac{1}{2}1.29\pi 0.6896^2 5.1444^3 = 131.1796$$

$$\text{Power Coefficient: } C_p = \frac{P_E}{P_T} = \frac{47.6475}{131.1796} = 0.3632 \text{ as required.}$$

4 Appendices

4.1 Background and Product Requirements

Project Description

Background and Product Requirements

WindPac, a New Zealand producer of wind turbines would like to develop a product for sale to remote communities in the higher windy latitudes of the South Pacific and the World (remote Africa, South America etc.).

The product will consist of:

1. A generator assembly that will fit onto a locally constructed turbine tower.
2. A rotor kitset consisting of pre-cut rotor blade profiles that insert into rods, a hub for inserting the rods into, materials for covering the blades, gauges and jigs for correct assembly of the blade profiles onto the rods.

WindPac wants a prototype rotor and a mock-up of the rotor kitset. The kitset will consist of:

a) The parts required for the rotor blades;
b) Jigs and fixtures for correct assembly;
c) Instructions for assembly.

Rotor design specifications are set down below:

Air speed	8 – 12 knots, with a target at 10 knots
Torque-speed curve	Refer to Figure 1
Number of blades	3 to 8
Preferred turbine speed at 10 knots	~140 rpm

In your project groups, you will design, build, test and report on the construction of your prototype.

Project groups will be announced on **Friday 27 July.**

Figure 1: Background and Product Reqiurements in the Project Brief

4.2 Final Design Dimensions

Name	sd7003.dat
Number of Blades	6
Number of Sections	15
Windspeed (m/s)	5.1444
Alpha (rad)	0.087266
Reynold's Number	60000
Plot	3
Thickness	8.5%
Camber	1.2%
Shape	3
Minimum Thickness (mm)	16.65

Section	1	2	3	4	5	6	7	8
Radii (m)	0.26	0.290683	0.321367	0.35205	0.382733	0.413417	0.4441	0.474783
Chordlength (m)	0.283274403	0.282054	0.278453	0.273203	0.266861	0.259838	0.252436	0.244874
Blade Setting Angle (Degrees)	30.6420011	28.57435	26.67673	24.93728	23.34338	21.8824	20.54216	19.31121
Section	9	10	11	12	13	14	15	
Radii (m)	0.505466617	0.53615	0.566833	0.597517	0.6282	0.658883	0.689567	
Chordlength (m)	0.237306432	0.229841	0.222551	0.215485	0.208672	0.202128	0.19586	
Blade Setting Angle (Degrees)	18.17896164	17.13578	16.17293	15.28259	14.45771	13.69203	12.97994	

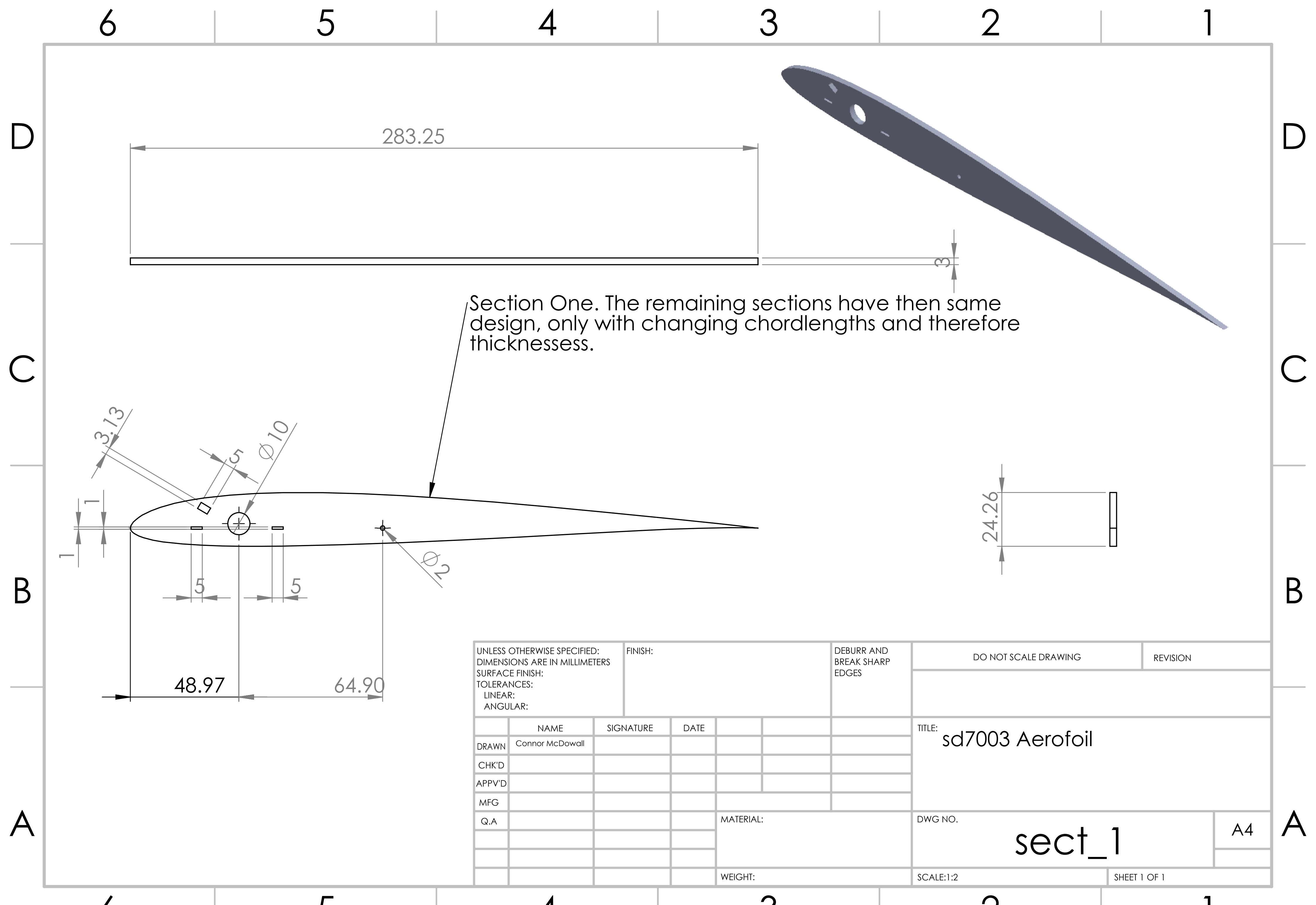
Figure 2: Exact Dimensions for the Final Design

4.3 Costs

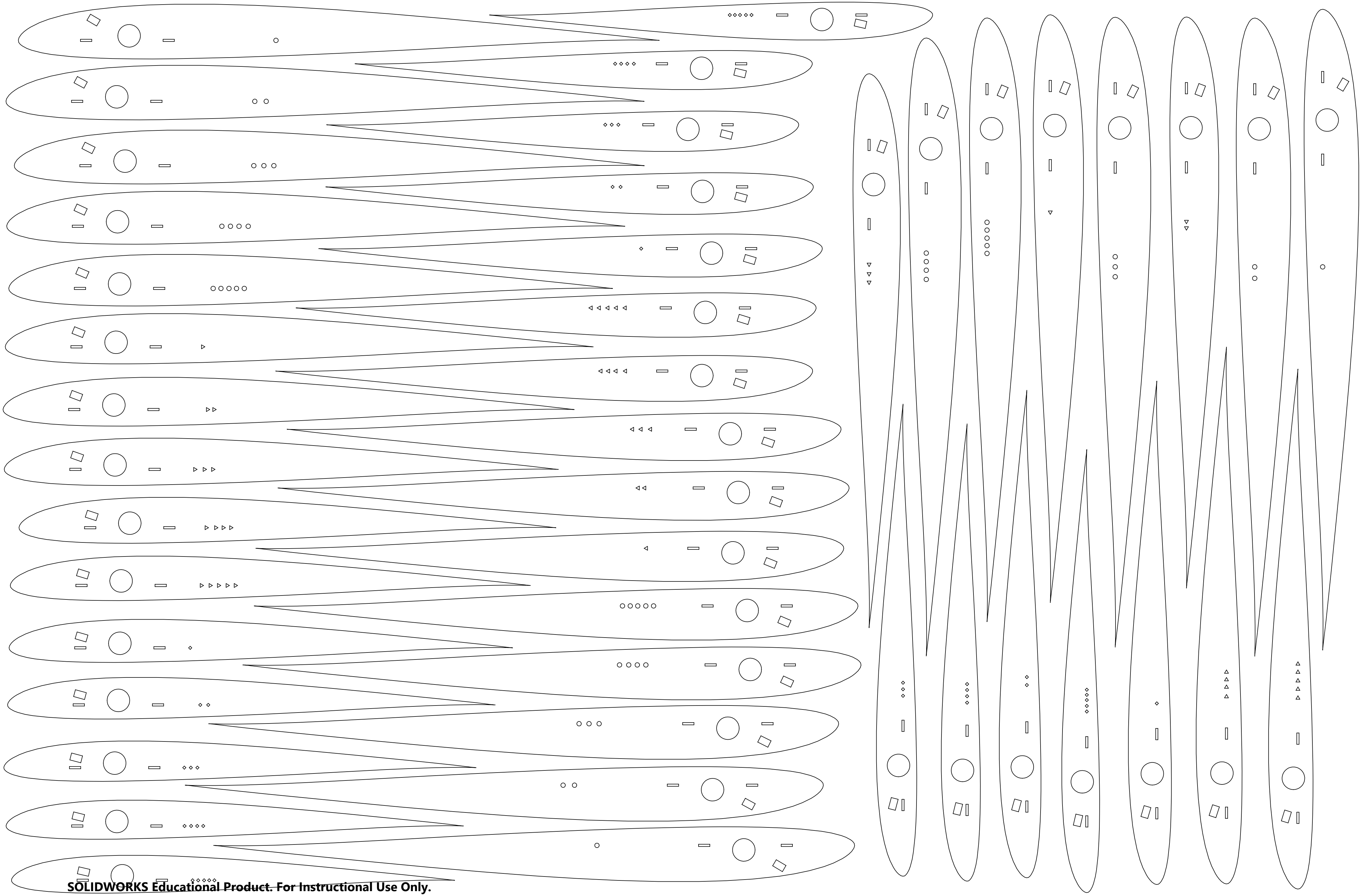
Item	Cost/Unit ($)	Cost/Length ($/M)	# of Units	Length/blade (m)	# of Blades	Cost ($)	Notes
Hub	150	#N/A	1	#N/A	#N/A	150	Only one hub
Aluminium rods	#N/A	3.20	#N/A	0.648	7	2.07	Includes an extra rod for one extra aerofoil blade. The rod doesn't go to the centre so is slightly shorter than the blade length. The rod length is blade length - hub radius (50mm) + rod connector hub insert length (8mm)
Acrylic sheets (PMMA)	6.36	#N/A	3	#N/A		19.08	Includes two blade setting rigs and an extra set of aerofoils for one extra blade.
Rod-Hub connection fitting	5.71	#N/A	7	#N/A		39.97	Includes an extra connector for a spare blade.
Cellotape	4.35	#N/A	3	#N/A		13.05	Two and a half blades per roll.
Duct Tape	10.89	#N/A	1	#N/A		10.89	For the blade edges
Hot Glue	6.73		1	#N/A		6.73	One for gluing all blades.
Total						241.79	Total cost for rotor + one spare blade and two blade setting rigs for the kitset.

Figure 3: Costs for the Final Design including Two Blade Setting rigs and One Spare Blade

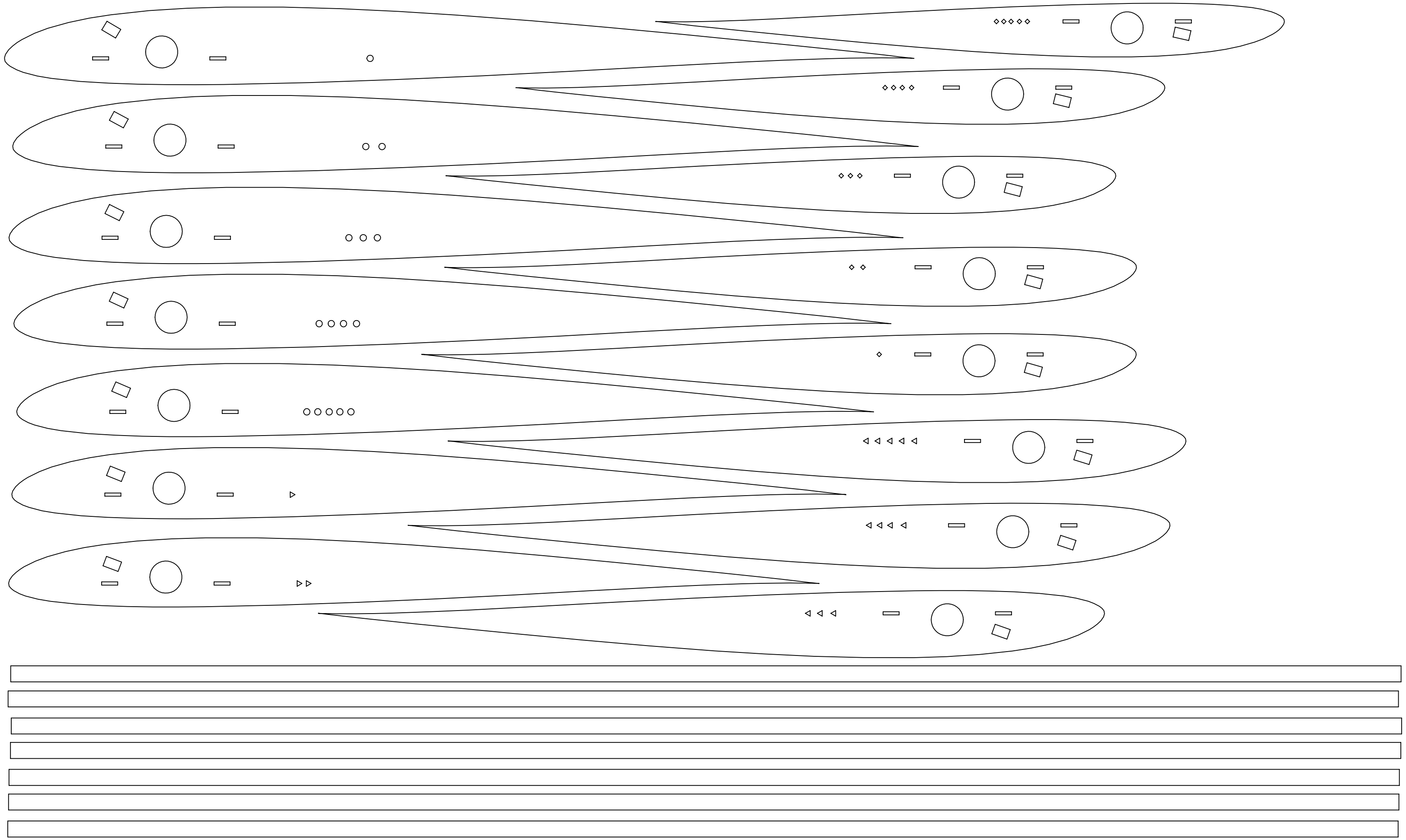
4.4 Models and Drawings : Parts, Rigs and Assemblies



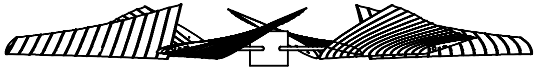
SOLIDWORKS Educational Product. For Instructional Use Only.

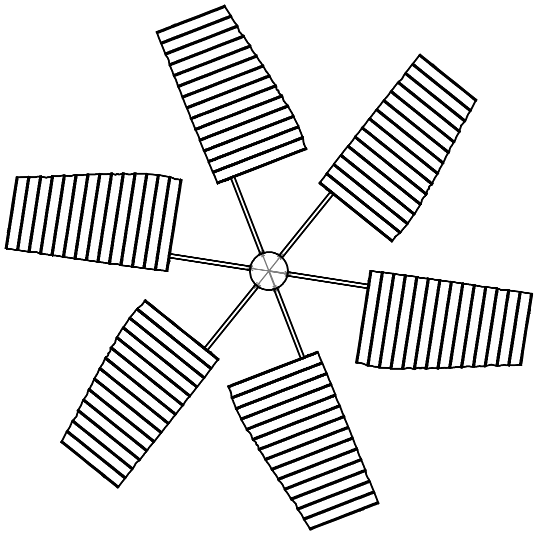


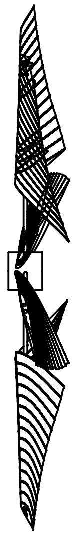
SOLIDWORKS Educational Product. For Instructional Use Only.

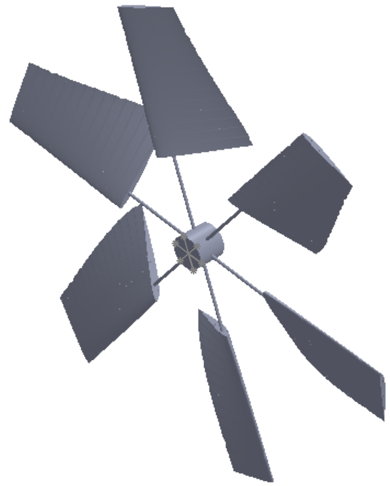


SOLIDWORKS Educational Product. For Instructional Use Only.







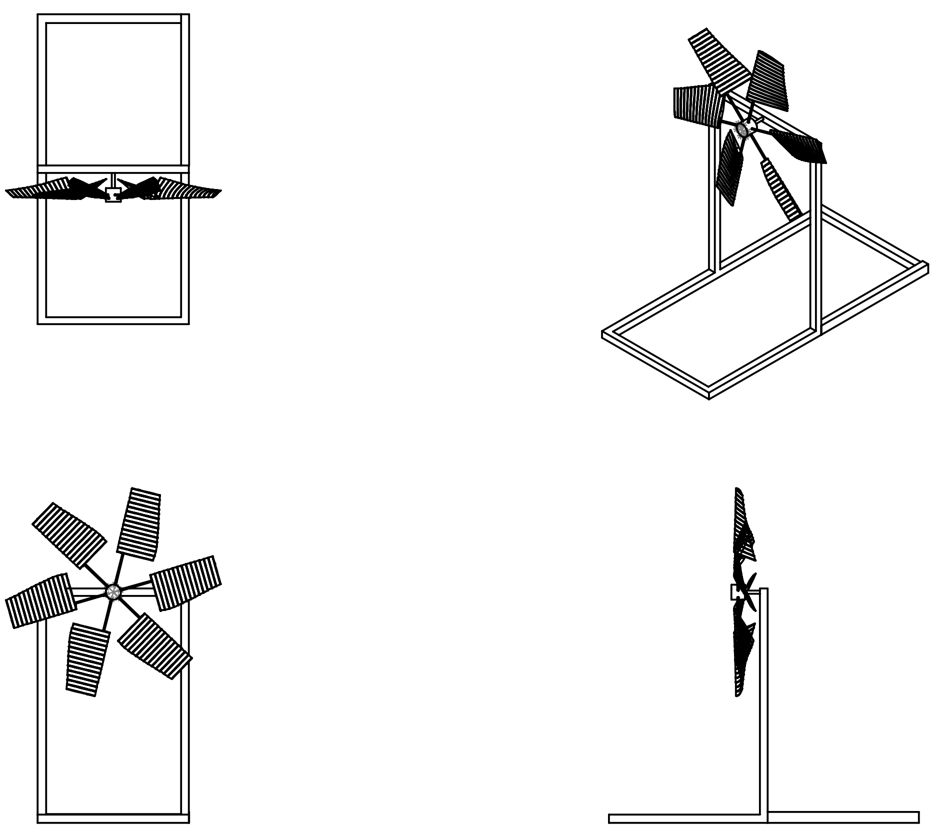


UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:	DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING
			REVISION

	NAME	SIGNATURE	DATE
DRAWN	Connor McDowall		
CHK'D			
APPV'D			
MFG			
Q.A			

TITLE:	Rotor Turbine
MATERIAL:	
DWG NO.	turbine
SIZE	A4
WEIGHT:	
SCALE:1:20	SHEET 1 OF 1

SOLIDWORKS Educational Product. For Instructional Use Only.



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:	DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING
			REVISION

	NAME	SIGNATURE	DATE
DRAWN	Connor McDowall		
CHK'D			
APPV'D			
MFG			
Q.A			

TITLE: Turbine on the rig

MATERIAL:

WEIGHT:

DWG NO. FinalAssembly

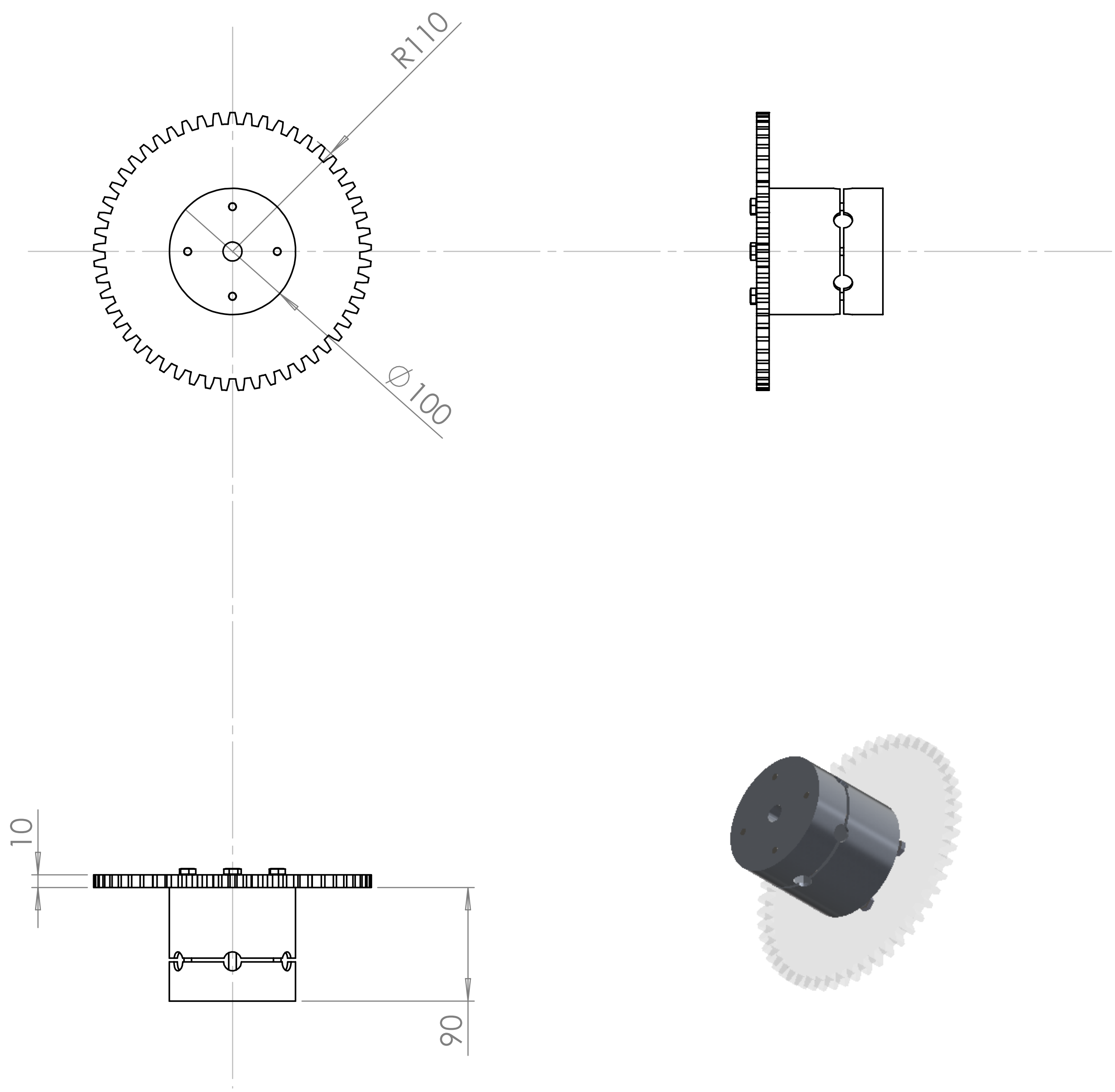
A4

SCALE:1:50

SHEET 1 OF 1

SOLIDWORKS Educational Product. For Instructional Use Only.

Appendix A.1: Technical Drawing - Turbine Hub



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:

FINISH:

DEBURR AND BREAK SHARP EDGES

DO NOT SCALE DRAWING

REVISION

	NAME	SIGNATURE	DATE
DRAWN	Oliver Wilson		
CHK'D	Kevin Jia		
APPV'D			
MFG			
Q.A			

TITLE:

MATERIAL:

DWG NO.

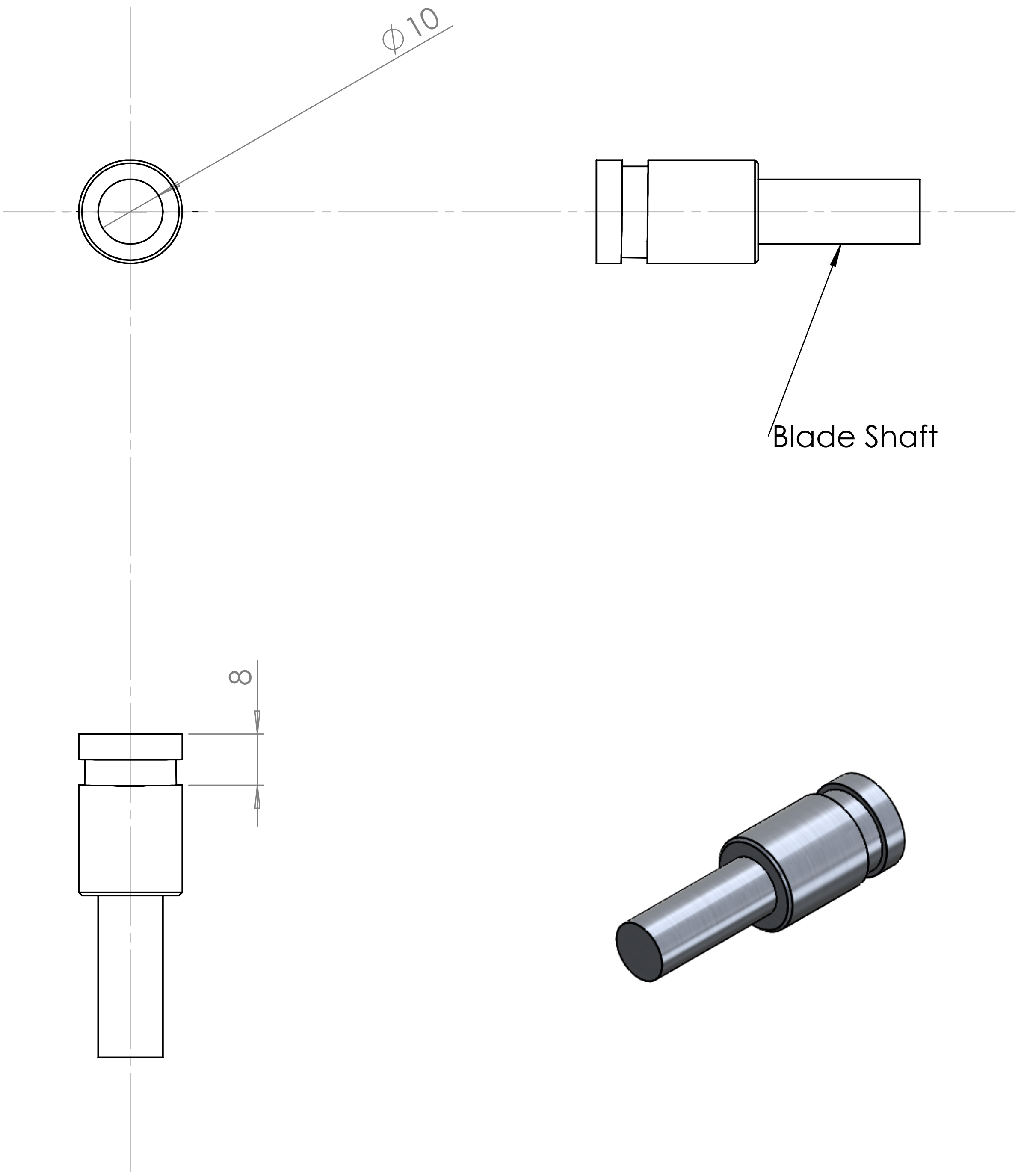
A4

WEIGHT:

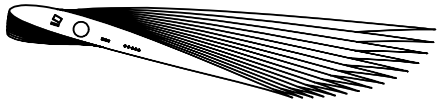
SCALE:1:5

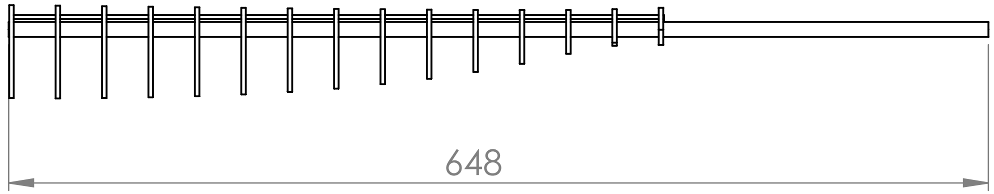
SHEET 1 OF 1

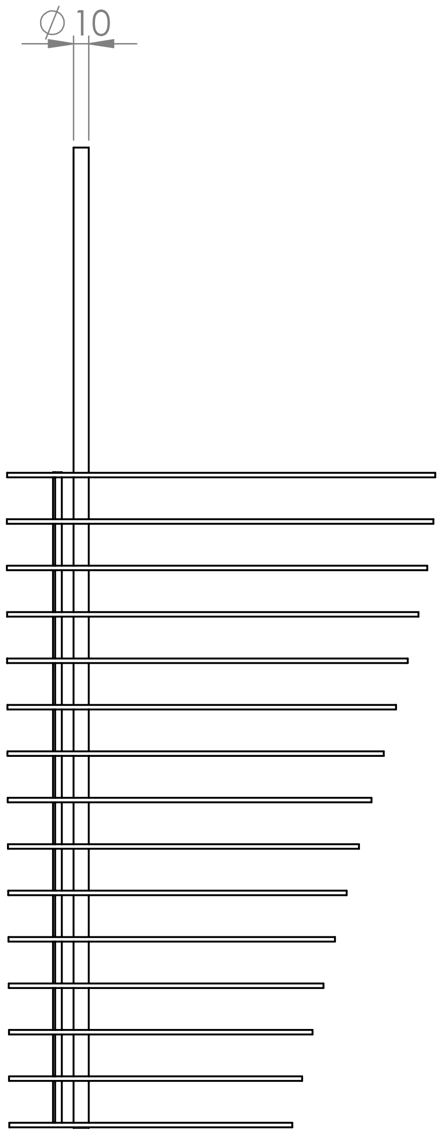
Appendix A.2: Technical Drawing - Blade Shaft / Connector



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:		DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE		TITLE:
DRAWN	Oliver Wilson				
CHK'D	Kevin Jia				
APPV'D					
MFG					
Q.A				MATERIAL:	DWG NO. A4
				WEIGHT:	SCALE:1:1 SHEET 1 OF 1







UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:	DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION

	NAME	SIGNATURE	DATE		
DRAWN	Connor McDowall				
CHK'D					
APPV'D					
MFG					
Q.A				MATERIAL:	
				WEIGHT:	

TITLE: Blade Assembly

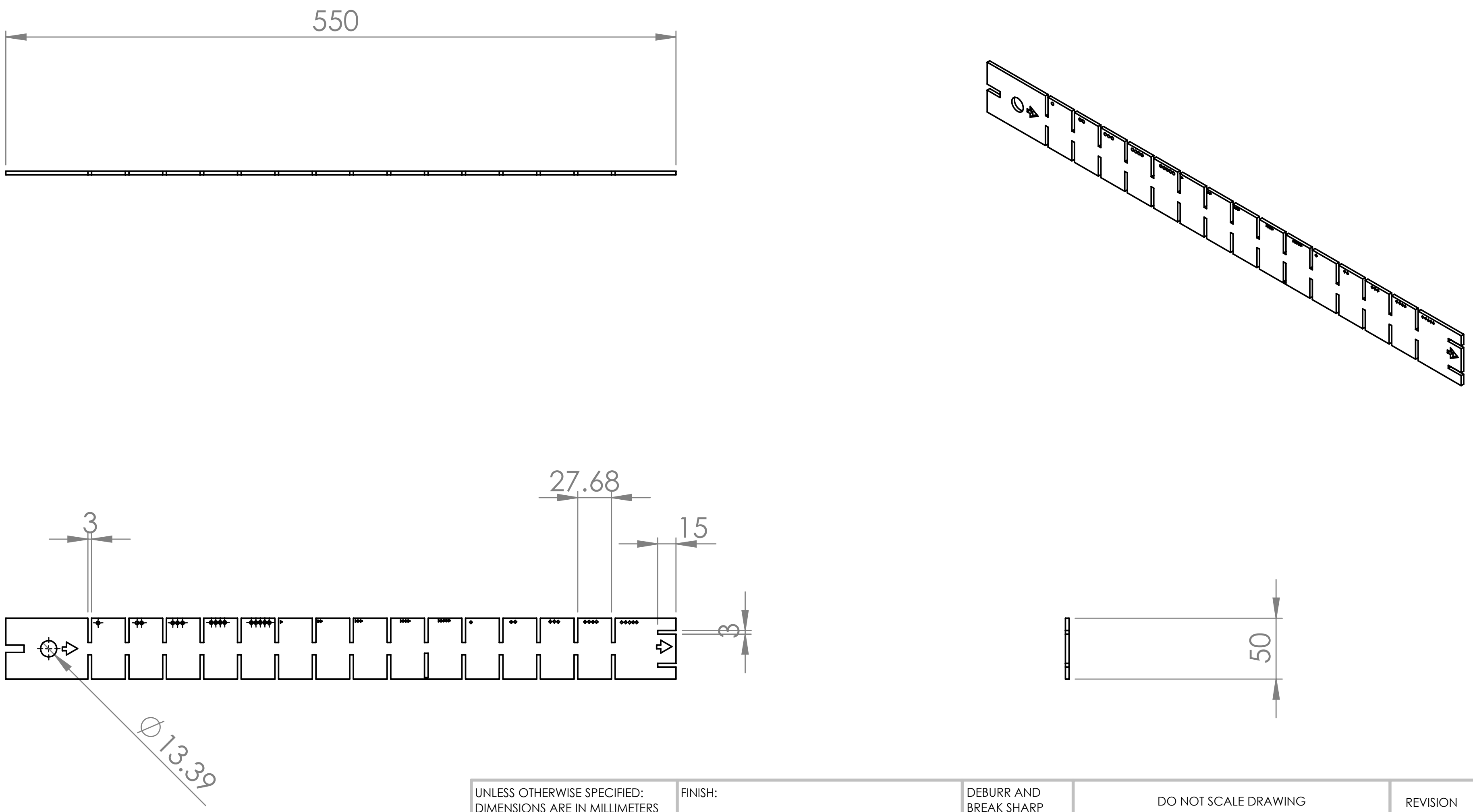
DWG NO. blade_assem

A4

SCALE:1:10

SHEET 1 OF 1

SOLIDWORKS Educational Product. For Instructional Use Only.



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:	DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING / REVISION

	NAME	SIGNATURE	DATE
DRAWN	Connor McDowall		
CHK'D			
APPV'D			
MFG			
Q.A			

TITLE: Blade Distance Setting Tool

MATERIAL:

DWG NO. Guide

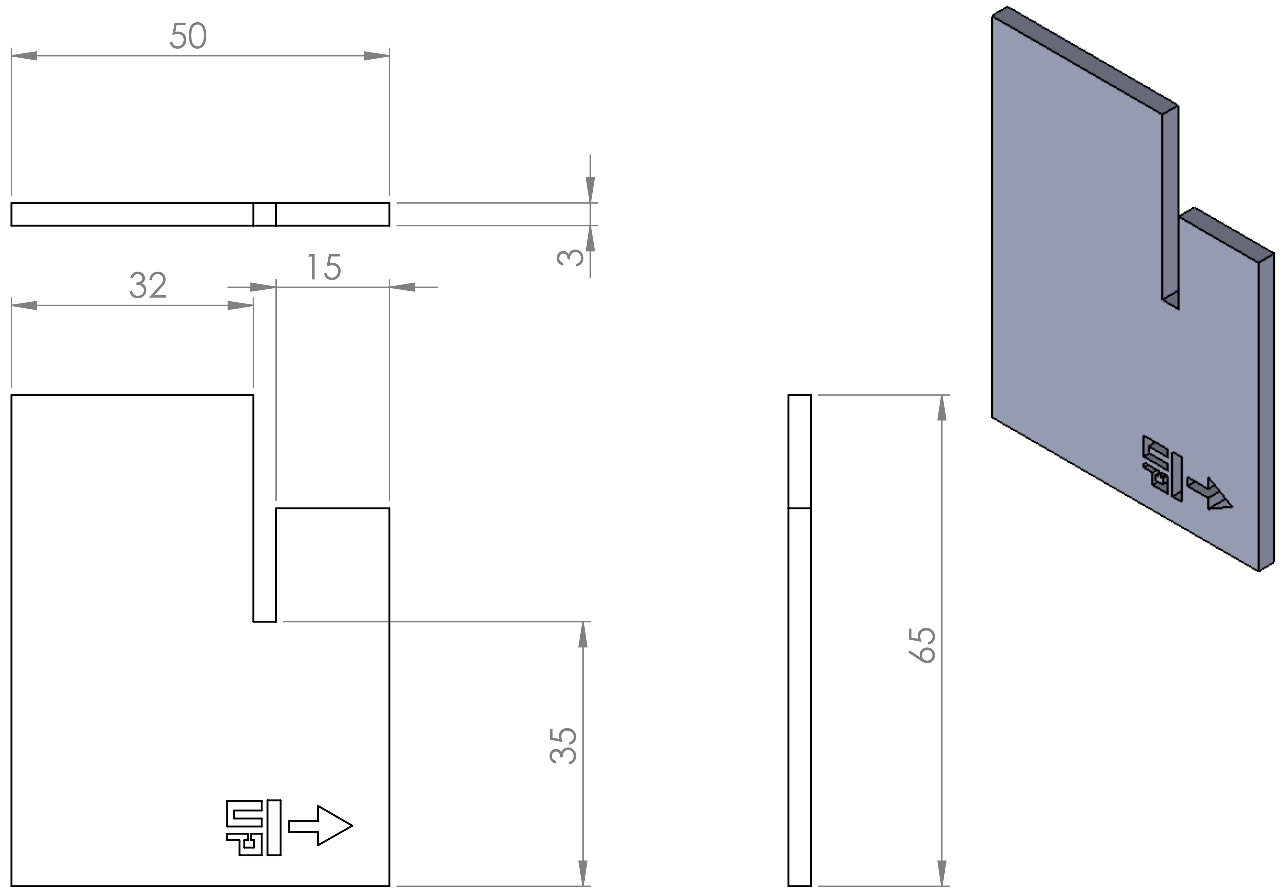
A4

WEIGHT:

SCALE:1:5

SHEET 1 OF 1

SOLIDWORKS Educational Product. For Instructional Use Only.



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:

FINISH:

DEBURR AND BREAK SHARP EDGES

DO NOT SCALE DRAWING

REVISION

	NAME	SIGNATURE	DATE
DRAWN	Connor McDowall		
CHK'D			
APPV'D			
MFG			
Q.A			

TITLE: Stand for the Blade Distance Setting Tool

MATERIAL:

DWG NO. Stand

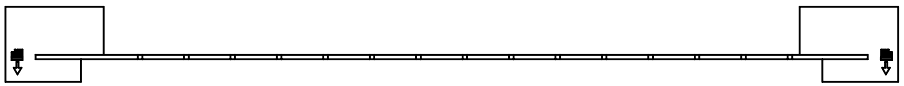
A4

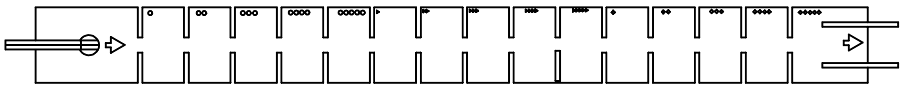
WEIGHT:

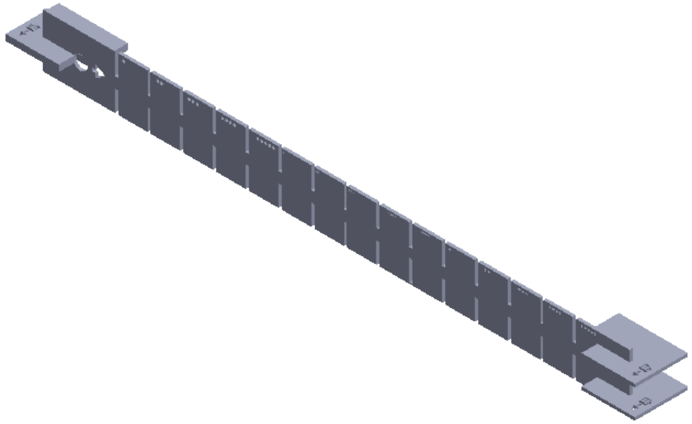
SCALE:1:1

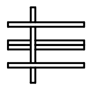
SHEET 1 OF 1

SOLIDWORKS Educational Product. For Instructional Use Only.









UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:			DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE		TITLE: Blade Setting Tool Assembly	
DRAWN	Connor McDowall					
CHK'D						
APPV'D						
MFG						
Q.A				MATERIAL:	DWG NO. GuideAssem	A4
				WEIGHT:	SCALE:1:5	SHEET 1 OF 1

SOLIDWORKS Educational Product. For Instructional Use Only.

4.5 Analysis: A Nine-Step Process

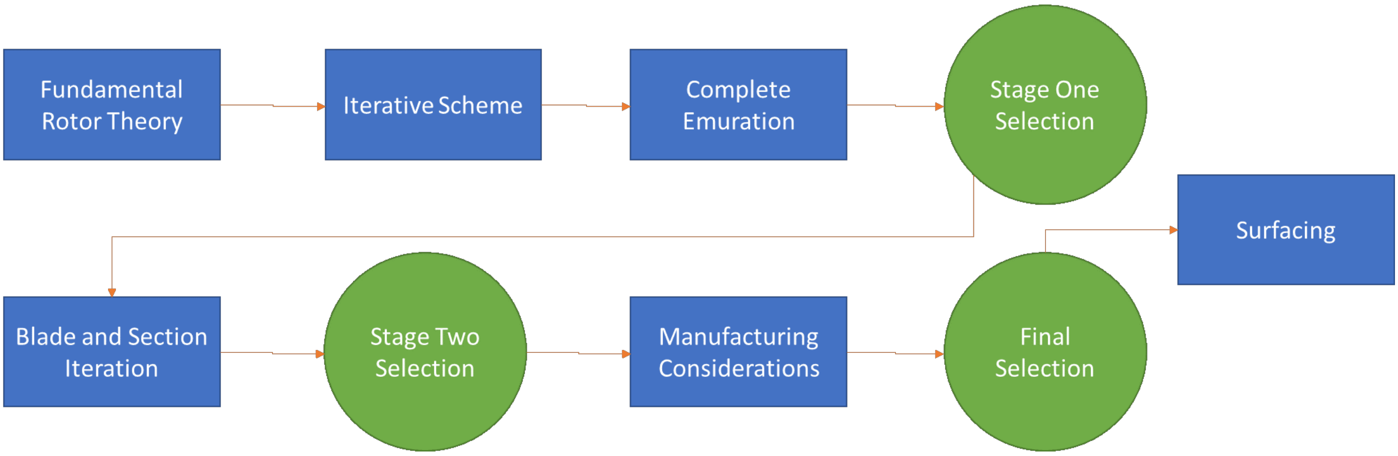


Figure 4: Method of Analysis: A Nine Step Process

5 Fundamental Rotor Theory

See the pdf files on canvas for lectures three to eight covering the fundamental rotor theory. Follow the link below.
"https://canvas.auckland.ac.nz/courses/30227/files/folder/Lecture%20Slides?"

5.1 Iterative Scheme

Listing 1: XfoilIteration

```
% This Script iterates through all aerofoil designs, trying to find the
% best combination for one aerofoil type.

% Convert Airfoil Bank (Toggle via commenting)
bank = dir('airfoil bank');
foilnames = {bank.name};

% Set up the good aerofoils (Toggle via commenting)
good = dir('Good');
goodfoils = {good.name};

% Read adjust names to get the data files
for i = 1:length(goodfoils)
    temp = strsplit(goodfoils{i},'.');
    goodfoils{i} = strcat(convertCharsToStrings(temp(1)),'.dat');
end

% Chosen design (Toggle via Commenting)
goodfoils = {'sd7003.dat','sd7003.dat','sd7003.dat'};

% Set the global variables
global Vu RPM rho torque eta nSections hubRadius B Re ReRange Storage

```

```
Vu = 10* 0.51444;  % Design speed, e.g. 10 knots an hour (Range: 8/10/12
    knots).
RPM = 140;          % Target RPM
rho = 1.29;         % Density of air
torque = 3.25;      % Target torque, from datasheet
eta = 1;            % System efficiency
nSections = 15;
hubRadius = 0.26;   % Radius of hub + any further radius with no blade
   allowed
B = 6;              % Number of blades (Set the appropriate number of
   blades).
Re = 60000; % Approximate Reynolds number for design,
ReRange = 40000:10000:80000;
Storage = 'H:\ENGSCI 363 BEM\BEM\clcdfigures'; %Storage String Directory

% Saving variables
SheetNum = 2;
iteration =6;
Count = 1;
max = 40;

% Create a for loop for setp through evaluate (changed as the design
% process, currently set up to text final design).
for i = 3:length(goodfoils)
    disp(goodfoils{i})
    % Call evaluate turbine
    try
        % Create the figure for the aerofoil
        [savefile] = evaluatePlot(foilnames{i});
        % Get parameters fro the aerofoil.
        [~, design1] = evaluateTurbine(goodfoils{i});
        % Create a Table
        name = goodfoils{i};
        T = table({name},design1.Cp, design1.blades ,design1.nSections,
           design1.Windspeed,design1.alpha, design1.Re, design1.r,
           design1.chord, design1.beta);
        T(1,:);
        % Save this table to an excel file
        filename = 'FinalIterations.xlsx';
        writetable(T,filename,'Sheet',SheetNum,'Range',strcat('A',
           num2str(i+1+iteration*max)), 'WriteVariableNames', false)
        Count = Count + 1;
    catch
    end
end
```

Listing 2: evaluateTurbine

```
function [obj, design] = evaluateTurbine(varargin)
% This function evaluates a turbine design, given a set of properties in
   a
% format suitable for optimisation using the metaheuristics code by Yang
   .
% 1 Input:  x - either a string that contains the ONE aerofoil type to
```

```
be
%               used, or else a cell array that contains the aerofoil type
%               at each cross-section.
%            -- OR --
% 4 Inputs: for aerofoil information already found from elsewhere
%           Inputs must be in correct order.
%           1. x = 1 x n matrix containing the sequence number (1...k) of the
%                aerofoil to be used at each section. n == nSections
%           2. CL = 1 x k matrix containing lift coefficient for each of k
%           aerofoils
%           3. CD = 1 x k matrix containing drag coefficient at optimal angle
%           of attack for each of k aerofoils.
%           4. alpha = 1 x k matrix containing optimal angle of attack for
%           each of k aerofoils
% Outputs:  obj = objective value of interest, to be defined for the application.
%           design = structure with turbine design features, specifically
%               r = cross-sectional radii
%               chord = chord length
%               Cp = power coefficient
%               alpha = angle of attack
%               beta = local twist angles

% Connor McDowall cmcd398 530913386

% Constants (to go into params)
global Vu RPM rho torque eta nSections hubRadius B Re
mach = 0;
% Extract names if one cell input.
if nargin == 1
    x = varargin{1};
    if isa(x, 'cell')
        assert(nSections == length(x))
    else
        x = cellstr(x);
    end

    % Find the unique items in the cell array
    unique_xfoil = unique(x);
    % Use a loop to step through and the optimum ratios.
    % Initialise alpha values.
    alpha_loop = 0:0.5:15;
    for j = 1:length(unique_xfoil)
        % Call X foil to get alpha, CD and CL of that cell array.
       [pol,~] = callXfoil(unique_xfoil{j},alpha_loop , Re, mach);
        % Find index position of the max ratio
        [~,i] = max((pol.CL./pol.CD));
```

```
        % Save the CD, CL and alpha values of the max CL/CD ratio position
        Cdsave = pol.CD(i);
        Clsave = pol.CL(i);
        alphasave = deg2rad(alpha_loop(i));
        % Use a loop to step through the length array and place CD, CL and
        % alpha values of unique_xfoil of the same type.
        for k =  1:length(x)
            if x{k} == unique_xfoil{j}
                Cd(k) = Cdsave;
                Cl(k) = Clsave;
                alpha(k) = alphasave;
            end
        end
    end

elseif nargin == 4
    % Aerofoil info already pre-generated and is read in.
    x = varargin{1};
    assert(nSections == length(x));
    LiftCoeffs = varargin{2};
    DragCoeffs = varargin{3};
    Alphas = varargin{4};

    Cl = LiftCoeffs(x); % Lift coeff for EACH cross-section
    Cd = DragCoeffs(x); % Drag coeff for EACH cross-section
    alpha = Alphas(x); % angle of attack for each aerofoil at EACH cross-section
else
    error("Incorrect number of inputs")
end

% BEM Calculations
% Establish Power requirement and estimate Cp.
omega = (RPM.*2.*pi()./60);
Ps = omega.*torque;
iterate = true;
Cp = 0.593; % Benz Limit
% Create the loop to iterate until Cp converges.
while iterate == true
    % Calculate the overall rotor radius
    R = sqrt((2.*Ps./(Cp.*eta.*rho.*pi().*Vu.^3)));

    % Individual Blade Radius, radia of all aerofoil locations.
    rind = linspace(hubRadius, R,nSections) ;

    % Calculate the individual wind speed ratios, from the centre of the
    % blade
    lambda_ind = (omega.*(rind)./Vu);

    % Calculate the local wind angle and chord length off each section.
    % This is with wake rotation.
```

```
    phiw = (2./3)*atan(1./lambda_ind); % radians
    cwake =((8.*pi().*(rind))./(B.*Cl)).*(1 - cos(phiw));

    % Calculate the blade setting angles in degrees
    beta = (180./pi).*(phiw- alpha) ;

    % Calculate the axial induction factors
    % Get the constants
    Cn = Cl.*cos(phiw) + Cd.*sin(phiw);
    Ct = Cl.*sin(phiw) - Cd.*cos(phiw);
    f = ((B*(R - rind))./(2.*(rind).*sin(phiw))); % Check r = Rotor Radius/ Number of sections
    F = (2./pi()).*acos(exp(-f));

    % Calculate the blade solidarity
    sigmaprime = (B.*cwake./(2.*pi().*rind));

    % Axial induction factor
    a = (sigmaprime.*Cn)./(4.*F.*(sin(phiw)).^2 + sigmaprime.*Cn);

    % Calculate the tangential load at each cross section
    pt = (0.5.*rho.*((Vu.^2).*((1-a).^2)./(sin(phiw).^2))).*Ct.*cwake;

    % Integrate between sections to find the total torque between the
    % cross sections.
    Q = B.*trapz(rind,pt.*rind);

    % Find the power extracted and the power available to the wind, therefore
    % finding the power co-efficient Cp
    Pe = Q.*omega;
    Pt = 0.5.*rho.*pi().*(R.^2).*(Vu.^3);
    Cp_iterate = Pe./Pt;

    % Control the looping
    if abs(Cp_iterate - Cp)<=eps
        % Reset looping variable if it has converged.
        iterate = false;
    end
    Cp = Cp_iterate;
end

% Store the things in a structure
% Define the objective function
obj = Cp;
% Blade design parameters
design = struct('r',rind,'chord',cwake,'Cp',Cp,'alpha',alpha,'beta',beta,'blades',B,'Re',Re,'nSections',nSections,'Windspeed',Vu,'Torque',Q,'Power',Pe,'PowerSystem',Pt);
%performance = struct('Reynolds Number',Re,'Wind speed',Vu,'Torgue',Q, 'Power',Pe,'Power Coefficient',Cp);
return
```

Listing 3: evaluatePlot

```
function [savefile] = evaluatePlot(varargin)
% This function evaluates the Cl and Cd, given a set of properties in a
% format suitable for optimisation using the metaheuristics code by Yang.
% 1 Input:  x - either a string that contains the ONE aerofoil type to be
%                 used, or else a cell array that contains the aerofoil type
%                 at each cross-section.
%              -- OR --
% 4 Inputs: for aerofoil information already found from elsewhere
%           Inputs must be in correct order.
%           1. x = 1 x n matrix containing the sequence number (1...k) of the
%                  aerofoil to be used at each section. n == nSections
%           2. CL = 1 x k matrix containing lift coefficient for each of k
%           aerofoils
%           3. CD = 1 x k matrix containing drag coefficient at optimal angle
%           of attack for each of k aerofoils.
%           4. alpha = 1 x k matrix containing optimal angle of attack for
%           each of k aerofoils
% Outputs:  Location of the saved image for the matlab
%           savefile = File Location of saved image

% Connor McDowall cmcd398 530913386
% Clear Figure
clf;

% Constants (to go into params)
global nSections ReRange Storage
mach = 0;
% Extract names if one cell input.
if nargin == 1
    x = varargin{1};
    if isa(x, 'cell')
        assert(nSections == length(x))
    else
        x = cellstr(x);
    end

    % Find the unique items in the cell array
    unique_xfoil = unique(x);
    % Use a loop to step through and the optimum ratios.
    % Initialise alpha values and legend
    plotLegend = {};
    alpha_loop = 0:1:15;
    for p = 1:length(ReRange)
        for j = 1:length(unique_xfoil)
            % Call X foil to get alpha, CD and CL of that cell array.
```

```
            [pol,~] = callXfoil(unique_xfoil{j},alpha_loop , ReRange(p), mach);
            % Plot the Cl/CD Function in each iteration of the loop.
            plot(pol.alpha,(pol.CL./pol.CD))
            plotLegend{p} = num2str(ReRange(p));
            hold on
        end
    end

    %Name and Label plot
    ylabel('Ratio: Cl/Cd')
    xlabel('Alpha (Degrees)')
    split = strsplit(unique_xfoil{1},'.');
    foil = strcat(split{1},'.png');
    title(split) % Assuming only one aerofoil is passed in.
    legend(plotLegend);
    % Show the figure
    figure(gcf);
    % Save the figure to the current directory.
    savefile = fullfile(Storage, foil);
    saveas(gcf,savefile);

else
    error("Incorrect number of inputs")
end
```

Listing 4: callXfoil

```
function [pol, foil] = callXfoil(coord, alpha, Re, Mach)
% This function acts as a general interface to xfoil.m by Louis Edelmann:
% https://au.mathworks.com/matlabcentral/fileexchange/49706-xfoil-interface-updated
%
% Inputs: coord = coordinates of aerofoil.
%                      3 cases: 1. 'NACAxxxxx' for NACA 4 or 5 digits
%                               2. 'xxxx.dat' for an aerofoil in the airfoil
%                               bank.
%                               3. an n by 2 array of x and y
%                               coordinates.
%                    For non NACA cases, 300 panel points are interpolated to
%                    hopefully give convergence. In all cases, max 100
%                    iterations are run.
%          alpha = angle(s) of attack to consider Re = Reynold's number of
%          interest. Mach = Mach number of interest. Typically 0 for
%          non-turbulent flow.
% Outputs: see xfoil.m
%
% This code is supplied with an adapted version of the UIUC Airfoil
% Database (http://m-selig.ae.illinois.edu/ads/coord_database.html) by
% Michael Selig. Divahar Jayaraman adapted these to a consistent format
```

```
in
% her code for 2D Potential Flows, as found below...
% http://au.mathworks.com/matlabcentral/fileexchange/12790-panel-method-based-2-d-potential-flow-simulator
% This is used by loading the 'airfoil bank' folder into the path when
% needed.
%
% Kevin Jia, UoA EngSci, 2017.

if isa(coord, 'char') == true && strcmp(coord(1:4), 'NACA') == true
    % A built in NACA airfoil.
    [pol,foil] = xfoil(coord, alpha, Re, Mach,'PLOP G', 'pane ppar n 300/', 'oper/iter 200');
elseif isa(coord, 'char') == true && strcmp(coord(length(coord)-3:length(coord)), '.dat')
    % An aerofoil in the airfoil bank folder.
    addpath('airfoil bank')
    if exist(coord)
        % Import data, first remove heading.
        data = importdata(coord, ' ', 1);
        coordinates = data.data;
        [pol,foil] = xfoil(coordinates, alpha, Re, Mach, 'PLOP G', 'pane n 300/', 'oper/iter 200');
    else
        error('.dat file does not exist')
    end

else
    % 2D array of coordinates
    [pol,foil] = xfoil(coord, alpha, Re, Mach,'PLOP G', 'pane n 200/', 'oper/iter 100');
end

pol = checkPol(pol, alpha);

return

function polOut = checkPol(polIn, alpha)

if length(polIn.alpha) ~= length(alpha)
    % Lengths do not match, need to do an interpolation for some angle(s).

    % Copy structure
    polOut = polIn;
    % Find missing angles
    existingInds = ismember(alpha', polIn.alpha);
    missingInds = ~ismember(alpha', polIn.alpha);
    needToFind = alpha(missingInds);
    missingLifts = interp1(polIn.alpha, polIn.CL, needToFind);
    missingDrags = interp1(polIn.alpha, polIn.CD, needToFind);

    allLifts(existingInds) = polIn.CL;
```

```
    allLifts(missingInds) = missingLifts;

    allDrags(existingInds) = polIn.CD;
    allDrags(missingInds) = missingDrags;

    polOut.CL = allLifts';
    polOut.CD = allDrags';
    polOut.alpha = alpha';

else
    polOut = polIn;
end

return
```

Listing 5: xfoil.m

```
function [pol,foil] = xfoil(coord,alpha,Re,Mach,varargin)
% Run XFoil and return the results.
% [polar,foil] = xfoil(coord,alpha,Re,Mach,{extra commands})
%
% Xfoil.exe needs to be in the same directory as this m function.
% For more information on XFoil visit these websites;
%  http://web.mit.edu/drela/Public/web/xfoil
%
% Inputs:
%    coord: Normalised foil co-ordinates (n by 2 array, of x & y
%           from the TE-top passed the LE to the TE bottom)
%           or a filename of the XFoil co-ordinate file
%           or a NACA 4 or 5 digit descriptor (e.g. 'NACA0012')
%    alpha: Angle-of-attack, can be a vector for an alpha polar
%       Re: Reynolds number (use Re=0 for inviscid mode)
%     Mach: Mach number
% extra commands: Extra XFoil commands
%           The extra XFoil commands need to be proper xfoil commands
%           in a character array. e.g. 'oper/iter 150'
%
% The transition criterion Ncrit can be specified using the
% 'extra commands' option as follows,
% foil = xfoil('NACA0012',10,1e6,0.2,'oper/vpar n 12')
%
%   Situation           Ncrit
%   -----------------   -----
%   sailplane           12-14
%   motorglider         11-13
%   clean wind tunnel   10-12
%   average wind tunnel    9 <= standard "e^9 method"
%   dirty wind tunnel    4-8
%
% A flap deflection can be added using the following command,
% 'gdes flap {xhinge} {yhinge} {flap_defelction} exec'
%
% Outputs:
%  polar: structure with the polar coefficients (alpha,CL,CD,CDp,CM,
```

```
%           Top_Xtr,Bot_Xtr)
%   foil: stucture with the specific aoa values (s,x,y,UeVinf,
%           Dstar,Theta,Cf,H,cpx,cp) each column corresponds to a different
%           angle-of-attack.
%          If only one left hand operator is specified, only the polar will be parsed and output
%
% If there are different sized output arrays for the different incidence
% angles then they will be stored in a structured array, foil(1),foil(2)...
%
% If the output array does not have all alphas in it, that indicates a convergence failure in Xfoil.
% In that event, increase the iteration count with 'oper iter ##;
%
% Examples:
%    % Single AoA with a different number of panels
%    [pol foil] = xfoil('NACA0012',10,1e6,0.0,'panels n 330')
%
%    % Change the maximum number of iterations
%    [pol foil] = xfoil('NACA0012',5,1e6,0.2,'oper iter 50')
%
%    % Deflect the trailing edge by 20deg at 60% chord and run multiple incidence angles
%    [pol foil] = xfoil('NACA0012',[-5:15],1e6,0.2,'oper iter 150','gdes flap 0.6 0 5 exec')
%    % Deflect the trailing edge by 20deg at 60% chord and run multiple incidence angles and only
%    parse or output a polar.
%    pol = xfoil('NACA0012',[-5:15],1e6,0.2,'oper iter 150','gdes flap 0.6 0 5 exec')
%    % Plot the results
%    figure;
%    plot(pol.alpha,pol.CL); xlabel('alpha [\circ]'); ylabel('C_L'); title(pol.name);
%    figure; subplot(3,1,[1 2]);
%    plot(foil(1).xcp(:,end),foil(1).cp(:,end)); xlabel('x');
%    ylabel('C_p'); title(sprintf('%s @ %g\\circ',pol.name,foil(1).alpha(end)));
%    set(gca,'ydir','reverse');
%    subplot(3,1,3);
%    I = (foil(1).x(:,end)<=1);
%    plot(foil(1).x(I,end),foil(1).y(I,end)); xlabel('x');
%    ylabel('y'); axis('equal');
%

% Some default values
if ~exist('coord','var'), coord = 'NACA0012'; end;
if ~exist('alpha','var'), alpha = 0;     end;
if ~exist('Re','var'),     Re = 1e6;      end;
if ~exist('Mach','var'),   Mach = 0.2;    end;
Nalpha = length(alpha); % Number of alphas swept
```

```
% default foil name
foil_name = mfilename;

% default filenames
wd = fileparts(which(mfilename)); % working directory, where xfoil.exe needs to be
fname = mfilename;
file_coord= [foil_name '.foil'];

% Save coordinates
if ischar(coord),  % Either a NACA string or a filename
  if isempty(regexpi(coord,'^NACA *[0-9]{4,5}$')) % Check if a NACA string
%      foil_name = coord; % some redundant code removed to go green ( ~isempty if uncommented)
%    else                 % Filename supplied
    % set coord file
    file_coord = coord;
  end;
else
  % Write foil ordinate file
  if exist(file_coord,'file'),  delete(file_coord); end;
  fid = fopen(file_coord,'w');
  if (fid<=0),
    error([mfilename ':io'],'Unable to create file %s',file_coord);
  else
    fprintf(fid,'%s\n',foil_name);
    fprintf(fid,'%9.5f   %9.5f\n',coord');
    fclose(fid);
  end;
end;

% Write xfoil command file
fid = fopen([wd filesep fname '.inp'],'w');
if (fid<=0),
  error([mfilename ':io'],'Unable to create xfoil.inp file');
else
  if ischar(coord),
    if ~isempty(regexpi(coord,'^NACA *[0-9]{4,5}$')),  % NACA string supplied
      fprintf(fid,'naca %s\n',coord(5:end));
    else  % filename supplied
      fprintf(fid,'load %s\n',file_coord);
    end;
  else % Coordinates supplied, use the default filename
    fprintf(fid,'load %s\n',file_coord);
  end;
  % Extra Xfoil commands
  for ii = 1:length(varargin),
    txt = varargin{ii};
    txt = regexprep(txt,'[ \\\/]+','\n');
    fprintf(fid,'%s\n\n',txt);
  end;
```

```
  fprintf(fid,'\n\noper\n');
  % set Reynolds and Mach
  fprintf(fid,'re %g\n',Re);
  fprintf(fid,'mach %g\n',Mach);

  % Switch to viscous mode
  if (Re>0)
    fprintf(fid,'visc\n');
  end;

  % Polar accumulation
  fprintf(fid,'pacc\n\n\n');
  % Xfoil alpha calculations
  [file_dump, file_cpwr] = deal(cell(1,Nalpha)); % Preallocate cell arrays

  for ii = 1:Nalpha
    % Individual output filenames
    file_dump{ii} = sprintf('%s_a%06.3f_dump.dat',fname,alpha(ii));
    file_cpwr{ii} = sprintf('%s_a%06.3f_cpwr.dat',fname,alpha(ii));
    % Commands
    fprintf(fid,'alfa %g\n',alpha(ii));
    fprintf(fid,'dump %s\n',file_dump{ii});
    fprintf(fid,'cpwr %s\n',file_cpwr{ii});
  end;
  % Polar output filename
  file_pwrt = sprintf('%s_pwrt.dat',fname);
  fprintf(fid,'pwrt\n%s\n',file_pwrt);
  fprintf(fid,'plis\n');
  fprintf(fid,'\nquit\n');
  fclose(fid);

  % execute xfoil
  cmd = sprintf('cd %s && xfoil.exe < xfoil.inp > xfoil.out',wd);
  [status,result] = system(cmd);
  if (status~=0),
    disp(result);
    error([mfilename ':system'],'Xfoil execution failed! %s',cmd);
  end;

  % Read dump file
  %    #     s        x        y      Ue/Vinf    Dstar     Theta      Cf       H
  jj = 0;
  ind = 1;
% Note that
foil.alpha = zeros(1,Nalpha); % Preallocate alphas
% Find the number of panels with an inital run
only = nargout; % Number of outputs checked. If only one left hand operator then only do polar

if only >1 % Only do the foil calculations if more than one left hand operator is specificed
```

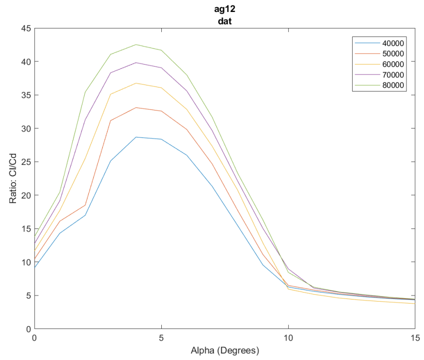
```
  for ii = 1:Nalpha
    jj = jj + 1;

    fid = fopen([wd filesep file_dump{ii}],'r');
    if (fid<=0),
      error([mfilename ':io'],'Unable to read xfoil output file %s', file_dump{ii});
    else
      D = textscan(fid,'%f%f%f%f%f%f%f%f','Delimiter',' ','MultipleDelimsAsOne',true,'CollectOutput',1,'HeaderLines',1);
      fclose(fid);
      delete([wd filesep file_dump{ii}]);

      if ii == 1 % Use first run to determine number of panels (so that NACA airfoils work without vector input)
         Npanel = length(D{1}); % Number of airfoil panels pulled from the first angle tested
         % Preallocate Outputs
         [foil.s, foil.x, foil.y, foil.UeVinf, foil.Dstar, foil.Theta, foil.Cf, foil.H] = deal(zeros(Npanel,Nalpha));
      end

      % store data
      if ((jj>1) && (size(D{1},1)~=length(foil(ind).x)) && sum(abs(foil(ind).x(:,1)-size(D{1},1)))>1e-6 ),
        ind = ind + 1;
        jj = 1;
      end;
      foil.s(:,jj) = D{1}(:,1);
      foil.x(:,jj) = D{1}(:,2);
      foil.y(:,jj) = D{1}(:,3);
      foil.UeVinf(:,jj) = D{1}(:,4);
      foil.Dstar(:,jj) = D{1}(:,5);
      foil.Theta(:,jj) = D{1}(:,6);
      foil.Cf(:,jj) = D{1}(:,7);
      foil.H (:,jj)= D{1}(:,8);
    end;

    foil.alpha(1,jj) = alpha(jj);

    % Read cp file
    fid = fopen([wd filesep file_cpwr{ii}],'r');
    if (fid<=0),
      error([mfilename ':io'],'Unable to read xfoil output file %s', file_cpwr{ii});
    else
      C = textscan(fid, '%10f%9f%f', 'Delimiter', '', 'WhiteSpace', '', 'HeaderLines' ,3, 'ReturnOnError', false);
      fclose(fid);
      delete([wd filesep file_cpwr{ii}]);
      % store data
      if ii == 1 % Use first run to determine number of panels (so that NACA airfoils work without vector input)
```

```
         NCp = length(C{1}); % Number of points Cp is listed for pulled from the first angle tested
         % Preallocate Outputs
         [foil.xcp, foil.cp] = deal(zeros(NCp,Nalpha));
         foil.xcp = C{1}(:,1);
      end
      foil.cp(:,jj) = C{3}(:,1);
    end;
  end;
end

if only <= 1% clear files for default run
  for ii=1:Nalpha % Clear out the xfoil dump files not used
      delete([wd filesep file_dump{ii}]);
      delete([wd filesep file_cpwr{ii}]);
  end
end

  % Read polar file
  %
  %       XFOIL         Version 6.96
  %
  % Calculated polar for: NACA 0012
  %
  % 1 1 Reynolds number fixed          Mach number fixed
  %
  % xtrf =   1.000 (top)        1.000 (bottom)
  % Mach =   0.000     Re =     1.000 e 6     Ncrit =  12.000
  %
  %   alpha    CL        CD       CDp       CM     Top_Xtr  Bot_Xtr
  %  ------ -------- --------- --------- -------- -------- --------
  fid = fopen([wd filesep file_pwrt],'r');
  if (fid<=0),
    error([mfilename ':io'],'Unable to read xfoil polar file %s',file_pwrt);
  else
    % Header
    % Calculated polar for: NACA 0012
    P = textscan(fid,' Calculated polar for: %[^\n]','Delimiter',' ','MultipleDelimsAsOne',true,'HeaderLines',3);
    pol.name = strtrim(P{1}{1});
    % xtrf =   1.000 (top)        1.000 (bottom)
    P = textscan(fid, '%*s%*s%f%*s%f%s%s%s%s%s%s', 1, 'Delimiter', ' ', 'MultipleDelimsAsOne', true, 'HeaderLines', 2, 'ReturnOnError', false);
    pol.xtrf_top = P{1}(1);
    pol.xtrf_bot = P{2}(1);
    % Mach =   0.000     Re =     1.000 e 6     Ncrit =  12.000
    P = textscan(fid, '%*s%*s%f%*s%*s%f%*s%f%*s%*s%f', 1, 'Delimiter', ' ', 'MultipleDelimsAsOne', true, 'HeaderLines', 0, 'ReturnOnError', false);
    pol.Re = P{2}(1) * 10^P{3}(1);
    pol.Ncrit = P{4}(1);
```

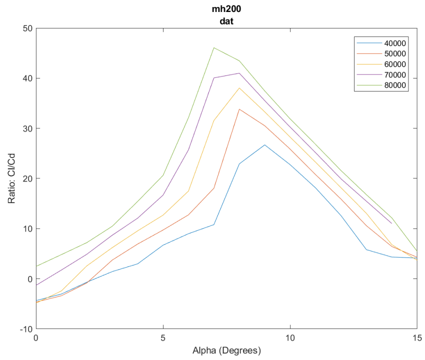
```

    % data
    P = textscan(fid, '%f%f%f%f%f%f%f%*s%*s%*s%*s', 'Delimiter',  ' ', '
       MultipleDelimsAsOne', true, 'HeaderLines' , 4, 'ReturnOnError',
       false);
    fclose(fid);
    delete([wd filesep file_pwrt]);
    % store data
    pol.alpha = P{1}(:,1);
    pol.CL  = P{2}(:,1);
    pol.CD  = P{3}(:,1);
    pol.CDp = P{4}(:,1);
    pol.Cm  = P{5}(:,1);
    pol.Top_xtr = P{6}(:,1);
    pol.Bot_Xtr = P{7}(:,1);
  end
  if length(pol.alpha) ~= Nalpha % Check if xfoil failed to converge
     warning('One or more alpha values failed to converge. Last
        converged was alpha = %f. Rerun with ''oper iter ##'' command.\n'
        , pol.alpha(end))
  end

end
```

5.2 Complete Enumeration

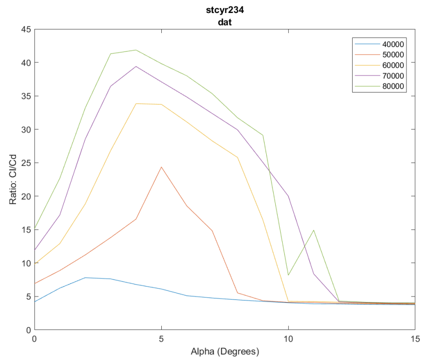
Over 1000 plots were generated. Below is a very small subset to illustrate the concept.



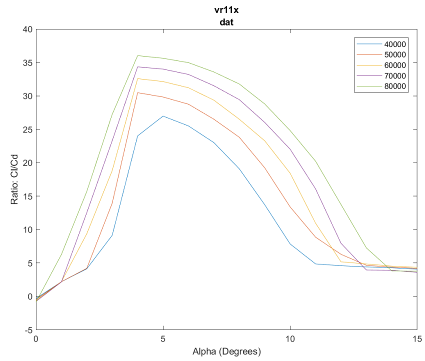
(a) ag12



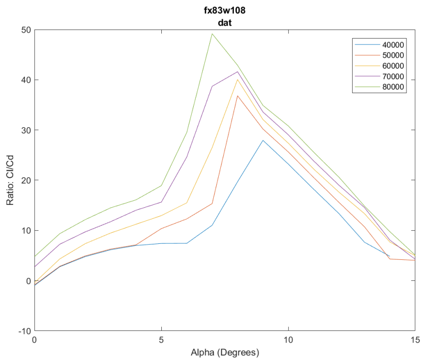
(b) mh200



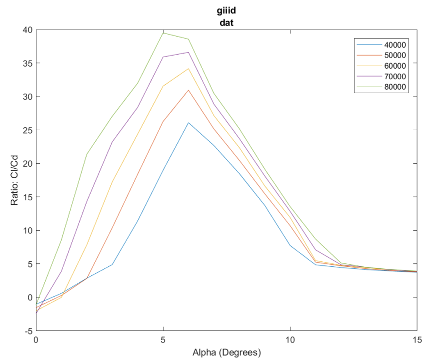
(c) stcyr234



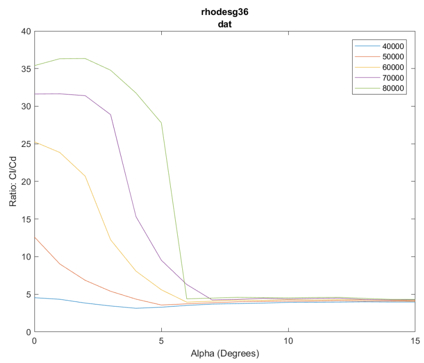
(d) vr11x



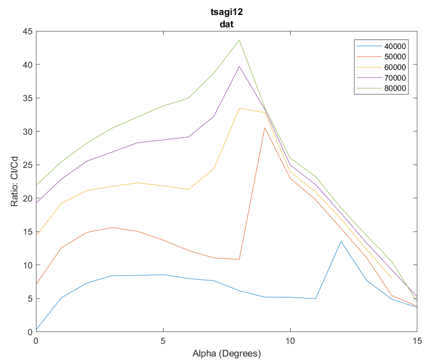
(e) fx83w108



(f) giiid



(g) rhodesg36



(h) tsagi12

Figure 5: A Small Subset of the Plots generated during Complete Enumeration

5.3 Stage One Selection

Stage One Selection	
ag03.dat	giiic.dat
ag11.dat	hh02.dat
ag16.dat	ht22.dat
ag455ct02r.dat	isa962.dat
ag45ct02r.dat	lg10sc.dat
ag46c03.dat	mh32.dat
ah7476.dat	mh44.dat
ames01.dat	naca1410.dat
arad10.dat	naca23012.dat
be50.dat	naca23015.dat
boe103.dat	ncambre.dat
defcnd3.dat	npl9615.dat
df101.dat	npl9660.dat
e212.dat	rae100.dat
e216.dat	s8025.dat
e222.dat	s9000.dat
fg1.dat	sd7003.dat
fx69h098.dat	v13009.dat
fx76120.dat	

Figure 6: 37 Selected Aerofoils

5.4 Blade and Section Iteration

Name	Cp	# of Blades	# of Sections	Cp	# of Blades	# of Sections	Cp	# of Blades	# of Sections
ag16.dat	0.382113123	4	15	0.398471	6	15	0.415932714	6	15
ag11.dat	0.381574269	4	15	0.398616	6	15	0.416108346	6	15
ag03.dat	0.381436383	4	15	0.39918	6	15	0.416794554	6	15
ag45ct02r.d	0.379999074	4	15	0.396907	6	15	0.414030026	6	15
ag455ct02r.	0.379943415	4	15	0.396965	6	15	0.414100994	6	15
sd7003.dat	0.379062469	4	15	0.397078	6	15	0.414238553	6	15
s9000.dat	0.383338597	4	15	0.406572	6	15	0.425757362	6	15
arad10.dat	0.383072989	4	15	0.391169	6	15	0.407033612	6	15
mh44.dat	0.379380573	4	15	0.400186	6	15	0.418016305	6	15
npl9615.da	0.378773944	4	15	0.402837	6	15	0.421233543	6	15
npl9660.da	0.378076386	4	15	0.399868	6	15	0.417630082	6	15
be50.dat	0.385603462	4	15	0.394314	6	15	0.410871864	6	15
isa962.dat	0.384534587	4	15	0.398732	6	15	0.416249713	6	15
mh32.dat	0.383587864	4	15	0.402648	6	15	0.421004085	6	15
ag46c03.da	0.380106966	4	15	0.404296	6	15	0.423000952	6	15
defcnd3.da	0.377468631	4	15	0.401544	6	15	0.419664783	6	15
fx69h098.d	0.3774145	4	15	0.402117	6	15	0.420360276	6	15
ncambre.d	0.376452056	4	15	0.394257	6	15	0.410802726	6	15
v13009.dat	0.375373694	4	15	0.390909	6	15	0.406716021	6	15
lg10sc.dat	0.37464165	4	15	0.396483	6	15	0.413513605	6	15
ames01.da	0.374466594	4	15	0.397494	6	15	0.414743697	6	15
fx76120.dat	0.374218478	4	15	0.390368	6	15	0.406054985	6	15
ht22.dat	0.373702189	4	15	0.401717	6	15	0.419875215	6	15
s8025.dat	0.371289808	4	15	0.391352	6	15	0.407257652	6	15
ah7476.dat	0.389168471	4	15	0.400725	6	15	0.41867134	6	15
e216.dat	0.38699533	4	15	0.396317	6	15	0.413312228	6	15
e212.dat	0.38542285	4	15	0.397365	6	15	0.414586757	6	15
fg1.dat	0.384916201	4	15	0.390573	6	15	0.406305696	6	15
e222.dat	0.384369063	4	15	0.38958	6	15	0.405091495	6	15
boe103.dat	0.382769489	4	15	0.393249	6	15	0.409573023	6	15
df101.dat	0.381685264	4	15	0.395682	6	15	0.412538283	6	15
hh02.dat	0.380503232	4	15	0.394951	6	15	0.411647935	6	15
naca1410.d	0.380380107	4	15	0.390386	6	15	0.40607767	6	15
giiic.dat	0.379538456	4	15	0.38784	6	15	0.402963004	6	15
naca23012.	0.373897979	4	15	0.400464	6	15	0.418354244	6	15
rae100.dat	0.373719903	4	15	0.395984	6	15	0.412906426	6	15
naca23015.	0.372950016	4	15	0.392119	6	15	0.408194236	6	15

(a) One set of changing the Number of Blades with a Fixed Number of Sections

Name	Cp	# of Blades	# of Sections	Cp	# of Blades	# of Sections	Cp	# of Blades	# of Sections
ag03.dat	0.402759219	6	14	0.403931	6	15	0.40490877	6	16
ag11.dat	0.402905735	6	14	0.404078	6	15	0.405055981	6	16
ag16.dat	0.403478333	6	14	0.404652	6	15	0.405631294	6	16
ag455ct02r.	0.401172931	6	14	0.40234	6	15	0.403314963	6	16
ag45ct02r.d	0.401232065	6	14	0.4024	6	15	0.403374377	6	16
ag46c03.da	0.401346694	6	14	0.402515	6	15	0.403489549	6	16
ah7476.dat	0.410978179	6	14	0.412171	6	15	0.4131667	6	16
ames01.da	0.395355723	6	14	0.396508	6	15	0.39747018	6	16
arad10.dat	0.404498381	6	14	0.405675	6	15	0.406656177	6	16
be50.dat	0.407187951	6	14	0.408371	6	15	0.409358499	6	16
boe103.dat	0.404175842	6	14	0.405351	6	15	0.40633211	6	16
defcnd3.da	0.398543958	6	14	0.399705	6	15	0.400673527	6	16
df101.dat	0.403023678	6	14	0.404196	6	15	0.405174483	6	16
e212.dat	0.406995962	6	14	0.408179	6	15	0.4091656	6	16
e216.dat	0.4086676	6	14	0.409855	6	15	0.410845164	6	16
e222.dat	0.405875862	6	14	0.407056	6	15	0.408040189	6	16
fg1.dat	0.406457416	6	14	0.407639	6	15	0.408624501	6	16
fx69h098.d	0.398486461	6	14	0.399647	6	15	0.400615758	6	16
fx76120.dat	0.39509226	6	14	0.396244	6	15	0.397205468	6	16
giiic.dat	0.400742698	6	14	0.401909	6	15	0.402882691	6	16
hh02.dat	0.401767713	6	14	0.402937	6	15	0.403912565	6	16
ht22.dat	0.394544056	6	14	0.395694	6	15	0.396654665	6	16
isa962.dat	0.406051795	6	14	0.407232	6	15	0.408216957	6	16
lg10sc.dat	0.395541611	6	14	0.396695	6	15	0.397656949	6	16
mh32.dat	0.405045575	6	14	0.406223	6	15	0.407205966	6	16
mh44.dat	0.400574967	6	14	0.401741	6	15	0.402714164	6	16
naca1410.d	0.401636896	6	14	0.402806	6	15	0.403781127	6	16
naca23012.	0.394751946	6	14	0.395903	6	15	0.396863541	6	16
naca23015.	0.393745436	6	14	0.394894	6	15	0.39585226	6	16
ncambre.d	0.397464226	6	14	0.398622	6	15	0.399588678	6	16
npl9615.da	0.399930519	6	14	0.401095	6	15	0.402066662	6	16
npl9660.da	0.39918952	6	14	0.400352	6	15	0.40132215	6	16
rae100.dat	0.394562865	6	14	0.395713	6	15	0.396673563	6	16
s8025.dat	0.39198292	6	14	0.393127	6	15	0.394081388	6	16
s9000.dat	0.404780657	6	14	0.405958	6	15	0.406939792	6	16
sd7003.dat	0.400237026	6	14	0.401402	6	15	0.402374622	6	16
v13009.dat	0.396318987	6	14	0.397474	6	15	0.39843801	6	16

(b) One set of changing the Number of Sections with a Fixed Number of Blades

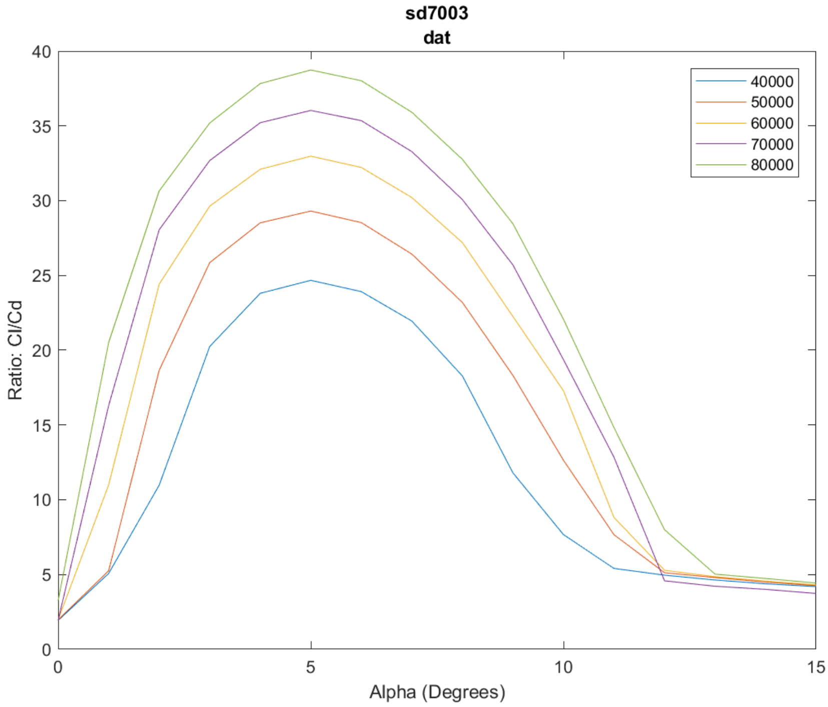
Figure 7: One third of all iterations

5.5 Manufacturing Considerations

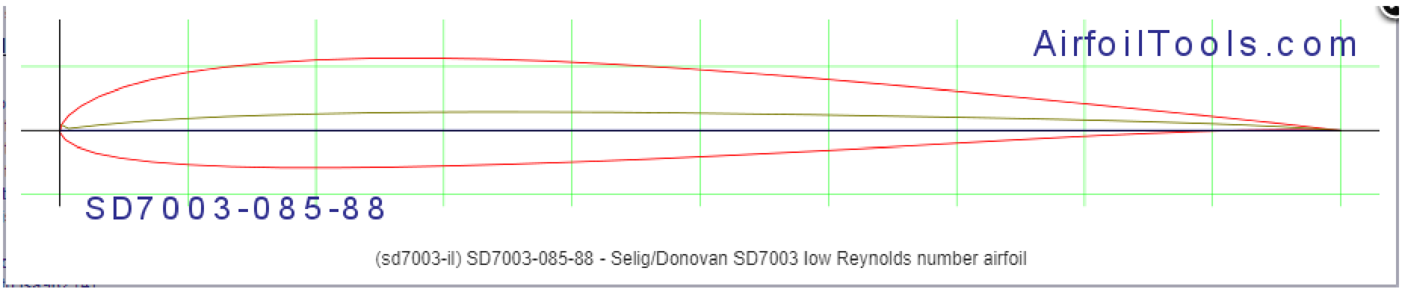
Name	Cp	Number of Blades	Number of Sections	Plot	Thickness	Cambre	Thickness2
ah7476.dat	0.406572	6	15	0	5.9		5.928195341
e216.dat	0.404296	6	15	0	10.4		10.54111723
be50.dat	0.402837	6	15	1	7.3		10.91979226
e212.dat	0.402648	6	15	0	10.6		14.35801718
fg1.dat	0.402117	6	15	0	8.2		13.45957095
isa962.dat	0.401717	6	15	1	9.6		12.29594233
e222.dat	0.401544	6	15	0	10.2		14.83437748
mh32.dat	0.400725	6	15	1	8.7		13.99001317
s9000.dat	0.400464	6	15	2	9		14.84379605
arad10.dat	0.400186	6	15	2	10		15.76392454
boe103.dat	0.399868	6	15	0	12.7		16.32408669
ag16.dat	0.39918	6	15	3	7.1		14.25860086
df101.dat	0.398732	6	15	0	11		17.23135917
ag11.dat	0.398616	6	15	3	5.8		12.66495861
ag03.dat	0.398471	6	15	3	6.2		12.95046043
hh02.dat	0.397494	6	15	0	9.6		16.61410868
naca1410.dat	0.397365	6	15	0	10		20.1849255
ag46c03.dat	0.397078	6	15	1	6		13.09624328
ag45ct02r.dat	0.396965	6	15	3	6.9		12.90264785
ag455ct02r.dat	0.396907	6	15	3	6.5		13.01754555
giiic.dat	0.396483	6	15	0	9.9		20.63785387
mh44.dat	0.396317	6	15	2	9.6		16.70661847
sd7003.dat	0.395984	6	15	3	8.5	1.2	17.15351248
npl9615.dat	0.395682	6	15	2	11.3		22.23923002
npl9660.dat	0.394951	6	15	2	11.3		23.18947427
defcnd3.dat	0.394314	6	15	1	11.5		15.14803011
fx69h098.dat	0.394257	6	15	1	9.9		20.73202467
ncambre.dat	0.393249	6	15	1	11.5		22.55469383
v13009.dat	0.392119	6	15	1	9		21.11584075
lg10sc.dat	0.391352	6	15	1	0		0
ames01.dat	0.391169	6	15	1	10.3		22.18069083
fx76120.dat	0.390909	6	15	1	12.1	0	28.16289896
naca23012.dat	0.390573	6	15	0	12		25.46063535
rae100.dat	0.390386	6	15	0	10		24.93279763
ht22.dat	0.390368	6	15	1	5.5		15.13183209
naca23015.dat	0.38958	6	15	0	15		31.19409281
s8025.dat	0.38784	6	15	1	8		19.64748684

Figure 8: Evaluation of Manufacturing Considerations

5.6 Final Selection



(a) sd7003 plot of CL/CD ratios with margins for changing alpha and Reynold's numbers



(b) sd7003 Profile Design

Figure 9: sd7003 Plot and Shape