A program to compute compressor blade geometries from multiple-circular-arc parameters with sweep and lean (v1.0)

Elsa Piollet (elsa.piollet@polymtl.ca) and Alain Batailly (alain.batailly@polymtl.ca)

Department of Mechanical Engineering, École Polytechnique de Montréal, P.O. Box 6079, Succ. Centre-Ville, Montréal, Québec, Canada H3C 3A7
elsa.piollet@polymtl.ca

May, 2019

Contents

1 Introduction

The aim of this program is to build compressor blade geometries from a set of parameters. Parameterizing a blade geometry allows exploring different design strategies by varying specific parameters. Specifically, this program was developed as part of a study on blade design for robustness to contact interactions [1]. This program is made publicly available along with the input parameters and output geometries to serve as a basis for comparative work between different research teams on blade dynamics.

The source code should be cited as follows: Piollet, E. and Batailly, A. (2019) "A program to compute compressor blade geometries from multiple-circular-arc parameters with sweep and lean (v1.0) [source code]", available at hal.archives-ouvertes

2 Blade parameterization

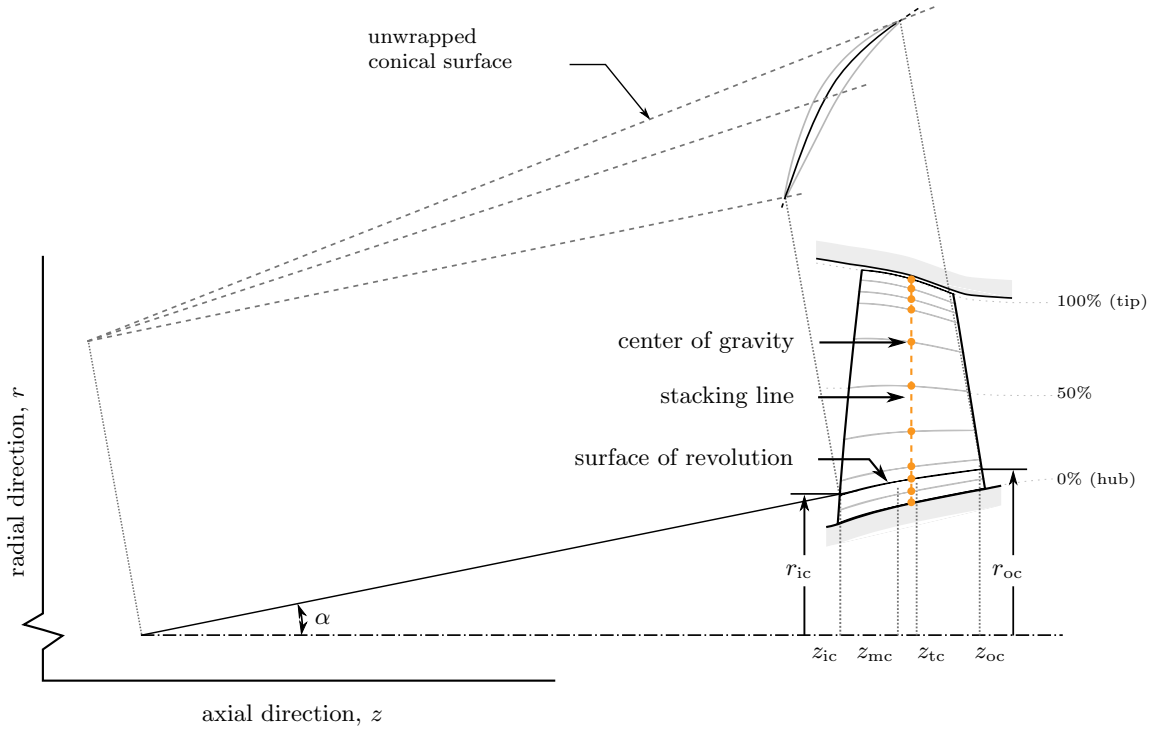
2.1 Overview

The parameterization is based on the original parameterization of the NASA rotor 37 [2], a compressor blade widely used as a test case in aerodynamic studies, and which was used as the reference case in [1]. The general principle of this parameterization is presented in figure 1. The blade is described by N profiles stacked above one another from the hub (0% of blade span) to the tip of the blade (100% of blade span). These profiles are defined on conical surfaces which approximate the stream flow surfaces at different positions along the blade span. The profiles are multiple-circular-arc (MCA) profiles: on the unwrapped conical surface, the chordline, the suction surface and the pressure surface of each profile are composed of two segments of constant turning rate. The two segments are joined at the transition point. Parameters are given at the inlet point, the transition point, the point of maximal thickness and the outlet point. Details of this parameterization, including a thorough derivation of corresponding equations, can be found in a NASA report from 1969 [3], along with a Fortran IV program (see section 4.2.1). The profiles can be stacked vertically along a straight line, but the stacking law can also include sweep and lean, in which case the profiles are translated and/or rotated with respect to one another. The original parameterization described in [3] considers constant sweep and lean along the blade span; more complex blade geometries can be obtained by applying varying sweep and lean along the blade span.

2.2 Multiple-circular-arc parameters

Based on the algorithm proposed by Crouse [3], the following 11 parameters are required for each profile, as described in figure 1:

- radius (r_{ic}) and local blade angle (κ_{ic}) for the blade centerline at the inlet/leading edge,
- radius (r_{oc}) and local blade angle (κ_{oc}) for the blade centerline at the outlet/trailing edge,
- local blade angle for the blade centerline at the transition point (κ_{tc}),
- thickness at the inlet, maximum thickness point and outlet (t_{i}, t_{m} and t_{o}),
- axial (z) position of the maximum thickness point, transition point and outlet with respect to the inlet (z_{mc}-z_{ic}, z_{tc}-z_{ic} and z_{oc}-z_{ic}).



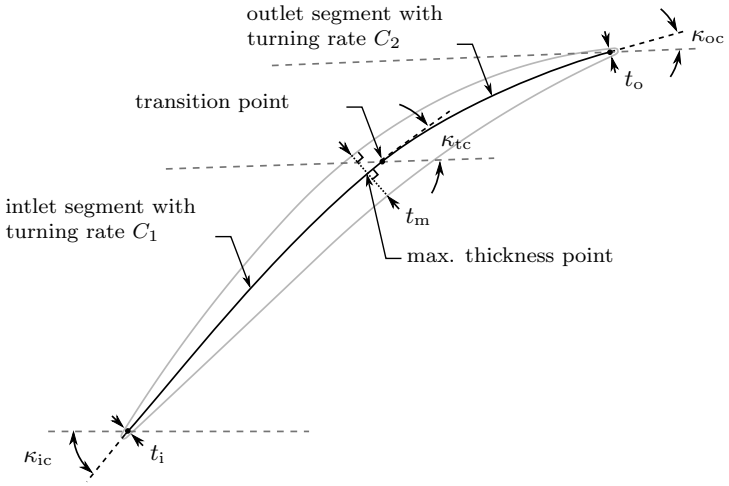


Figure 1: blade parameterization, adapted from [4, 5]

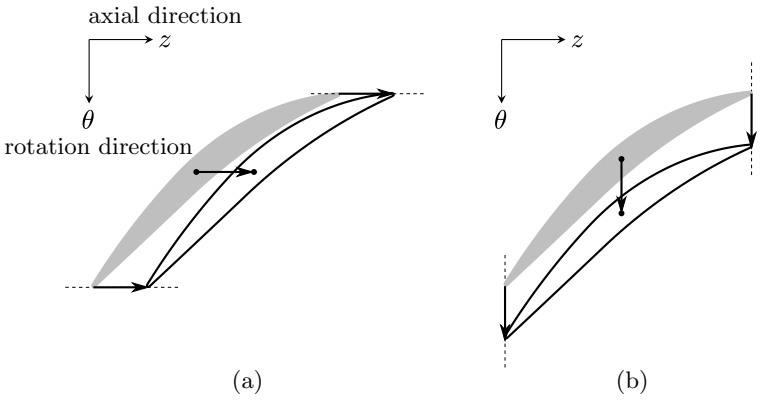


Figure 2: sweep and lean definitions: (a) axial sweep, and (b) tangential lean

More parameters are generally provided in NASA reports, but any supplementary parameter can be computed from the 11 parameters. **For consistency with parameters found in NASA reports, the parameters' units are in degrees and in centimeters.**

2.3 Sweep and lean

While the profiles of the original rotor 37 are stacked vertically along a straight line, recent studies have shown that modifying the stacking line is an interesting design avenue both from an aerodynamic and a structural point of view [6, 7]. Therefore, the present program allows applying varying sweep and lean in the blade span to modify the blade geometry. Sweep and lean definitions used are illustrated in figure 2: applying sweep to a profile corresponds to moving it in the z direction (in the axial direction), while applying lean corresponds to moving it in the θ direction (in the direction of rotation). Sweep is positive in the downstream direction (backward sweep); lean is positive in the direction of rotation (forward lean). In practice, the stacking line is described by choosing values for sweep and lean at four span values, and interpolating these values with a cubic B-spline to all profile levels. The values can be expressed either as a displacement or as an angle; the angle can be defined with the hub as the reference, or with the rotor axis as the reference. An example is presented in figure 3.

3 Overview of the program

3.1 Organization and use of the program

The code is organized in three modules, which can all be launched from main.py or used individually:

parameter_smoothing: smoothing of the MCA parameters along the blade span, in order to obtain a smoother geometry,

profile_coordinates: computation of cartesian point coordinates for each profile from the profile parameters and the sweep and lean, therefore describing the whole blade geometry as a 3D point cloud,

blade_cad: generation of a CAD (Computer-Aided-Design) file and a finite element mesh from the profile point coordinates with Salomé [8].

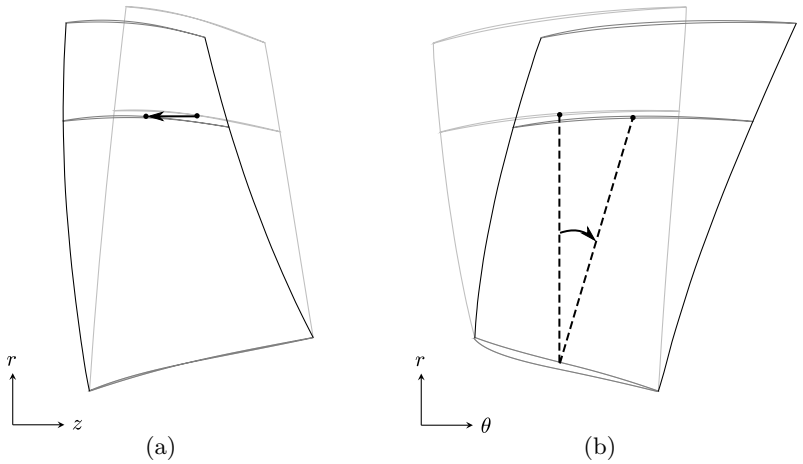


Figure 3: sweep and lean examples: (a) negative sweep as a displacement, and (b) positive lean as an angle with the hub as the reference

The general principle of the program is described in figure 4. All parameters should be entered in the file `parameters.ini` under the header corresponding to the module of interest. A typical use would follow these steps:

1. write MCA parameters for each profile in a `.csv` file in the folder `input/profiles`,
2. write sweep and lean parameters in two `.json` files in the folder `input/angles` with file names `blade_name_sweep.json` and `blade_name_lean.json`, where `blade_name` is the desired blade name,
3. write parameters in `parameters.ini`:
   - section `[main]` allows chosing which module should be run (depending on whether the user wants parameter smoothing or not, or has access to Salomé or not for example),
   - sections `[parameter_smoothing]`, `[profile_coordinates]` and `[blade_cad]` include parameters for the modules of the same name: see sections 4.1, 4.2 and 4.3 for details on available parameters,
4. run `main.py`.

3.2 Pre-requisites

3.2.1 Python

The code was developed in Python 2.7, and was tested with the following module versions:

- numpy 1.16.2
- scipy 1.2.1
- json 2.0.9
- matplotlib 2.1.1

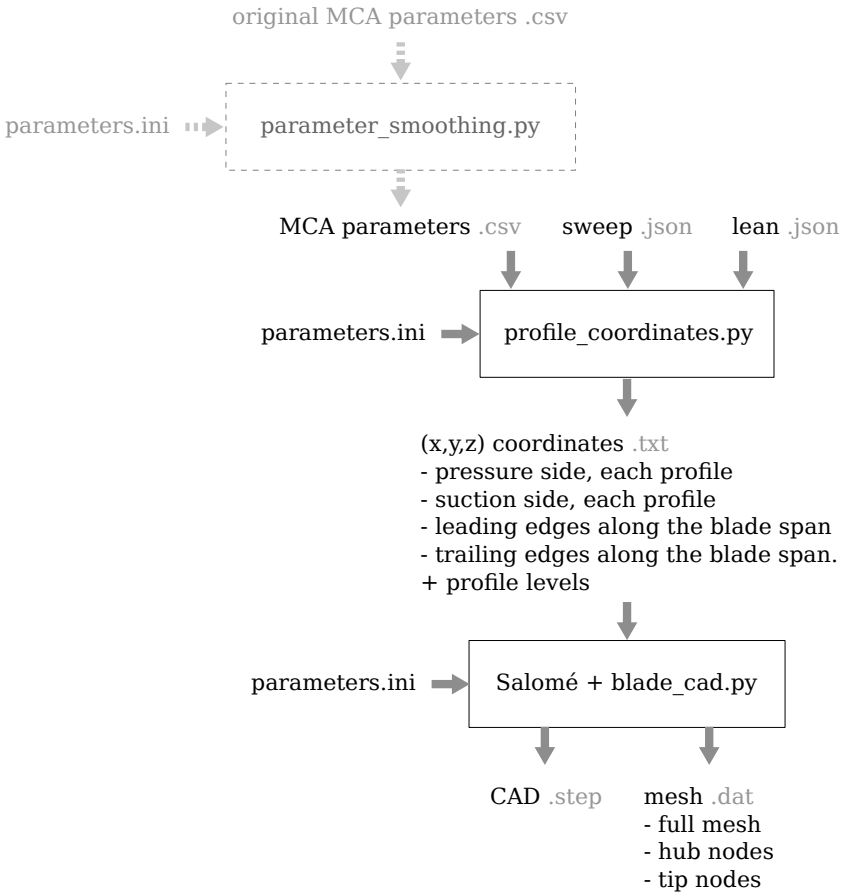


Figure 4: general principle of the program

Other modules used include: os, ConfigParser and inspect. Specific Salomé modules are also used, see below.

3.2.2 Salomé

The script blade_cad.py is intended to be run from Salomé, and cannot be run directly from Python. Salomé is an open-source pre- and post-processing platform for numerical simulation. More information can be found in [8] and on the website www.salome-platform.org. The script blade_cad.py was successfully tested in Salomé 8.2.0, 8.3.0 and 8.4.0. Salomé-related Python modules used include: salome, salome_notebook and SMESH.

4 Module details

4.1 Smoothing module - parameter_smoothing.py

The parameters for the smoothing script are under the section `[parameter_smoothing]` in `parameters.ini`. Available parameters and example values are:

```
profile_folder = input/profiles
original_filename = rotor37_original.csv
smoothed_filename = rotor37_smoothed.csv
```

The script loads the MCA parameters from the file profile_folder/original_filename, and smoothes the parameters along the blade span with a cubic polynomial. In practice, the polynomial fitting is applied for each group of points: inlet point, transition point, maximum thickness point and outlet point. For each group, the corresponding parameters (κ, z, t) are smoothed with respect to the radial position r. Note that the path of profile_folder is relative to the path of the executed python file, which is main.py generally (the module can also be executed directly as a script, in which case the folder path must be changed accordingly).

4.2 Coordinate module - profile_coordinates.py

4.2.1 Relationship with the original report and Fortran program

This module is based on the equations presented in [3]. The same report also provides a Fortran IV computer program to compute the blade geometries from profile parameters, which served as a basis for the present program. However, because this original program is 50 years old, it was not simply translated to Python, but also adapted: for example, some subroutines were replaced with functions from modern libraries such as scipy in order to improve the overall efficiency of the program. In the present program, the names of functions and variables are also more explicit for a better legibility. The general organization of the main function is still consistent with the original report and Fortran program, and comments in the sources give the names of the original Fortran functions and subroutines, and references to the equations in the report. The main modification of the original code is the possibility to apply sweep and lean angles to each cross-section, which modifies the general geometry of the blade. The original equations only allowed the inclusion of constant lean or sweep angles in the blade geometry.

The general procedure described in [3] and implemented in the module follows these steps:

- profile computation:

  - centerline: for the four main points (inlet, transition point, maximum thickness point, outlet), computation of (R, ϵ) coordinates, and computation of constants C, ζ for the two segments of the centerline,
  - suction surface and pressure surface: computation of the same quantities as the centerline, along with the slope κ at the main points,

- stacking procedure:
  - step 1: initial positioning of the profiles along the stacking line, with computation of the (R, θ, z) coordinates of the stacking points,
  - iterations on the following steps for alignment along the stacking line:
    * step 2: calculation of stacking points relative to a common reference (the stacking point at the hub), taking into account lean and sweep,
    * step 3: calculation of (x, y, z) coordinates for the main points from (R, ϵ) coordinates,
    * step 4: calculation of the coordinates of horizontal sections of the blade (sections with constant x),
    * step 5: calculation of the centers of area of the horizontal blade sections,
    * step 6: calculation of the new profile stacking points, as the intersections between the profiles and the curve of the centers of area of the horizontal sections, and evaluation of the variation in position of the stacking points; if repositioning is necessary, the procedure gets back to step 2,

- computation of the final (x, y, z) coordinates for the profiles with the converged stacking points.

4.2.2 Coordinate systems

The computed coordinates stored in the text files generated by the module are cartesian coordinates (x, y, z), where the z direction corresponds the direction of the rotation axis, and the x direction corresponds to the radial direction of the blade. Consistently with the input data, the coordinates are expressed **in centimeters**.

4.2.3 Input

Parameters The parameters for the coordinate module are under the section `[profile_coordinates]` in `parameters.ini`. Available parameters and example values are:

```
blade_name = blade_ref
profile_folder = input/profiles
profile_filename = rotor37_smoothed.csv
sweep_lean = true
sweep_lean_folder = input/sweep_lean
output_folder = output/coordinates
display_fig = false
```

Parameter "sweep_lean_folder" can be omitted if parameter "sweep_lean" is false, in which case no lean or sweep is applied to the geometry, and the stacking line is straight and vertical.

CSV file: profile parameters The MCA parameters should be provided in a CSV (comma-separated values) file in the location `profile_folder/profile_filename` provided in `parameters.ini`. Parameters should be organized as follows:

- the first line is not read, and it can contain a commented header with each parameter name for example,
- a new line should be created for each profile, going from the hub to the tip or from the tip to the hub (both orders are accepted, but profiles should be sorted along the blade span),
- for each profile, the following parameters should be provided, in this order: profile level as a percentage of the blade span, r_{ic}, r_{oc}, κ_{ic}, κ_{tc}, κ_{oc}, t_{i}, t_{m}, t_{o}, $z_{\mathrm{mc}} - z_{\mathrm{ic}}$, $z_{\mathrm{tc}} - z_{\mathrm{ic}}$, $z_{\mathrm{oc}} - z_{\mathrm{ic}}$,
- percentage of the blade span should be 0 at the hub and 100 at the tip (a correction may need to be applied to the data from NASA reports to follow this rule).

An example of CSV file content (here, for the original rotor 37 [2]:

```
# percent span, ric, roc, kic, ktc, koc, ti, tm, to, zmc-zic, ztc-zic, zoc-zic
100,25.23,24.506,62.53,62.83,49.98,0.025,0.175,0.025,1.717,1.686,2.659
95,24.935,24.218,61.66,61.86,49.07,0.026,0.186,0.026,1.725,1.707,2.759
90,24.597,23.929,60.76,60.86,48.18,0.028,0.199,0.028,1.731,1.719,2.86
85,24.254,23.641,60.07,60.09,47.34,0.029,0.211,0.029,1.73,1.706,2.946
70,23.211,22.775,58.48,58.09,44.22,0.032,0.25,0.033,1.759,1.628,3.178
50,21.761,21.622,56.53,54.49,38.87,0.037,0.303,0.038,1.847,1.611,3.505
30,20.246,20.468,54.24,50.48,32.37,0.042,0.36,0.043,1.893,1.597,3.839
15,19.03,19.603,52.67,47.6,25.28,0.047,0.407,0.047,1.931,1.654,4.074
10,18.603,19.314,52.37,46.87,22.68,0.048,0.425,0.049,1.936,1.612,4.15
5,18.161,19.026,52.18,46.39,19.75,0.05,0.443,0.05,1.936,1.57,4.22
0,17.78,18.738,52.04,46.03,16.75,0.051,0.458,0.051,1.933,1.53,4.283
```

JSON files: sweep and lean Sweep and lean can be provided as displacements (in centimeters) or as angles (in degrees). Angles can be measured from the rotor axis or from the blade hub. Two json files should be created, `sweep_lean_folder/blade_name_sweep.json` and `sweep_lean_folder/blade_name_lean.json`, where `sweep_lean_folder` and `blade_name` are entry parameters of the program that should be entered in `parameters.ini` (see above). The parameters and their possible values are:

- "type": "angle" or "disp",
- "angle_ref" [only used if "type"=="angle"]: "rotor_axis" or "hub",
- "percent_span": list of 4 span levels in percent, from 0 to 100,
- "values": list of the values at the 4 span levels.

Examples:

- lean, as an angle measured in degrees from the rotor axis:

```
{
"type": "angle",
"angle_ref": "rotor_axis",
"percent_span": [0,30,70,100],
"values": [0,0.764,1.721,2.366]
}
```

- sweep, as a displacement in centimeters:

```
{
"type": "disp",
"percent_span": [0,30,70,100],
"values": [0, -0.257, -0.479, -0.563]
}
```

4.2.4 Output

The module yields the following files:

- for each profile, at a span level of i%:
  - `profile_`i`p.txt`: pressure side point coordinates in a file,
  - `profile_`i`s.txt`: suction side point coordinates in a file,
- `profile_le.txt`: leading edge point coordinates along the blade span - one point per cross-section,
- `profile_te.txt`: trailing edge point coordinates along the blade span - one point per cross-section,
- `percent_span.txt`: available profile levels, in percentage of the blade span.

4.3 CAD module - blade_cad.py

4.3.1 Using the script

This script necessitates the Salomé program (see section 3.2.2). It can be launched from main.py, or used directly in Salomé via terminal or via the GUI. In the terminal, the command is `$PATH/salome -t blade_cad.py` where `$PATH` is the path to your Salomé installation. In the GUI, use File > Load script... and load the script `blade_cad.py`.

4.3.2 Input

Parameters The parameters for the CAD script are under the section `[blade_cad]` in `parameters.ini`. Available parameters and example values are:

```
blade_name = blade_ref
coordinate_folder = output/coordinates
export_cad = true
cad_folder = output/CAD
compute_mesh = true
quadratic = true
number_segments = 34
export_mesh = true
mesh_folder = output/mesh
salome_directory = /path/salome/8.4.0/salome
```

TXT files: profile coordinates The inputs of the CAD script are exactly the output of the coordinate module, see above and figure 4.

4.3.3 Output

The module yields the following files, depending on the export options chosen in `parameters.ini`:

- if CAD export is selected, the blade CAD is exported as a `.step` file,
- if mesh computation and export are selected, three files are exported: `Mesh.dat` contains the whole finite element mesh, `Group_hub.dat` contains the nodes of the hub and `Group_tip.dat` contains the nodes of the tip; in the contact simulations presented in [1], all the nodes in Group_hub are clamped, and the boundary nodes on which contact is managed are chosen among the nodes in Group_tip.

Warning: consistently with the data used from NASA reports, all distances are **in centimeters**. It is therefore necessary to convert from centimeters to meters when using the meshes produced by the program.

5 Provided files

Along with this program, all input and output files related to the seven blades studied in [1] are provided:

Profile parameters: parameters in the file `input/profiles/rotor37_original.csv` are from the original NASA report from Moore [2]; `input/profiles/rotor37_smoothed.csv` contains the smoothed parameters used to define the reference blade in [1],

Sweep and lean: values of sweep and lean provided in `input/sweep_lean` correspond to the six modified blades studied in [1]; the values are based on [6, 9],

Coordinates: resulting $(x,\ y,\ z)$ coordinates for each profile, for each blade are provided in `output/coordinates/blade_name` where "name" is the name of each blade as defined in [1],

CAD: the blade geometries for the reference blade and six modified blades are provided as `.step` files, which can be imported in most CAD and finite element software.

Mesh: the finite element meshes for the reference blade and six modified blades are provided as `.dat` files containing the nodes and the elements of the meshes; a translation is necessary for specific finite element software.

References

[1] Piollet, E., Nyssen, F., and Batailly, A., 2019. "Blade/casing rubbing interactions in aircraft engines: numerical benchmark and design guidelines based on NASA rotor 37 (under minor revision)". *Journal of Sound and Vibration*.

[2] Moore, R. D., and Reid, L., 1980. Performance of single-stage axial-flow transonic compressor with rotor and stator aspect ratios of 1.19 and 1.26, respectively, and with design pressure ratio of 1.82. Tech. rep., NASA Lewis Research Center.

[3] Crouse, J. E., Janetzke, D. C., and Schwirian, R. E., 1969. A computer program for composing compressor blading from simulated circular-arc elements on conical surfaces. Tech. rep., NASA.

[4] Reid, L., and Moore, R. D., 1978. Design and overall performance of four highly loaded, high-speed inlet stages for and advanced high-pressure-ratio core compressor. Tech. rep., NASA Lewis Research Center.

[5] Dunham, J., 1998. CFD validation for propulsion system components (la validation CFD des organes des propulseurs). Tech. rep., Advisory Group for Aerospace Research and Development (AGARD), Neuilly-sur-Seine (France).

[6] Benini, E., and Biollo, R., 2007. "Aerodynamics of swept and leaned transonic compressor-rotors". *Applied Energy,* **84**(10), pp. 1012–1027. doi:10.1016/j.apenergy.2007.03.003.

[7] Lainé, J., Piollet, E., Nyssen, F., and Batailly, A., 2019. "Blackbox optimization for aircraft engine blades with contact interfaces". *Journal of Engineering for Gas Turbines and Power,* **141**(6), feb, p. 061016. doi:10.1115/1.4042808 - oai:hal-02059582.

[8] Ribes, A., and Caremoli, C., 2007. "Salome platform component model for numerical simulation". In Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International, Vol. 2, IEEE, pp. 553–564. Salome can be downloaded from www.salome-platform.org.

[9] Biollo, R., and Benini, E., 2009. "Shock/boundary-layer/tip-clearance interaction in a transonic rotor blade". *Journal of Propulsion and Power,* **25**(3), pp. 668–677. doi:10.2514/1.39541.