Università degli Studi di Napoli Federico II

Scuola Politecnica e delle Scienze di Base
Corso di Laurea Magistrale in Ingegneria Aerospaziale

Documentation about the *geometryreader.m* class

Authors:
Davide Mennella
Alessandro Carmine Esposito

Anno Accademico 2020/2021

1 | Documentation

1.1 Introduction

The aim of this function is to read a standard text file (which the user has to compile with the geometry informations about the propeller) and to allocate the geometry parameters in local variables. Moreover, the class has been provided with two more functions that allow to analyse the propeller via *xrotor* and also to plot the main results.

1.2 User guide

For the correct usage of the tool, user has to put in the same folder:

- the *"geometryreader.m"* class (whose Matlab script is fully reported in appendix A);
- a standard text file;
- the *xrotor* executable;
- a *"main.m"* Matlab file (whose script is showed in the listing below) in which user can choose the desired operations.

```
PropDataFileName='propgeometry.txt';    %.txt standard geometry file
myProp=geometryreader(PropDataFileName);       %class call
myProp.xRotorAnalysis;
myProp.plotResults;
```

The user has three usage options:

- by typing the first two lines of the listing, the class will just read the input file and allocate the parameters in local variables;
- by typing the first three lines of the listing, the class will interface with *xrotor.exe*, analyze the geometry and allocate the results in local variables;
- by typing the first two and the last lines of the listing, the class will also plot the results of the analysis.

In figure 1.1 on the following page is showed the standard text file that the user has to compile with the features of the propeller.

```
PROPELLER MAIN GEOMETRIC DATA
----------------------------------------------------------------------
2       Number of blades
0.81    Tip radius, (m)
0.14    Hub radius, (m)
0       Hub wake displacement body radius, (m)

----------------------------------------------------------------------
PROPELLER PERFORMANCE
----------------------------------------------------------------------
0       Thrust, (N)
55000   Power, (W)

----------------------------------------------------------------------
AERODYNAMICS
----------------------------------------------------------------------
1       Medium lift coefficient, (adimensional)
0       Lock number
0       Equivalent parasite area

----------------------------------------------------------------------
FLIGHT CONDITIONS
----------------------------------------------------------------------
45      Airspeed (m/s)
0.21    Advance Ratio
4       Height asl,(km)

----------------------------------------------------------------------
BLADE SECTIONS
----------------------------------------------------------------------
1      2      3      4      5      6      7      8      9      10    SEZ
0.25   0.5    0.75   0.97                                            r/R
0.13   0.17   0.11   0.02                                            c/R
55     35     25     0                                               theta(deg)

----------------------------------------------------------------------
XROTOR ANALYSIS PARAMETERS
----------------------------------------------------------------------
500      Number of iterations
0        Power choice (set 0 for power and 1 for thrust)
0.143    First advance ratio value
0.363    Last advance ratio value
0.006    Advance ratio step
```

Figure 1.1. Standart text file example

Finally, in the listing below, taken from the *geometryreader.m* script, the user can find any variable associated with their name.

```
%-------------------------------------------------------------------
% Geometry
%-------------------------------------------------------------------
N       %Number of blades
t_r     %Tip radius
h_r     %Hub radius
hwb     %Hub wake displacement body radius, (m)
%-------------------------------------------------------------------
% Propeller performance
%-------------------------------------------------------------------
T       %Thrust, (N)
P       %Power, (W)

%-------------------------------------------------------------------
% Aerodynamics
%-------------------------------------------------------------------
Cl_ave      %Average lift coefficient
lock_number %Lock number
f

%-------------------------------------------------------------------
% Flight conditions
%-------------------------------------------------------------------
v_inf   %Airspeed, (m/s)
adv     %Advance ratio
h       %Height asl, (km)

%-------------------------------------------------------------------
%Propeller sections
%-------------------------------------------------------------------
r           %Adimensional radius
c           %Adimensional chord
theta       %Pitch angle

%-------------------------------------------------------------------
%ANALYSIS PARAMETERS
%-------------------------------------------------------------------
iter          %Number of iterations
Powerchoice   %Choice between Power and Thrust
firstadv      %first advance ratio value
lastadv       %last advance ratio value
advstep       %advance ratio step

%-------------------------------------------------------------------
%ANALYSIS RESULTS
%-------------------------------------------------------------------
%Results of the analysis performed at the project advance ratio
chords         %chords distribuition along the radius
pitch          %pitch distribuition along the radius (deg)
radius         %adimensional radius vector
Reynolds       %Reynolds number along the radius
Mach           %Mach number along the radius
Cd             %drag coefficient along the radius

%Results of the analysis performed for different advance ratios
advanceratio   %advance ratio vector
Thrust         %Thrust vector (N)
Power          %Power vector(kW)
Torque         %Torque vector
Efficiency     %Efficiency vector
rpm            %rpm vector
```

A | Appendix

A.1 *geometryreader.m* script

The *geometryreader.m* script is shown in full in the following listing.

```
classdef geometryreader < handle
properties

Name = 'geometryreader';
err = 0;
%-------------------------------------------------------------------
% Geometry
%-------------------------------------------------------------------
N       %Number of blades
t_r     %Tip radius
h_r     %Hub radius
hwb     %Hub wake displacement body radius, (m)
%-------------------------------------------------------------------
% Propeller performance
%-------------------------------------------------------------------
T       %Thrust, (N)
P       %Power, (W)

%-------------------------------------------------------------------
% Aerodynamics
%-------------------------------------------------------------------
Cl_ave      %Average lift coefficient
lock_number %Lock number
f

%-------------------------------------------------------------------
% Flight conditions
%-------------------------------------------------------------------
v_inf   %Airspeed, (m/s)
adv     %Advance ratio
h       %Height asl, (km)

%-------------------------------------------------------------------
%Propeller sections
%-------------------------------------------------------------------
r           %Adimensional radius
c           %Adimensional chord
theta       %Pitch

%-------------------------------------------------------------------
%ANALYSIS PARAMETERS
%-------------------------------------------------------------------
iter          %Number of iterations
Powerchoice   %Choice between Power and Thrust
firstadv      %first advance ratio value
lastadv       %last advance ratio value
advstep       %advance ratio step

%-------------------------------------------------------------------
%ANALYSIS RESULTS
%-------------------------------------------------------------------
%Results of the analysis performed at the project advance ratio
chords         %chords distribuition along the radius
pitch          %pitch distribuition along the radius (deg)
radius         %adimensional radius vector
Reynolds       %Reynolds number along the radius
```

```
Mach           %Mach number along the radius
Cd             %drag coefficient along the radius

%Results of the analysis performed for different advance ratios
advanceratio   %advance ratio vector
Thrust         %Thrust vector (N)
Power          %Power vector(kW)
Torque         %Torque vector
Efficiency     %Efficiency vector
rpm            %rpm vector

end

methods
%% Constructor, populates class properties reading from  propgeometry file
function obj = geometryreader (dataFileName)

f_id = fopen(dataFileName,'r');
    % verifies the .txt file opening
if (f_id==-1)
 obj.err = -1;
 disp(['geometryreader␣::␣initFromFile␣___␣Could␣NOT␣open␣file␣', ...
 dataFileName, '␣...'])
else
 disp(['geometryreader␣::␣initFromFile␣___␣Opening␣file␣', ...
 dataFileName, '␣...␣OK.'])
end
%% File open, OK
for i=1:3
 fgetl(f_id);
end
%% Geometric data
obj.N = fscanf(f_id,'%f'); fgetl(f_id);
obj.t_r = fscanf(f_id,'%f') ;fgetl(f_id);
obj.h_r = fscanf(f_id,'%f');fgetl(f_id);
obj.hwb = fscanf(f_id,'%f');fgetl(f_id);
for i=1:4
 fgetl(f_id);
end
%% Performance data
obj.T = fscanf(f_id,'%f␣'); fgetl(f_id);
obj.P = fscanf(f_id,'%f␣'); fgetl(f_id);
for i=1:4
 fgetl(f_id);
end
%% Aerodynamics
obj.Cl_ave = fscanf(f_id,'%f'); fgetl(f_id);
obj.lock_number= fscanf(f_id,'%f'); fgetl(f_id);
obj.f=fscanf(f_id,'%f'); fgetl(f_id);
for i=1:4
 fgetl(f_id);
end
%% Flight conditions
obj.v_inf = fscanf(f_id,'%f'); fgetl(f_id);
obj.adv = fscanf(f_id,'%f');fgetl(f_id);
obj.h = fscanf(f_id,'%f');
for i=1:6
  fgetl(f_id);
end

%% Propeller sections
%% radius
dataBuffer = textscan(f_id,'%s\n','CollectOutput',1,...
                                 'Delimiter','');
r=dataBuffer{:};
newStr = split(r);
newStr=newStr(1:end-1);
obj.r=str2double(newStr)';
fgetl(f_id);

%% chord
dataBuffer = textscan(f_id,'%s\n','CollectOutput',1,...
                                 'Delimiter','');
c=dataBuffer{:};
newStr = split(c);
newStr=newStr(1:end-1);
```

```
obj.c=str2double(newStr)';
fgetl(f_id);

%% theta
dataBuffer = textscan(f_id,'%s\n','CollectOutput',1,...
                                  'Delimiter','');
theta=dataBuffer{:};
newStr = split(theta);
newStr=newStr(1:end-1);
obj.theta=str2double(newStr)';
fgetl(f_id);

%% Xrotor analysis parameter
for i=1:4
 fgetl(f_id);
end
obj.iter = fscanf(f_id,'%f');fgetl(f_id);
obj.Powerchoice = fscanf(f_id,'%f');fgetl(f_id);
obj.firstadv = fscanf(f_id,'%f');fgetl(f_id);
obj.lastadv = fscanf(f_id,'%f');fgetl(f_id);
obj.advstep = fscanf(f_id,'%f');fgetl(f_id);
%% finally, sets the error tag
obj.err = 0;
end

%% performs analysis via xrotor, if requested from user

function  xRotorAnalysis(obj)

%% creates .txt input file for xrotor
fid = fopen('xrotor_input.txt','w');
fprintf(fid,'ATMO\n');
fprintf(fid,'%f\n',obj.h);
fprintf(fid,'DESI\n');
fprintf(fid,'INPU\n');
fprintf(fid,'%d\n',obj.N);
fprintf(fid,'%f\n',obj.t_r);
fprintf(fid,'%f\n',obj.h_r);
fprintf(fid,'%f\n',obj.hwb);
fprintf(fid,'%f\n',obj.v_inf);
fprintf(fid,'%f\n',obj.adv);
fprintf(fid,'%d\n',obj.Powerchoice);
fprintf(fid,'%f\n',obj.P);
fprintf(fid,'%f\n',obj.Cl_ave);
fprintf(fid,'\n');
fprintf(fid,'\n');
fprintf(fid,'\n');
fprintf(fid,'OPER\n');
fprintf(fid,'ITER\n');
fprintf(fid,'%d\n',obj.iter);
fprintf(fid,'ADVA\n');
fprintf(fid,'%f\n',obj.adv);
fprintf(fid,'WRIT\n');
fprintf(fid,'newprop.txt\n');
fprintf(fid,'ASEQ\n');
fprintf(fid,'%f\n',obj.firstadv);
fprintf(fid,'%f\n',obj.lastadv);
fprintf(fid,'%f\n',obj.advstep);
fprintf(fid,'CPUT\n');
fprintf(fid,'prest\n');
% Close file
fclose(fid);

%% Runs Xrotor using input file, if requested from user in the file 'main'
cmd = 'xrotor.exe␣<␣xrotor_input.txt';
[status,result] = system(cmd);

%% Reads results saved from xrotor and allocates them in local variables
newprop='newprop.txt';
fidprop = fopen(newprop);                    % Open file for reading

dataBuffer = textscan(fidprop,'%f%f%f%f%f%f%f%f%f%f%f','CollectOutput',1,... % Read data from file
                                  'Delimiter','','HeaderLines',17);
fclose(fidprop);                                                             % Close file
delete('newprop.txt');
A=dataBuffer{:};
```

```
obj.radius=A(:,2);
obj.chords=A(:,3);
obj.pitch=A(:,4);
obj.Reynolds = A(:,7);
obj.Mach=A(:,8);
obj.Cd=A(:,6);

prest='prest.txt';
fidprop1 = fopen('prest');
dataBuffer1 = textscan(fidprop1,'%f%f%f%f%f%f%f%f%f%f%f%f%f','CollectOutput',1,...
                                 'Delimiter','','HeaderLines',3);
fclose(fidprop1);
delete('prest');
B=dataBuffer1{:};
obj.advanceratio=B(:,2);
obj.rpm=B(:,5);
obj.Power=B(:,10);
obj.Thrust=B(:,11);
obj.Torque=B(:,12);
obj.Efficiency=B(:,13);

end

%% Generates plots, if requested from user in the file 'main'
function plotResults(obj)

%% creates .txt input file for xrotor
fid = fopen('xrotor_input.txt','w');
fprintf(fid,'ATMO\n');
fprintf(fid,'%f\n',obj.h);
fprintf(fid,'DESI\n');
fprintf(fid,'INPU\n');
fprintf(fid,'%d\n',obj.N);
fprintf(fid,'%f\n',obj.t_r);
fprintf(fid,'%f\n',obj.h_r);
fprintf(fid,'%f\n',obj.hwb);
fprintf(fid,'%f\n',obj.v_inf);
fprintf(fid,'%f\n',obj.adv);
fprintf(fid,'%d\n',obj.Powerchoice);
fprintf(fid,'%f\n',obj.P);
fprintf(fid,'%f\n',obj.Cl_ave);
fprintf(fid,'\n');
fprintf(fid,'\n');
fprintf(fid,'\n');
fprintf(fid,'OPER\n');
fprintf(fid,'ITER\n');
fprintf(fid,'%d\n',obj.iter);
fprintf(fid,'ADVA\n');
fprintf(fid,'%f\n',obj.adv);
fprintf(fid,'WRIT\n');
fprintf(fid,'newprop.txt\n');
fprintf(fid,'ASEQ\n');
fprintf(fid,'%f\n',obj.firstadv);
fprintf(fid,'%f\n',obj.lastadv);
fprintf(fid,'%f\n',obj.advstep);
fprintf(fid,'CPUT\n');
fprintf(fid,'prest\n');
% Close file
fclose(fid);

%% Runs Xrotor using input file, if requested from user in the file 'main'
cmd = 'xrotor.exe␣<␣xrotor_input.txt';
[status,result] = system(cmd);

%% Reads results saved from xrotor and allocates them in local variables
newprop='newprop.txt';
fidprop = fopen(newprop);                    % Open file for reading
dataBuffer = textscan(fidprop,'%f%f%f%f%f%f%f%f%f%f','CollectOutput',1,... % Read data from file
                                 'Delimiter','','HeaderLines',17);
fclose(fidprop);                                                           % Close file
delete('newprop.txt');
A=dataBuffer{:};
obj.radius=A(:,2);
obj.chords=A(:,3);
obj.pitch=A(:,4);
obj.Reynolds=A(:,7);
obj.Mach=A(:,8);
obj.Cd=A(:,6);

prest='prest.txt';
fidprop1 = fopen('prest');
dataBuffer1 = textscan(fidprop1,'%f%f%f%f%f%f%f%f%f%f%f%f%f','CollectOutput',1,...
                                'Delimiter','','HeaderLines',3);
fclose(fidprop1);
delete('prest');
B=dataBuffer1{:};
obj.advanceratio=B(:,2);
obj.rpm=B(:,5);
obj.Power=B(:,10);
obj.Thrust=B(:,11);
obj.Torque=B(:,12);
obj.Efficiency=B(:,13);

figure(1)
title('\theta␣(deg)');
plot(obj.radius,obj.pitch);
grid on;
xlabel('r/R');
ylabel('\theta␣(deg)');

figure(2)
title('chord');
plot(obj.radius,obj.chords);
grid on;
xlabel('r/R');
ylabel('c/R');

figure(3)
title('Reynolds␣Number');
plot(obj.radius,obj.Reynolds);
grid on;
xlabel('r/R');
ylabel('Re*10^3');

figure(4)
title('Mach␣Number');
plot(obj.radius,obj.Mach);
grid on;
xlabel('r/R');
ylabel('M');

figure(5)
title('Cd');
plot(obj.radius,obj.Cd);
grid on;
xlabel('r/R');
ylabel('C_d');

figure(6)
title('efficienza');
plot(obj.advanceratio,obj.Efficiency);
grid on;
xlabel('J');
ylabel('\eta');

figure(7)
title('Power');
plot(obj.advanceratio,obj.Power);
grid on;
xlabel('J');
ylabel('P␣(kW)');

figure(8)
title('Thrust');
plot(obj.advanceratio,obj.Thrust);
grid on;
xlabel('J');
ylabel('T␣(N)');

end %plotResults
end %methods
end %constructor of geometryreader
```