Università degli Studi di Napoli
Federico II

OPTIMUM PROPELLER DESIGN FUNCTION

Autore: **Francesco Gervasio**
Nicola Russo

Contents

List of Figures

1 Theory

This function allows to design an optimum propeller by considering the approximated optimum propeller theory developed by Prandtl. It is possible to suppose that the propeller is lightly loaded, the number of blades has been consider $N >> 1$ in order to neglect the wake contraction; the viscous losses are neglected as well.By considering the overmentioned hypothesis we can define the thrust T and the power P distributions along the non-dimensional radius as functions of the rotational speed, the number of blades and two induction factors a and $a^{'}$ that will be analyzed in the following sections.

$$\begin{cases} dT = N\rho\Omega\bar{r}(1-a^{'})\Gamma R_{tip}^2 d(\bar{r}) \\ dP = N\rho\Omega\bar{r}V_\infty(1+a)\Gamma R_{tip}^2 d(\bar{r}) \end{cases} \tag{1.1}$$

It is important to notice that a, $a\prime$ represent the axial and rotational induction distributions along the non dimensional radius χ which can be defined as:

$$a = \frac{w_0}{V_\infty}\frac{\chi^2}{(1+\frac{w_0}{V_\infty})^2+\chi^2} \tag{1.2}$$

$$a^{'} = \frac{w_0}{V_\infty}\frac{1+\frac{w_0}{V_\infty}}{(1+\frac{w_0}{V_\infty})^2+\chi^2} \tag{1.3}$$

$$\chi = \frac{\Omega r}{V_\infty} \tag{1.4}$$

We can now define the circulation along the blade :

$$\frac{N\Gamma}{\Omega R_{tip}^2} = F(\bar{r})4\pi\bar{r}^2 a(\bar{r})\prime \tag{1.5}$$

In this expression it is possible to emphatize the F function which represents the Prandtl correction function that allows to take into account the effect related to a finite number of blades.

$$F(\bar{r}) = \frac{2}{\pi}\arccos[e^{\frac{N}{2\lambda}(\frac{r-R_{tip}}{R_{tip}})}]; \qquad \lambda = \frac{V_\infty}{\Omega R_{tip}} \tag{1.6}$$

2 Input and Output

In this section a list of both input and output parameters will be provided. The whole function consists of two different subfunctions:

- **Opti__Prop__T.m** which solves the Euler constrained minimum problem. In this case we are supposing to fix the thrust coefficient C_T and obtaining the maximum C_P coefficient.
- **Opti__Prop__P.m** which solves the Euler constrained minimum problem. In this case we are supposing to fix the power coefficient C_P and obtaining the maximum C_T coefficient.

2.1 Opti__Prop__T

2.1.1 Input

This function requires the following input values

- **N__blade** = blade's number
- **R__hub** = Hub percentage value with respect to the radius
- **R__tip** = Tip Radius
- **N__rpm** = Revolutions per minute
- **V__inf** = Asymptotic speed
- **C__T** = Thrust coefficient
- **h** = Altitude

2.1.2 Output

This function provides the following output values

- **r__adim__T** = non dimensional radius
- **chi__T** = non dimensional radius
- **a__corr__chi__T** = corrected axial induction vs χ

- **a_first_corr_chi_T** = corrected rotational induction vs χ
- **dCt_dradim_T** = Thrust coefficient distribution along the non dimensional radius
- **dCp_dradim_T** = Power coefficient distribution along the non dimensional radius
- **Cp** = Power coefficient

2.2 Opti_Prop_P

2.2.1 Input

This function requires the following input values

- **N_blade** = blade's number
- **R_hub** = Hub percentage value with respect to the radius
- **R_tip** = Tip Radius
- **N_rpm** = Revolutions per minute
- **V_inf** = Asymptotic speed
- **C_p** = Power coefficient
- **h** = Altitude

2.2.2 Output

This function provides the following output values

- **r_adim_P** = non dimensional radius
- **chi_P** = non dimensional radius
- **a_corr_chi_P** = corrected axial induction vs χ
- **a_first_corr_chi_P** = corrected rotational induction vs χ
- **dCp_dradim_P** = Power coefficient distribution along the non dimensional radius
- **dCt_dradim_P** = Thrust coefficient distribution along the non dimensional radius
- **Ct** = Thrust coefficient

2.3 Plot and Datasheet Output

2.3.1 Datasheet

Both functions provide an output datasheet file in which all these parameters are collected together with the efficiency calculated at the project advance ratio and the convergency induction velocity w. By default, the datasheets are named as

- **Data_Opti_Prop_T.txt**
- **Data_Opti_Prop_P.txt**

2.3.2 Plot - Opti_Prop_T(P)

Both functions provides the following charts that are reported just for an illustrative purpose; further information will be provided in the Test Cases section.

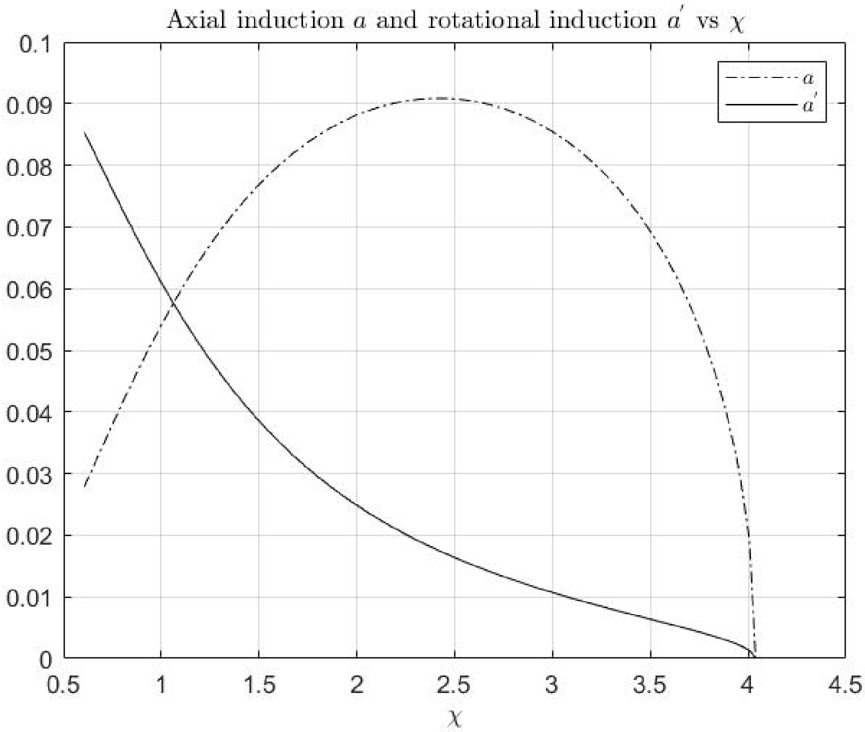

Figure 2.1: Axial a and rotational $a^{'}$ distribution scaled with the F Prandtl funciont vs χ

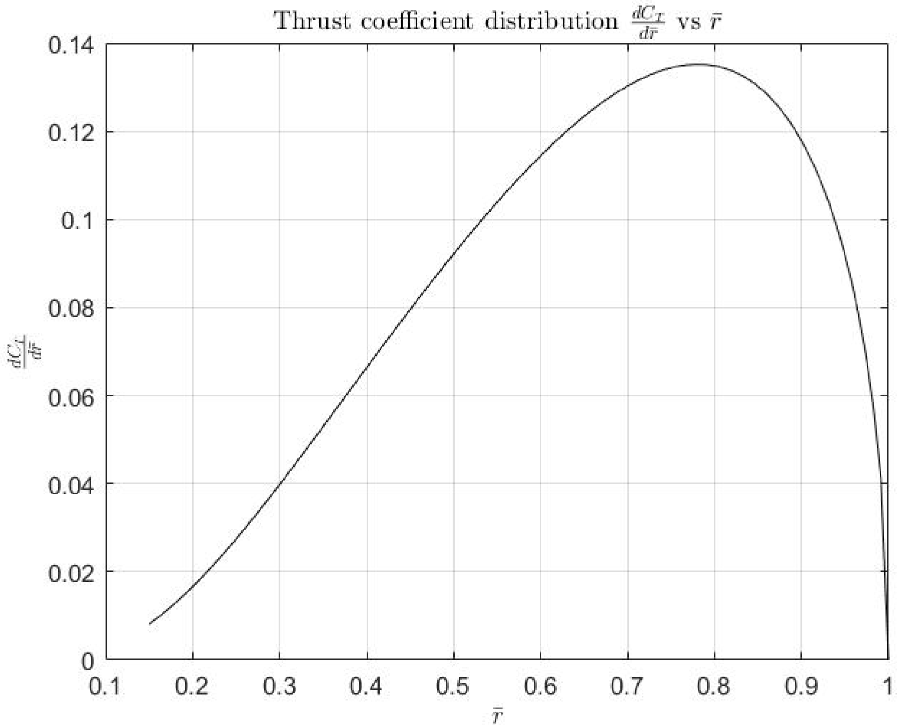

Figure 2.2: C_T thrust coefficient distribution vs non dimensional radius $\bar{r}$

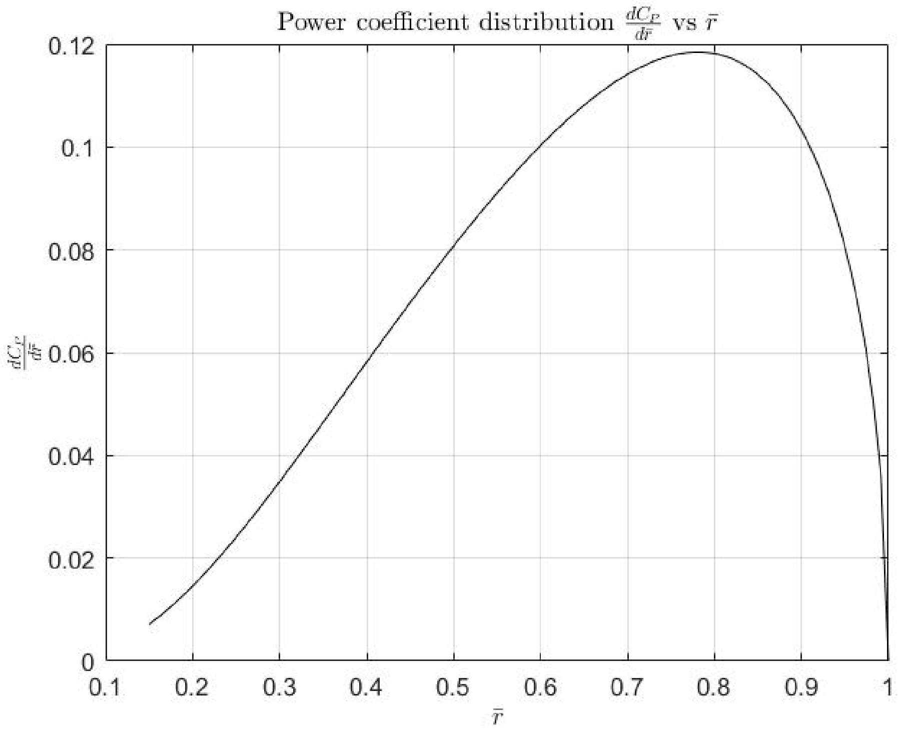

Figure 2.3: C_P power coefficient distribution vs non dimensional radius $\bar{r}$

3 Algorithm Description

In this chapter a detailed description of the algorithm will be provided. Since both subfunctions are similar, only the Opti_Prop_T subfunction will be analyzed while the Opti_Prop_P subfunction will be highlighted only in its peculiar aspetcs.

3.1 Opti_Prop_T

The two first attempt values of the axial induction w_0 and w_1 have been set. $w_0 = aV_\infty$ where a is the axial induction and $w_1 = 2w_0$. Two different first attempt values are necessary in order to initialize the false position method.

```
T     = Ct*rho*(n_rps^2)*(D^4);
syms w0;
eqn1 = w0 > 0;
eqn2 = T == 2*rho*pi*(R_tip^2)*(V_inf+w0)*w0;
eqn  = [eqn1 eqn2];
S    = solve(eqn,w0,'ReturnConditions',true);
w_0  = double(S.w0);
w_1  = 1.5*w_0;
T_1  = 2*rho*pi*(R_tip^2)*(V_inf+w_1)*w_1;
Ct_1 = T_1/(rho*(n_rps^2)*(D^4));
Ct_0 = 0;
Ct_new = 0;
```

The false position method has been chosen in ordert to implement the iterative cycle. The first value of the axial induction speed, has been calculated according to the classical method formulation by using the first attempt axial induction speed values. A while loop cycle has been implemented in order to exploit the false position method.

```
w = (w_1*error_0-w_0*error_1)/(error_0-error_1);
k = 0; %cycle counter
while abs((Ct-Ct_new)/Ct) > tao
```

The Prandtl correction function for finite blade number has been implemented. $\lambda = \chi^{-1}$. According to the momentum theory: $w_j = 2w$;

```
lambda = (n_rads*R_tip/V_inf)^-1;
F       = (2/pi)*acos(exp((N_blade/(2*lambda))*(r_adim-1)));
a = (w/V_inf).*((chi.^2)./((1+w/V_inf)^2+(chi.^2)));
a_first = (w/V_inf).*((1+w/V_inf)./((1+w/V_inf)^2+chi.^2));
```

The non-dimensional aerodynamic optimal load Gamma, already scaled with the Prandtl correction function is implemented.

```
GAMMA = (4*pi.*F.*(r_adim*R_tip).^2.*a_first*n_rads)/N_blade';
```

At each step, $\frac{dT}{d\bar{r}}$ or $\frac{dP}{d\bar{r}}$ are calculated. These values are lately integrated along the non dimensional radius in order to calculate a step T or P value that must be confronted with the design T or P value.

```
dT_dradim = N_blade*rho*n_rads*(R_tip^2)*r_adim.*(1-a_first).*GAMMA;
dCt_dradim= dT_dradim/(rho*(n_rps^2)*(D^4));
Ct_new     = trapz(r_adim,dCt_dradim);
error_new = Ct_new - Ct;
```

In this step, the induction speed values and errors are updated at each cycle and then the new induction speed value is calculated in order to reiterate the calculation until the while loop exit condition is verified.

```
w_0      = w_1;
w_1      = w;
error_0 = error_1;
error_1 = error_new;
w        = (w_1*error_0-w_0*error_1)/(error_0-error_1);
k        = k+1;
end
```

The corrected inductions are defined and the power distribution and coefficients are calculated as well. The efficiency η and the design advance ratio are then summared in the output file.

```
a_corr = a.*F;
a_first_corr = a_first.*F;
dCp_dradim = dT_dradim*(V_inf+w)/((rho*(n_rps^3)*(D^5)));
Cp   = trapz(r_adim,dCp_dradim)
J    = (V_inf/(n_rps*D));
eta = J*(Ct_new/Cp);
error_perc_Ct = abs((error_new))/Ct*100
```

3.2 Opti_Prop_P

It is possible to highlight the 3 different sections with respect to the previously analyzed script. The following code block is substitutive of the last 3 steps of the previous section.

```
%% Step 2
%{At each step, dT/dr_adim or dP/dr_adim are calculated. These ...
    values are lately integrated along the non dimensional ...
    radius in order to calculate a step T or P value that must ...
    be confronted with the design T or P value.
%}
dP_dradim = N_blade*rho*V_inf*n_rads*(R_tip^2)*r_adim.*(1+a).*GAMMA;
dCp_dradim = dP_dradim/(rho*(n_rps^3)*(D^5));
Cp_new = trapz(r_adim,dCp_dradim);
error_new = Cp_new - Cp;


%% Step 3 - FALSE POSITION METHOD
%{In this step, the induction speed values and errors are ...
    updated at each cycle and then the new induction speed value ...
    is calculated in order to reiterate the calculation until ...
    the while loop exit condition is verified.
%}
w_0 = w_1;
w_1 = w;
error_0 = error_1;
error_1 = error_new;
w = (w_1*error_0-w_0*error_1)/(error_0-error_1);
k = k+1;

end

a_corr = a.*F;
a_first_corr = a_first.*F;
dCt_dradim = dP_dradim/(((V_inf+w)*(rho*(n_rps^2)*(D^4))));
Ct  = trapz(r_adim,dCt_dradim);
J   = (V_inf/(n_rps*D));
eta = J*(Ct/Cp_new);
error_perc_Cp = abs((error_new))/Cp*100
```

4 Test Cases

In this chapter a test case is considered in order to validate both the Opti_Prop_T and Opti_Prop_P functions.

```
%% INPUT
N_blade = 2;          %[ ]
R_hub   = 0.15;       %[%]
R_tip   = 0.9;        %[m]
n_rpm   = 2500;       %[rpm]
V_inf   = 58.33 ;     %[m/s]
Ct = 0.0740;          %[]
h = 4510;             %[m]

r = linspace(R_hub*R_tip,R_tip,1000)/R_tip;
[r_adim_T,chi_T,a_corr_T,a_first_corr_T,..
dCt_dradim_T,dCp_dradim_T,Cp]= ...
    Opti_prop_T(N_blade,R_hub,R_tip,n_rpm,V_inf,Ct,h);
[r_adim_P,chi_P,a_corr_P,a_first_corr_P....
,dCp_dradim_P,dCt_dradim_P,Ct]= ...
    Opti_prop_P(N_blade,R_hub,R_tip,n_rpm,V_inf,Cp,h);
```

In order to verify that both functions provide almost the same output valus, a generic input data set has been definied. The C_p coefficient has been calculated by using the Opti_Prop_T; this value has been lately been given as an input parameter to the Opti_Prop_P function.

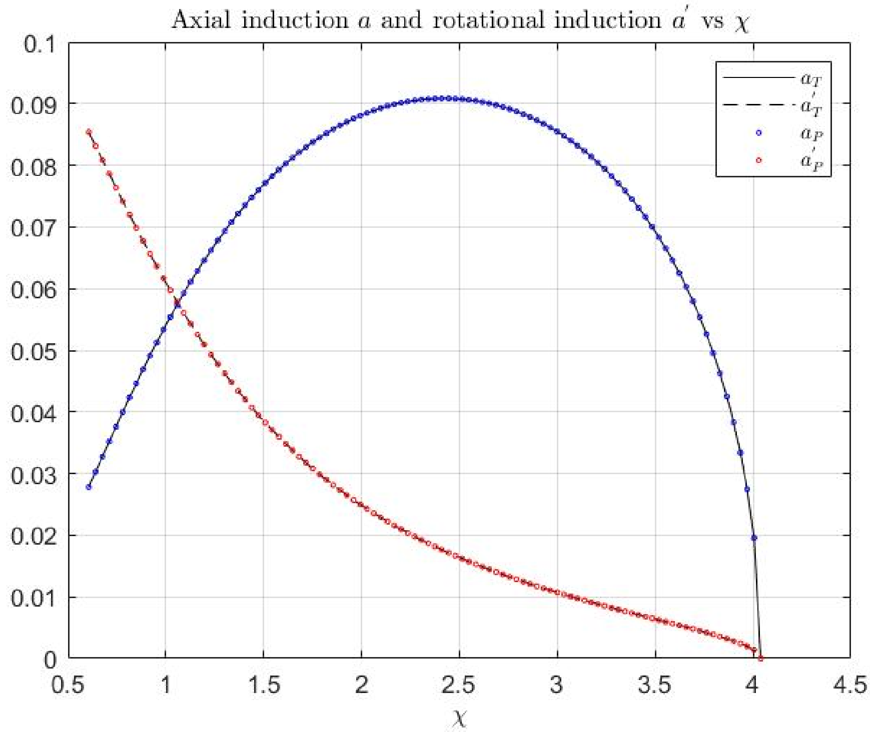

Figure 4.1: Axial a and rotational $a^{'}$ distribution scaled with the F Prandtl funciont vs χ

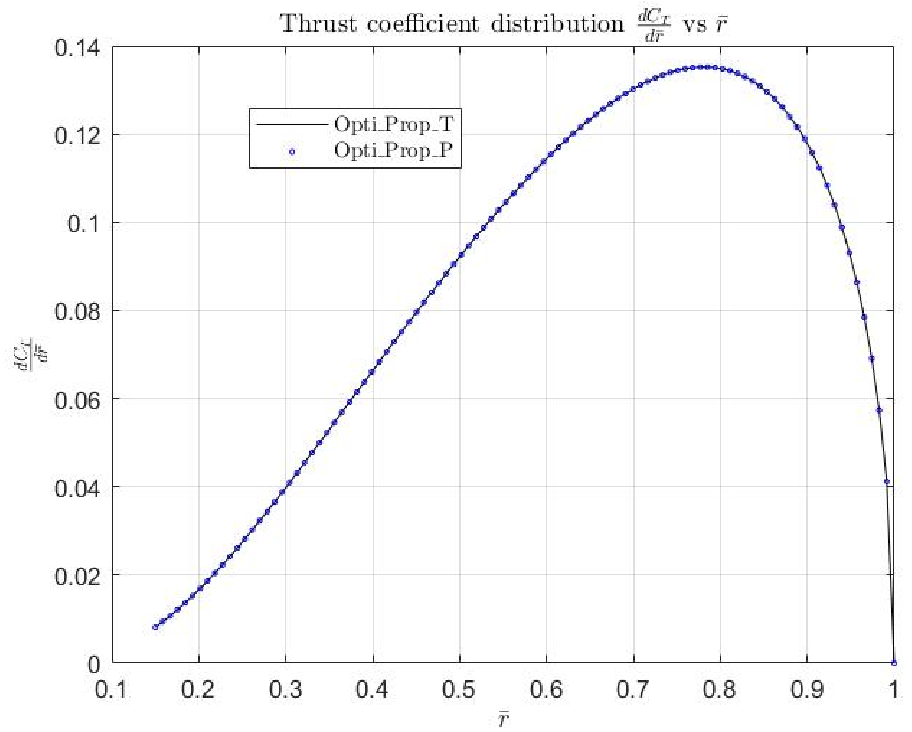

Figure 4.2: C_T thrust coefficient distribution vs non dimensional radius $\bar{r}$

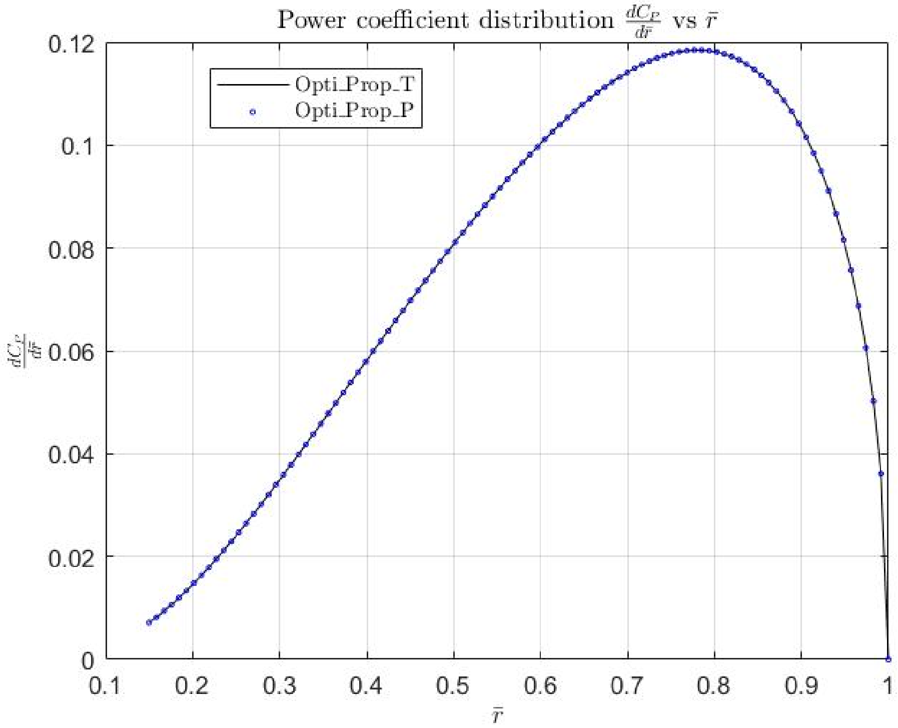

Figure 4.3: C_P power coefficient distribution vs non dimensional radius $\bar{r}$

Bibliography

[1] Tognaccini R., (2019), "Lezioni di Aerodinamica dell'ala rotante".