Documentation

Contents

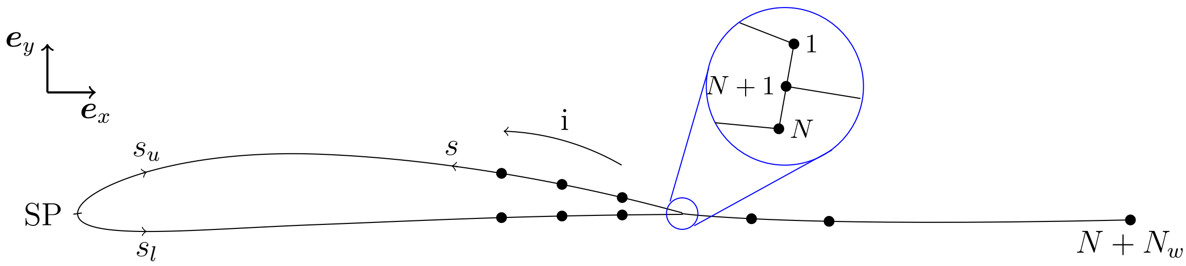
1 General description

The matlab-code provides a coupled boundary layer and potential flow solver for airfoils. The solver for the flow is related to Xfoil [2, 3, 1] and extends it in order to incorporate uniform blowing/suction. A script for the optimization of blowing parameters with the particle swarm optimization [5, 4] is also provided (requires the optimization tool box of MATLAB).

All parameters, that are necessary are set and explained in the parameters-script. In order to calculate the solution, the main-script can be run. It containes already a example and the exlainations of the plotting functions avaivable. Examples for the optimization can be found in the script called OptimizationRuns.

2 Coordinate and index definition

The indexing and total arc length start from the upper part of the trailing edge and go around the airfoil. The arc lengths of the two boundary layers on the suction side s_u and pressure side s_l start from the stagnation point (SP).



3 Calculation of Forces

The lift and drag coefficients are calculated integrating the pressure and friction coefficient over the airfoil

$$C_L = \int C_p \boldsymbol{n} + C_{f\infty} \boldsymbol{t} \mathrm{d}s \cdot \boldsymbol{e}_{\hat{y}} \tag{1}$$

$$C_d = \int C_p \boldsymbol{n} + C_{f\infty} \boldsymbol{t} \mathrm{d}s \cdot \boldsymbol{e}_{\hat{x}} \tag{2}$$

with

$$\boldsymbol{e}_{\hat{x}} = \cos(\alpha)\boldsymbol{e}_x + \sin(\alpha)\boldsymbol{e}_y \tag{3}$$

$$\boldsymbol{e}_{\hat{y}} = -\sin(\alpha)\boldsymbol{e}_x + \cos(\alpha)\boldsymbol{e}_y. \tag{4}$$

The vectors $\boldsymbol{t}$ and $\boldsymbol{n}$ denote the tangential and the normal vector of the profile and $C_{f\infty}$ the friction coefficient normalized using the free stream velocity (not the boundary layer edge velocity). The integration is done numerically using the simpson law.

4 Variable names

In this section a explaination of the most important variables is given.

solution struct (sol)

c	Vector with amplification exponent n for laminar nodes and shear coefficient C_τ for turbulent nodes
Cdrag	Drag coefficient C_d
CD	Dissipationcoefficient
Cf	friction coefficient normalized using $\frac{\rho}{2}U^2$
CI_U	Integral wall shearstress $\int_{s_u} \tau_w/\rho U_\infty^2 \mathrm{d}s$ on suction side
CI_L	Integral wall shearstress $\int_{s_l} \tau_w/\rho U_\infty^2 \mathrm{d}s$ on pressure side
CL	Lift coefficient C_L
Cnu	Friction part of drag coefficient
Cp	Pressure coefficient $C_p = 1 - (U/U_\infty)^2$
D	Displacement thickness δ^*
HK	Shape parameter θ/δ^*
HS	Shape parameter θ^*/θ
itmax	Maximal iterations for solution
iTran	Index of first turbulent node
m	Mass defect $\delta^* U$
PowerInput	Power input needed for uniform blowing $\int C_p v_w/U_\infty \mathrm{d}s$
Ret	Reynoldsnumber $Re_\theta = U\theta/\nu$
sT	Arc length of transition point starting from stagnation point
T	Momentum thickness θ
tau	Normalized wall shear stress $\tau_w/\rho U_\infty^2$
U	Boundary layer edge velocity
US	Velocity at the edge of the inner layer
Vb	Wall normal blowing velocity
xT	x-coordinate where transition occurs
xseperation	x-coordinate where separation starts
xreattach	x-coordinate where reattachement occurs

profile struct (prf)

nodes.X/Y	x/y-coord of nodes
nodes.e	Tangential vector at nodes (1. line $\rightarrow$ x-component, 2. line y-component)
nodes.n	Normal vector at nodes (showing inside the profile)
panel.X/Y	x/y-coord of panel start and end point
panel.L	Panel length
panel.e	Tangential vector of panel
panel.n	Normal vector of panel
panel.theta	Angle between panel and y-axis
gap	TE thickness
noSkew	Defines if profile skewness is neglected or not
sharpTE	Defines if profile is adjusted to have a sharp trailing edge
c	Chord of profile
M	Index where lower side begins
N	Number of nodes on profile
Nle	Index of leading edge node
LE1	Distance of stagnation point to first node on suction side
LE2	Distance of stagnation point to first node on pressure side
s	Total arc length vector
sLE	Arc length of leading edge
sL	Arc length vector of pressure side beginning from stagnation point (index starting from LE)
sU	Arc length vector of suction side beginning from stagnation point (index starting from TE)
xL	x-coord vector of pressure side (index starting from LE)
xU	x-coord vector of suction side (index starting from TE)

flow struct (flo)

A	Matrix of the potential flows equation system
Ages	Matrix A including the Kutta-condition
nkrit	Critical amplification exponent for transition
ui	x-component of $\boldsymbol{u}_i nfty$
vi	y-component of $\boldsymbol{u}_i nfty$
t	Right hand side of eq sys $t_i = U_\infty \cos(\alpha) y_i + U_\infty \sin(\alpha) y_i$
wake	Struct with wake node information (same as in prf struct)
wake.gap	Thickness of dead air region

coefficient matrix struct (CoeffMatrix)

A	Matrix of the potential flows equation system
B	Influence of profile sources on profile node equations
Bw	Influence of wake sources on profile node equations
Bges	Total B-matrix $B^{ges} = [B, B_w]$
Btilde	$-A^{-1}B^{ges}$
Cg	Influence of circulation on the wake node equations
Cq	Influence of all sources on wake node equations
Cq2	$C_q - C_g\tilde{B}$
D	Total matrix for mass defect from eq $U_i = U_i^{inv} + \sum_j \operatorname{sgn} D_{ij}m_j$ consisting of Bges in the upper part and Cq2 in the lower part

5 Solution process

This section sketches the solution process and names the functions responsible for each step.

Determination of the inviscid solution (function InviscidSolution)

1. Solve the equation system for the circulation on the profile

$$\sum_{j=1}^{N} A_{ij}\gamma_j + \psi_0 = U_\infty \cos(\alpha)y_i + U_\infty \sin(\alpha)y_i$$

   (function Potential)

2. Determinate the stagnation point finding $\gamma = 0$ using a linear approximation on the panel with $\gamma_i > 0 \wedge \gamma_{i+1} < 0$ (function getStagnationPoint)

3. Calculate the nodes of the wake streamline, where $\psi = \psi_0$ holds using a Newton-method (function getWakeStreamline)

4. Calculate the coefficient matrices needed for the determination of the boundary layer edge velocity U from the node circulation γ_i and the node mass defect $m_i = U_i\delta_i^*$ (functions Qlin, GradPsiN)

5. Calculate the inviscid part U^{inv} of U, that is independent of m

Determination of the viscid solution (function airfoil)

- Calculate the initial boundary layer (BL) solution using U^{inv}. Solves the initial value problem for both the BLs on suction and pressure side starting from the stagnation point (function GetInitialSolution)

- Use the Newton-method for the coupled solution of potential flow and BL (function NewtonEQ). Each iteration consists of the following steps (the superscript (k) denotes the index of the current iteration)

  1. Calculate the new BL edge velocity with the current mass defect

  $$U_i^{(k)} = U_i^{inv} + \sum_{j=1}^{N+N^w} D_{ij} m_i^{(k)}$$

  2. Set up the right hand side f and the Jacobi-matrix J for the current step (function JacobiM)
  3. Calculate the correction for the solution vector $z = [\theta_i, n_i/C_{\tau,i}, m_i]^T$ solving

  $$J \Delta z = -f$$

     and update it with a proper relaxation factor r (function Update)

  $$z^{(k+1)} = z^{(k)} + r \Delta z$$

  4. Recalculate the stagnation point and adjust the sign for the D-matrix
  5. Recalculate the transition points and adjust the solution in terms of separation (function Refresh)

References

[1] Drela, M: Integral boundary layer formulation for blunt trailing edges. In: *7th Applied Aerodynamics Conference*, 1989, S. 2166

[2] Drela, M: XFoil: An analysis and design system for low Reynolds number airfoil aerodynamics. In: *Conference on Low Reynolds Number Airfoil Aerodynamics, University of Notre Dame*, 1989

[3] Drela, M ; Giles, M: Viscous-inviscid analysis of transonic and low Reynolds number airfoils. In: *AIAA journal* 25 (1987), Nr. 10, S. 1347–1355. `http://dx.doi.org/10.2514/3.9789`. – DOI 10.2514/3.9789

[4] Mezura-Montes, E ; Coello Coello, C: Constraint-handling in nature-inspired numerical optimization: past, present and future. In: *Swarm and Evolutionary Computation* 1 (2011), Nr. 4, S. 173–194

[5] Venter, G ; Sobieszczanski-Sobieski, J: Particle swarm optimization. In: *AIAA journal* 41 (2003), Nr. 8, S. 1583–1589. `http://dx.doi.org/10.2514/2.2111`. – DOI 10.2514/2.2111