User's Manual for polaraRun Python Script

version 2.06

Abstract

This document describes the use of a Python script that is designated to produce a polar-sweep computation in a single command. The script uses the SUmb code in a Linux system. The script can be applied effectively only with a good quality mesh and a suitable SUmb input file. Naturally the use of this script does not guarantee convergence of the solution for each polar-sweep point. This is the responsibility of the qualified user. This document will be updated as the script evolves.

1. Concept

The script performs sweeps in angle of attack* (α) or roll angle (ϕ). Reading the required sweep type and a list of angles from a control file, it produces a set of SUmb input files, with the relevant flow directions. Those files are made out of an SUmb input file that is supplied by the user. Through a loop, each polar-sweep point is computed, using the previous sweep point as initial conditions. After each polar-sweep point computation is concluded, a CBD (Components Break Down) file is produced+.

Once **the whole set of computations** is completed, the script **polaraPlotPrep.py** is automatically initiated. It reads the CBD output files and groups the results together for files of main coefficients versus the sweep angle. Furthermore, following user specifications of required surfaces names, secondary coefficients polar-sweep files are prepared for them.

For older versions of SUmb (older than version *ADL01E02*) or for ultra-conservative users who stick to old practices, the SUmb can be applied in a post-process mode to produce component-break-down printout. This process is applied only if -c was added in the application of the script (e.g. *polararun.py -f junk.dat -c*) . If CBD post-processing was not run during the computation sweep, it can be applied as a standalone using script CBDrun.py (no parameters are needed, it reads its input

* The user also designates the *pitch axis*, which is the AOA- normal axis. Thus, selecting the axis pointing upward or downward, the polar-sweep can be actually in β, the side-slip angle.

\+ This is the case only for version *ADL01E02* and higher of SUmb. Using older versions, the user should explicitly generate the CBD output files, using CBDrun.py script, described below

from a hidden file produced by the computation sweep). This separation between computation and post-processing sweeps was added since there have been observed certain cases were the post-processing was hanged-up. Till this issue is resolved, it is better not to let it disrupt the computation sweep[#].

For each sweep point the volume and surface solution files, the convergence printout and the CBD printout are conserved. All output files are designated by the angle of attack or roll angle (such as *Sol.a32.cgns* or *Sol.phi35.cgns*). That way, if a sweep run was not completed, the missing cases can be run at the same directory, without rerunning or overwriting the results that were completed already.

Three **types of polar-sweeps** are now possible:

1. Polar sweep in angle-of-attack α for a given roll angle ϕ
2. Polar sweep in α for a given side slip angle β
3. Polar sweep in roll angle ϕ for a given angle-of-attack α

Formulation:

The Sumb input includes velocity vector definition, instead of angle (of attack, roll or side-slip) in order to avoid ambiguities in the definitions of angles. In the polar-sweep applications, definitions of angles are back, live and kicking. In order to reduce the number of errors in the definitions of angles, the relations between the velocity vector in input (designated by $\vec{V}_\infty = dv_1\hat{i} + dv_2\hat{j} + dv_3\hat{k}$) and the various angles is listed here. Note that the *pitch axis* definition is also part of the polar-sweep input, and it influences the formulation:

<u>Polar-sweep type = 1,3</u>

if *Pitch Axis = Z* (X in flow direction, Y pointing up, Z to the left in pilot-view)

$dv1 = 1.0\ ;\ dv2 = \tan(\alpha)\cos(\phi)\ ;\ dv3 = \tan(\alpha)\sin(\phi)$

if *Pitch Axis = Y* (X in flow direction, Y pointing right in pilot-view, Z pointing up)

Until proved differently, it appears that the hangup problem of CBD run was resolved in version *ADL01E02*

$dv1 = 1.0\ ;\ dv2 = \tan(\alpha)\sin(\phi)\ ;\ dv3 = \tan(\alpha)\cos(\phi)$

Polar-sweep type = 2

if *Pitch Axis = Z* (X in flow direction, Y pointing up, Z to the left in pilot-view)

$dv1 = \cos(\beta)\ ;\ dv2 = \tan(\alpha)\cos(\beta)\ ;\ dv3 = \sin(\beta)$

if P*itch Axis = Y* (X in flow direction, Y pointing right in pilot-view, Z pointing up)

$dv1 = \cos(\beta)\ ;\ dv2 = \sin(\beta)\ ;\ dv3 = \tan(\alpha)\cos(\beta)$

Limitations and warnings:

1. No mixed type polar-sweep
2. One can specify either ϕ or β . Not both
3. For *polar-sweep-type*=1,3, with *pitch-axis=Z*, proper right-hand roll definition would have been right-wing up (right in pilot view). However, in the common aircraft systems, positive roll is defined as right-wing down (since the axes are defined differently). Thus, here the formulation for roll definition is also right-wing down, though this formulation is inconsistent with the axis definition.

2. New Features

version 2.06 SUmb input parameter, controlling the contribution of components (families) to forces and moments (yes/no) is now parsed by the polar-sweep script. The script removes the contribution of the components (families) that were marked in the input file in the following manner:

Boundary family DNOSE: Contribute to forces: no

It is possible to assign several components (families) as non-contributing. Each component (family) in a different line of the same format. Also note that contribution assignment within the SUmb code is still not fully functional (active only for explicit user boundary families with wall BC, not for direct wall BC). This affects only the printout of forces and moments during convergence. This limitation does not apply for the polar-sweep scripts: Contribution of any wall-type BC can be controlled.

version 2.05: Convergence quality grade is included in each line of the main polar-sweep output. Definition of grades is presented in the SUMB input description:

Grade	Convergence type
10	Relative residual smaller than *convergence criterion*
6	Coefficients are uniform for 1x*windowSize* iterations
4	Coefficients are uniform for 10xwindowSize
2	Coefficients are uniform for 100xwindowSize iterations;
0	Solution stopped without reaching any convergence criterion

convergence criterion and *windowSize* are input parameters described in SUMB input description. Matlab script *polaraPlot.m* reads the polar-sweep output and set point colors according to convergence quality. This script can be run even before the polar-sweep is completed. It invokes *polaraPlotPrep.py* to update the polar-sweep output file, before plotting it.

version 2.03: Improve convergence/resolution

For a given polar-sweep, resolution (in α or ϕ) or convergence can be improved by adding, interactively a list of angles. This option is activated by

polaraRun.py ..... -a $\alpha 1,\alpha 2,\alpha 3$

3. Prerequisites

a. Python version 2.3 or newer.

b. A working MPI

c. A control file which defines the sweep parameters

d. An SUmb input file which fits the required flow field

e. A proper CGNS format multi-block mesh

f. A machinefile, with list of the relevant compute-nodes

g. Add /usr/local/mb directory to Linux path. Can be done by adding the following line to *.bashrc*

export PATH=$PATH:/usr/local/mb

h. mpd daemons in the air, as required by MPI

Note a change here: No need to create a local link for the SUmb code. This link exists in the */usr/local/mb* directory and is used by the scripts

4. Features

Feature	Applied	comments
Run a polar sweep for α	Yes (v 1.01)	
Run a polar sweep for ϕ	Yes (v 1.02)	
Produce CBD output	Yes (v 1.01)	
Group CBD output to a sweep file of main contributions	Yes (v 1.03)	
Group CBD output to a sweep file of specific components contribution	Yes (v 1.05)	The user supply list of relevant components, using families' names defined in the mesh
Run a polar sweep in α for a given β	Yes (v 1.06)	
Select ADL/BDF versions	Yes (v 1.07)	
Polar sweep of a single case	Yes (v 1.08)	
Standard sweep definition in headers	Yes (v 1.08)	Intended for standard CFD output procedure
Fractions possible in angles list (0.5,-0.82)	Yes (v 1.09)	Possible in alpha, phi or beta
Automatic CBDOUT production	Yes (v 2.01)	CBDrun.py still available but not needed anymore
Use solver previous version	Yes (v 2.01)	Use parameter -o
Improve convergence/resolution	Yes (v 2.03)	Use -a [list of angles]
Convergence quality grade for each polar-sweep line	Yes (v 2.05)	Use *polarPlot.m* to view results and convergence quality
Control over contribution of families to forces and moments	Yes(v 2.06)	Control by SUmb input file

5. How to use

List of scripts:

polaraRun.py : Main script; Entering the command

polaraRun.py -h

Provides the following help screen:

```
Usage: polaraRun.py [options] (enter polaraRun.py -h for a list of valid otions)

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -f FILE, --file=FILE  reads polara control parameters from FILE
                        (default:polaraCtrl.dat)
  -v, --verbose         Print debug messages (def=false)
  -n NPROCS, --nprocs=NPROCS
                        Number of MPI procs (def=32)
  -m MACHINEFILE, --machinefile=MACHINEFILE
                        machinefile name (default: machinefile in
                        current directory)
  -e ENV, --env=ENV     MPI parameters, such as device (def: rdma,
                        local :  ignored)
  -c, --cbd             Perform Component Break Down (def=false)
  -o, --old             Use older (previous) version of solver (def=false)
  -a ADDRUNSTR, --add=ADDRUNSTR
                        List of additional/Rerun angles to be
                        added/to improve convergence of already
                        completed list (a1,a2,...)
```

with -c it activates CBDrun.py; At the end of polar-sweep it activates *polaraPlotPep.py*; uses user's parameters control file

CBDru.py: Compute component break down for the polar-sweep. No input parameters needed. For version 2.01 (version *ADL01E02* of SUmb) and higher, it is not needed anymore, since CBDOUT

files are automatically produced at the end of each run. It was observed that sometimes the CBD run get hang-up, and can work with a lower number of processors. For such cases, CBDrun.py can be applied with *-n* parameter (use *-h* for online help), specifying. number of processors. The CBD computation will be performed using the minimum of this parameter and the number of processors used in the main polar-sweep computation. This hang-up problem seems to be resolved in version *ADL01E02* of SUmb (until proven differently).

polaraPlotPep.py: Group the results of the polar sweep onto single files of the sweep (main and secondary). No input parameters needed.

A new feature at version 2.01: While computation over the polar-sweep are performed, *polaraPlotPep.py* can be applied. The scripts reads the incomplete list of CBD output files, and creates the sweep sum-up files. That way the user can monitor the advance of computation and recognize peculiar results before everything is completed. In cases of application over incomplete sweep-list, an warning message is issued.

A new feature in version 2.03: Improve convergence/resolution

For a given polar-sweep, resolution (in α or ϕ) can be improved by adding, interactively a list of angles. This option is activated by

polaraRun.py ..... -a α1,α2,α3

A list of angles (either α or ϕ) separated by commas (without space !!!).

For cases where an angle (or several of them) is equal to one (or several) of the original angles in the polar-sweep (close means that they differ by less than 0.1 degrees) this is a continuation run restarting from the results of the same angle (designated for convergence improvement, where it needed). In cases of convergence improvement run, the user might find it appropriate to modify some numerical control parameters in the input base file. The polar-sweep output file will include the complete sorted list of angles. The control file is updated, and a copy of the original control file is saved under [control file].bck.

Note: It is possible to add a list in an ascending or descending order. However, the list is sorted in an ascending order and the smallest added angle will be first computed, even if it was not first on the

added list.

User's control file:

Prepare a control file. Any name goes. Its structure is equivalent to the SUmb input file:

designates a comment, to be ignores by parser

: designates a data line with the structure keyword : value

Order of input has no effect. Case (upper/lower) has no effect.

List of keywords:

Keyword	data	Applied	Comment
input base file	Input file name	yes	Base input can designate a restart file for first point
Pitch axis	Y or Z	yes	Only these value are permitted
angles of attack	alpha1, alpha2,....,alphan	yes	, (comma) should separate between the angles
roll angles	phi1, phi2, ...,phin	yes	Same as above
Side slip angle	beta	yes	Either beta or phi. Not both
Base surface	Base surface name as it appears in mesh file	yes	If not specifies, cxbase=0
components polara required	[component1 name],[ ],...	yes	, (comma) should separate between the name

The script is applied in a Linux manner. Use polaraRun -h for available parameters

A typical application looks as follows:

polaraRun.py -f polaraCRTL.dat -n 1012 -m machinefile

Polar sweep can be performed either in α (β) or in roll (ϕ), but not in a mixed manner. Thus a single value at the *angles of attack* can be introduced together with a list at *roll angles* line to produce a polar-sweep in ϕ, using a given α (β) . Or vice-versa, single value at *roll angles* line with a list of values at the *angles of attack* line means a polar sweep in α (β) for a given roll (ϕ).

Not all the lines must appear in the file. The defaults that are used if a line is missing are the

following:

α = 0 ϕ = 0 β = 0 *pitch axis* = z

Warning: Do not use command script (creating typescript) when applying. The polaraRun script invokes these commands.

An example of a control file is also located at */usr/local/mb* directory.

Code versions

New at version 1.07: The scripts can run either the old mb_bdf (version 180) or the new mbadl code. The user can control this option by definition of environment variable USE_ADL. Value Y means mbadl. Any other value (or no definition) means mb_bdf (version 180). Easy way to define is adding the following line to .bashrc file:

```
export  USE_ADL=Y
```

Note: upper case Y counts here.

New at version 2.01: using SUmb version ADL01E02 separated CBDrun is not needed anymore. *polaraPlotPep.py* is automatically activated at the end of a polar-sweep.

6. Standard output text

For the use of the automated, standard CFD-output generator, which is on preparation by now, the polar-sweep output files (both primary and secondary) include a comment line (starting by % sign) of te following form:

% polarSweepType : [1/2/3] sweepAngle : [alpha/phi] singleAngle : [phi/beta/alpha]= [X] deg

In square brackets appear possible values. [X] represents the single-angle value.