Xoptfoil-JX v1.52.0

Reference

Jochen Guenzel 2020

This brief reference describes only the enhancements and changes made to the original Xoptfoil.

Information about the capabilities and options of Xoptfoil can be found in the ‘Xoptfoil User Guide’ by Dan Prosser.

Aerodynamic target values

With the new optimization types ‘target-drag’ and ‘target-moment‘ one can define a certain ‘target_value‘ for an operating point. With this the Xoptfoil optimization tries to get to the value of the design variable as close as possible to the defined target value. As with the normal optimization types the gravitational force towards the target value is controlled by the weighting of the operating point.

`&operating_conditions`	common parameters for all operating points
`optimization_type(n)`	= `'target-drag'` Tries to achieve the target_value for the drag (cd) in this operating point = `'target-moment'` Tries to achieve the target_value for the moment (cm) in this operating point = `'target-lift'` Tries to achieve the target_value for the lift (cl) in this operating point. Makes only sense for op_mode ‘spec-cl’. This target type is helpful to achieve a certain lift-slope.
`target_value(n)`	Numerical value which should be achieved. A negative value will be taken to multiply the value of the seed airfoil to get the target value – so “-1.0” would mean: Achieve the value of the seed airfoil.

Example:

```
op_mode(7) = 'spec-cl'
op_point(7) = 0.82
optimization_type(7) = 'target-drag'
target_value(7) = 0.0142
reynolds(7) = 100E+03
weighting(7) = 3
```

Remarks

Aerodynamic target values are especially usefully when an existing airfoil should be “tweaked” at certain operating points. In this case all operating points which should be kept to their existing value, will have a ‘target_value’ of -1, the other operating points will be optimized for example to ‘min_drag’.

Additional operating point options

`&operating_conditions`	common parameters for all operating points
`re_default`	= `xxxxxxx` Default value for reynolds numer which is taken if a reynolds(i) for an operating point is not defined. This is useful if all operating points do have the same reynolds number – see also command line argument '-r' (equals to re_default)
`re_default_as_resqrtcl`	= .true. take `re_default` as the value of re*sqrt(cl) which equals to the Type 2 polar. This is convenient if you have Type 2 based operating points = .false `re_default` is taken as it is (default)
`optimization_type()`	= `'min-lift-slope'` Minimize slope of cl(alpha) curve at operating point. This type can be used to get cl-max at a certain angle of attack. Take a little care when using this option: There should be a helper (dummy) operating point to the left and to the right with a distance of e.g. 0.5 degrees. These helper points are needed to calculate the slope at the current operating point. = `'min-glide-slope'` Minimize slope of glide ration (cl/cd)/cl curve at operating point. This type can be used to get the best glide ration at a certain cl value. Take a little care when using this option: There should be a helper (dummy) operating point to the left and to the right with a delta cl of 0.1. These helper points are needed to calculate the slope at the current operating point.

Additional optimization options

`&optimization options`	high-level optimization parameters
`show_details`	= .true. show more detailed information during optimization (default) = .false do not show (default) This option allows to get more detailed information about the contribution of each operating point to the overall objective function. It's very helpful when tweaking the operating points and their weighting within the objective function. When activated multi threading will be switched off.
`echo_input_parms`	= .true. echo all input parameter = .false suppress echoing (default)
`restart_write_freq`	= 0 if set to '0' no restart files are written at all (default) > 0 standard behavior

Geometric target values

If there is the need that the final airfoil should have a certain thickness it is more natural for the optimization process to define the thickness as an optimization target instead being a constraint.

Another use case is to have control over a sometimes bad deformation of the surface on the upper or lower side of the airfoil especially towards the trailing edge.

`&geometric_targets`	New namelist
`ngeotargets`	Number of geometric targets that shall be applied
`target_type(n)`	Type of geometric target – either 'zBot' z-value at the bottom side ‘zTop’ z-value on the top side ‘Thickness’ final thickness of the airfoil ‘Camber’ final camber of the airfoil
`target_geo(n)`	Numerical value which should be achieved. A negative value will be taken to multiply the value of the seed airfoil to get the target value – so “-1.0” would mean: Achieve the value of the seed airfoil.
`x_Pos(n)`	X-Position along the chord for this geometric target. Only needed for ‘zBot’ and ‘zTop’ (nor for ‘Thickness’ and ‘Camber’)
`weighting_geo(n)`	Weighing of this target within the objective function. Same meaning as ‘weighting(N)’ for an operating point

Example

```
&geometric_targets
   ngeotargets = 2

   target_type(1)   = 'zBot'
   target_geo(1)    = 0.012
   x_Pos(1)         = 0.7
   weighting_geo(1) = 0.5

   target_type(2)   = 'Thickness'
   target_geo(2)    = 0.08
   weighting_geo(2) = 1.0
```

Shape_functions ‘camb-thick’ and ‘camb-thick-plus’

Beside the shape_functions type 'naca' and 'hicks-henne' there is a new type implemented called 'camb-thick'. In this case the seed airfoil will be modified during optimization by the 6 airfoil parameters

- thickness
- x-location of max. thickness
- camber
- x-location of max. camber
- leading edge radius
- leading edge radius mixing distance

In case of shape functions ‘camb-thick-plus’ the top side and the bottom side of the airfoil are treated seperatly when applying the thickness, camber modifications. This doubles the design variables and increases the solution space for optimization.

This shape_functions type is quite convenient if only minor changes should be made to an existing airfoil for example to adopt the airfoil to different Re number situations. It avoids possible geometric problems like bumps.

For the modification of the airfoil the “xfoil’ routines THKCAM and HIPNT are used.

`&optimization options`	high-level optimization and parameterization parameters are

`shape_functions`	`= 'naca'`
	`= 'hicks-henne'`
	`= 'camb-thick'` **new**
	`= 'camb-thick-plus'` **new**

Additional command line options

`-r xxxxxxx`	Allows the definition of a default reynolds number `xxxxxxx` for the operating points. This value will be taken if reynolds() is not defined for an operating point. With this option a single inputs.txt can be used for different polar based optimizations.
`-a airfoil_file`	Filename of the seed airfoil. A ‘airfoil_file’ within inputs.txt will be overwritten.

Generation of polars for the final airfoil

This option allows to generate a polar set in xfoil-format at the end of the optimization. This is especially useful, if further analysis of the airfoil will be done in xflr5 or flow5, where the generated polars can be directly imported via menu function (flow5 also allows to import the polar files via scripting)

The polars will be stored in the subdirectory `<airfoil-name>_polars`.

`&polar_generation`	Namelist to define the polar set
`generate_polars`	`= .true.` Polars as defined in this namelist will be generated at the end of the optimization `= .false` (default) No polars will be generated
`type_of_polar`	`= 1` Type 1 polars (fixed speed) will be generated `= 2` Type 2 polars (fixed lift) will be generated. In this case the specified reynolds number will be taken as RE*sqrt(cl)
`polar_reynolds`	`= xxxxx, yyyyy, zzzzzz, …` List of reynolds numbers for the polars to be generated

`op_mode`	`= 'spec-al'`
	The operating points of the polars specified with `op_point_range` are based on alpha (aoa)
	
	`= 'spec-cl'`
	The operating points of the polars specified with `op_point_range are` based on cl

`op_point_range`	= <start>, <end>, <increment> Defines the series of operarting points of a polar. For `op_mode = 'spec-al'` this could be for example: `-3.0, 10.0, 0.5`

Improving the airfoil surface quality - Smoothing

Activate a simple smoothing algorithm to get rid off micro bumps (spikes) in the surface of the seed airfoil which may lead to curve reversal warnings. Smoothing will be done once at the beginning of the optimization to improve the seed airfoil

`&smoothing_options`	New namelist
`do_smoothing`	= .true. - activate smoothing of the surface for the seed airfoil
`spike_threshold`	Numerical value to detect reversals of the 3nd derivative which are discontinuities of the curvature called spikes. The smaller the value the more spikes will be detected. A good value to start is 2

Remarks on Paneling

At the very beginning of an optimization the seed airfoil will be repaneled using Xfoils PANGEN routine to get a solid and standardized point distribution of the surface. The number of data points (npan) is fixed to 200.

Afterwards the seed airfoil will be ‘normalized’ to have the leading edge at x,y = 0,0 and the trailing edge at x,y = 1,0. In case of a trailing edge gap the mean value of top and bottom surface is applied.

Typically there is no real leading edge point when Xfoils repaneling is done. In this case an additional real leading edge point at 0,0 will be inserted into the data points. Therefore most Xoptfoil generated airfoil .dat files will have 201 data points also 200 points are used for repaneling.

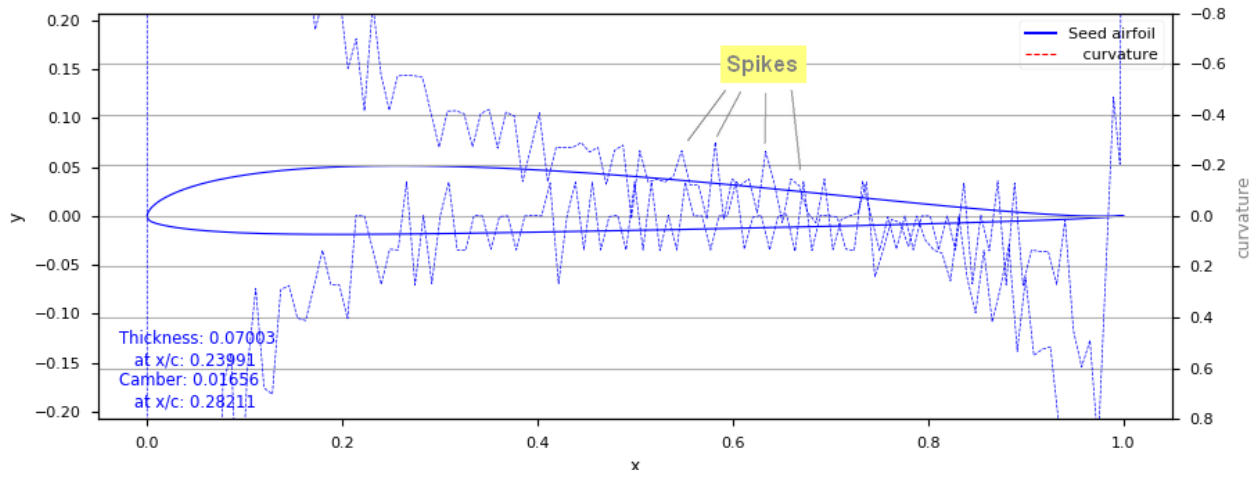
Be aware: Xoptfoil-JX will rotate the airfoil to have a chord which is equal to the x-axis. So the design airfoils could have a little rotated chord line compared to the original airfoil. Be aware if you make aoa (angle of attack) based comparisons e.g. with Xflr5. Use Xfoil-worker with the ‘norm’ or ‘smooth’ option if you want to repanel and normalize your seed airfoil.

`&xfoil_paneling_options`	Namelist for xfoil paneling parameters
`repanel`	`= .true.` each time a new design will be evaluated by xfoil the airfoil will be repaneled in advance. `= .false.` no repaneling during optimization (default) .
`spike_threshold`	Numerical value to detect reversals of the 3nd derivative which are discontinuities of the curvature called spikes. The smaller the value the more spikes will be detected. A good value to start is 2

Remarks on Smoothing

The assessment of the surface is based on the 2nd derivative (which is the curvature) and the 3rd derivative of the airfoils polyline. The visualizer allows to plot the value of this derivatives (see documentation of the visualizer-JX).

The picture shows a typical situation before smoothing is applied. The reason for the discontinuous curvature is typically based on the number of decimals of airfoils .dat file. In this case there were just 5 decimals used to describe the airfoil. These micro spikes have only little or no influence on the aerodynamic characteristics but make it very difficult to detect real curve reversals in this curvature noise.

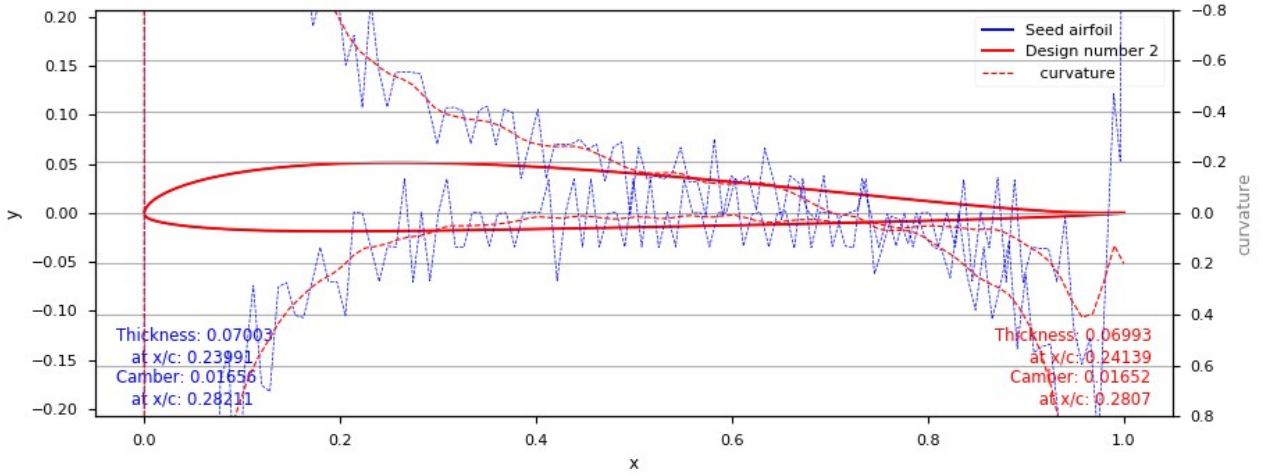


The implemented smoothing algorithm (‘Chaikin Corner Cut’) is quite defensive in changing the airfoils surface. It is able to smooth discontinuities between neighboring data points but it is not able to remove real bumps in the surface of the airfoil.

When smoothing is activated (`do_smoothing = .true.`) a simple assessment of the upper and lower side of the seed airfoil will be displayed showing the position of spkies, highlows and reversals.

```
Before smoothing ...
         Top   7R 34HL 32s   -------------------------LH---LH-LHsLsH-s-sLHs----LHsLsH-Ls-Hss-sssLHsLHsRRRRH-ssRLRR---LH---LHLHLHs--
         Bot  31R 39HL 42s   -R-L-----------------------ss--ssHLs--HLRRss-LRRRRLRR-L-RRRRRRHLs-RRRsRRRsRRsRRLRRs--RRRR--LH---L-s
Ater smoothing ...
         Top   1R  1HL   2s   ------------------------------------------------------------------------------R-----------------L-s-
         Bot   1R  2HL   4s   -R-s--------------------------------------------------------------------------H---------s-------Ls-s
```

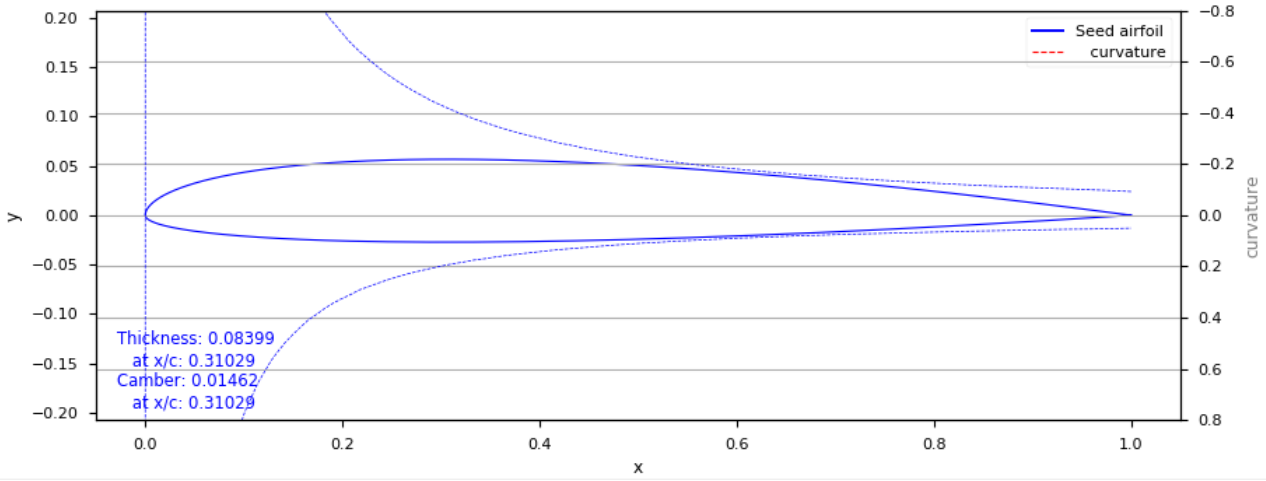
Xoptfoil_Visualizer-JX allows a much more detailed presentation of the curvature situation of the airfoil.



The new airfoil will be stored with 7 decimals of the x,y coordinates in the .dat file. So the quality and precision will be kept if used for further optimization run.

Unless the optimization use case is to tweak or to improve an existing airfoil, it is a good advice to take an arbitrary airfoil with a real smooth surface than taking a similar airfoil with a bad surface.

This is an example of a well suited seed airfoil as a starting point for intensive Hicks Henne based optimizations (see also in the ‘examples’ folder for additional seed airfoil)


Seed airfoil
curvature
Thickness: 0.08399
at x/c: 0.31029
Camber: 0.01462
at x/c: 0.31029
y
x
curvature
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
0.0
0.2
0.4
0.6
0.8
1.0
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8

Improving the airfoil surface quality – Bump prevention

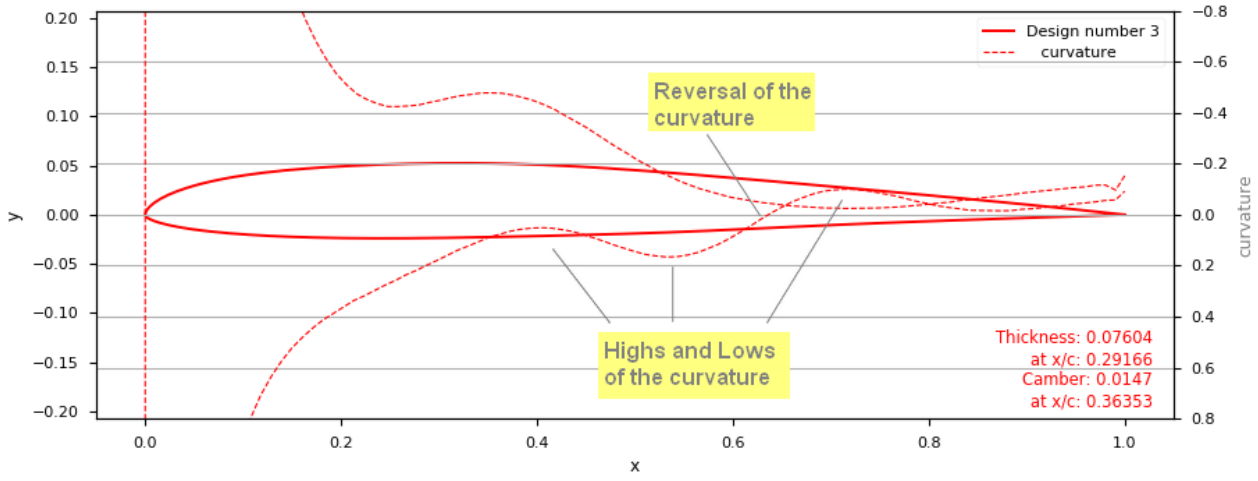
Additional constraints which help to avoid bumps of the surface and artefacts at the trailing edge.

`&constraints`	Namelist for geometric constraints
`highlow_treshold`	Numerical value to detect highs and lows of the 2nd derivative (equals curvature) of the surface curve. The number of high and lows are checked against the constraints `max_curv_highlow_top` and `max_curv_highlow_bot`. A good value to start is 0.05. Default value is 1.0 Only active if '`check_curvature = .true.`'
`max_curv_highlow_top` `max_curv_highlow_bot`	The maximum number of 'highs and lows' of the 2nd derivative (equals curvature) of the surface curve. This constraint can be used to avoid bumps on the surface. Default value is 1. Only active if '`check_curvature = .true.`'
`max_te_curvature`	Maximum value of curvature at trailing edge In the current Hicks Henne shape functions implementation, the last panel is forced to become the original trailing edge value which may lead to a thick trailing edge area with steep last panel(s). Have a look at the curvature plot in the visualizer to assess the situation. A value of 0.1 – 1.0 leads to well formed trailing edges. Default value is 1.0 Only active if '`check_curvature = .true.`'

Remarks

When `check_curvature = .true.` the build in curvature check becomes active. The minimum check is taking care of unwanted reversals of the curvature (equals to the 2nd derivative of the surface polyline) to identify 'bumps' of the surface. The sensitivity of the reversal detection is controlled by `curv_threshold`.

As curve reversals only indicate 'bigger' bumps an additional check is available to achieve a smooth, bump free surface of the airfoil. This check counts the highs and lows of the curvature on the upper and lower side and compares it to `max_curv_highlow` as the constraint.



The sensitivity of the highlow-detection is set by the parameter `highlow_treshold`.

Trailing edge artefact

Mainly because of the current implementation of Hicks Henne shape functions it may occur that the airfoil gets thickened close to the trailing edge with a steep last panel going down to the trailing edge point.

This artefact can be avoided with the parameter `max_te_curvature` which defines the maximum allowed curvature close to the trailing edge.