Xoptfoil_visualizer-JX is a modified version of Xoptfoil_visualizer version 1.11.1 by Daniel Prosser

Xoptfoil_visualizer-JX v1.50.2

Reference

Jochen Guenzel 2020

The Python script 'Xoptfoil_visualizer-JX.py' is used to visualize the data generated by the airfoil optimizer Xoptfoil-JX during or after an optimization run.

Although the JX version should be downward compatible to the original Xoptfoil_visualizer it is recommended to use both the optimizer and the visualizer in the JX version.

This brief reference describes only the enhancements and changes made to the original Xoptfoil_visualizer. More information can be found in the 'Xoptfoil User Guide' by Dan Prosser.

As Python is scripting language it is easy to make changes in the program without the need to compile and link to make it run. So feel encouraged to make you own changes as you need it

Command line options

Command line options ease the repeated use of the visualizer during optimization sessions. With this no manual input is needed each time the visualizer is re-started in the command shell.

Note:

In a Windows environment you have to set up once that Python scripts accept command line options. Set the following registry keys to point to **your** local Python installation.

```
HKEY_CLASSES_ROOT\Applications\python.exe\shell\open\command --> "C:\Anaconda3\python.exe" "%1" %*

HKEY_CLASSES_ROOT\py_auto_file\shell\open\command            --> "C:\Anaconda3\python.exe" "%1" %*
```

Following two options are available:

`-c` *`my_case_name`*	Sets the optimization case name to visualize to '*my_case_name*' (e.g. SD7003_optimized')
`-o [1,2,3,4]`	Equals to the menu option when starting the visualizer. For example, '-o 3' will immediately start the monitoring of your current optimization case.

Adapted size of display windows

Size and layout of the typical three visualizer windows have been adjusted to make best use of a monitor with a resolution of 1920x1024.

It is easy to adapt size and position of each single window to your needs. Search in the Python file for the string 'setGeometry' to find something like

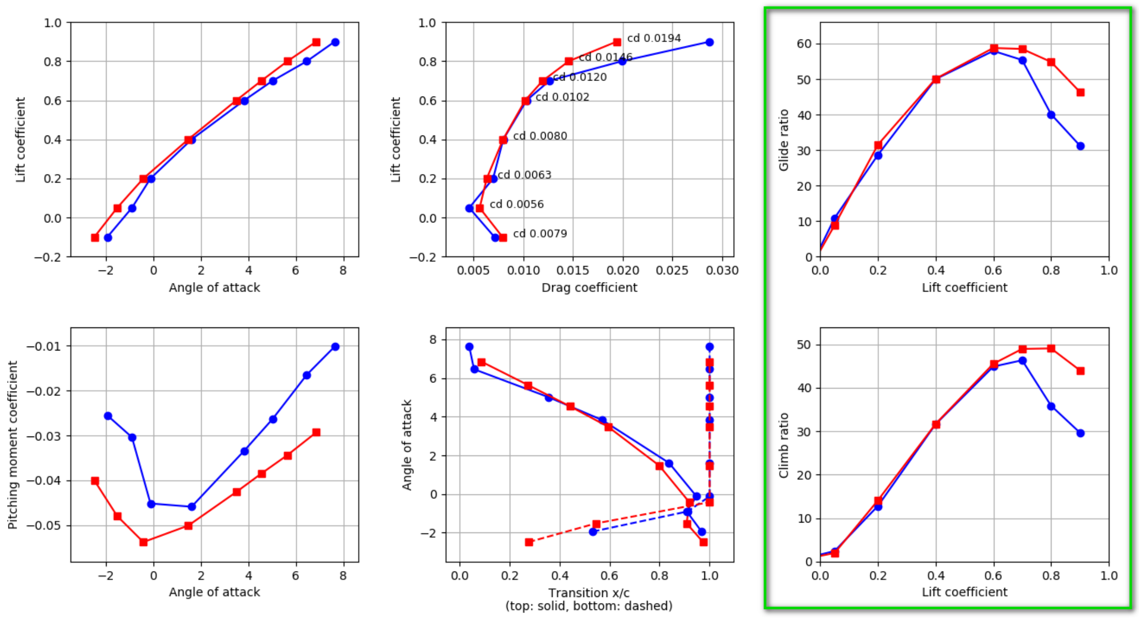
```
plt.get_current_fig_manager().window.setGeometry(100,30,1300,550)
```

The first two numbers are the x,y position and the second two numbers are the size of the output window. Just overwrite the numbers with your desired values, save it and restart the visualizer – ready.

Extensions of polar view

To get an earlier indication whether the optimized airfoil will have the desired aerodynamic properties the polar window of the visualizer is complemented by two sub views for glide ratio and climb ratio.

Especially the glide ration view is helpful for an early detection of optimizations resulting in poor polars.



Additionally the current values of cd at the operating points are displayed in the cl(cd) view.

When doing flap optimization, the actual flap angle at the operating point is taken instead the cd value. The value of the flap angle comes from an extension in the '…_design_polars.dat' file generated by Xoptfoil-JX.

If you don't want these values to be displayed, set the following value to 'False' (as described above)

```
show_flap_angle = True | False                    (default True)

show_cd_value   = True | False                    (default True)
```

Extensions of geometry view

There is a couple of additional information about the seed and the design airfoil which can be displayed during optimization. The additional, optional plots are:

– **delta y**: Is the difference between the y-coordinate of seed and design airfoil (within Xoptfoil the 'y' coordinate is the 'z' coordinate which can be a little confusing). By its nature 'delta y' is equal to the shape function values the optimizer is currently applying. As 'delta y' is scaled by factor 10 the work of the optimizer applying bumps (in case of Hicks Henne shape types) can be watched quite convenient. Display of delta y can be controlled by

```
plot_delta_y = True | False                 (default True)
```

- **curvature / 2nd derivative** : Plots the 2nd derivative or curvature of the airfoil surface. The 2nd derivative is very helpful to access the quality of the airfoil surface as it acts like magnifying glass to detect bumps or deformations. If the plot of the 2nd derivative crosses the x-axis for a certain amount, this equals to the '(max_)curve_reversals' parameter in the Xoptfoil input file. The values of the 2nd derivative are delivered by an extension in the '…_design_coordinates.dat' file generated by Xoptfoil-JX. Display of this plot can be controlled by:

  ```
  plot_2nd_deriv = True | False        (default False)
  ```

- **3rd derivative**: Plots the 3rd derivative of the airfoil surface which helps to detect smaller bumps in the surface – only needed for special cases. Display of this plot can be controlled by:

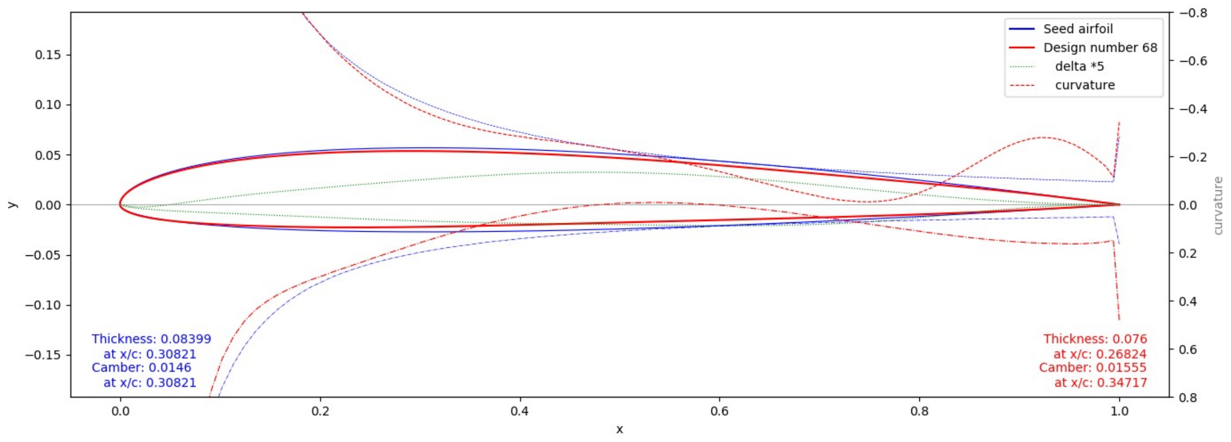
  ```
  plot_3rd_deriv = True | False        (default False)
  ```

- **point of transition**: Shows the point of laminar-turbulent transition on the upper and lower surface for each operating point. Tip: Interesting to experience the correlation between the curvature (2nd derivative) and the point of transition. Display of these points of transition can be switch off:

  ```
  show_transition = True | False       (default False)
  ```

- **matchfoil**: When using Xoptfoils special 'match_foils' mode (see Users Guide) 'delta y' shows the difference between design and match airfoil. The matchfoil mode is automatically activated when its coordinates are detected in '…_design_coordinates.dat' generated by Xoptfoil-JX.

If all plots are activated at the same time the display may become a little confusing. So normally it's better to focus on a certain aspect and set the other plots to 'False'.


Seed airfoil
Design number 68
delta *5
curvature
Thickness: 0.08399
at x/c: 0.30821
Camber: 0.0146
at x/c: 0.30821
Thickness: 0.076
at x/c: 0.26824
Camber: 0.01555
at x/c: 0.34717
y
x
curvature