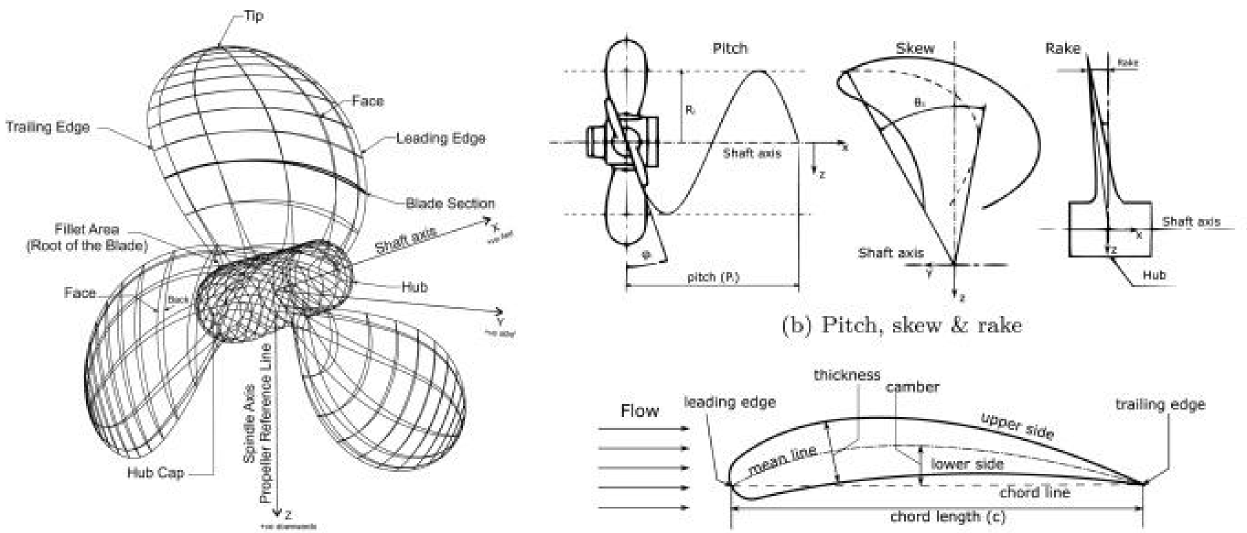
Marine propellers optimization with the use of software OpenProp



(a) Marine propeller model

(b) Pitch, skew & rake

(c) Hydrofoil geometry

Proposed by

Chris Gkiokas

As part of the final project
MIT 2.29 – Numerical Fluid Mechanics

Project Description and Background

Every design of a propeller for a specific application includes the problem of optimization. This means that the designer will have to choose the most appropriate propeller design from a wide variety of options. The optimization can be conducted based on different properties and objectives. In this project I studied the optimization of a propeller design using the software OpenProp. OpenProp is a propeller lifting line code which can be used to solve the propeller and wind turbine design optimization problem and also analyze these devices at operating points off-design conditions.

The program is based on Lifting Line Theory (LLT), a fundamental theory for the calculation of the lift force generated by a foil. The above calculation is difficult using analytical methods. However, the first step to solving this problem would be to realize that we could separate the foil into cross-sectional segments and exploit the individual lift force produced by each cross-section in order to compute the total lift. But opposed to what might be the first impression, the total lift will not be equal to the sum of the individual lifts generated by each segment. On the contrary, the lift on each cross-section has an impact on the lift of the two adjacent cross-sections.

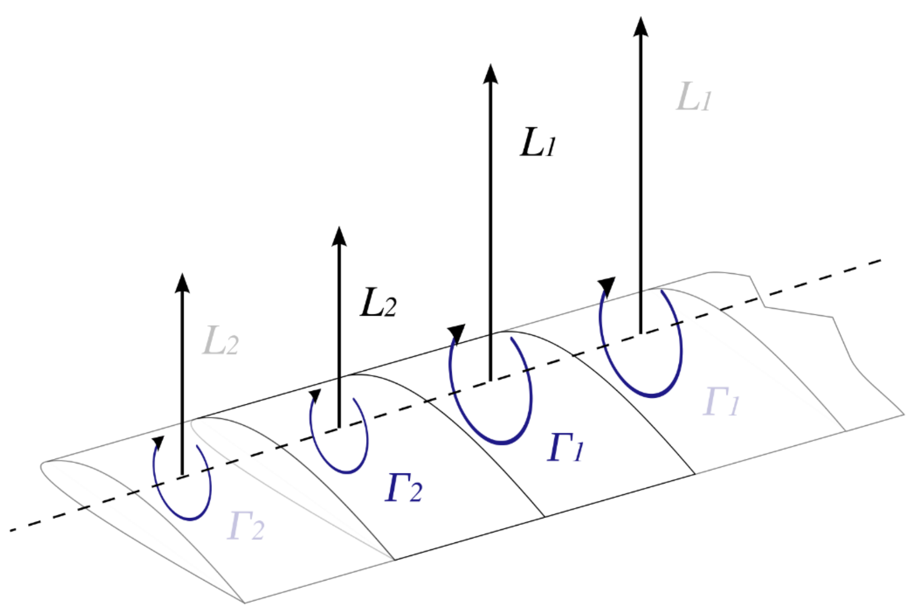


Figure 1: Lift distribution over a foil (www.ariadacapo.net)

This is where Lifting Line Theory is introduced in order to fill this gap and take into account the influence between the neighboring segments. Lifting Line Theory uses Kutta – Joukowski theorem {$L(y)=\rho V\Gamma(y)$} to transfer the problem, from finding the unknown lift to finding the unknown circulation on each cross-section. Thus, a change in the lift along the span of the foil is equivalent through the above formula, to a change in circulation. The concept of circulation implies the existence of vortices which create the circulation. So, the change in lift spanwise is translated into shedding vortices spanwise and downstream.

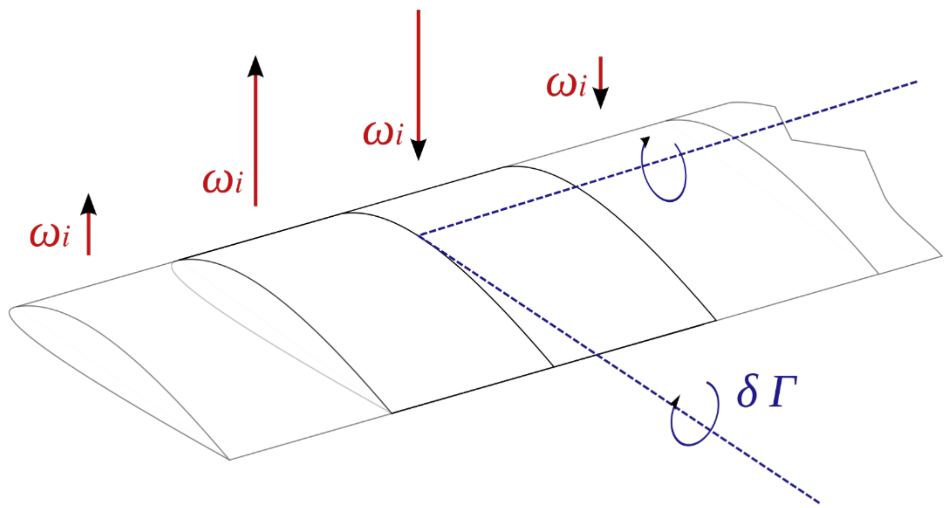


Figure 2: Shed vortex and induced velocities on the adjacent segments (www.ariadacapo.net)

The influences on the neighboring sections from each vortex are the basic ingredient of the LLT. If we know the change in lift distribution throughout a section then one can predict the induced velocities on the adjacent sections, using the velocity distribution inside the vortex combined with the change in effective angle of attack.

In the propeller LLT every blade of the propeller is represented by a lifting line, as shown in Figure 3. The next for the approach will be to divide the spanwise distance of the blade into M panels and consider constant circulation in the interior of each panel. The value of circulation Γ in each panel will be equal to the value of the continuous circulation at the control point that we select. Usually, the control points are selected to be at the center of the length of the panel. The selection of the control points is important since the induced velocities by the vortices will be calculated at the control points.

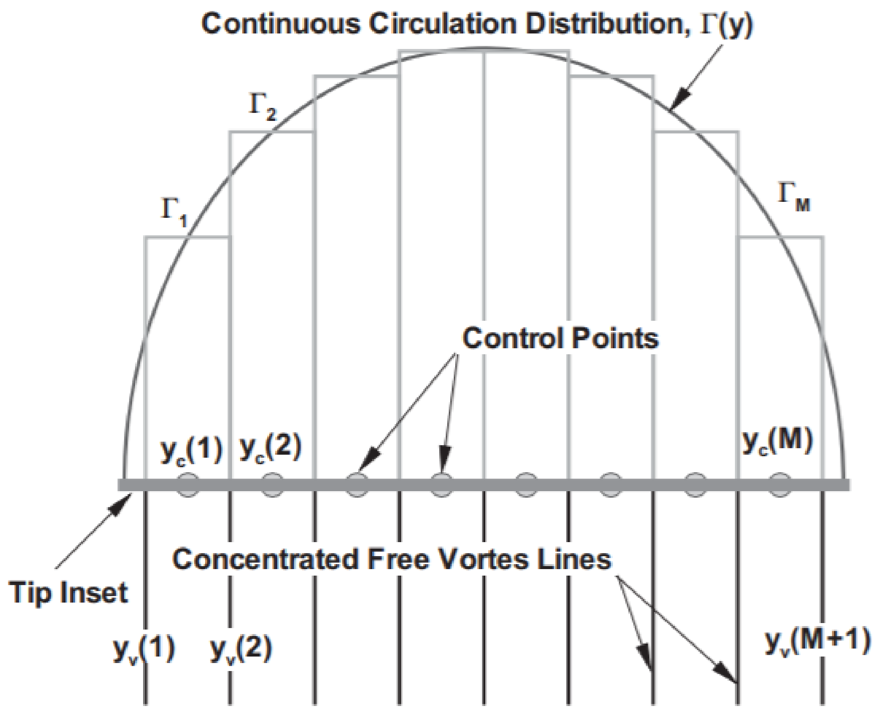


Figure 3: Lifting line with selected spacing and circulation distribution
(Kerwin and Handler (2010), PNA)

In Figure 4 we can see a velocity – force diagram for the blade section, as it can be viewed from the tip of the blade. In this diagram we can see the following:

$\mathbf{e_a}$: axis of the axial direction
$\mathbf{e_t}$**:** axis of the tangential direction
$\omega\mathbf{e_a}$: angular velocity of the propeller
V_a: axial inflow velocity
V_t: tangential inflow velocity
u_a^*: induced axial velocity
u_t^*: induced tangential velocity

$\boldsymbol{V^*} = \sqrt{(\boldsymbol{Va}+\boldsymbol{ua^*})^2+(\boldsymbol{\omega r}+\boldsymbol{Vt}+\boldsymbol{ut^*})^2}$ total resultant inflow velocity

$\boldsymbol{\beta_i} = \boldsymbol{arctan}(\frac{\boldsymbol{V_a+u_a^*}}{\boldsymbol{wr+V_t+u_t^*}})$ pitch angle

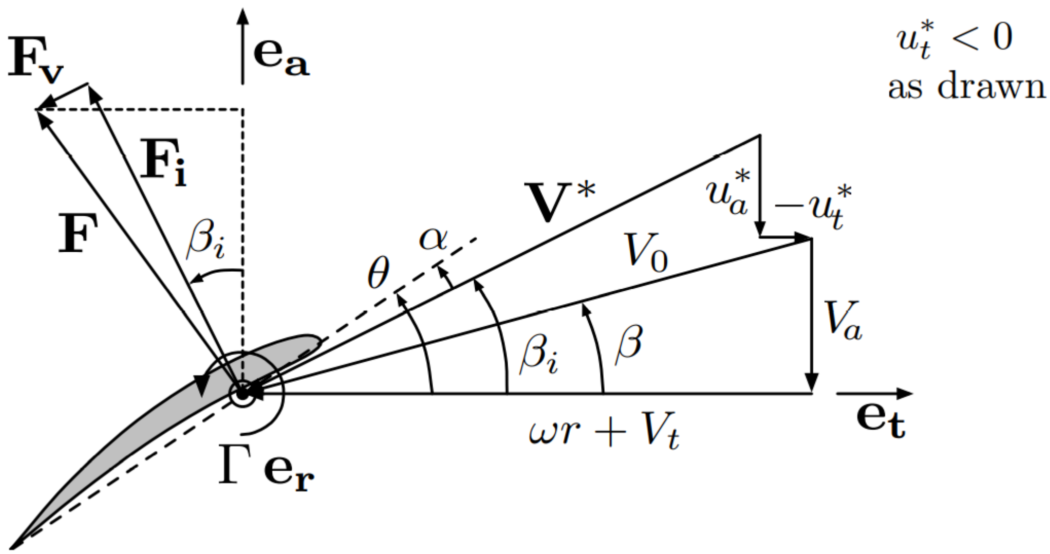


Figure 4: Velocity – force diagram for a blade section
(Kerwin and Handler (2010), PNA)

OpenProp and Optimization process

OpenProp is an open-source code suite that is written in MATLAB. It can be used for the design, analysis, construction, and optimization of a propeller. It can be used to analyze both marine propellers and wind turbines with horizontal axis. The basic assumptions of the program are that:

a) the inflow is steady and can vary radially but not circumferentially
b) the moderately-loaded propeller lifting line theory applies

The inputs for the program that the user must insert are:

- propeller and hub diameter
- number of blades
- ship speed and rotational speed of the propeller
- inflow profile (uniform or radially varying)
- duct (if present, with a thrust ratio τ)
- assumed blade 2D section profile – optionally a viscous 2D drag coefficient
- discretization in panels along the blade radius
- required thrust

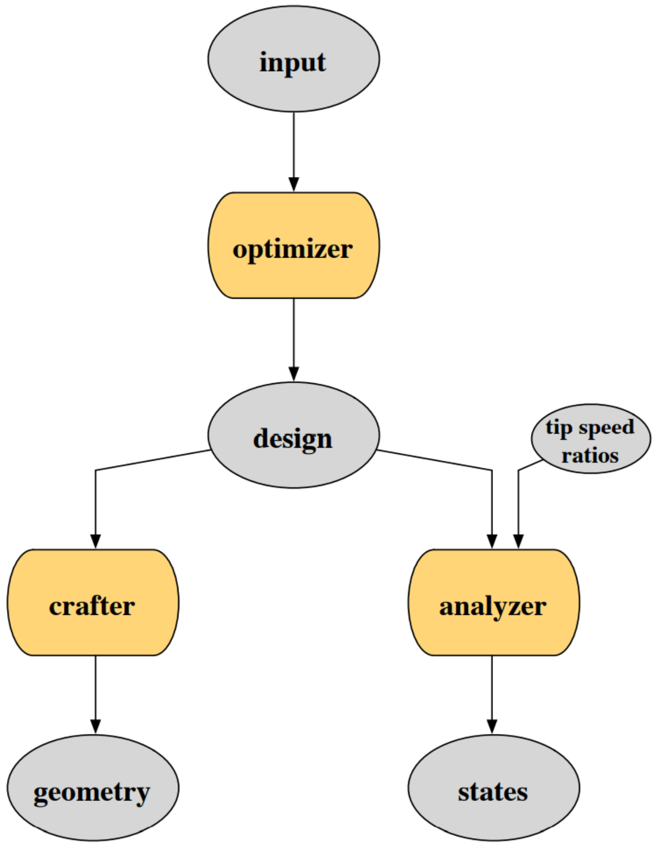


Figure 5: OpenProp information flow chart
(OpenProp v2.4 Theory Document)

In Figure 5 we can see the modular form of the flow chart of the program, which uses data structures to store the input, design, geometry and operating states. The basic modules are the optimizer, the crafter, and the analyzer. The optimizer is the module that determines the optimum propeller design. The analyzer determines the off-design operating states, while the crafter is responsible for the creation of the 3D geometry of the propeller and the export of the files for the visual depiction.

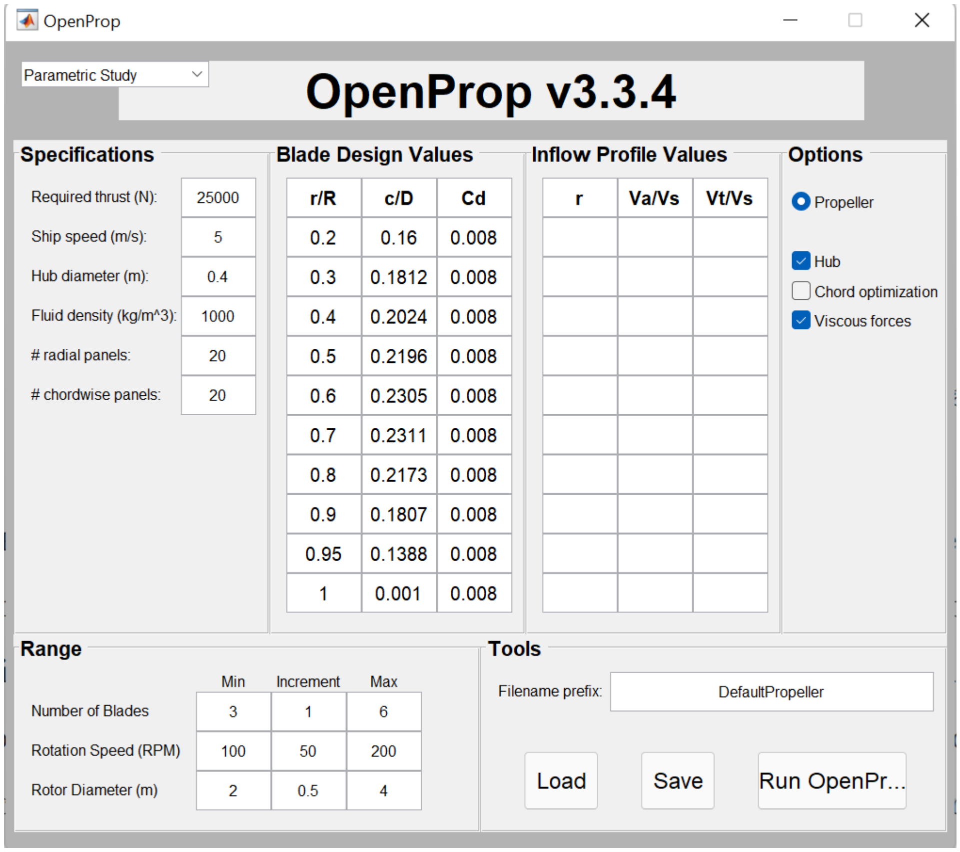


Figure 6: Parametric study screen

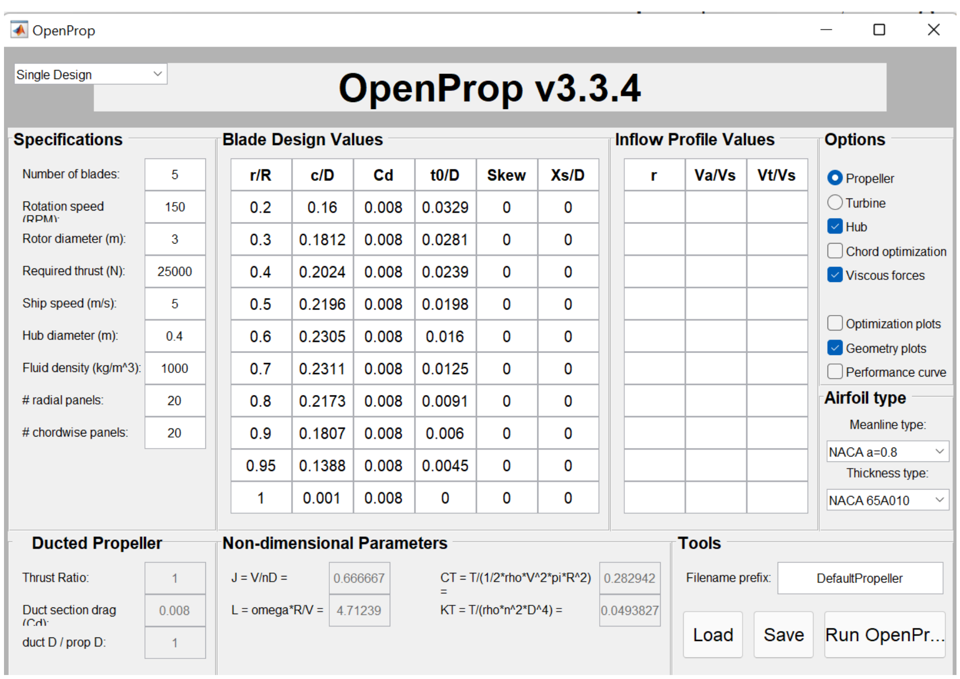


Figure 7: Single design screen

The program has two main options: parametric study and single propeller design. Screenshots of the two operations are shown in Figures 6 and 7. A parametric study is usually conducted in the beginning of the process as an assessment for the most suitable design for the specific application. In the parametric study the user can input the basic specifications of the propeller and then specify the range of the varying specifications, which are in particular the number of blades, the rotation speed and the propeller diameter. After completing the parametric study and selecting the propeller, a single design study must be conducted to analyze and optimize the design. After running the single design study, the program exports all the diagrams that are necessary for the evaluation of the propeller design, such as performance curves, circulation and induced velocities plots and efficiency plots.

The objective of the program is to find the propeller that maximizes the efficiency of the propeller. The efficiency of the propeller is defined as the ratio of the thrust power for a propeller operating in the wake of a ship over the delivered power to the propeller:

$$\eta = \frac{P_T}{P_D}$$

The optimization problem will eventually reduce to finding the optimum circulation distribution Γ. The program will find the set of M circulations of the vortex lattice panels which produce the least torque, using the following formulae for torque and thrust respectively:

$$Q = \rho Z \sum_{m=1}^{M} \left\{ [V_a + u_a^*]\Gamma + \tfrac{1}{2} V^* C_D c [\omega r_c + V_t + u_t^*] \right\} r_c \triangle r_v$$

$$T = \rho Z \sum_{m=1}^{M} \left\{ [\omega r_c + V_t + u_t^*]\Gamma - \tfrac{1}{2} V^* C_D c [V_a + u_a^*] \right\} \triangle r_v$$
$$- \text{Hflag} \cdot \tfrac{\rho Z^2}{16\pi} \left[\ln\left(\tfrac{r_h}{r_o}\right) + 3 \right] [\Gamma(1)]^2 = T_s$$

where Hflag is set to 1 for modelling a hub or 0 for no hub and T_s is the specified thrust.

For the solution of the optimization, the code employs the method of Lagrange multiplier. An auxiliary function, $H = Q + \lambda_1 (T - T_s)$, is introduced with λ_1 the unknown Lagrange multiplier which constrains the thrust. It is obvious that if T= T_s then a minimum value for H coincides with a minimum value for Q (torque). In order to find the minimum value, we can follow the known process of setting the derivatives with respect to the unknowns Γ, λ_1 equal to zero:

$$\frac{\partial H}{\partial \Gamma(\mathrm{i})} = 0$$, for i=1…M

$$\frac{\partial H}{\partial \lambda_1} = 0$$

This results to a non-linear system of M+1 equations which is solved iteratively up to convergence of all parameters using Newton's method. For each solution iteration the flow parameters (u_a*, u_t*, V*, $\frac{\partial V*}{\partial \Gamma(i)}$, λ_1) are kept constant to linearize the equations for torque and thrust. After this step the linearized unknowns $\breve{\Gamma}$, $\breve{\lambda_1}$ are taken, forming the following system:

$$\begin{aligned}
\frac{\partial H}{\partial \Gamma(i)} = \quad & \rho Z \sum_{m=1}^{M} \breve{\Gamma}(m) \cdot [\bar{u}_a^*(m,i) r_c(m) \triangle r_v(m) + \bar{u}_a^*(i,m) r_c(i) \triangle r_v(i)] \\
& + \rho Z V_a(i) r_c(i) \triangle r_v(i) \\
& + \rho Z \sum_{m=1}^{M} \tfrac{1}{2} C_D \tfrac{\partial V^*(m)}{\partial \Gamma(i)} c(m) [\omega r_c(m) + V_t(m) + u_t^*(m)] r_c(m) \triangle r_v(m) \\
& + \rho Z \sum_{m=1}^{M} \tfrac{1}{2} C_D V^*(m) c(m) [\bar{u}_t^*(m,i)] r_c(m) \triangle r_v(m) \\
& + \rho Z \lambda_1 \sum_{m=1}^{M} \breve{\Gamma}(m) \cdot [\bar{u}_t^*(m,i) \triangle r_v(m) + \bar{u}_t^*(i,m) \triangle r_v(i)] \\
& + \rho Z \breve{\lambda}_1 [\omega r_c(i) + V_t(i)] \triangle r_v(i) \\
& - \rho Z \breve{\lambda}_1 \sum_{m=1}^{M} \tfrac{1}{2} C_D \tfrac{\partial V^*(m)}{\partial \Gamma(i)} c(m) \left[V_a(m) + u_a^*(m) \right] \triangle r_v \\
& - \rho Z \breve{\lambda}_1 \sum_{m=1}^{M} \tfrac{1}{2} C_D V^*(m) c(m) [\bar{u}_a^*(m,i)] \triangle r_v \\
& - \text{Hflag} \cdot \frac{\partial \Gamma(1)}{\partial \Gamma(i)} \cdot \lambda_1 \frac{\rho Z^2}{8\pi} \left[\ln\left(\frac{r_h}{r_o}\right) + 3 \right] \breve{\Gamma}(1) \\
= 0 \quad & \text{for} \quad i = 1 \dots M
\end{aligned}$$

$$\begin{aligned}
\frac{\partial H}{\partial \lambda_1} = \quad & \rho Z \sum_{m=1}^{M} \breve{\Gamma}(m) \cdot [\omega\, r_c(m) + V_t(m) + u_t^*(m)] \triangle r_v(m) \\
& - \rho Z \sum_{m=1}^{M} \tfrac{1}{2} C_D V^*(m) c(m) [V_a(m) + u_a^*(m)] \triangle r_v(m) \\
& - \text{Hflag} \cdot \frac{\rho Z^2}{16\pi} \left[\ln\left(\frac{r_h}{r_o}\right) + 3 \right] \Gamma(1) \cdot \breve{\Gamma}(1) \\
& - T_s \\
= 0 &
\end{aligned}$$

The system will next be solved for the now linear unknowns, Γ and λ_1 are updated to the linearized values ($\Gamma = \breve{\Gamma}$ and $\lambda_1 = \breve{\lambda_1}$) and the new value of Γ is used to update all the flow parameters. One important feature here is that the wake is not iteratively aligned through the process, but instead, one new guess is made for the wake flow parameters (angle β_i) and then the main iterative loop of the code continues to find the next guess for Γ:

```
for j = 1:NJS
    j

    Js        = JsALL(j);       % advance ratio == Vs/nD

    i.N    = 60*Vs/(Js*D);    % propeller speed [RPM]
    i.Js   = Js;              % advance coefficient, Js = Vs/nD = pi/L
    i.L    = pi/Js;           % tip speed ratio, L = omega*R/V


    % Perform design optimization
    pt(j).filename = pt(1).filename;
    pt(j).date     = pt(1).date;
    pt(j).i        = i;

    pt(j).d = EppsOptimizer(i);

    % Off-design analysis
    pt(j).s = AnalyzeAuto(pt(j),deltaJ,Lmax);
end
```

Each iteration of the loop includes updating Γ using the system of the derivatives equations. The critical step in this process is to update the induced velocities u_a^*, u_t^* using the equations:

$$u_a^*(m) = \sum_{i=1}^{M} \Gamma(i)\, \bar{u}_a^*(m,i) \;+\; \Gamma_d\, \bar{u}_{a,d}^*(m)$$

$$u_t^*(m) = \sum_{i=1}^{M} \Gamma(i)\, \bar{u}_t^*(m,i)$$

and then correct them by smoothing the velocities at the blade ends, the hub and the tip. This smoothing is critical because this way the system of all the equations can converge. This is due to the fact that the alignment of the wake and the vortices influence functions which are given to the next iteration are sensitive to irregularities of the induced velocities.

Results

I ran the program to study the optimization of the propeller for a specific application, in particular the design of the propeller for REMUS 6000 AUV, which is shown in Figure 8.

Figure 8: REMUS 6000 AUV

I first conducted a parametric study to find the most efficient propeller for the application, varying the number of blades from 2 to 9, finding that 2-5 blades was the most efficient range. I also varied the rotational speed from 100 to 400 rpm and the propeller diameter from 0.1 to 0.5 m. The plots that the program generated were the ones shown in Figure 9.

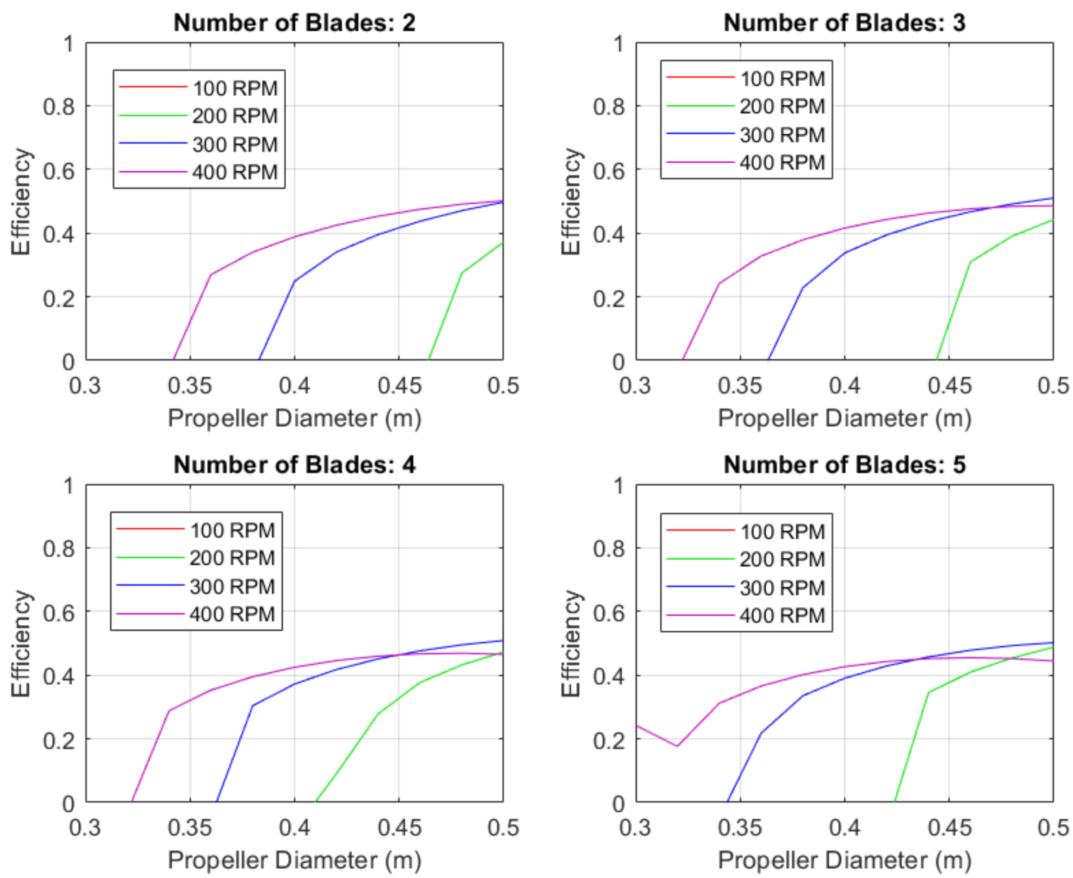


Figure 9: Plots generated after parametric study for a defined range of varying parameters

It is obvious in the above plots that the solution for efficiency of the propeller converges up to a value and then there is a rapid non-convergence which leads to zero value for efficiency, something that is not realistic.

The program also produces the following plots. Figure 10 illustrates the thrust coefficient K_T, torque coefficient K_Q and propeller efficiency η as a function of advance ratio J_s and Figure 11 shows the circulation as a function of the dimensionless radius along the blade r/R. As is evident from Figures 10 and 11, the circulation increases from the root as the radius increases, achieves its maximum value at about r/R = 0.7 and then decreases towards the propeller tips. The vehicle operates at an advance ratio of about J = 1.5 with an efficiency η= 0.51. At design conditions, the thrust coefficient assumes a value of K_T = 0.38 and a torque coefficient of K_Q = 0.066. Off-design conditions are also plotted in Figure 10.

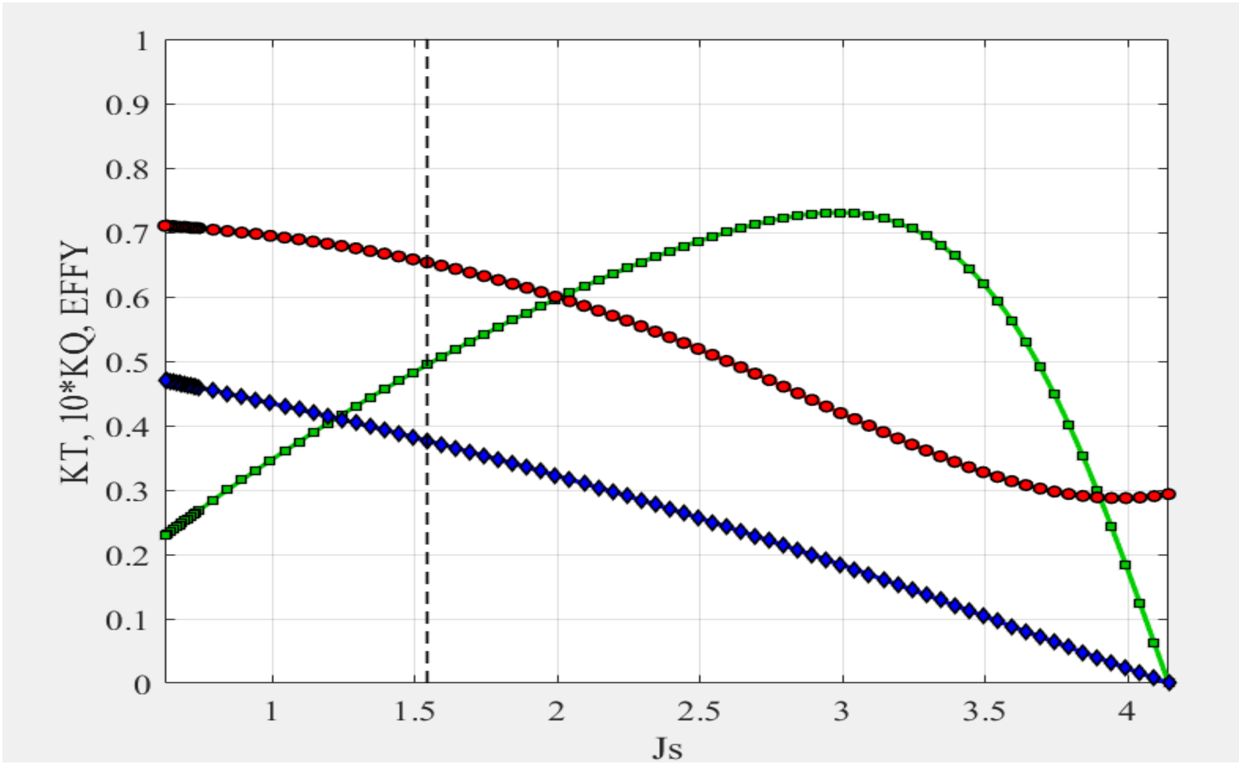


Figure 10: K_T, K_Q and propeller efficiency η as a function of advance ratio J_s

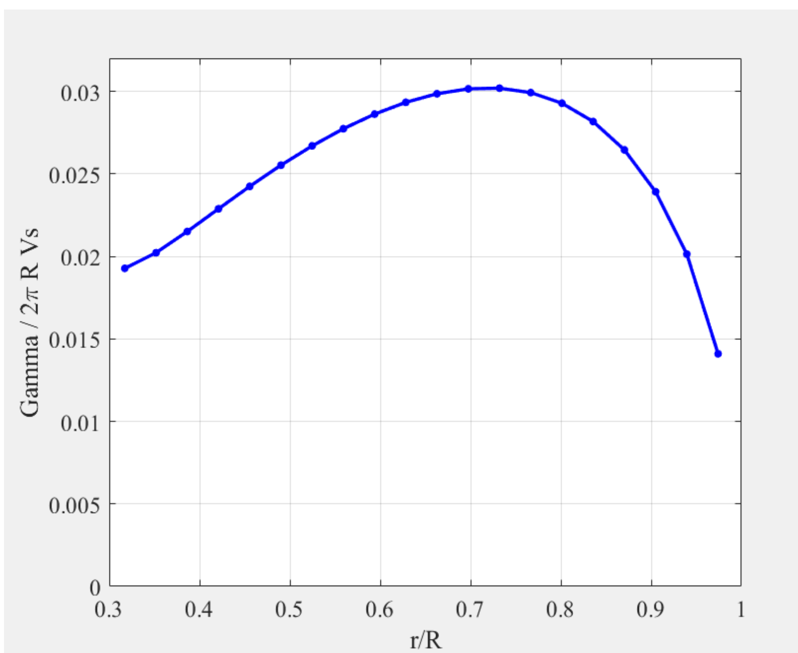


Figure 11: Normalized circulation Γ as a function of r/R

We saw above that the crafter module is responsible for the 3D geometry of the propeller and its visualization. In Figure 12 the same propeller is shown; the left propeller consists of 20x20 panels for the chord and span directions, while the right propeller consists of 5x5 panels. The difference in the shape is evident. It was observed that the program demands at least 3 panels for each direction in order to be able to generate the shape plot of the propeller.

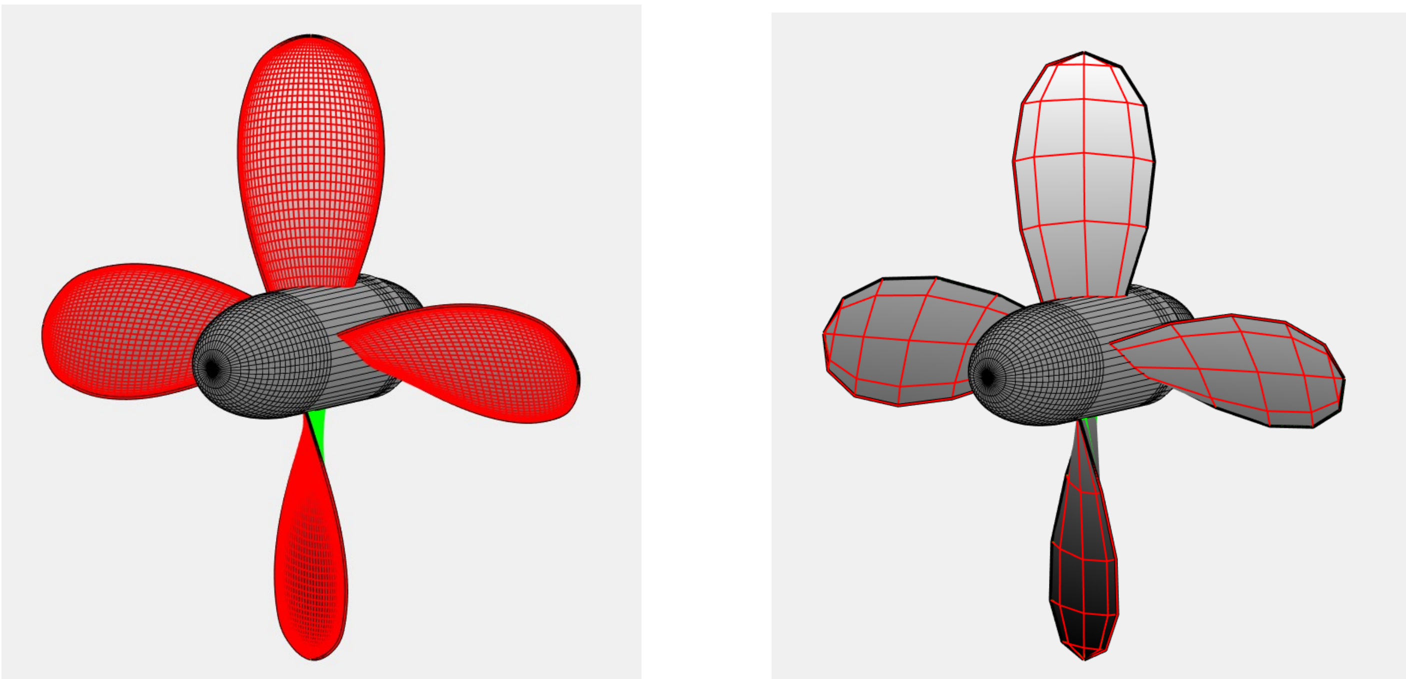


Figure 12: Same specifications propeller shape for 2 different discretizations: a) 20x20 panels on the left and b) 5x5 panels on the right

Finally, I plotted the solutions for efficiency of the propeller as a function of the number of panels to show the difference I encountered in the solutions. It is obvious that the solution converges to a specific value (approximately 0.725) as the number of panels increases. Moreover, the error in the calculation is decreased for increased number of panels. The convergence is of 2^{nd} order since the plot in Figure 13 can be simulated with a 2^{nd} order polynomial.

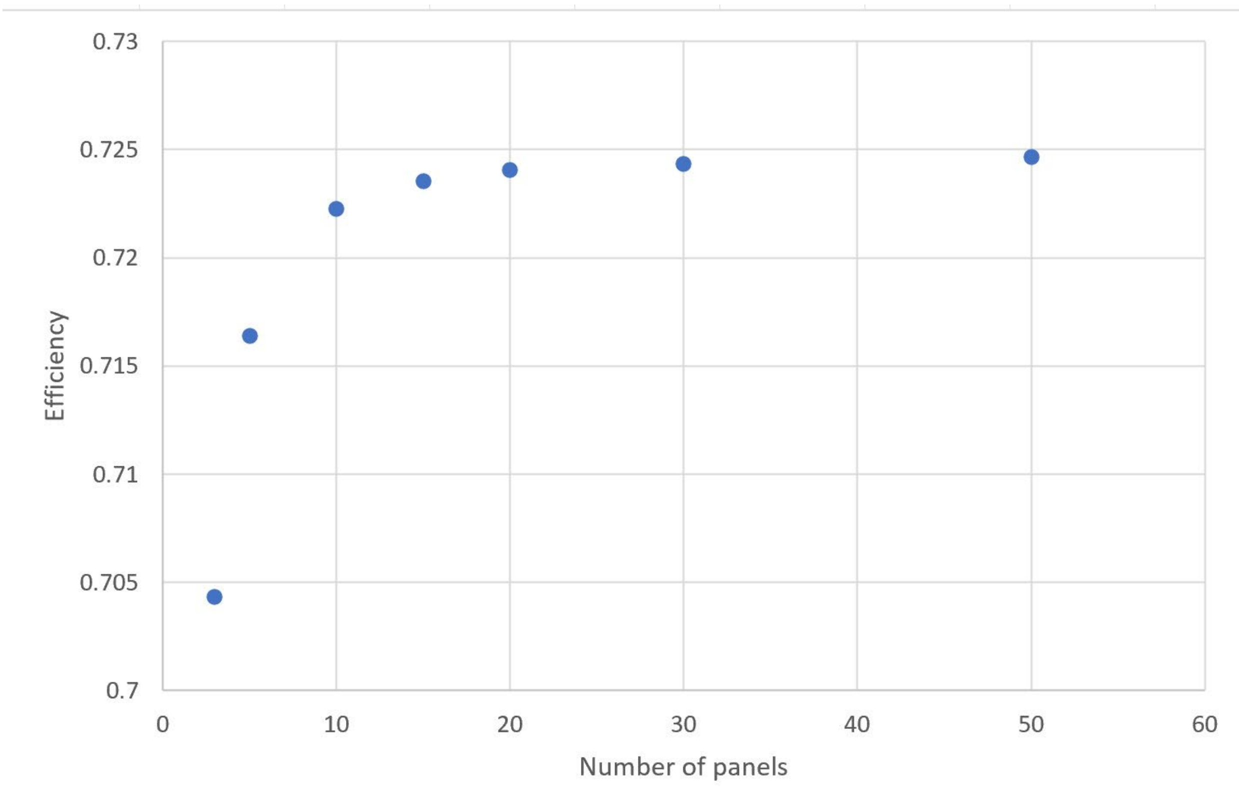


Figure 13: Efficiency solution as a function of the number of panels