Xoptfoil-JX is a modified version of Xoptfoil version 1.11.1 – the airfoil optimizer by Daniel Prosser

Xoptfoil-JX v1.50.2

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 for operating points

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

op_mode(4) = 'spec-al'
op_point(4) = 6.0
optimization_type(4) = 'target-lift'
target_value(4) = -1.0
reynolds(4) = 400E+03
```

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.

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

Additional constraints

`&constraints`	Namelist for geometric constraints
`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 0.5 – 1.0 leads to well formed trailing edges.

New shape_functions ‘camb-thick’

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

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 two ‘xfoil’ routines THKCAM and HIPNT are used.

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

`shape_functions`	`= 'naca'`
	`= 'hicks-henne'`
	`= 'camb-thick'` **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`	Define the filename of the seed airfoil. A ‘airfoil_file’ within inputs.txt will be overwritten.

Additional optimization options

`&optimization options`	high-level optimization parameters
`show_details`	= .true. show more detailed information during optimization (default) = .false do not show 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
`echo_input_parms`	= .true. echo all input parameter (default) = .false suppress echoing
`restart_write_freq`	= 0 if set to ‘0’ no restart files are written at all > 0 standard bevaiour

Generation of polars of the final, optimized 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`

Airfoils surface quality and smoothing

Activate a simple smoothing algorithm to get rid off micro waves (spikes) in the surface of the (seed) airfoil.

`&smoothing_options`	New namelist
`do_smoothing`	= .true. - activate smoothing of the surface
`show_smoothing`	= .true. - show additional informations during the optimization process
`highlow_treshold`	Numerical value to detect highs and lows of the 2nd derivative of the surface curve. The number of high and lows is taken into account to assess the quality of the surface. A good value to start is 0.05
`spike_threshold`	Numerical value to detect highs and lows of the 3nd derivative called spikes. The number of spikes is taken into account to assess the quality of the surface. A good value to start is 2
`weighting_smoothing`	Out of the number of highs, lows and spikes a “perturbation value” is calculated which can be an indication for the surface quality: ‘0’ is excellent, a value greater ‘1’ indicate some problems… If this value should be part of the objective function ‘weighting_smoothing’ should have a value > 0.0.

Example

```
&smoothing_options

  do_smoothing     = .true.
  show_smoothing   = .true.
  highlow_treshold = 0.04
  spike_threshold  = 1.5
  weighting_smoothing = 0.1
```