

DOCUMENTATION OF
AXIAL_DESCENT_ASCENT_OPERATING_
CURVES_ROTOR FUNCTION

Colledà Moreno M53001072
Veneruso Salvatore M53001082

21 December, 2020

Contents

1	Documentation	2
1.1	Algorithm Introduction	2
1.2	Algorithm Description	3
1.3	Input and Output	4
1.4	Test Case	5
	Appendices	8
A	Function's code	9

Chapter 1

Documentation

1.1 Algorithm Introduction

The function plots $w(V_\infty)$ and $P(V_\infty)$ curves according to rotor's simply impulsive theory.

To achieve this objective in the first step we defined the following vectors:

- ALTITUDE
- ASCENT VELOCITY
- DESCENT VELOCITY

Density is then calculated from the altitude vector with the MATLAB function *atmosisa*.

After calculated the density, the axial hovering induction is obtained from the following relation derived by rotor's simply impulsive theory:

$$w_h = \sqrt{\frac{Mg}{2\rho(h)\pi r^2}} \quad (1.1)$$

From this value, we calculated the non-dimensional variables:

- $\tilde{V} = \frac{V_\infty}{w_h}$
- $\tilde{w}_{\text{ascent}} = -\frac{\tilde{V}}{2} + \sqrt{\frac{\tilde{V}^2}{4} + 1}$
- $\tilde{w}_{\text{descent}} = -\frac{\tilde{V}}{2} + \sqrt{\frac{\tilde{V}^2}{4} - 1}$
- $\tilde{P}_{\text{ascent}} = \tilde{V} + \tilde{w}_{\text{ascent}}$
- $\tilde{P}_{\text{descent}} = \tilde{V} + \tilde{w}_{\text{descent}}$

Then, we used some cycles to obtain the matrices including the values of induction and power that are two variables functions. Subsequently these functions are plotted. In the code the user has also the possibility to insert the interest altitude.

1.2 Algorithm Description

The code begins at line 40 with the function call, where are defined the inputs as well as described in section 1.3.

Dimensional variables, as the altitude and the velocity, are defined in the lines [42-45].

The density as a function of altitude is calculated in line 49 through the MATLAB function *atmosisa*. In line 51 this variable is interpolated in a function handle to obtain a numerical law to use with any altitude that the user choice in line 152.

From the line 55 to line 59 the code calculates the non-dimensional variables as mentioned in section 1.1.

At lines [61-62] the matrices which will contain the numerical value of induction as a function of velocity and altitude are initialized. Subsequently (lines [66-81]) those matrices are filled with *for loops* as show below, where you can see that is respected the limit of simply impulsive rotor theory:

```
for i = 1 : length(hh)
    for j = 1 : length(VVs)
        WTS(i,j) = w_tilde_salita(VVs(j),hh(i));
    end
end

for i = 1 : length(hh)
    for j = 1 : length(VVd)
        if V_tilde(VVd(j),hh(i)) < -2
            WTD(i,j) = w_tilde_discesa(VVd(j),hh(i));
            aa(1,i) = j;
        else
            WTD(i,j) = 0;
        end
    end
end
```

Subsequently (lines [88-108]) the same thing is done for the non-dimensional power

The first type of outputs are the 3D plots of the induction and power as a function of altitude and velocity, which are in line [113-144]. After that, in lines [148-152] the user has the possibility to choose the altitude of interest to obtain the 2d plots of induction and power as function of only velocity at the entered altitude. In the following lines [154-185] the matrices containing the numerical value of induction and power are initialized and till as function of velocity at the entered altitude as show below:

```

prompt = {'Insert interest altitude in metres [min=0,Max=6000]: '};
dlgtitle = 'Altitude';
dims = [1 35];
answer = inputdlg(prompt,dlgtitle,dims);
hnew = str2double(answer{1});

WTSnew = zeros(1,length(VVs));
WTDnew = zeros(1,length(VVd));

% Matrices fill
for j = 1 : length(VVs)
    WTSnew(1,j) = w_tilde_salita(VVs(j),hnew);
end

for j = 1 : length(VVd)
    if V_tilde(VVd(j),hh(i)) < -2
        WTDnew(1,j) = w_tilde_discesa(VVd(j),hnew);
        aaneu = j;
    else
        break
    end
end

PTSnew = zeros(1,length(VVs));
PTDnew = zeros(1,length(VVd));

for j = 1 : length(VVs)
    PTSnew(1,j) = P_tilde_salita(VVs(j),hnew);
end

for j = 1 : length(VVd)
    if V_tilde(VVd(j),hh(i)) < -2
        PTDnew(1,j) = P_tilde_discesa(VVd(j),hnew);
        bbneu = j;
    else
        break
    end
end

```

The values of induction and power as function of velocity at the entered altitude are plotted through the command in line[188-207].

1.3 Input and Output

The function takes in input:

- ROTORCRAFT'S MASS
- ROTOR'S RADIUS

The outputs are the plots of induction and power and these are reported in section *Test Case*

1.4 Test Case

The numerical values in input for those plots are:

- ROTORCRAFT'S MASS = 5000kg
- ROTOR'S RADIUS = 7m

The outputs of the test case are the following plots:

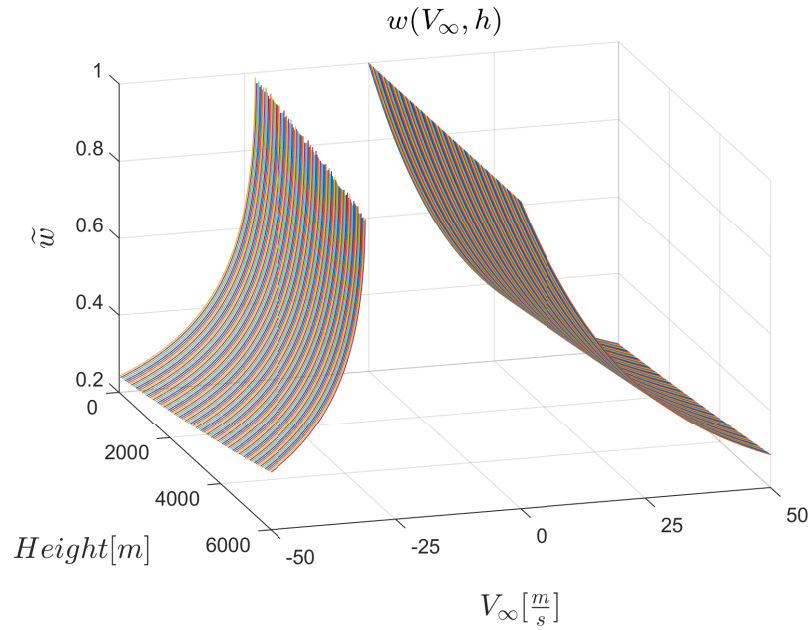


Figure 1.1: Induction

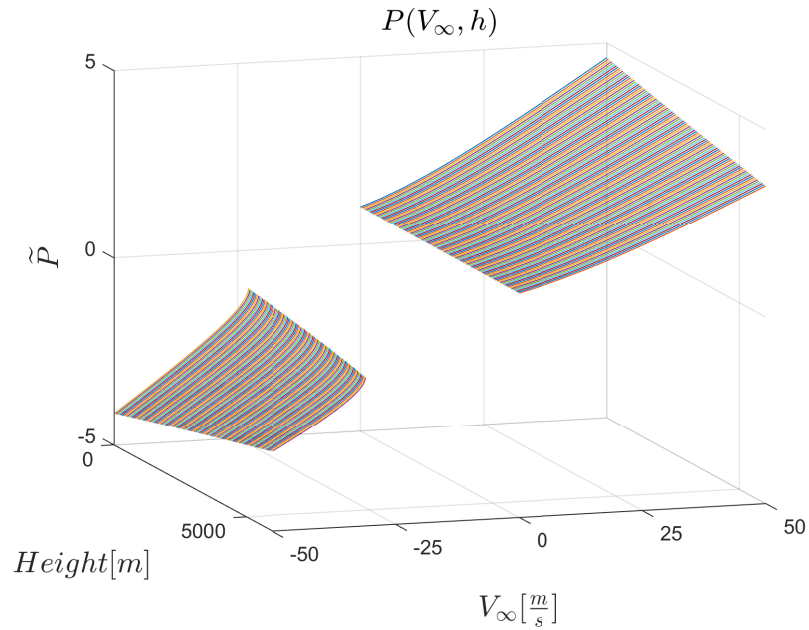


Figure 1.2: Power

In the code the user has also the possibility to insert the interest altitude: for example, at sea level we are obtained the following curves. In order to prove the validity of the code the curves were compared with some examples reported in [1] as follow.

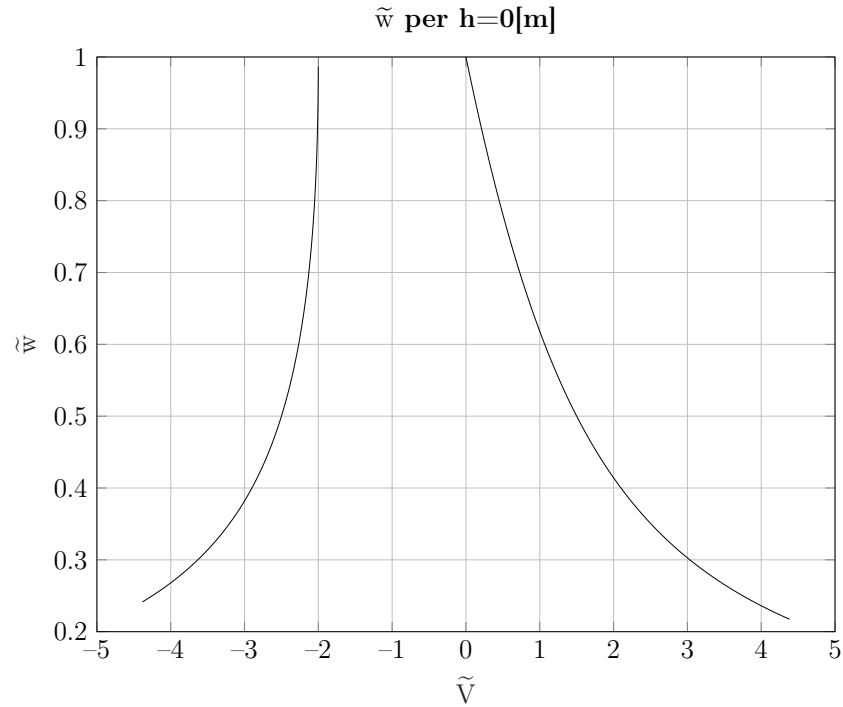
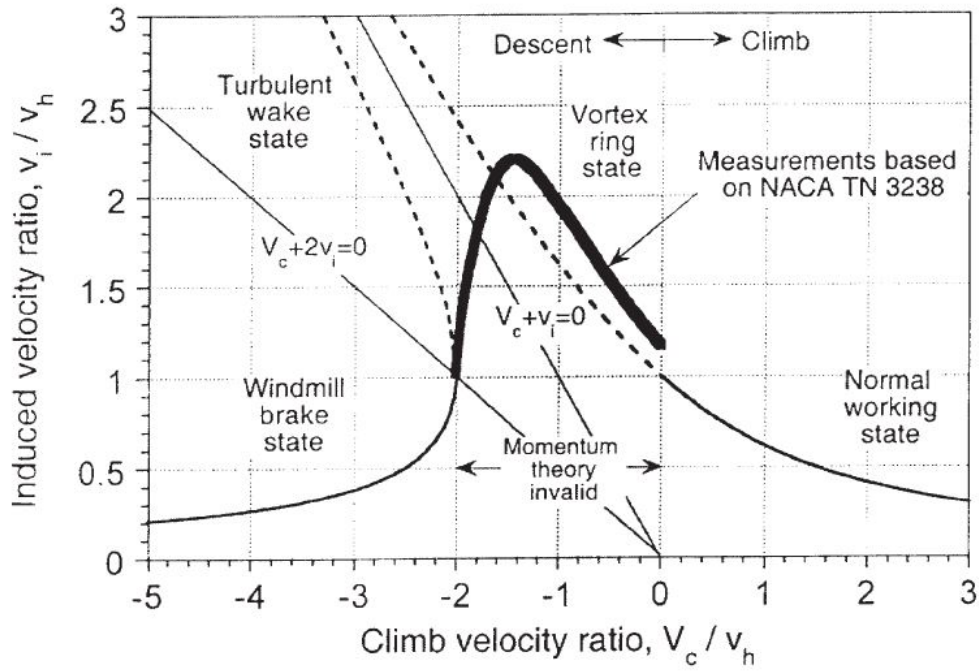


Figure 1.3: Induction



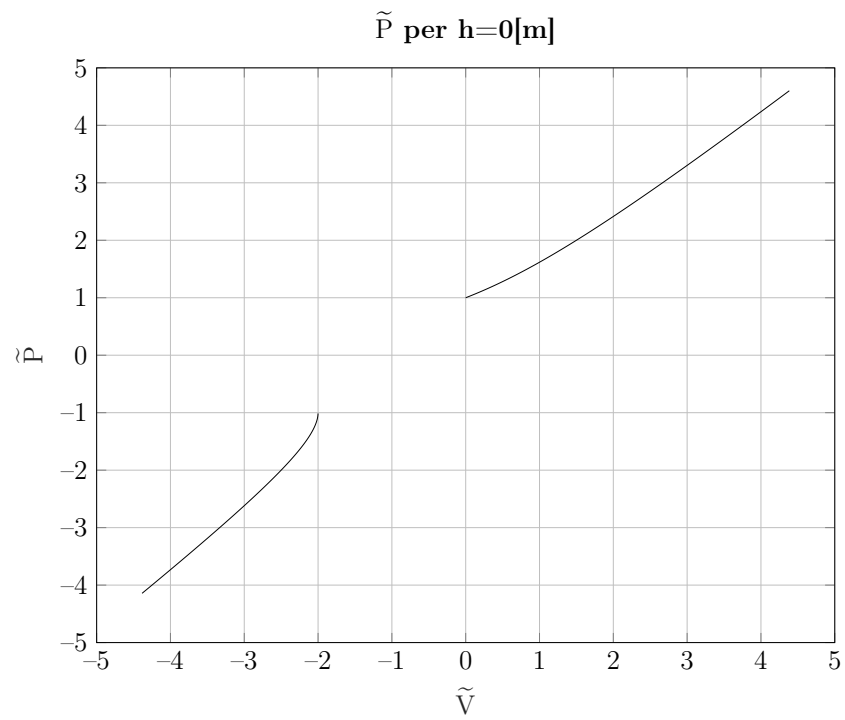
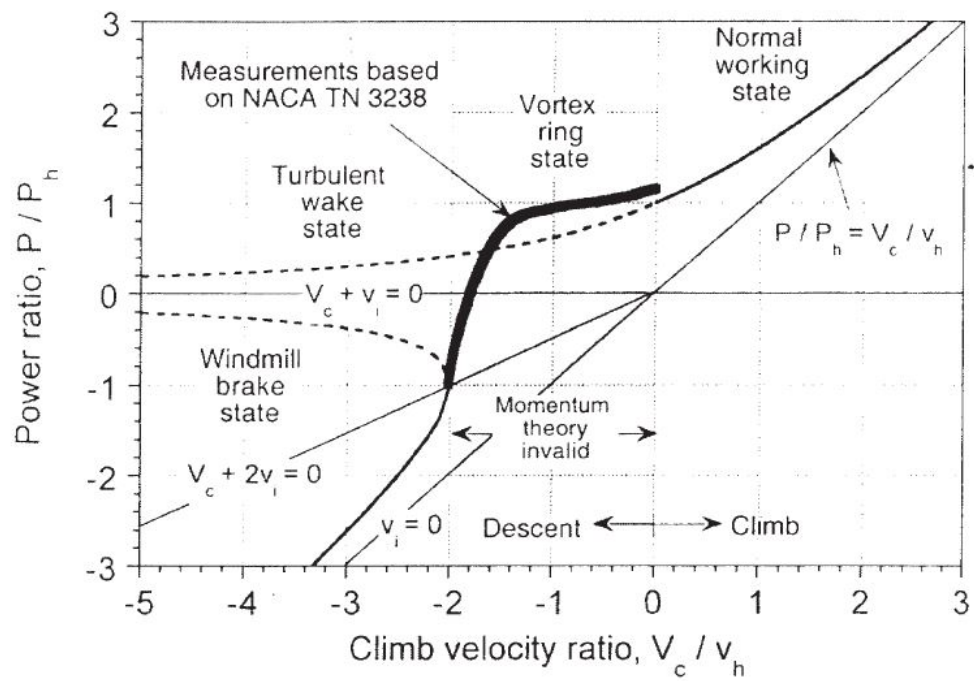


Figure 1.4: Power



Appendices

Appendix A

Function's code

```

1 %% \Axial_Descent_Ascent_Operating_Curves_Rotor.m
2 % \brief: the function plots w(V_infty) and P(V_infty) curves according to
3 % Impulsive theory.
4 % aerodynamic model
5 % \author: Colledà Moreno, Veneruso Salvatore
6 % \version: 1.00
7 %
8 % Eli-TAARG is free software; you can redistribute it and/or
9 % modify it under the terms of the GNU General Public
10 % License as published by the Free Software Foundation; either
11 % version 3 of the License, or (at your option) any later version.
12 %
13 % Eli-TAARG is developed by the TAARG Educational organization for
14 % educational purposes only.
15 % Theoretical and Applied Aerodynamic Research Group - University of Naples Federico II.
16 %
17 % Eli-TAARG GitHub link: <https://github.com/TAARG-Education/Eli-TAARG>
18 %
19 % Eli-TAARG is distributed in the hope that it will be useful,
20 % but WITHOUT ANY WARRANTY; without even the implied warranty of
21 % MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
22 % General Public License for more details.
23 % <http://www.gnu.org/licenses/>.
24 %
25 -----
26 % Name      : Axial_Descent_Ascent_Operating_Curves_Rotor.m
27 % Author    : Colledà Moreno, Veneruso Salvatore
28 %           : University of Naples Federico II.
29 % Version   : 1.00
30 % Date      : 21/12/2020
31 % Modified  : 21/12/2020
32 % Description: the function plots w(V_infty) and P(V_infty) curves according to
33 %           : Rotor Simply Impulsive theory.
34 % Reference  : Renato Tognaccini. Appunti Aerodinamica dell'ala rotante.
35 %           : Università degli studi di Napoli Federico II. a.a.2020/2021
36 % Input     : * the inputs must be Mass of rotorcraft and radius of rotor
37 % Output    : w(V_infty) and P(V_infty) plots
38 % Note      :
39 -----
40 function [Power,Induction]=Axial_Descent_Ascent_Operating_Curves_Rotor(M,R)
41 %Variables Definition
42 hh = linspace(0,6000,100);
43 VVs = linspace(0, 50, 100);
44 VVd = linspace(-50, 0, 7000);
45 g = 9.81;
46
47 %Axial Induction
48
49 [~, ~, ~, rho1] = atmosisa(hh);
50
51 rho = @(h) interp1(hh,rho1,h, 'pchip');
52 wh = @(h) sqrt((M*g)/(2*rho(h)*pi*R^2));
53
54 % Non-Dimensional Variables Definition
55 V_tilde = @(V,h) V/wh(h);
56
57 %Non Dimensional Induction
58 w_tilde_salita = @(V,h) (-V_tilde(V,h)/2) + sqrt((V_tilde(V,h)/2)^2+1);
59 w_tilde_discesa = @(V,h) (-V_tilde(V,h)/2) - sqrt((V_tilde(V,h)/2)^2-1);
60
61 WTS = zeros(length(hh),length(VVs)); %Ascent Induction
62 WTD = zeros(length(hh),length(VVd)); %Descent Induction
63 aa = zeros(1,length(hh)); %Control Parameter
64
65 % Matrices fill
66 for i = 1 : length(hh)
67     for j = 1 : length(VVs)
68         WTS(i,j) = w_tilde_salita(VVs(j),hh(i));
69     end
70 end
71
72 for i = 1 : length(hh)
73     for j = 1 : length(VVd)
74         if V_tilde(VVd(j),hh(i)) < -2 %Validity limit of simply impulsive theory;

```

```

75     WTD(i,j) = w_tilde_discesa(VVd(j),hh(i));
76     aa(1,i) = j;
77     else
78         WTD(i,j) = 0;
79     end
80 end
81 end
82
83 %Non-Dimensional Power
84
85 P_tilde_salita = @(V,h) V_tilde(V,h) + w_tilde_salita(V,h);
86 P_tilde_discesa = @(V,h) V_tilde(V,h) + w_tilde_discesa(V,h);
87
88 PTS = zeros(length(hh),length(VVs)); %Ascent Power
89 PTD = zeros(length(hh),length(VVd)); %Descent Power
90 bb = zeros(1,length(hh)); %Control Parameter
91
92 % Matrices fill
93 for i = 1 : length(hh)
94     for j = 1 : length(VVs)
95         PTS(i,j) = P_tilde_salita(VVs(j),hh(i));
96     end
97 end
98
99 for i = 1 : length(hh)
100     for j = 1 : length(VVd)
101         if V_tilde(VVd(j),hh(i)) < -2 %Validity limit of simply impulsive theory;
102             PTD(i,j) = P_tilde_discesa(VVd(j),hh(i));
103             bb(1,i) = j;
104         else
105             break
106         end
107     end
108 end
109
110 %%
111 % 3D Plots of Induction [output]
112 figure(1)
113 plot3(hh(1)*ones(1,length(WTS(1,:))),VVs,WTS(1,:))
114 hold on
115 for i = 2 : length(hh)
116     plot3(hh(i)*ones(1,length(WTS(i,:))),VVs,WTS(i,:))
117 end
118 for i = 1 : length(hh)
119     plot3(hh(i)*ones(1,length(WTD(i,1:aa(i)))),VVd(1,1:aa(i)),WTD(i,1:aa(i)))
120 end
121 xlabel('$Quota$', 'Interpreter', 'latex', 'FontSize', 15)
122 ylabel('$V_{\infty}$', 'Interpreter', 'latex', 'FontSize', 15)
123 zlabel('$w$', 'Interpreter', 'latex', 'FontSize', 15)
124 grid on
125 title('$w(V_{\infty}, h)$', 'Interpreter', 'latex', 'FontSize', 15)
126 view(71,32)
127
128 % 3D Plots of Power [output]
129 figure(2)
130 plot3(hh(1)*ones(1,length(PTS(1,:))),VVs,PTS(1,:))
131 hold on
132 for i = 2 : length(hh)
133     plot3(hh(i)*ones(1,length(PTS(i,:))),VVs,PTS(i,:))
134 end
135 for i = 1 : length(hh)
136     plot3(hh(i)*ones(1,length(PTD(i,1:bb(i)))),VVd(1,1:bb(i)),PTD(i,1:bb(i)))
137 end
138 xlabel('$Quota$', 'Interpreter', 'latex', 'FontSize', 15)
139 ylabel('$V_{\infty}$', 'Interpreter', 'latex', 'FontSize', 15)
140 zlabel('$P$', 'Interpreter', 'latex', 'FontSize', 15)
141 grid on
142 title('$P(V_{\infty}, h)$', 'Interpreter', 'latex', 'FontSize', 15)
143 view(71,32)
144
145 %% Insertion of Interest Altitude;
146
147 prompt = {'Insert interest altitude in metres [min=0,Max=6000]: '};
148 dlgtitle = 'Altitude';
149 dims = [1 35];
150 answer = inputdlg(prompt,dlgtitle,dims);
151 hnew = str2double(answer{1});
152
153 WTSnew = zeros(1,length(VVs)); %Vector Inizialization of axial ascent induction related to velocity
154 WTDnew = zeros(1,length(VVd)); %Vector Inizialization of axial descent induction related to velocity
155
156 % Matrices fill
157 for j = 1 : length(VVs)
158     WTSnew(1,j) = w_tilde_salita(VVs(j),hnew);
159 end
160
161 for j = 1 : length(VVd)
162     if V_tilde(VVd(j),hnew) < -2 %Validity limit of simply impulsive theory;
163         WTDnew(1,j) = w_tilde_discesa(VVd(j),hnew);
164         aanew = j;
165     else
166         break
167     end
168 end
169
170 PTSnew = zeros(1,length(VVs));
171 PTDnew = zeros(1,length(VVd));
172
173 for j = 1 : length(VVs)
174     PTSnew(1,j) = P_tilde_salita(VVs(j),hnew);
175 end
176
177

```

```

178 for j = 1 : length(VVd)
179     if V_tilde(VVd(j),hnew) < -2 %Validity limit of simply impulsive theory;
180         PTDnew(1,j) = P_tilde_discesa(VVd(j),hnew);
181         bbnew = j;
182     else
183         break
184     end
185 end
186 %%
187 % 2D Plots of induction [Output]
188 figure(3)
189 plot(V_tilde(VVs,hnew), WTSnew, '-k')
190 hold on
191 plot(V_tilde(VVd(1:aanew), WTDnew(1:aanew), '-k')
192 grid on
193 xlabel('$V_{\infty}$','Interpreter','latex','FontSize',15)
194 ylabel('$w$','Interpreter','latex','FontSize',15)
195 title(['$w$ per h=' num2str(hnew)],'Interpreter','latex')
196 matlab2tikz('D:\Universit \Magistrale\AerodinamicaAlaRotante\Matlab\EliiTAARG\Figure\Induzione2d.tex');
197
198 % 2D Plots of power [Output]
199 figure(4)
200 plot(V_tilde(VVs,hnew), PTSnew, '-k')
201 hold on
202 plot(V_tilde(VVd(1:aanew), PTDnew(1:bbnew), '-k')
203 grid on
204 xlabel('$V_{\infty}$','Interpreter','latex','FontSize',15)
205 ylabel('$P$','Interpreter','latex','FontSize',15)
206 title(['$P$ per h=' num2str(hnew)],'Interpreter','latex')
207 matlab2tikz('D:\Universit \Magistrale\AerodinamicaAlaRotante\Matlab\EliiTAARG\Figure\Potenza2d.tex');
208 %%
209 % Numerical Output of Power and Induction for the interest altitude;
210 Power = [PTDnew(1:bbnew), PTSnew];
211 Induction = [WTDnew(1:aanew), WTSnew];
212 end
213 -----

```

Bibliography

- [1] Renato Tognaccini, *Appunti Aerodinamica dell'Ala Rotante*.
Univeristà degli studi di Napoli Federico II, a.a. 2020/2021