xfoil_polar.m user guide

Lorenzo Frascino - Palma Caputo

Contents

1 Introduction

The function *xfoil_polar.m* allows to evaluate the $C_d - C_l$ polar of an airfoil by using the software *xfoil* [1]. The lift coefficient C_l is obtained by direct surface pressure integration

$$C_l = \int C_p d\bar{x} \tag{1}$$

where the integral is performed in the clockwise direction around the airfoil contour. The pressure coefficient C_p is calculated using the Karman-Tsien compressibility correction.
The drag coefficient C_d is obtained by applying the Squire-Young formula at the last point in the wake. For further information, please refer to [1].

2 I/O

Inputs and outputs are listed in the 2 following subsections.

2.1 Inputs

- **airfoil**: airfoil of interest. It can be a NACA airfoil or an external one. If NACA, it is required the number; if not, it is necessary to write the file name with the coordinates of the external airfoil.
  Example: `2412` for NACA 2412; `'myairfoil.txt'` for external file.
- **numPanel**: number of panels in which the airfoil is divided;
- **Re_number**: Reynolds number;
- **FirstAlfa**: first value of the angle of attack;
- **LastAlfa**: last value of the angle of attack;
- **DeltaAlfa**: pace between one angle of attack and the following one.

Number of iterations is set to 100. Number of panels is still one of the inputs as to allow user to control convergence.

2.2 Outputs

- C_l: lift coefficient;
- C_d: drag coefficient.

3 Code description

After having inserted inputs, the code creates an array of angle of attack `Alfa_vec` that will then be given to xfoil. Iterations are set to 100.
In order not to having overwriting with existing files, it is then checked the possibility and eventually existing files are deleted.
Some of the inputs are converted into strings so that it is possible to write the input file through `printf`.

```
numPanel_st   = num2str(numPanel);
Re_number_st = num2str(Re_number);
FirstAlfa_st = num2str(FirstAlfa);
LastAlfa_st   = num2str(LastAlfa);
DeltaAlfa_st = num2str(DeltaAlfa);

iter = '100';
Alfa_vec = FirstAlfa:DeltaAlfa:LastAlfa;

saveGeometry = 'Airfoil_geometry.txt';  % Create .txt file to ...
    save airfoil coordinates
savePolar    = 'Polar.txt';             % Create .txt file to ...
    save the polar


% Delete files if they exist
if (exist(saveGeometry,'file'))
    delete(saveGeometry);
end

if (exist(savePolar,'file'))
    delete(savePolar);
end
```

The following step is the creation of a `.txt` file with commands which have to be passed to xfoil.

```
%% WRITING XFOIL COMMANDS
% Create the airfoil
f_input = fopen('xfoil_input.txt','w');            % Create ...
    input file for xfoil
fprintf(f_input,'y\n');

if isnumeric(airfoil)
    airfoil = num2str(airfoil);
    fprintf(f_input,['naca ' airfoil '\n']);
else
    fprintf(f_input,['load ' airfoil '\n']);
end

fprintf(f_input,'PPAR\n');
fprintf(f_input,['N ' numPanel_st '\n']);
fprintf(f_input,'\n\n');

% Data for the polar
fprintf(f_input,'OPER\n');
fprintf(f_input,'visc\n');
fprintf(f_input,[Re_number_st '\n']);
fprintf(f_input,['iter ' iter '\n']);
```

```
fprintf(f_input,'pacc\n');
fprintf(f_input,[savePolar '\n\n']);

for i = 1:length(Alfa_vec)
    fprintf(f_input,'a %2.4f\n', Alfa_vec(i));
end

fprintf(f_input,'pacc\n\n');

% Save the airfoil data points
fprintf(f_input,['PSAV ' saveGeometry '\n']);

% Close file
fclose(f_input);
```

After having run the software, data file is read and lift and drag coefficient imported from the file. In addition, it is possible to plot the polar in the main with the new-obtained values of C_l and C_d.

```
%% RUNNING XFOIL (MUST BE IN THE SAME DIRECTORY!)
cmd = 'xfoil.exe < xfoil_input.txt';
[status,result] = system(cmd);

%% READ DATA FILE
filePol = fopen(savePolar);
A = textscan(filePol,'%f %f %f %f %f %f %f', 'Headerlines',12);
fclose(filePol);
alfa = A{1}(:,1);
Cl   = A{2}(:,1);
Cd   = A{3}(:,1);

figure(1);
plot(Cd,Cl,'k.-')
xlabel('Drag coefficient C_d');
ylabel('Lift coefficient C_l');
grid on;
```

In order to work, the main and function must be in the same directory as the software xfoil! Same goes for external airfoils data files (if needed).

4 Test cases

4.1 Case 1 - `NACA 2412` airfoil

The inputs that had been chosen for the test case are listed below. An example for a `main` is the following:

```
clc; close all; clear all;

%% INPUT DATA
airfoil     = 2412;      % NACA number or external file ...
    ('myairfoil.txt')
numPanel    = 120;       % Number of panels
```

```
Re_number  = 1e6;       % Reynolds number
FirstAlfa  = -10;       % First value of the angle of attack
LastAlfa   = 10;        % Last value of the angle of attack
DeltaAlfa  = 0.5;       % Pace

[Cl, Cd] = CdCl_xfoil(airfoil, numPanel, Re_number, FirstAlfa, ...
    LastAlfa, DeltaAlfa);
```

The code provides the $C_d - C_l$ polar in figure 1.

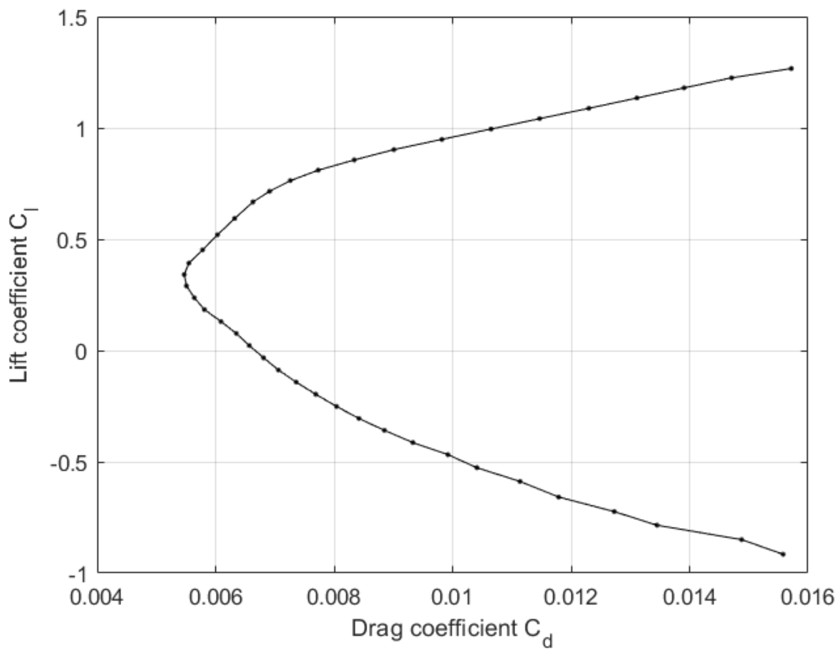

Figure 1: $C_d - C_l$ polar for the test case 1. NACA 2412 airfoil.

4.2 Case 2 - ONERA 212 airfoil

In the second test is for an external airfoil. It has been chosen the ONERA 212 airfoil (OA212) which coordinates are available on airfoiltools website.

```
%% INPUT DATA
airfoil    = 'oa212.txt';      % NACA number or external file ...
    ('myairfoil.txt')
numPanel   = 120;       % Number of panels
Re_number  = 1e6;       % Reynolds number
FirstAlfa  = -10;       % First value of the angle of attack
LastAlfa   = 10;        % Last value of the angle of attack
DeltaAlfa  = 0.5;       % Pace

```

```
[Cl, Cd] = CdCl_xfoil(airfoil, numPanel, Re_number, FirstAlfa, ...
    LastAlfa, DeltaAlfa);
```

Results for the case 2 are shown in figure 2.

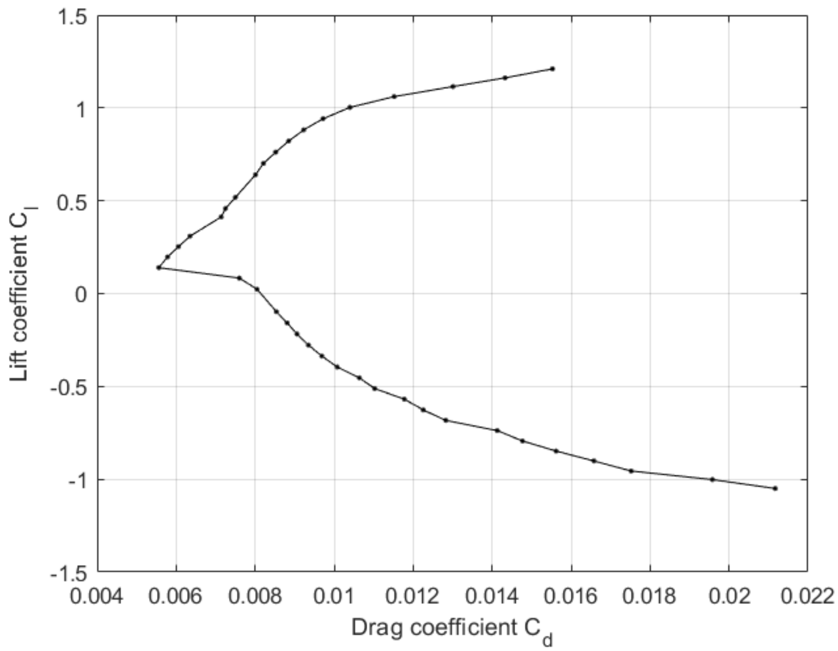

Figure 2: $C_d - C_l$ polar for the test case 2. ONERA 212 airfoil.

References

[1] *xfoil User's Guide*. MIT. URL: https://web.mit.edu/drela/Public/web/xfoil/xfoil_doc.txt.