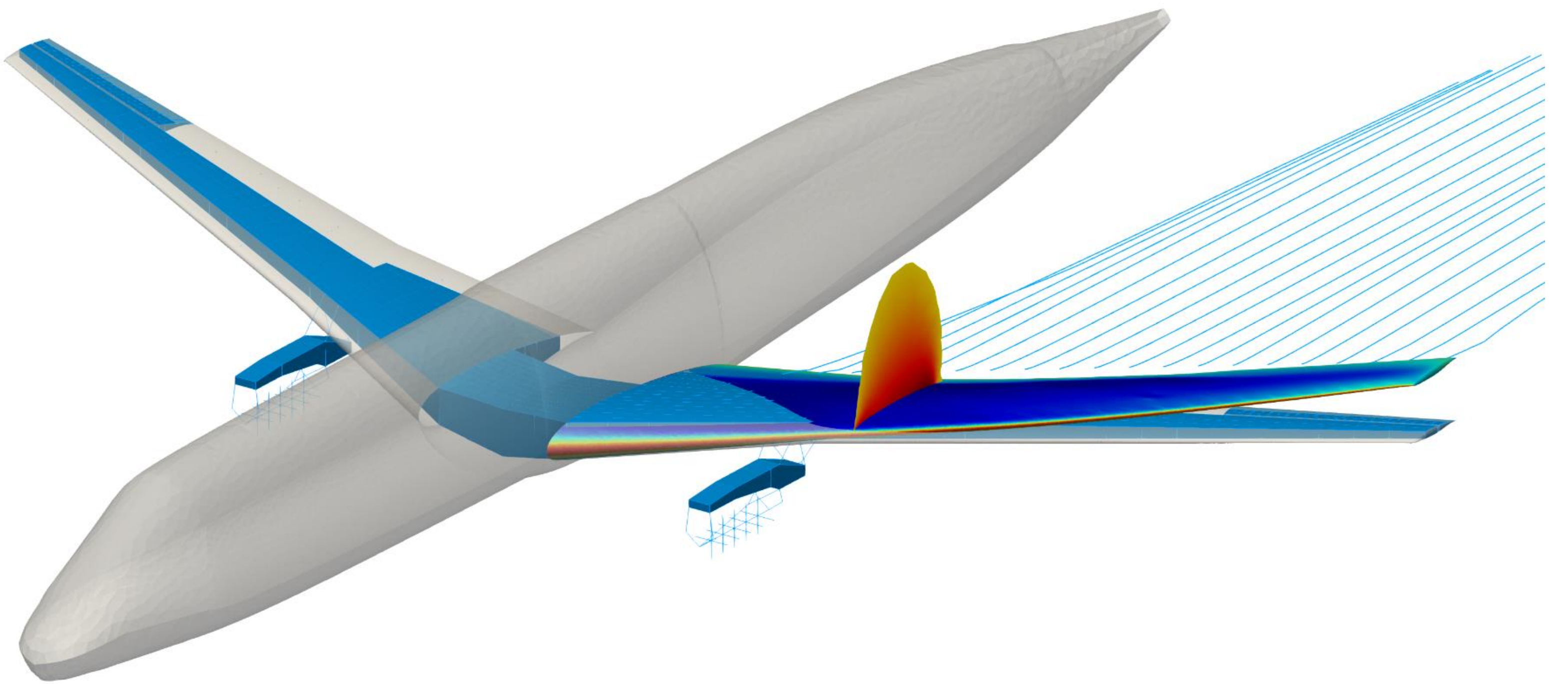
Steady transonic aerodynamic and aeroelastic modeling for preliminary aircraft design

Adrien Crovato



Liège, December 2020

The state of aviation

More traffic

$3 \rightarrow 16\ \mathrm{Gp}$

$46 \rightarrow 400\ \mathrm{Mt}$

Less emissions

$-75\%\ CO_2$

$-90\%\ NO_x$

Faster aircraft

4h door-to-door

max. 1min late

More profit

best 3.8% $\uparrow$

avg. 0.1% $\uparrow$

Fuel consumption must be reduced!

Reducing fuel consumption

Flight dynamics

Computer science

Aerodynamics

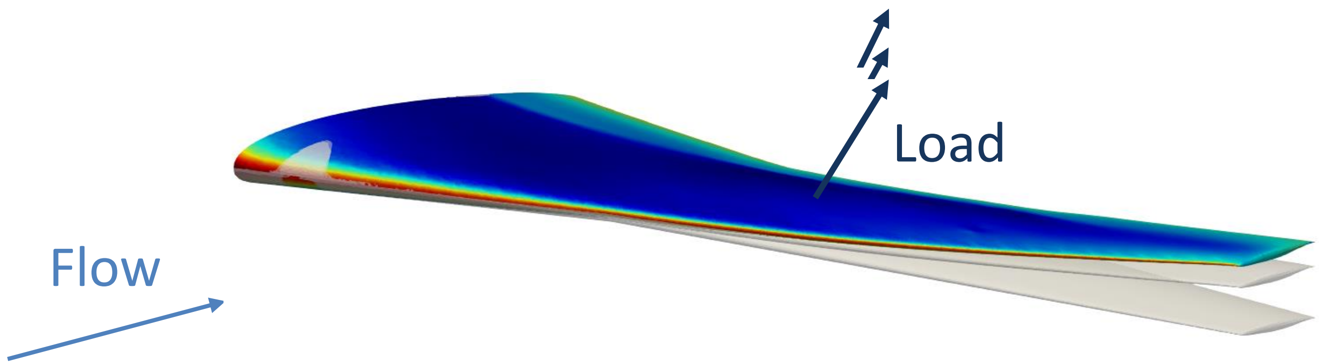
Structure and materials

Propulsion and fuel

Air traffic management

…

Aeroelasticity



Structural dynamics
Inertial forces

Aerodynamic stability

Structural vibrations

Dynamic aeroelasticity

Fluid mechanics
Aerodynamic forces

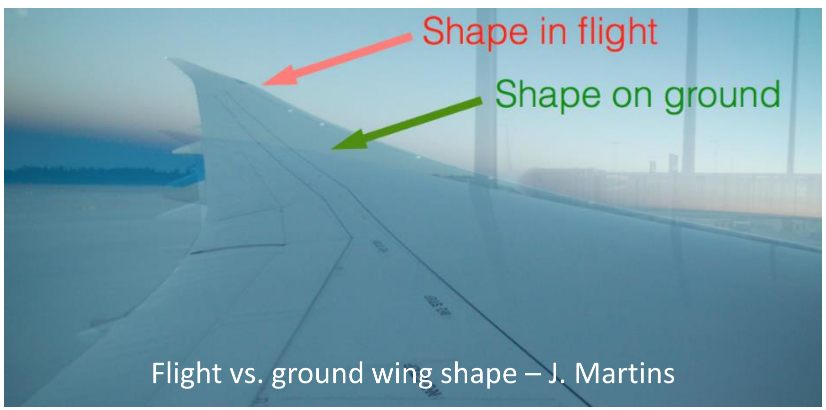
Static aeroelasticity

Structural mechanics
Elastic forces

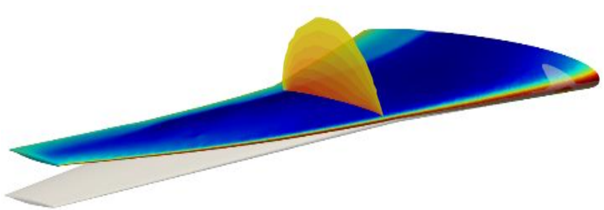
Aeroelasticity in aircraft design

Static aeroelasticity

- Divergence speed
- Wing shapes



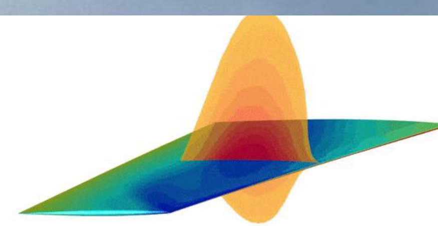
Flight vs. ground wing shape – J. Martins



Dynamic aeroelasiticty

- Flutter speed
- Buffeting

Tail flutter on PA-38 – NASA

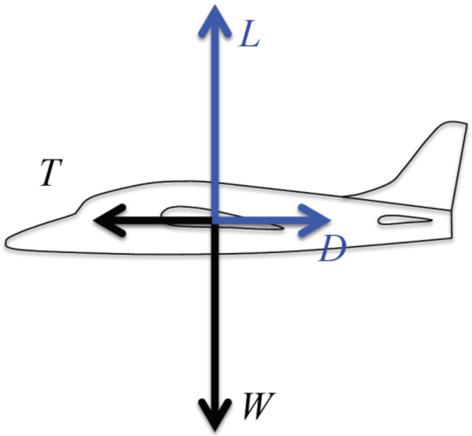


Flutter on AGARD wing – D. Thomas

Aircraft design process

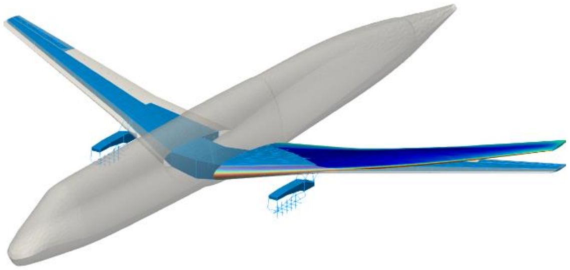
Conceptual

Concept (1%)
- Configuration
- Mission & cost



Preliminary

Model (9%)
- Global design
- Optimization
- Performance

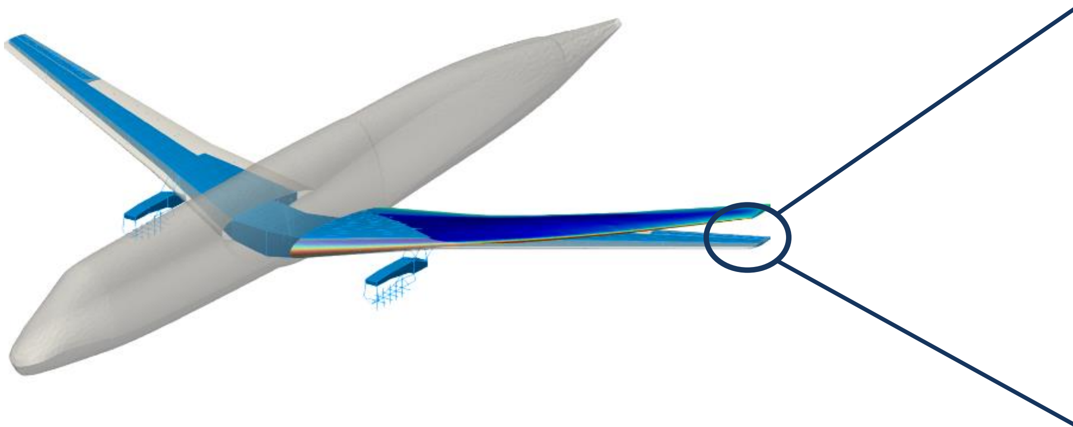


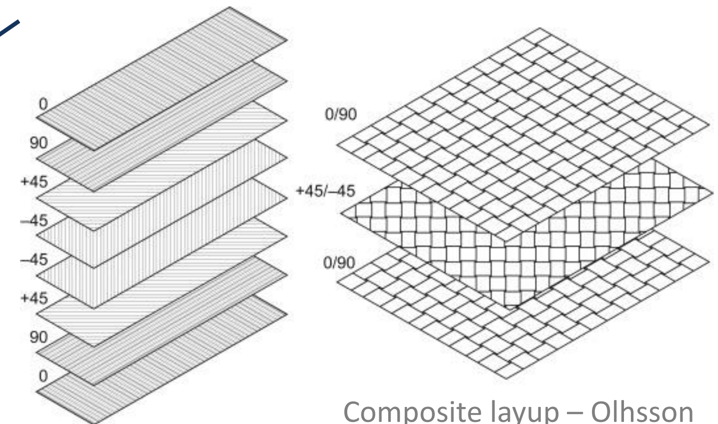
Detail

Prototype (90%)
- Local design
- Manufacturing
- Testing
- Certification

ERJ-190-300-STD © C. Hines (airliners.net)

Aeroelastic tailoring





Composite layup – Olhsson

Optimize wing

- Light structure

Such that

- Sustain ultimate loads
- Recover flight shape
- Stable aeroelastic behavior

EMBRAER

LIÈGE université

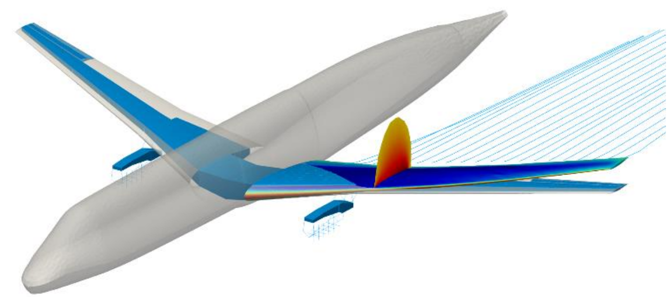
Investigate **aerodynamic** modeling solutions for **steady** and unsteady flows

Thesis objectives

Development

Develop an aerodynamic solver for fast transonic load computation and **integrate** it into a fluid-structure framework

Benchmark



Evaluate and **compare** the aerodynamic models in the context of rigid and static transonic aeroelastic computations

Outline – development

Modeling

- Flow physics
- Aerodynamic models
- Potential equation

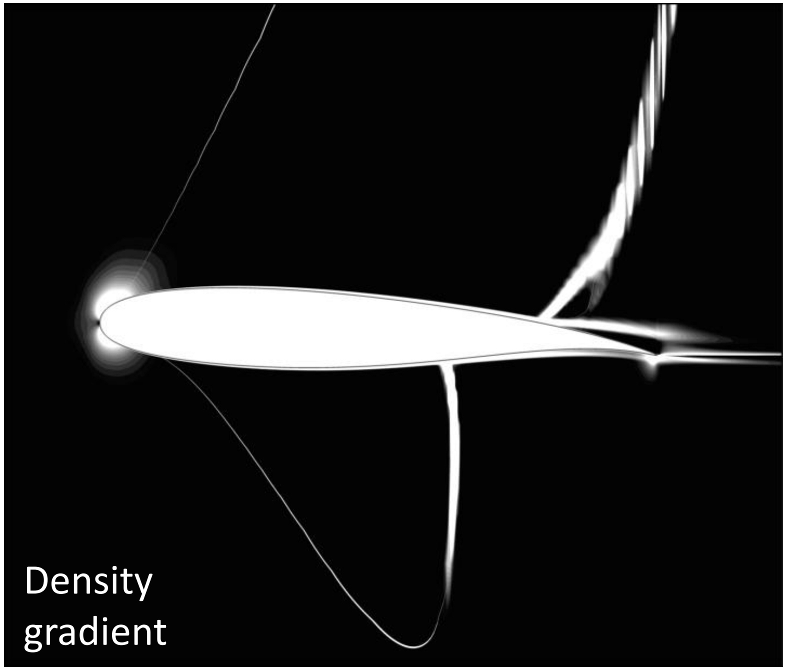
Methods

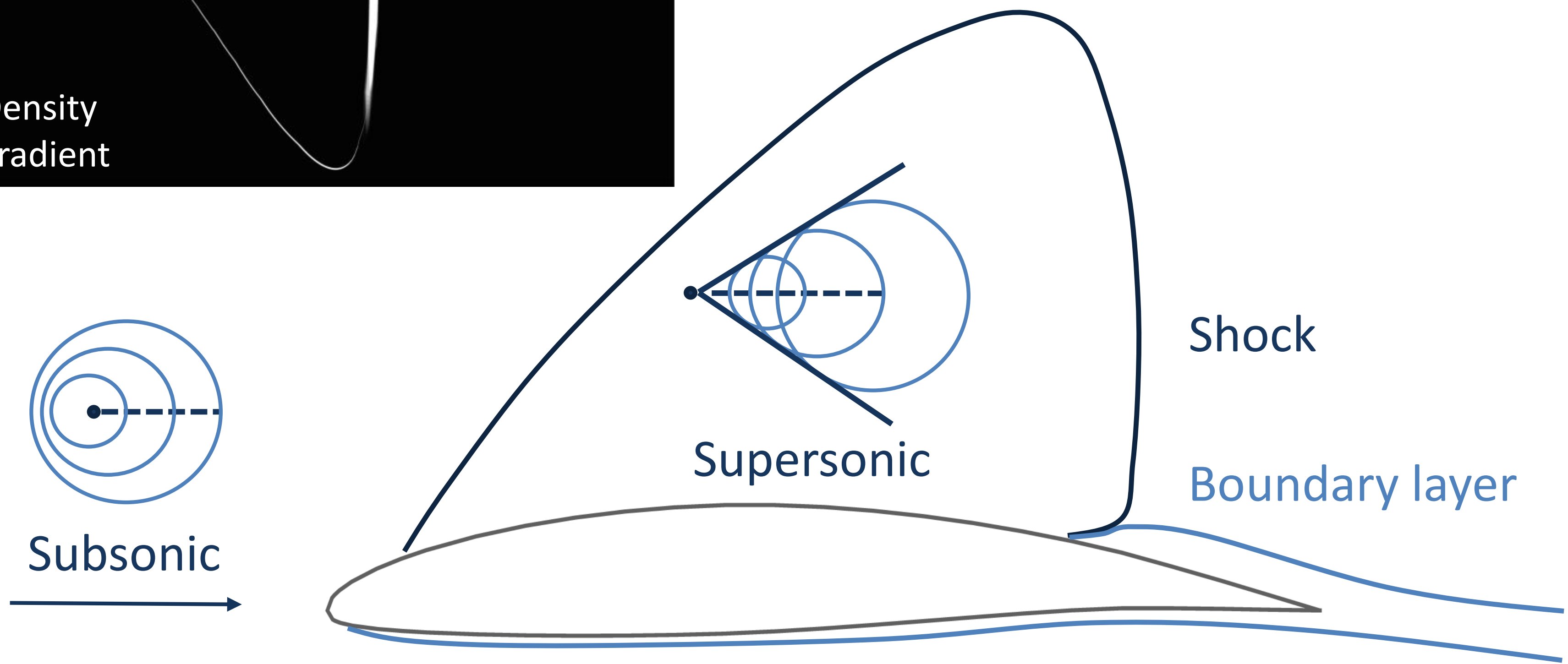
- Field panel method
- Finite element method

Fluid-structure interaction framework

- Structural model
- Framework
- CUPyDO

Transonic viscous flows

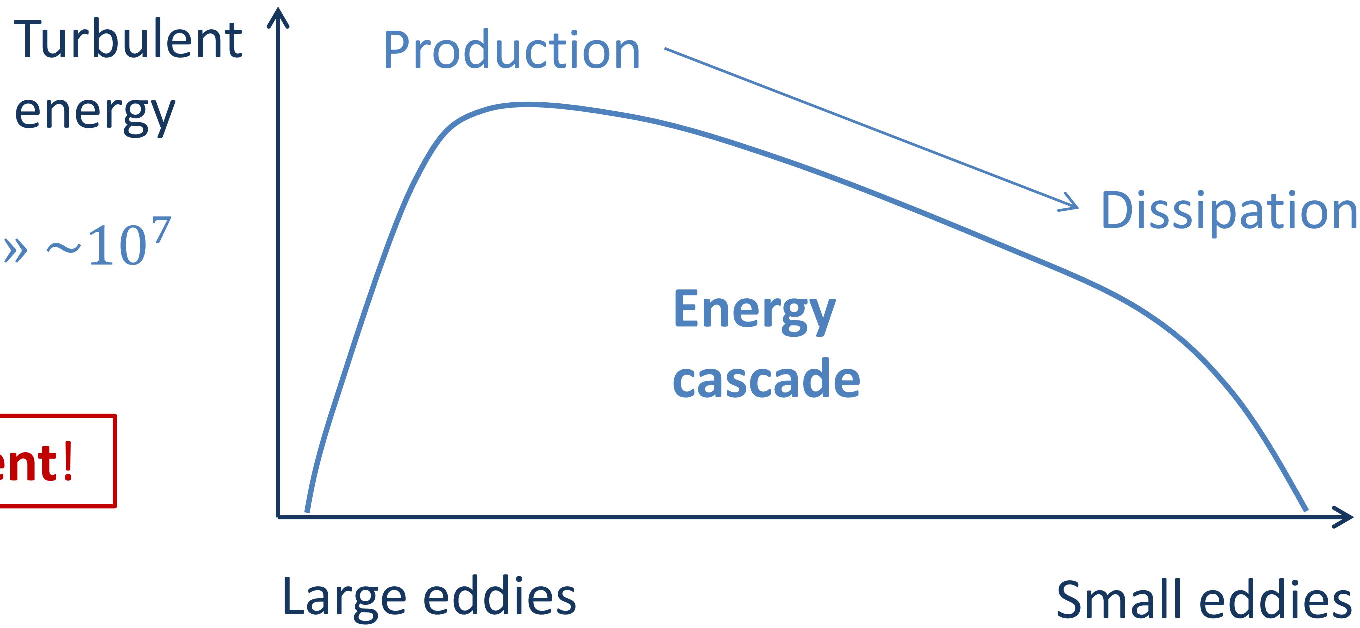




Aerodynamic models

$$Re = « \frac{inertia}{viscosity} » \sim 10^7$$

Flow is **turbulent**!

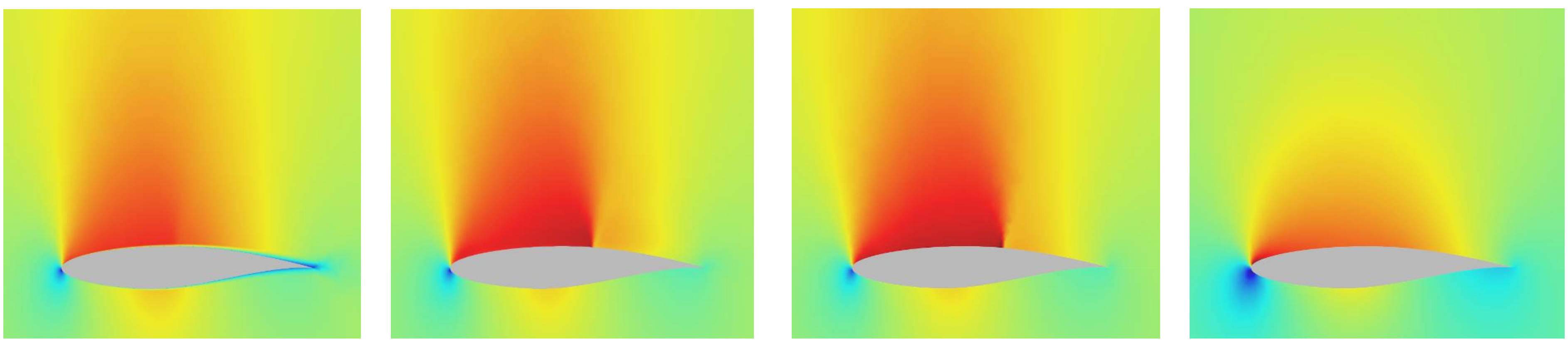


Direct Numerical Simulation	Resolved	
Large Eddy Simulation	Resolved	Modeled
Reynolds-Averaged N-S	Modeled	

Aerodynamic models in aircraft design

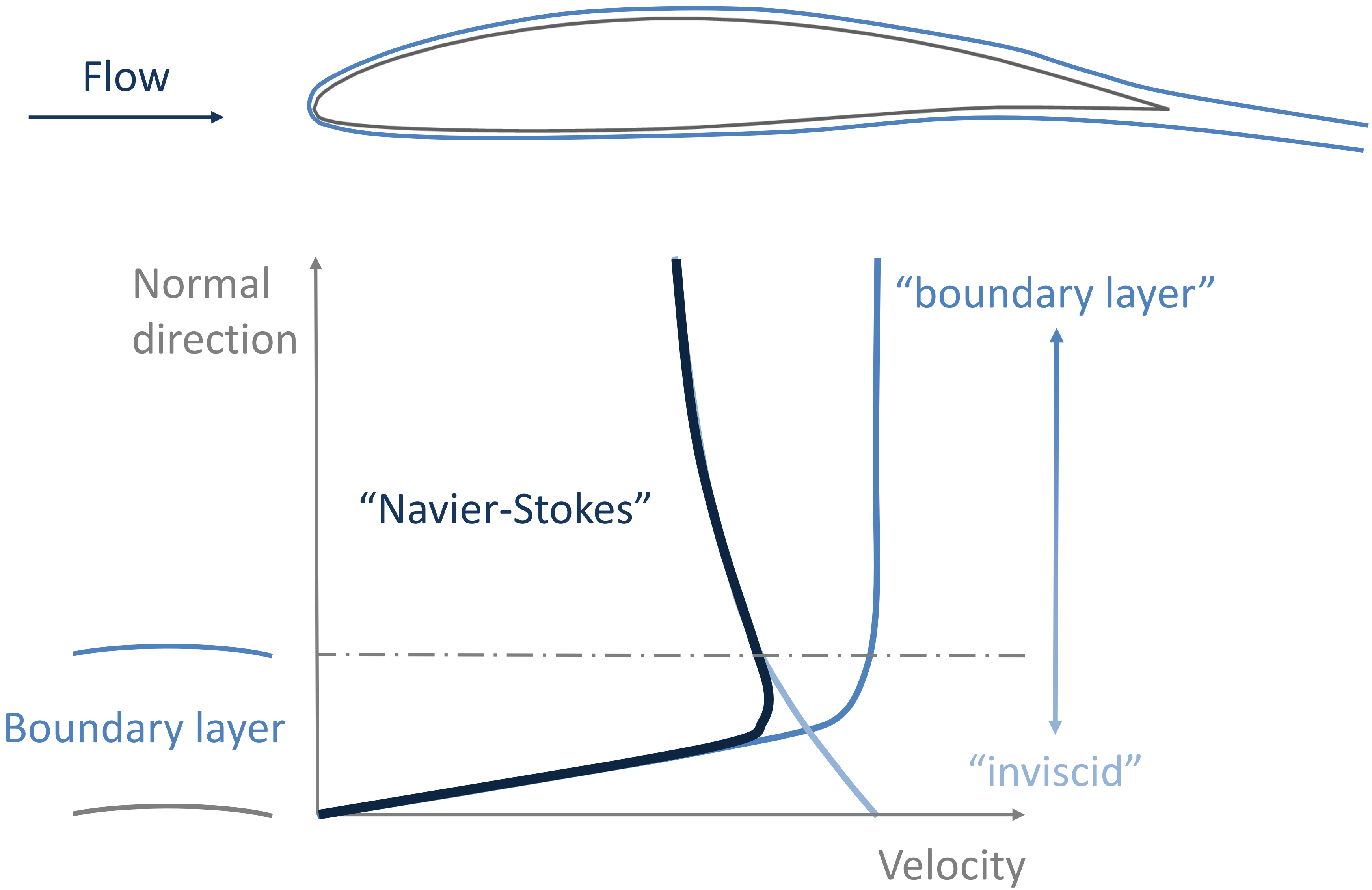
RANS equations	Euler equations	Full potential equation	Linear potential equation
• Subsonic	• Subsonic	• Subsonic	• **∼Subsonic**
• Supersonic	• Supersonic	• Supersonic	• **∼Supersonic**
• Transonic	• Transonic	• **∼Transonic**	• ~~**Transonic**~~
• Viscous	• **Inviscid**	• Inviscid	• Inviscid

→ Inviscid → Isentropic → Linear

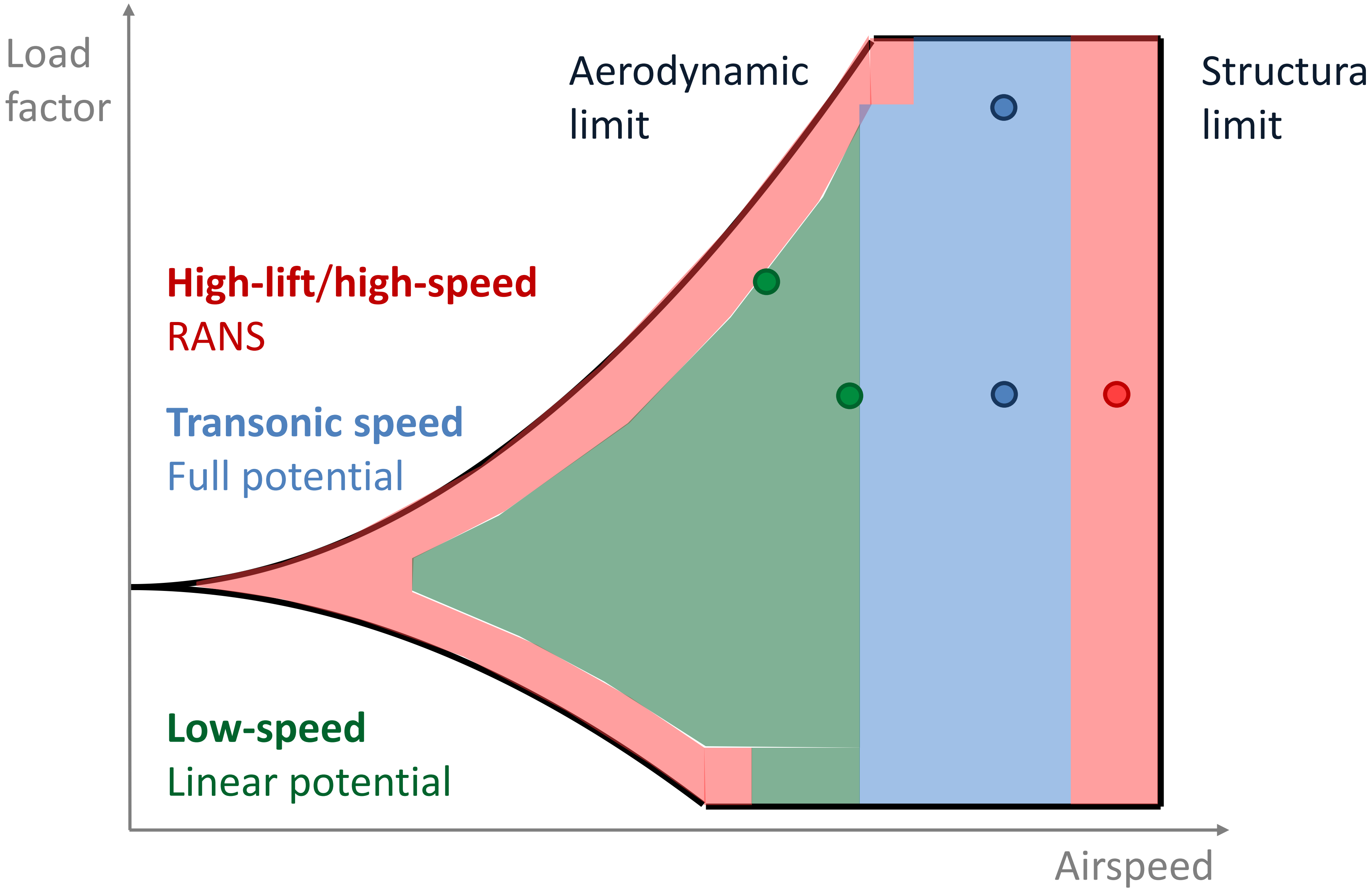


Mach number

Viscous-inviscid interaction



Range of validity



The potential equation

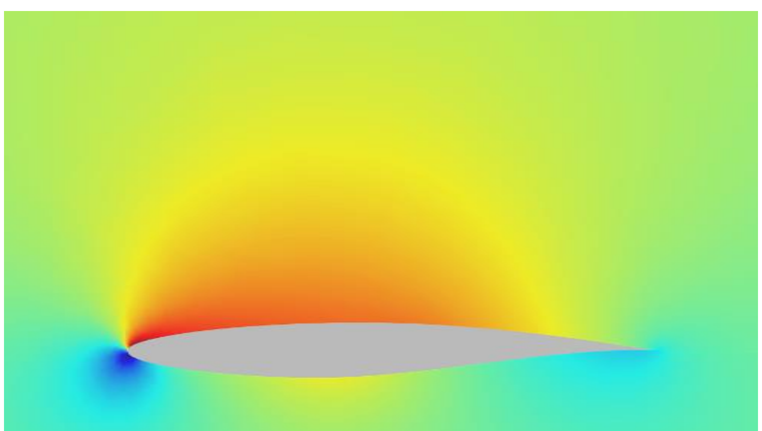
$$\nabla \cdot (\rho \nabla \phi) = 0$$

$\rho \sim \rho(\nabla \phi)$

$\rho = \rho(\nabla \phi)$

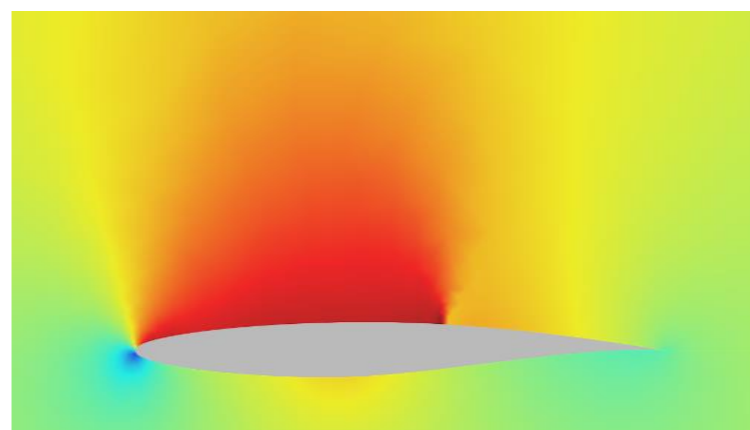
Linear

Flow
- Scaled incompressible
- Subsonic or supersonic
- ~~Transonic~~



Full (nonlinear)

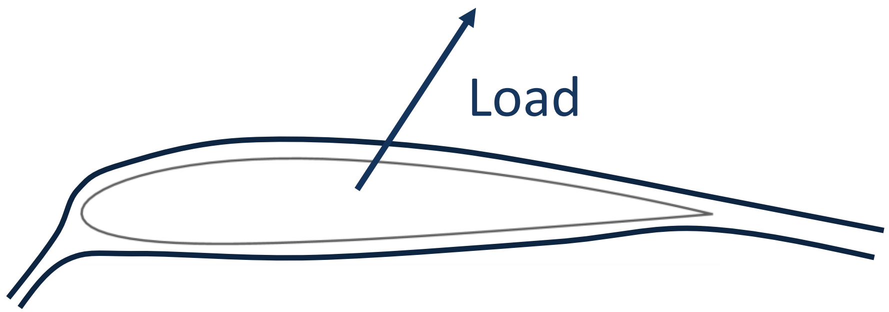
Flow
- Compressible
- Subsonic or supersonic
- **Weak transonic**



Challenges of potential flow – loads

Real fluid

- viscous, finite velocity
- no turn around sharp corners

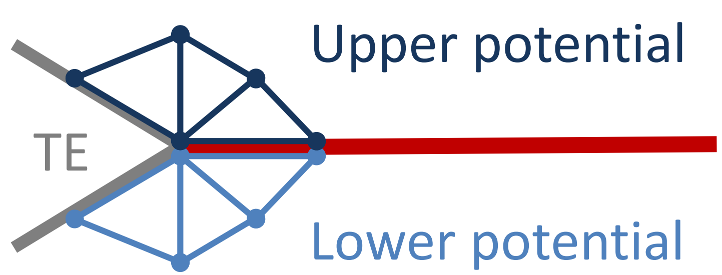


Irrotational flow

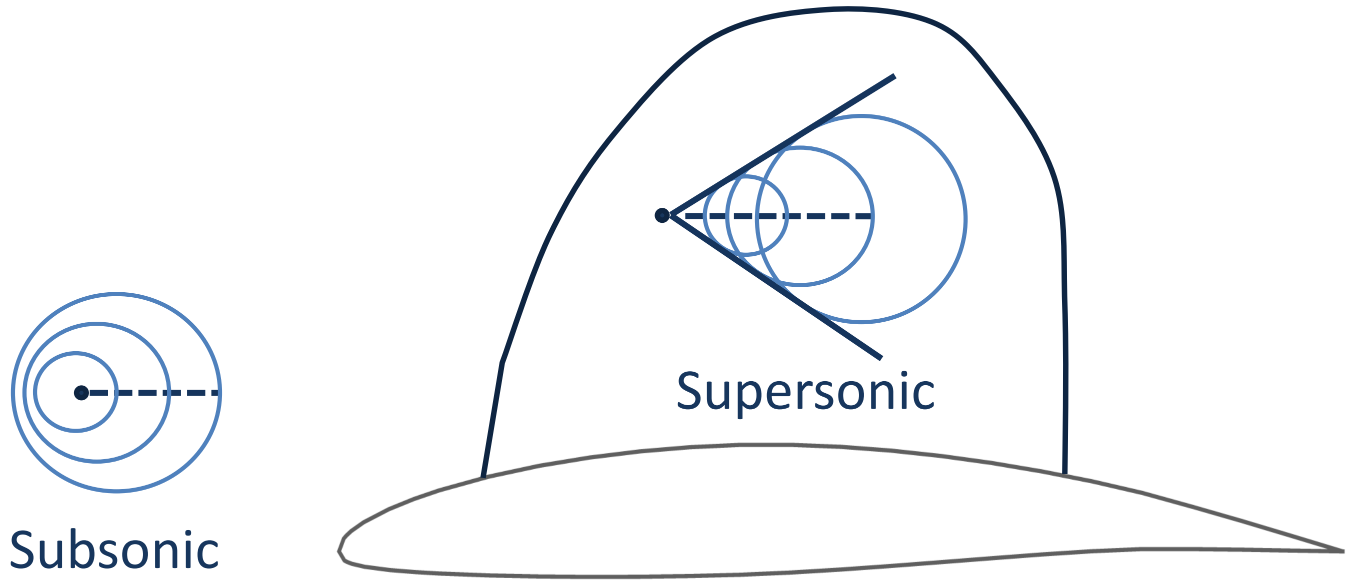
- no deviation
- no load

Numerical model

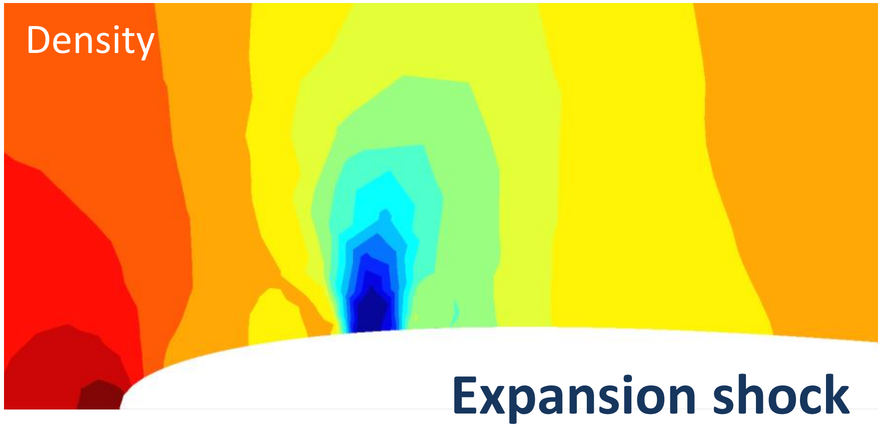
- prescribe wake
- enforce continuity in physical variables



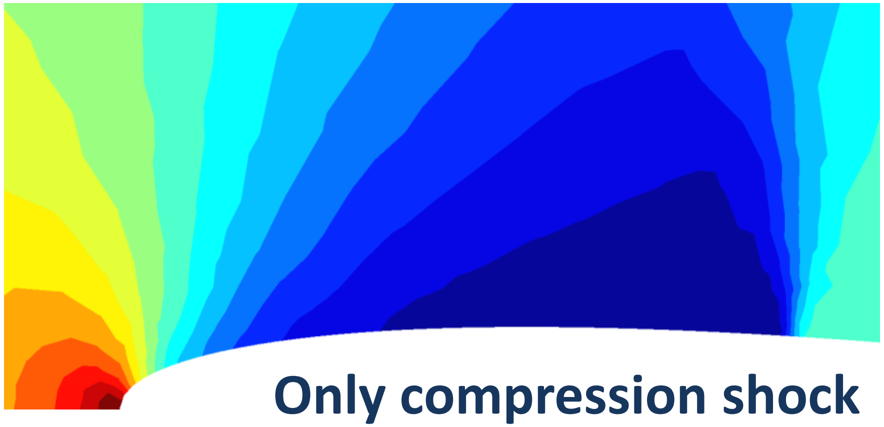
Challenges of potential flow – stabilization



Without stabilization



With stabilization



Outline – development

Modeling

- Flow physics
- Aerodynamic models
- Potential equation

Methods

- Field panel method
- Finite element method

Fluid-structure interaction framework

- Structural model
- Framework
- CUPyDO

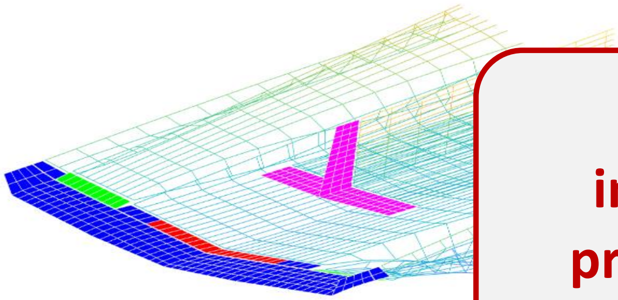
Traditional solution methods

Boundary element method

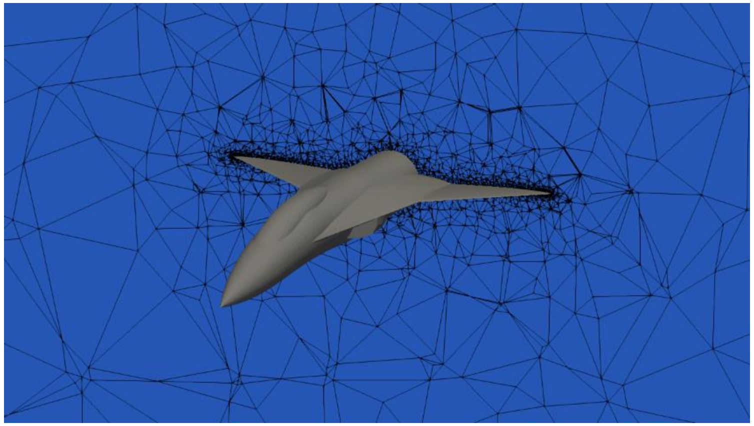
- Only **boundary** is discretized
- **Linear** equations
- Panel/lattice methods

Field method

- Whole **field** is discretized
- Linear and **nonlinear** equations
- Finite volume/element methods



Current industrial practice for aeroelastic computations

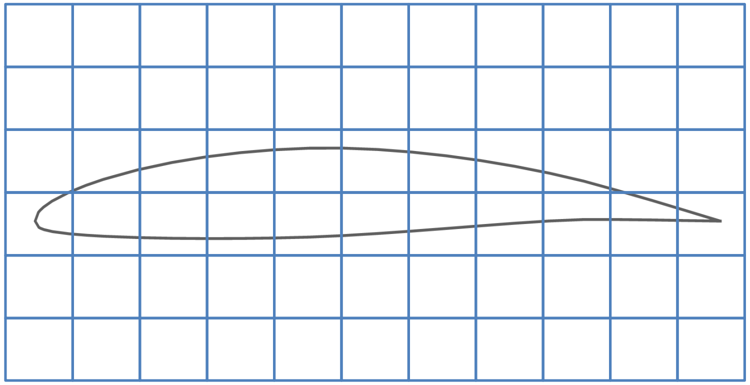


Field panel method

Combination

Boundary element method
- linear part
- on the wing surface

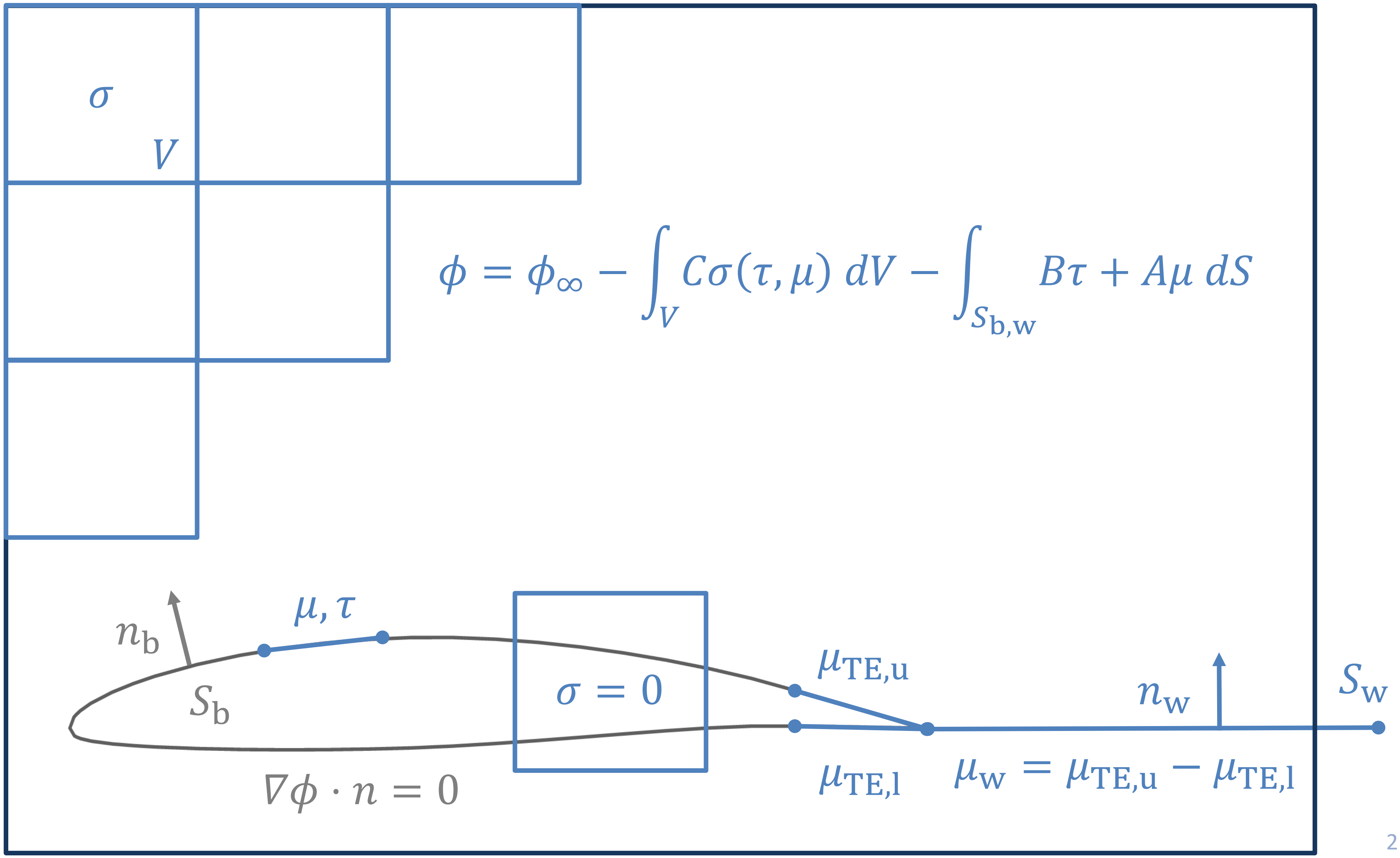
Field method
- nonlinear part
- in the field



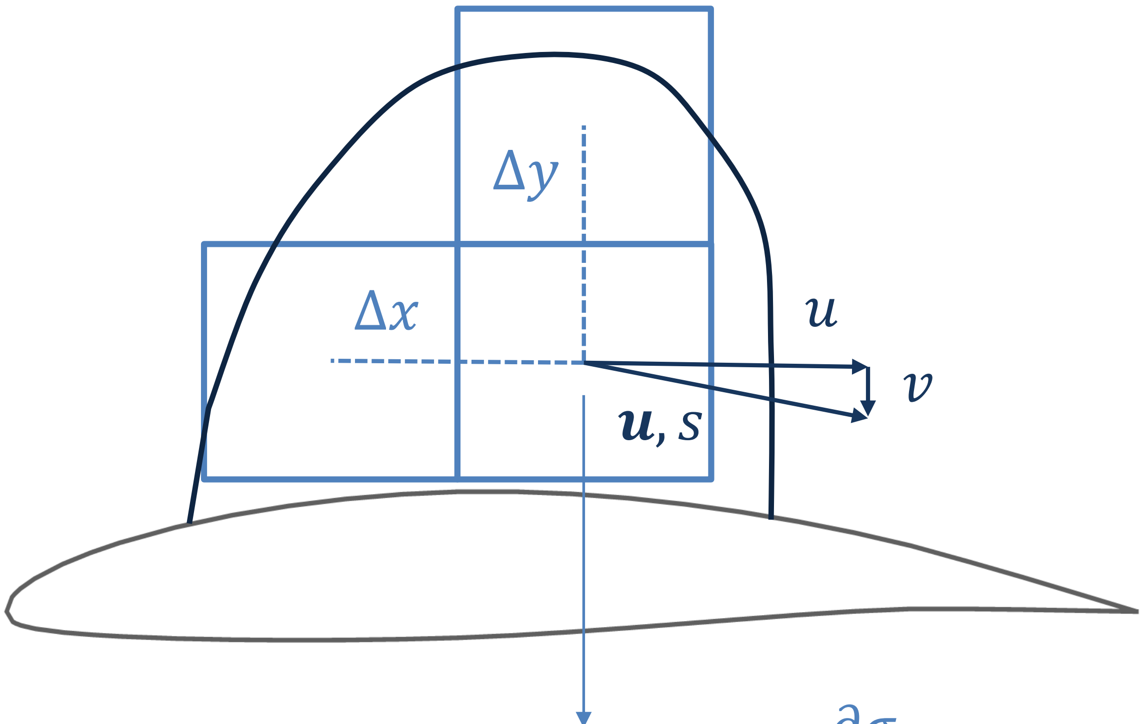
Advantages
- extension to panel method
- simple grid generation

Disadvantages
- high memory requirement
- disagreement in literature

Field panel method – overview



Field panel method – stabilization



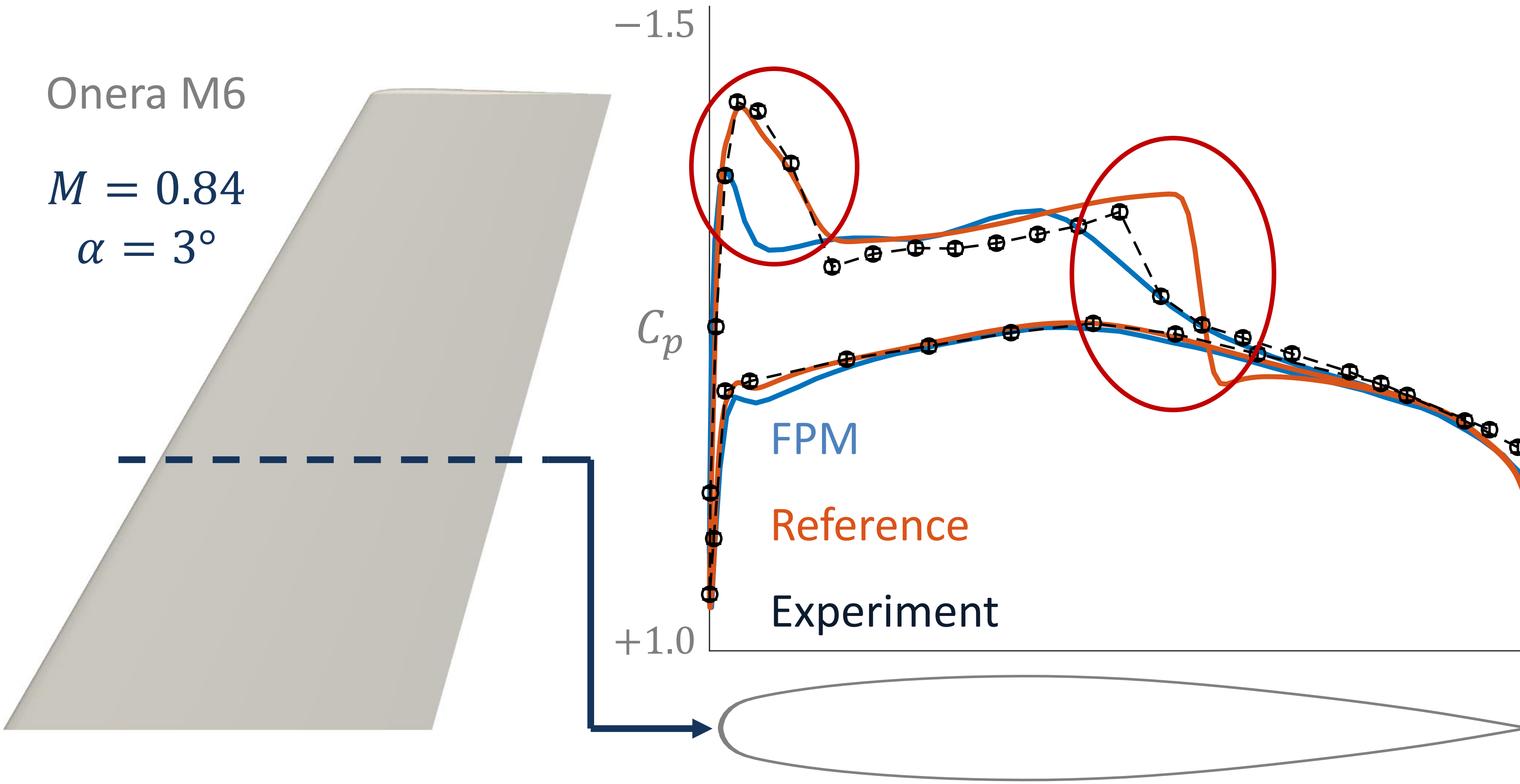
$$\sigma \leftarrow \sigma + \nu \frac{\partial \sigma}{\partial s} \Delta s$$

$$\sigma \leftarrow \sigma \left[1 + \max\left(0{,}1 - \frac{M_C^2}{M^2} \right) \frac{1}{|\boldsymbol{u}|} \left(u \overleftarrow{\Delta} x + v \overleftarrow{\Delta} y \right) \right]$$

Field panel method – verification

Onera M6

$M = 0.84$
$\alpha = 3°$



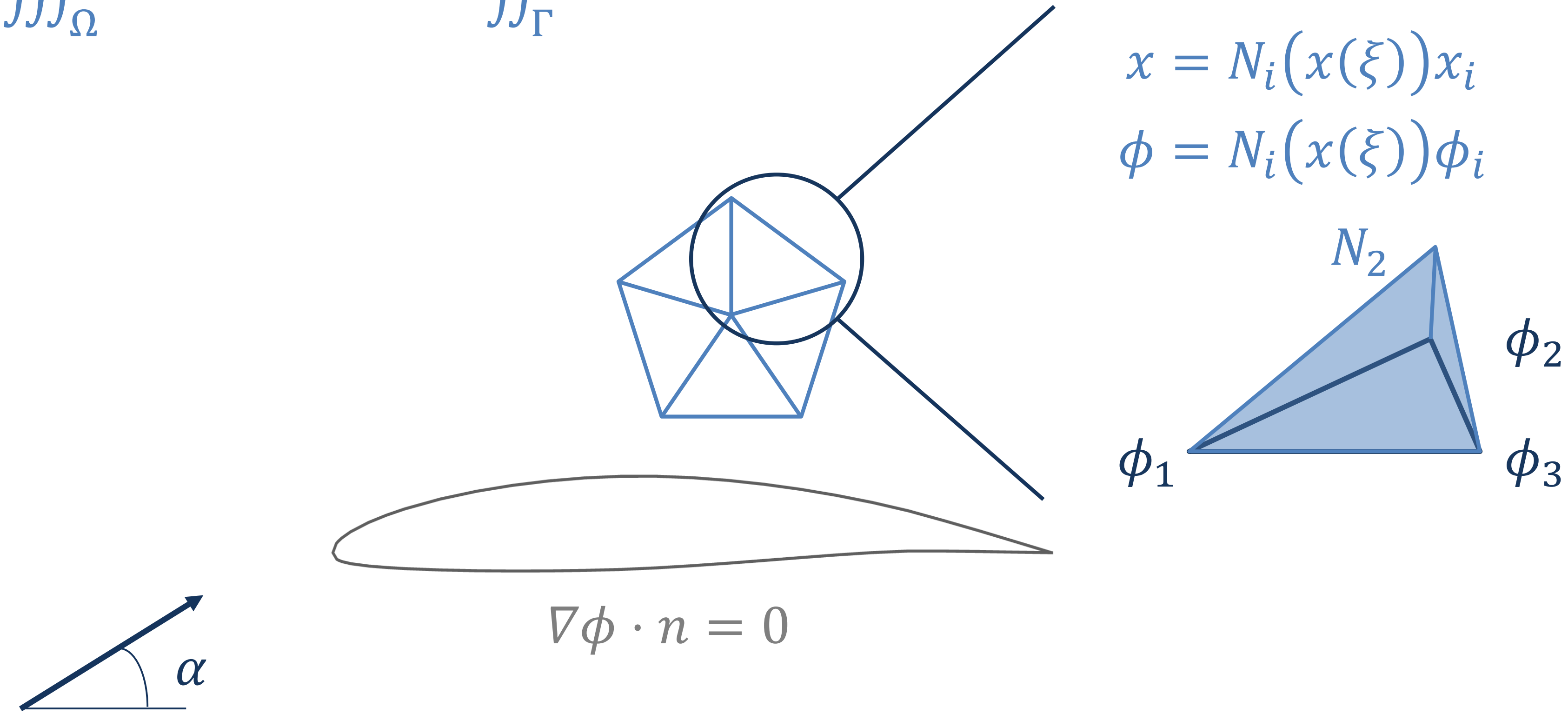
Shocks smeared and displaced upstream
Discontinued

Finite element method – overview

$$\iiint_\Omega \rho \nabla\phi \cdot \nabla\psi \, dV - \iint_\Gamma \rho \nabla\phi \cdot n \, \psi \, dS = 0$$

$$x = N_i\big(x(\xi)\big)x_i$$

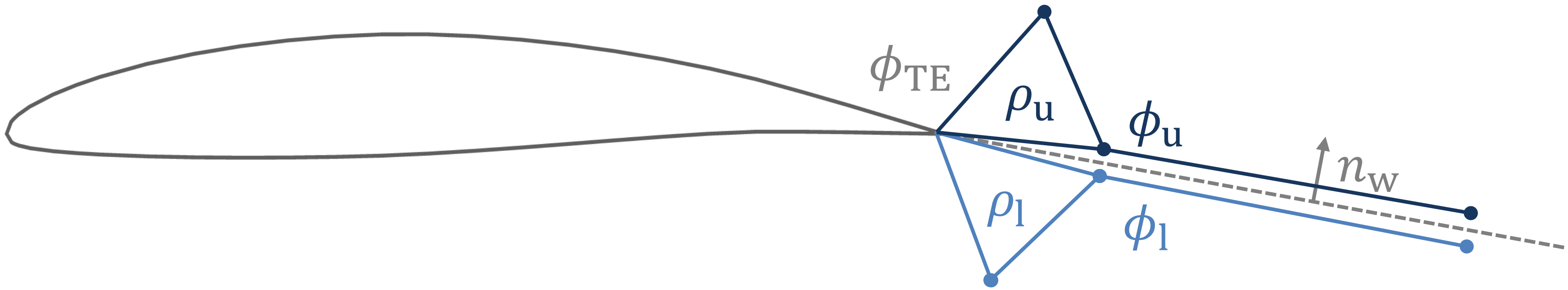
$$\phi = N_i\big(x(\xi)\big)\phi_i$$



$$\nabla\phi_\infty = \{\cos\alpha, \sin\alpha\}$$

$$\nabla\phi \cdot n = \nabla\phi_\infty \cdot n$$

Finite element method – wake model



Formulation

$$\rho_{\mathrm{u}} \nabla_{\mathrm{n}} \phi_{\mathrm{u}} = \rho_{\mathrm{l}} \nabla_{\mathrm{n}} \phi_{\mathrm{l}} \qquad \rightarrow \iint \psi \, [[\rho \nabla \phi \cdot n]] \, dS = 0$$

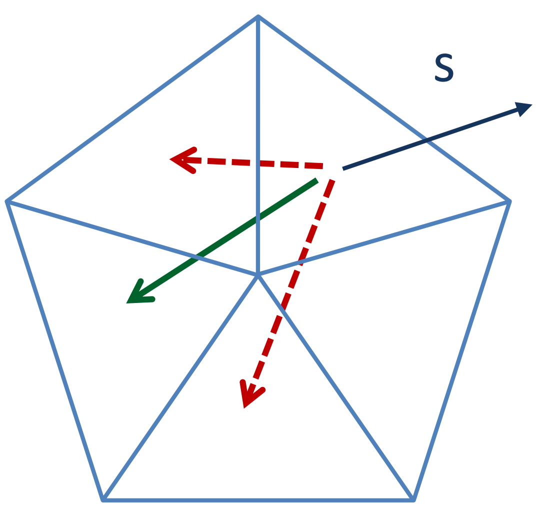
$$p_{\mathrm{u}} = p_{\mathrm{l}} \qquad \rightarrow \iint \left(\psi + \frac{h}{2} U_{\infty} \cdot \nabla \psi \right) [[|\nabla \phi|^2]] \, dS = 0$$

Finite element method – stabilization

Density upwinding + **Newton-Raphson procedure**

$$\rho \leftarrow \rho - \nu \frac{\partial \rho}{\partial s} \Delta s$$

$$\rho \leftarrow \rho - \nu \overleftarrow{\Delta} \rho$$

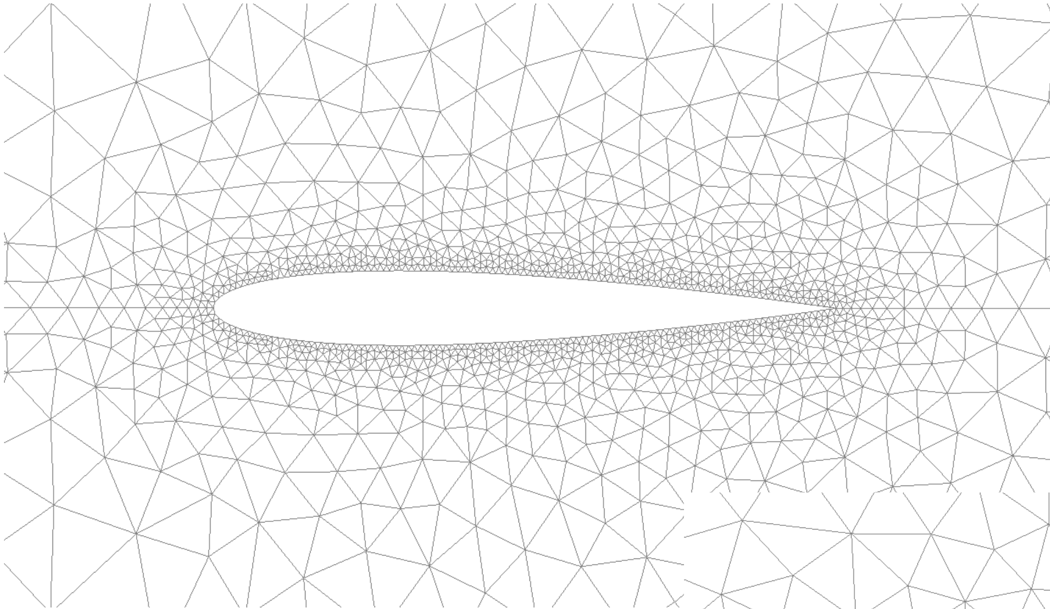


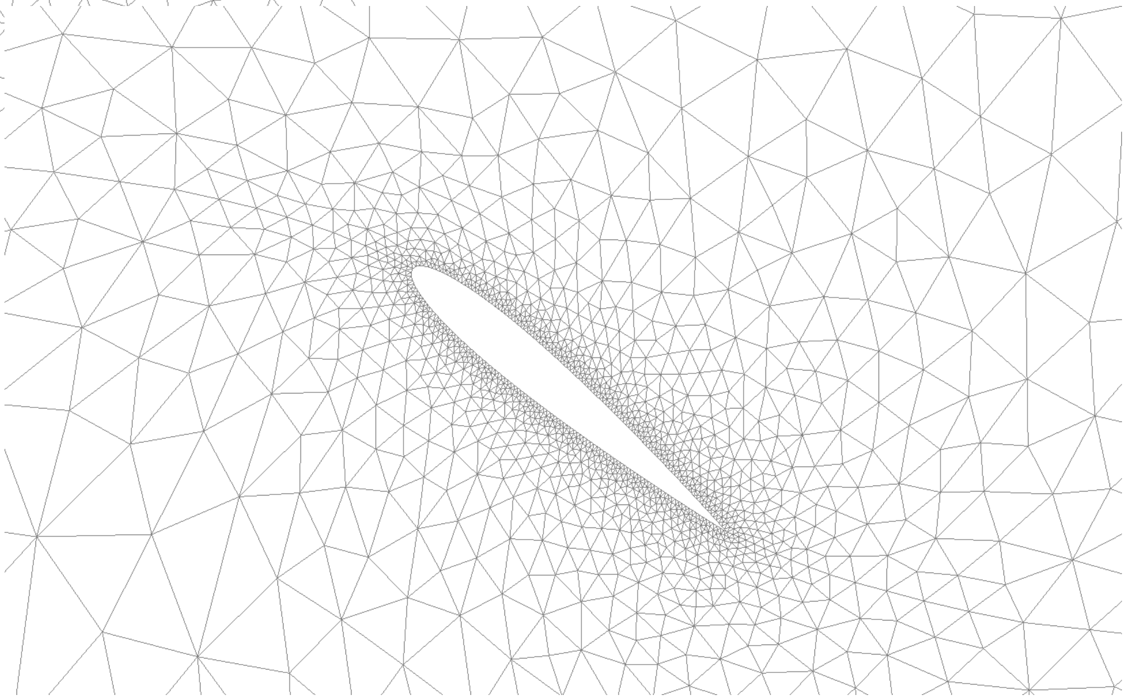
$$F(\phi) = 0 \Rightarrow \frac{\partial F}{\partial \phi} \Delta \phi + F \approx 0$$

- ✓ Analytical tangent matrix
- ✓ Rose & Bank line search
- ✓ Adaptive viscosity ramping

$$\nu = \max \left[0, \boldsymbol{\nu}_{\mathbf{c}\downarrow} \left(1 - \frac{\boldsymbol{M}_{\mathbf{c}\uparrow}^{\mathbf{2}}}{M^2} \right) \right]$$

Finite element method – mesh morphing





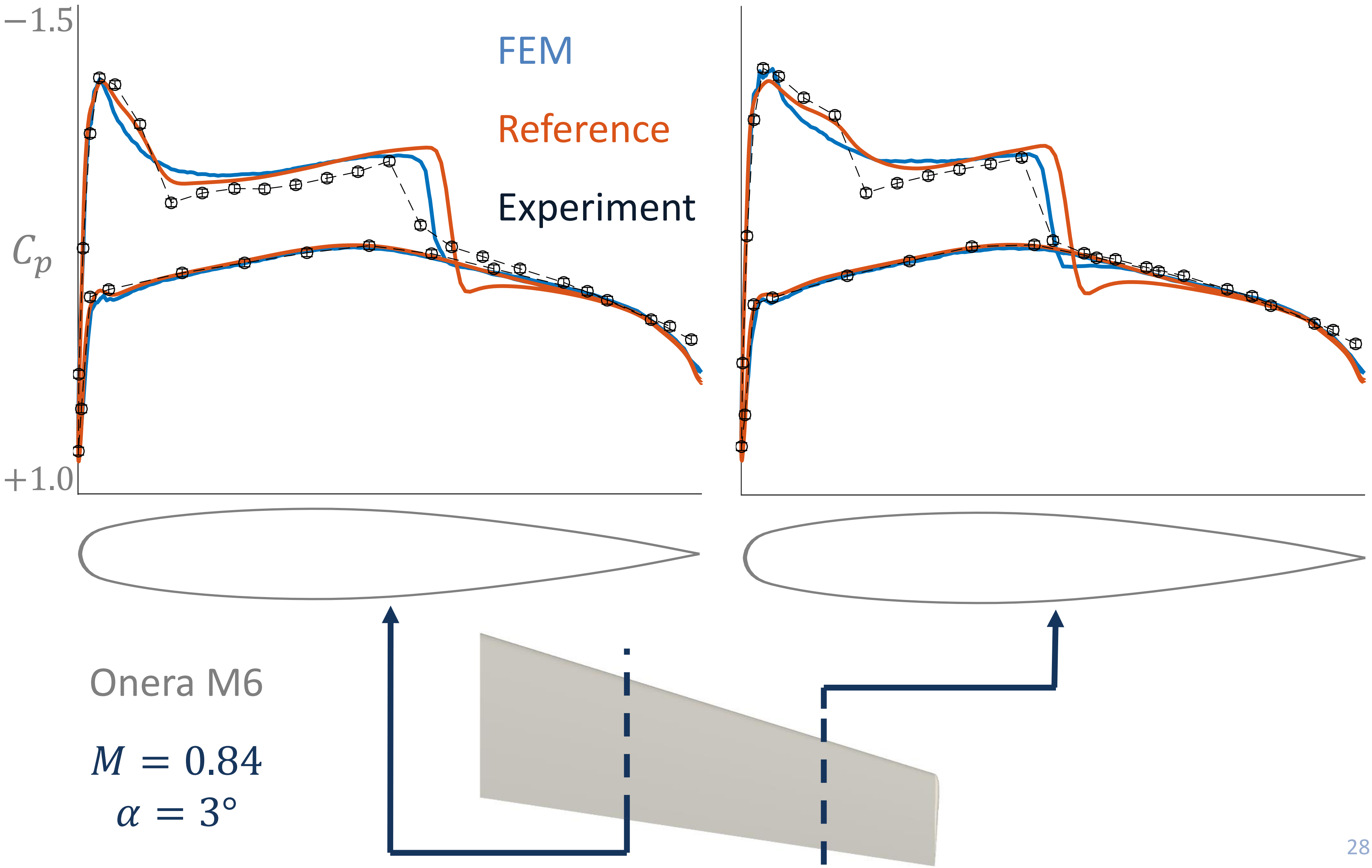
Mesh morphing

- Laplace smoothing

✓ Linear elasticity

$$\boldsymbol{\varepsilon} = \boldsymbol{H}(E, 0)\boldsymbol{\sigma}$$
$$E = V^{-1}$$

Finite element method – verification



Onera M6

$M = 0.84$
$\alpha = 3°$

Finite element method – cost

Intel i7
2.8 Ghz

FEM	$n_{\text{CPU}} = 1$	$t_{\text{real}} = 245\ \text{s}$
	$n_{\text{CPU}} = 2$	$t_{\text{real}} = 170\ \text{s}$
	$n_{\text{CPU}} = 4$	$t_{\text{real}} = 105\ \text{s}$
Reference	$n_{\text{CPU}} = 1$	$t_{\text{real}} = 240\ \text{s}$

As fast as **reference** solver (Tranair), but **scaling** has to be **investigated**!

Outline – development

Modeling

- Flow physics
- Aerodynamic models
- Potential equation

Methods

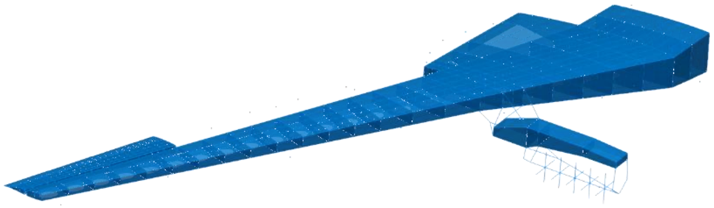
- Field panel method
- Finite element method

Fluid-structure interaction framework

- Structural model
- Framework
- CUPyDO

Structural model

Structural modal solver (NASTRAN)

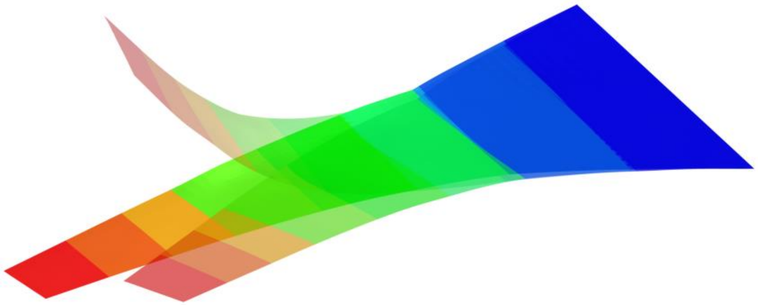


$$\boldsymbol{M}\ddot{\boldsymbol{u}} - \boldsymbol{K}\boldsymbol{u} = \boldsymbol{f} \longleftarrow \boldsymbol{u} = \boldsymbol{w}\mathrm{e}^{j\omega t}$$

$$(\boldsymbol{K} - \omega^2 \boldsymbol{M})\boldsymbol{w} = 0 \longrightarrow \begin{cases} \omega_k \\ \boldsymbol{w}_k \end{cases}$$

Modal interpreter (modali)

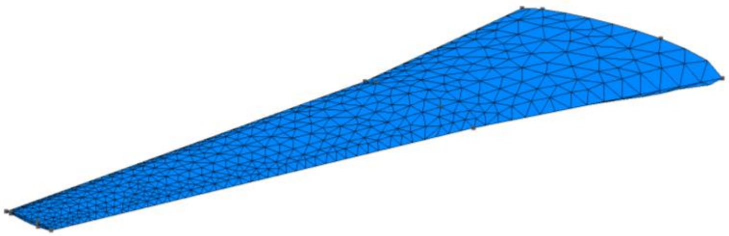
first mode shapes: $\boldsymbol{w}_k$



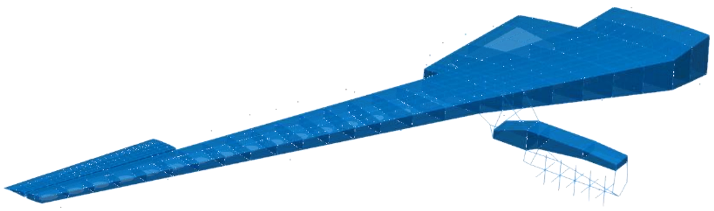
$$\longrightarrow \begin{cases} \boldsymbol{w} = \boldsymbol{W}\boldsymbol{q} \\ \boldsymbol{K}_\mathrm{q} = \boldsymbol{W}^\mathrm{T}\boldsymbol{K}\boldsymbol{W} \\ \boldsymbol{f}_\mathrm{q} = \boldsymbol{W}^\mathrm{T}\boldsymbol{f} \end{cases} \longrightarrow \boldsymbol{K}_\mathrm{q}\boldsymbol{q} = -\boldsymbol{f}_\mathrm{q}$$

Framework – toolchain

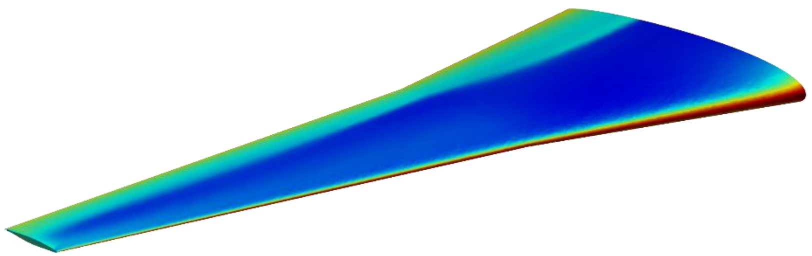
Geometry – gmsh



Structural solver – NASTRAN

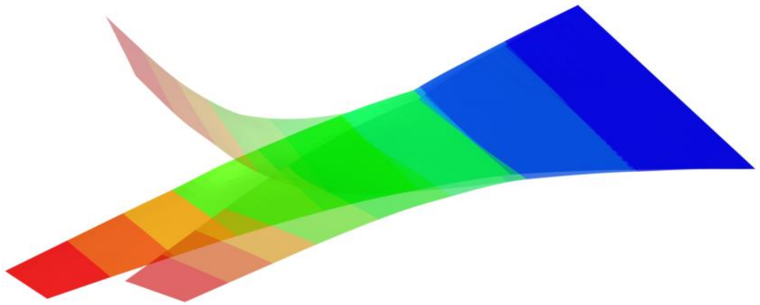


Aerodynamics – Flow

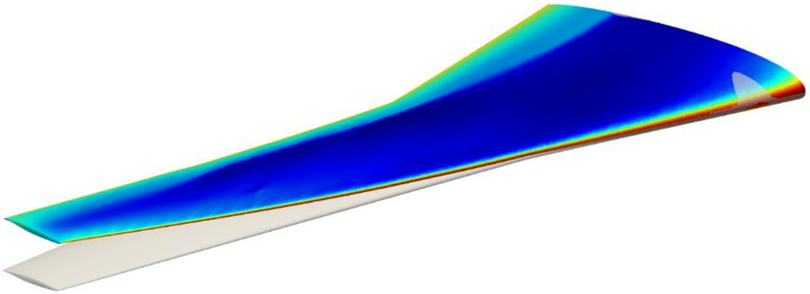


+

Modal interpreter – modali



Aeroelasticity – CUPyDO



CUPyDO – architecture

Fluid solver

Core code (C/C++)

SWIG

Interface (python)

FSI coupler

CUPyDO (python)

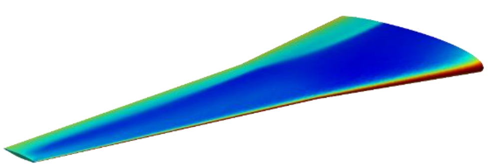
```
# FSI loop
while not convrgd:
  l = fluid.run()
  solid.setLoads(l)
  d = solid.run()
  fluid.setDefo(d)
```

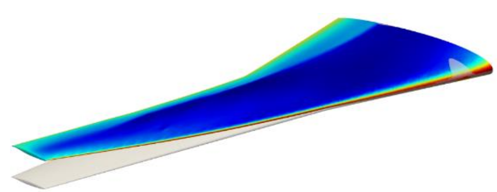
Solid solver

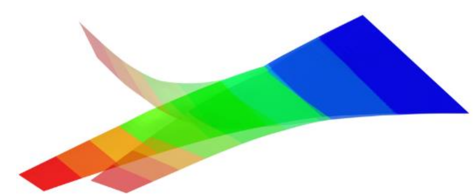
Core code (C/C++)

SWIG

Interface (python)





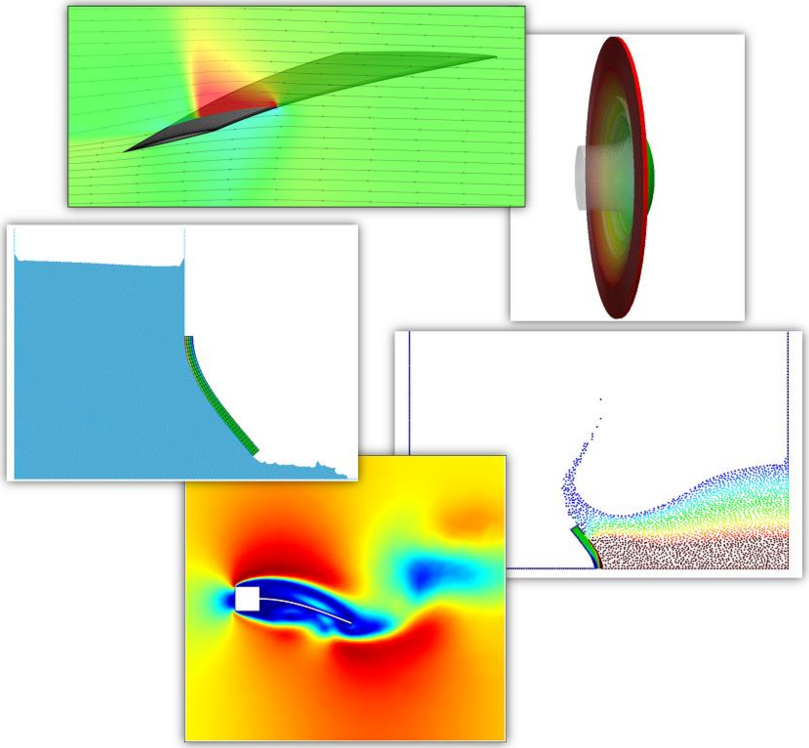


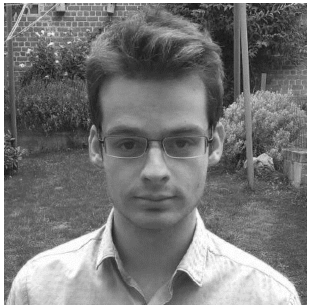
CUPyDO – contributors

Romain Boman

David Thomas

Marco L. Cerquaglia





Adrien Crovato

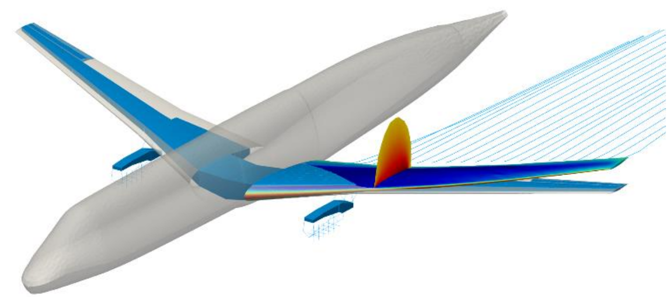
Mariano Sanchez M.

Thesis objectives

Development

Develop an aerodynamic solver for fast transonic load computation and **integrate** it into a fluid-structure framework

Benchmark



Evaluate and **compare** the aerodynamic models in the context of rigid and static transonic aeroelastic computations

Outline – benchmark

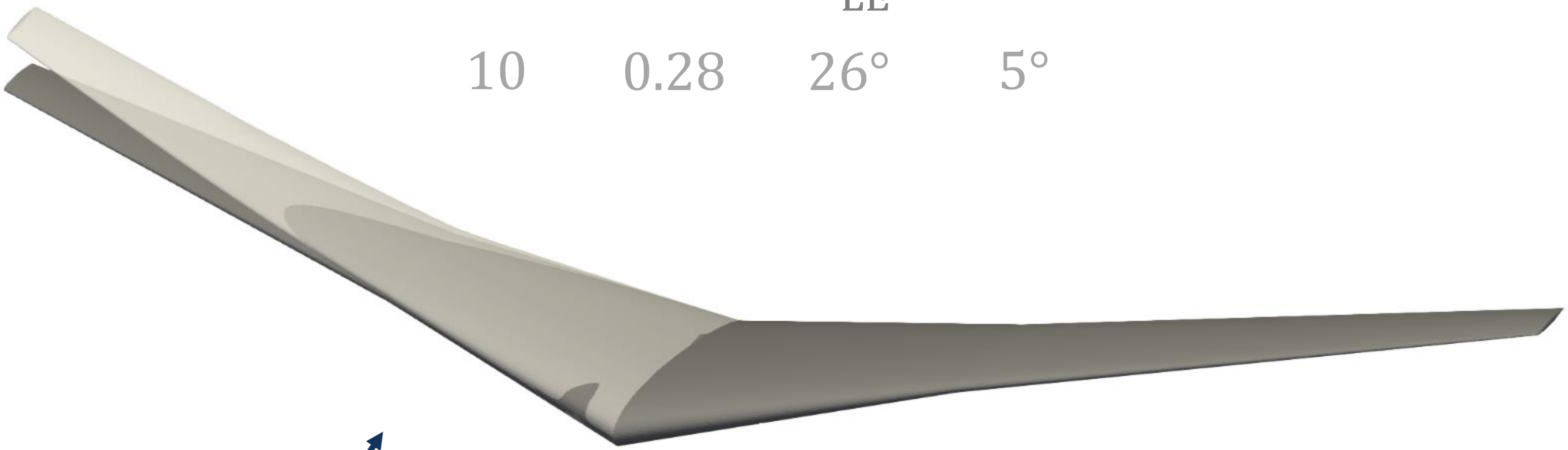
Case definition

- Benchmark wing
- Fluid solvers

Analyses

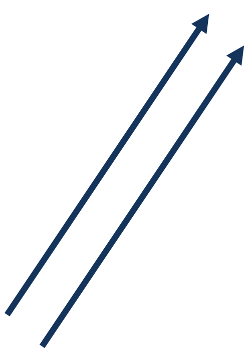
- Rigid aerodynamic
- Static aeroelastic

Benchmark wing



Embraer benchmark wing

AR	λ	Λ_{LE}	Γ
10	0.28	26°	5°



Flow

M	FL	C_L	n_z
0.78	270	0.60	2.0

Solvers

SU2	RANS	Finite volume	RANS
Tranair	Full potential viscous-inviscid	Finite element	FPV
SU2	Euler	Finite volume	EUL
Flow	Full potential	Finite element	FP
Panair	Linear potential	Doublet + source	LP
NASTRAN +F2AJ	Linear potential corr. by Euler	Doublet lattice	LP LPC

Outline – benchmark

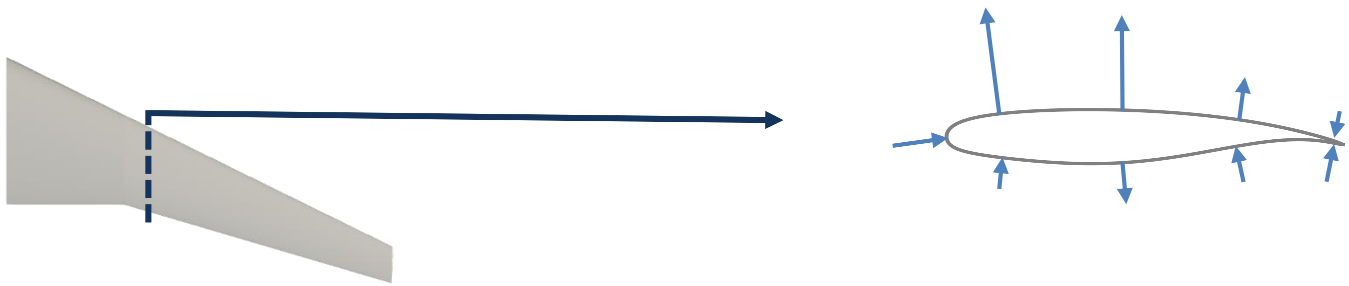
Case definition

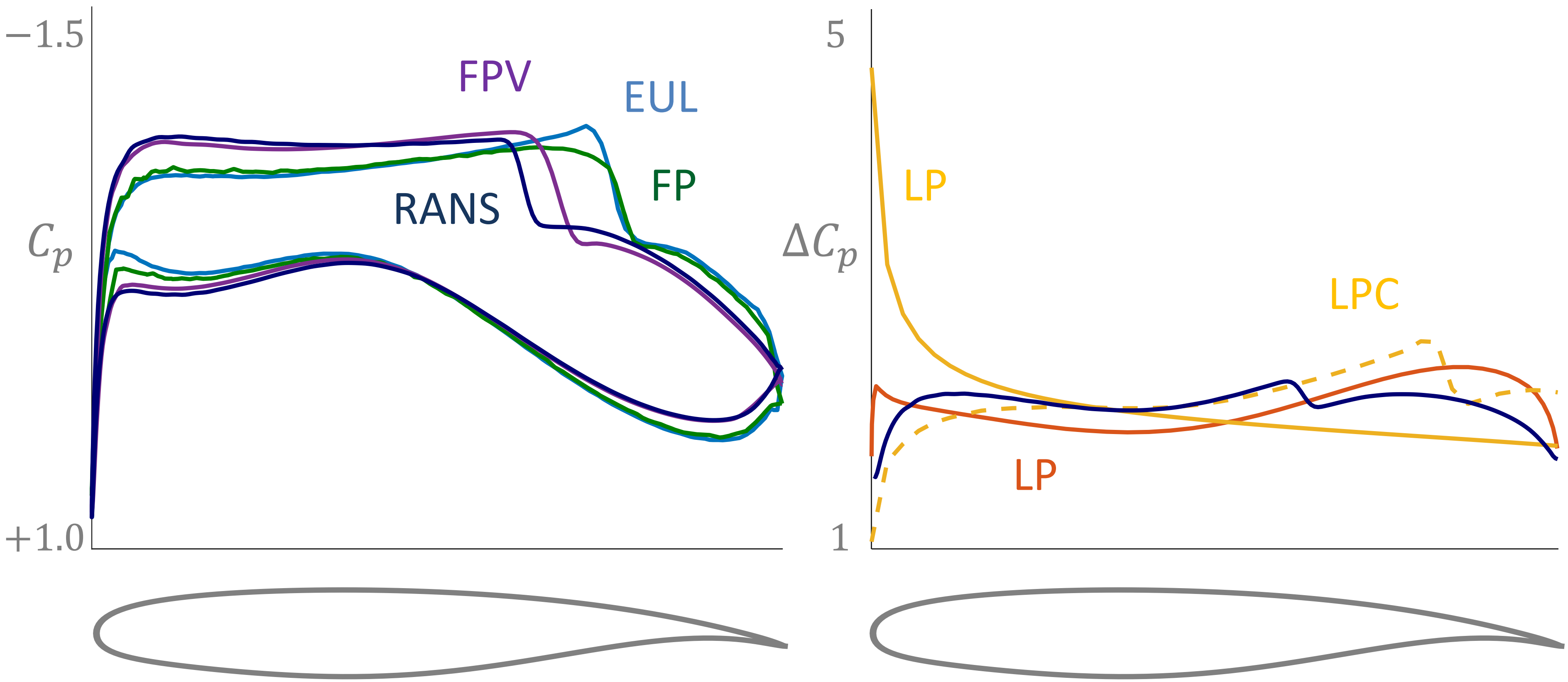
- Benchmark wing
- Fluid solvers

Analyses

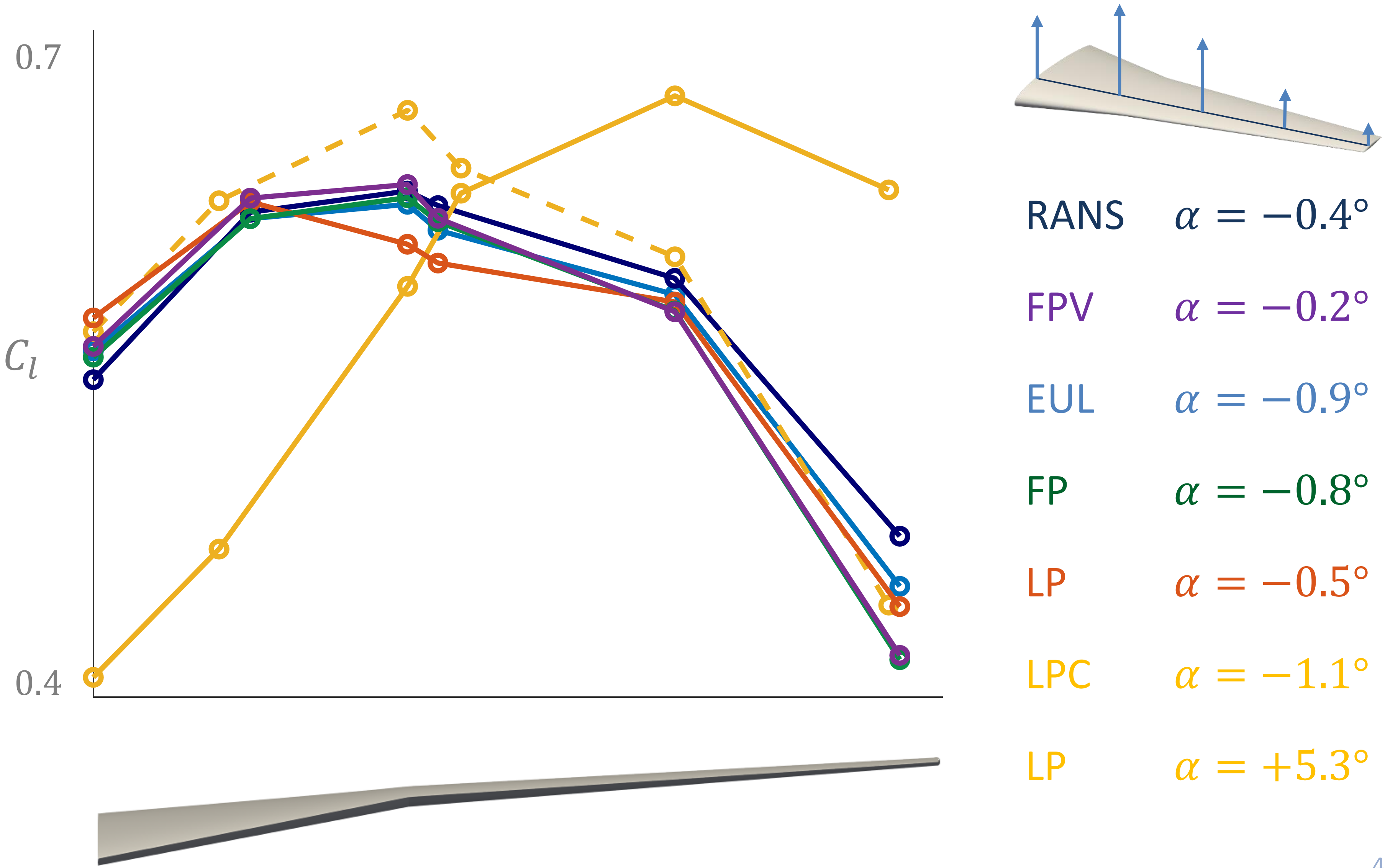
- Rigid aerodynamic
- Static aeroelastic

Aerodynamic computations – pressure

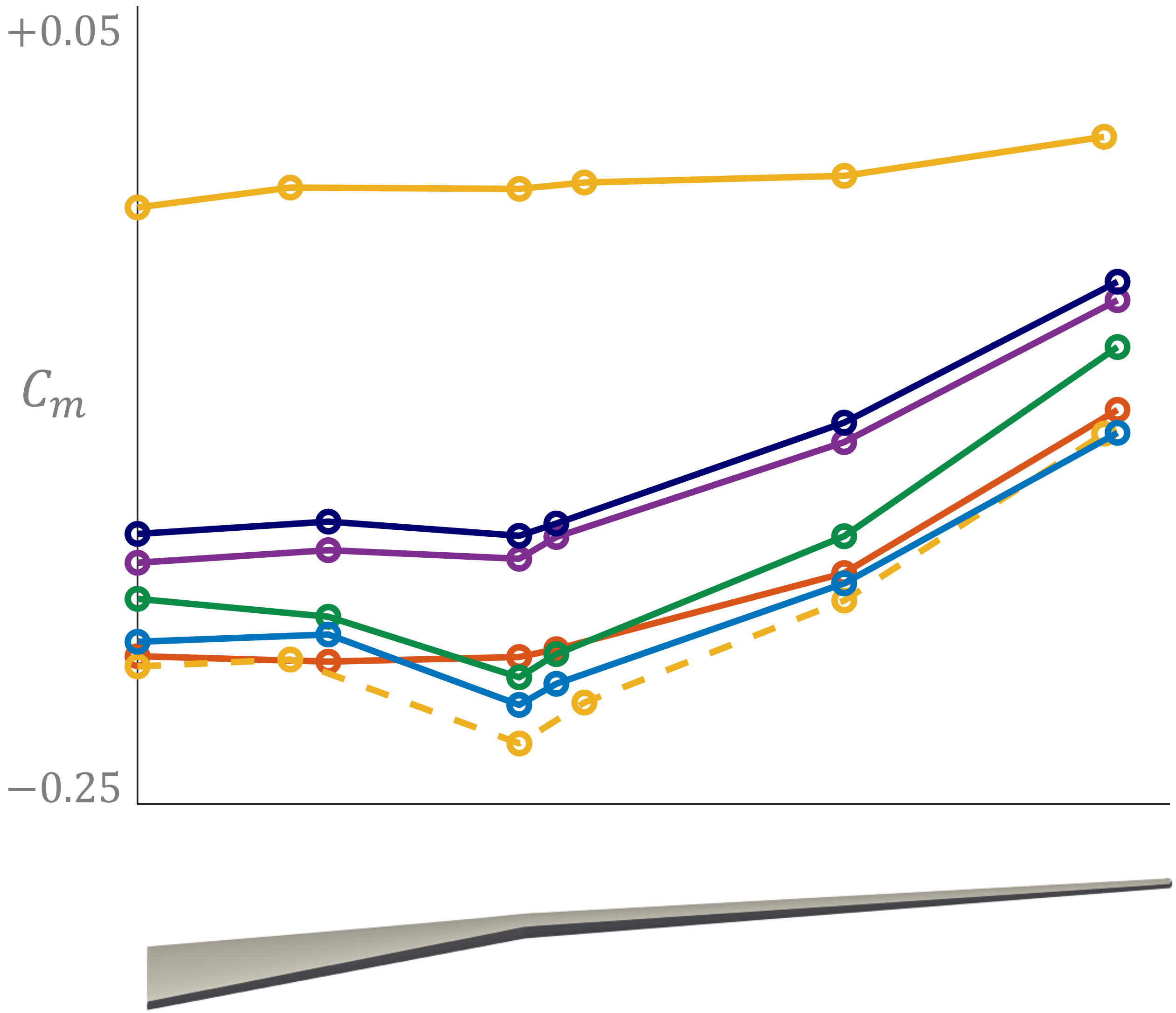


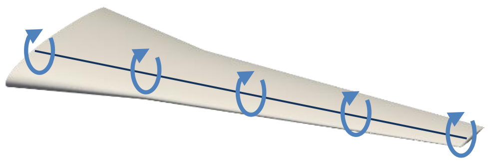


Aerodynamic computations – lift



Aerodynamic computations – moment





RANS $\alpha = -0.4°$

FPV $\alpha = -0.2°$

EUL $\alpha = -0.9°$

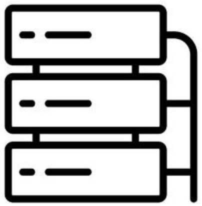
FP $\alpha = -0.8°$

LP $\alpha = -0.5°$

LPC $\alpha = -1.1°$

LP $\alpha = +5.3°$

Aerodynamic computations – cost



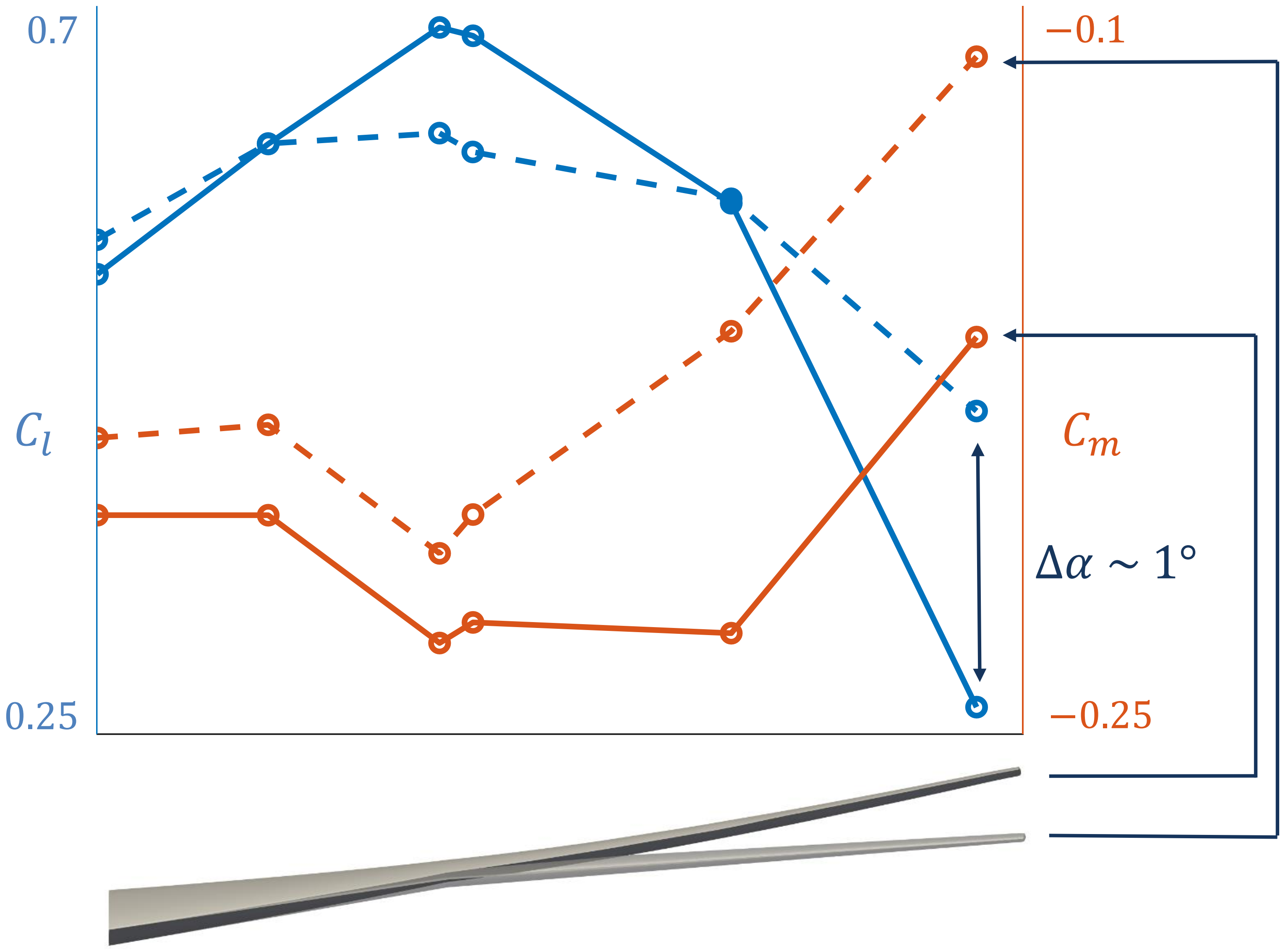
Intel Xeon
2.0 GHz

RANS	$n_{\mathrm{CPU}} = 60$	$t_{\mathrm{real}} = 48\ \mathrm{h}$
EUL	$n_{\mathrm{CPU}} = 12$	$t_{\mathrm{real}} = 30\ \mathrm{min}$

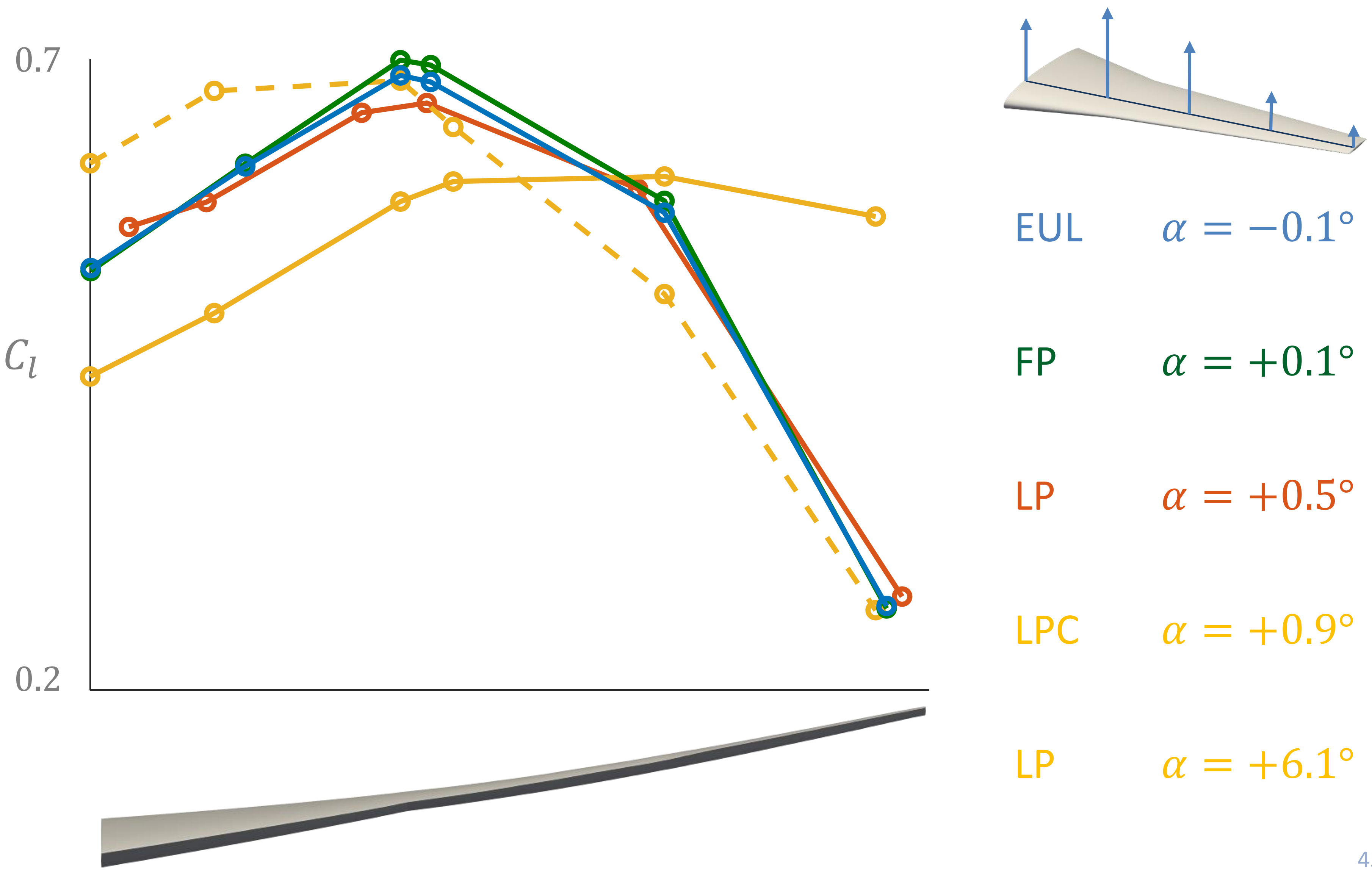
Intel i7
2.8 GHz

FPV	$n_{\mathrm{CPU}} = 1$	$t_{\mathrm{real}} = 12\ \mathrm{min}$
FP	$n_{\mathrm{CPU}} = 1$	$t_{\mathrm{real}} = 5\ \mathrm{min}$
LP	$n_{\mathrm{CPU}} = 1$	$t_{\mathrm{real}} = 10\ \mathrm{s}$
LPC	$n_{\mathrm{CPU}} = 1$	$t_{\mathrm{real}} = 25\ \mathrm{s}\ (+\mathrm{t_{hf}})$
LP	$n_{\mathrm{CPU}} = 1$	$t_{\mathrm{real}} = 25\ \mathrm{s}$

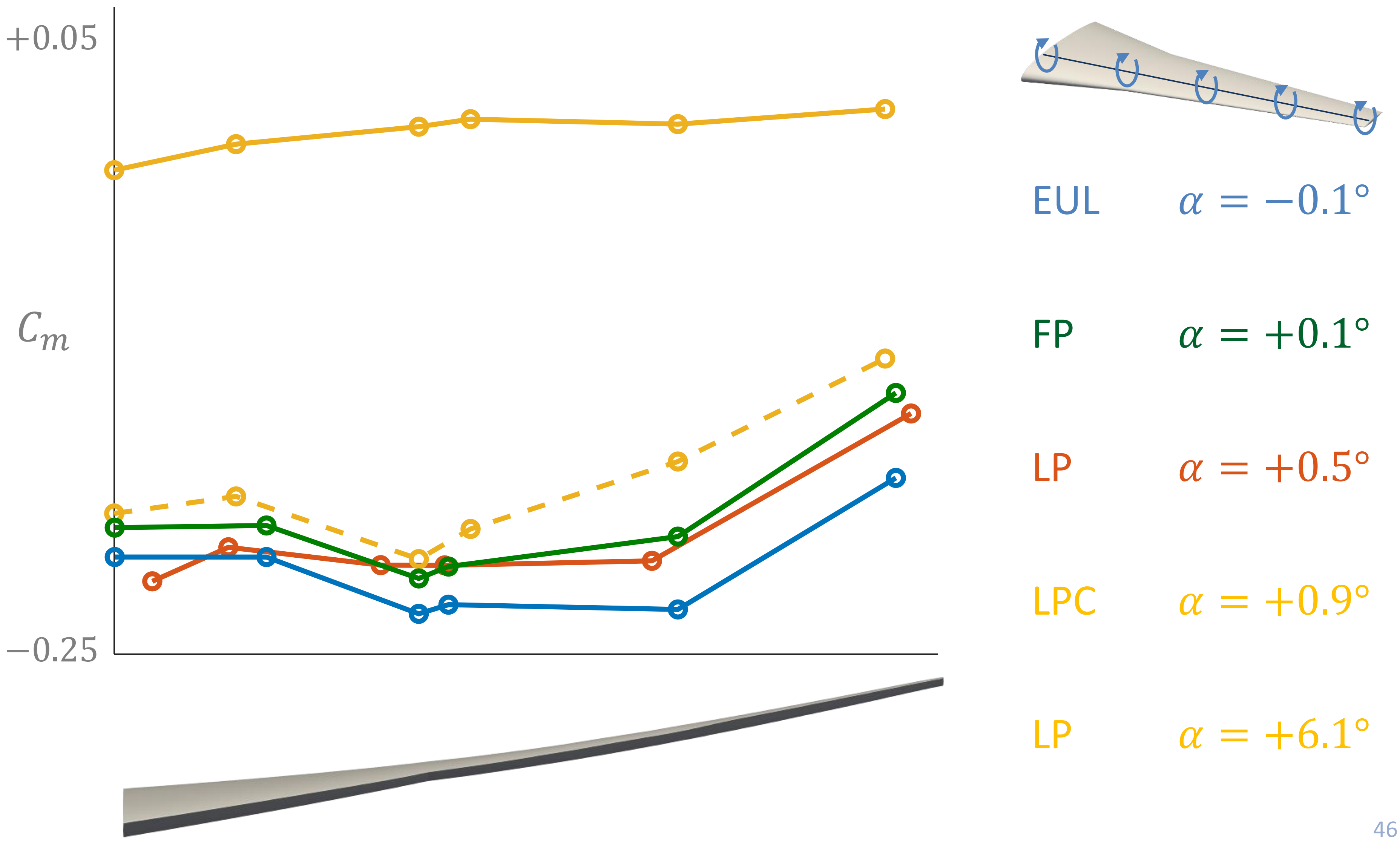
From cruise to maneuver flight shape



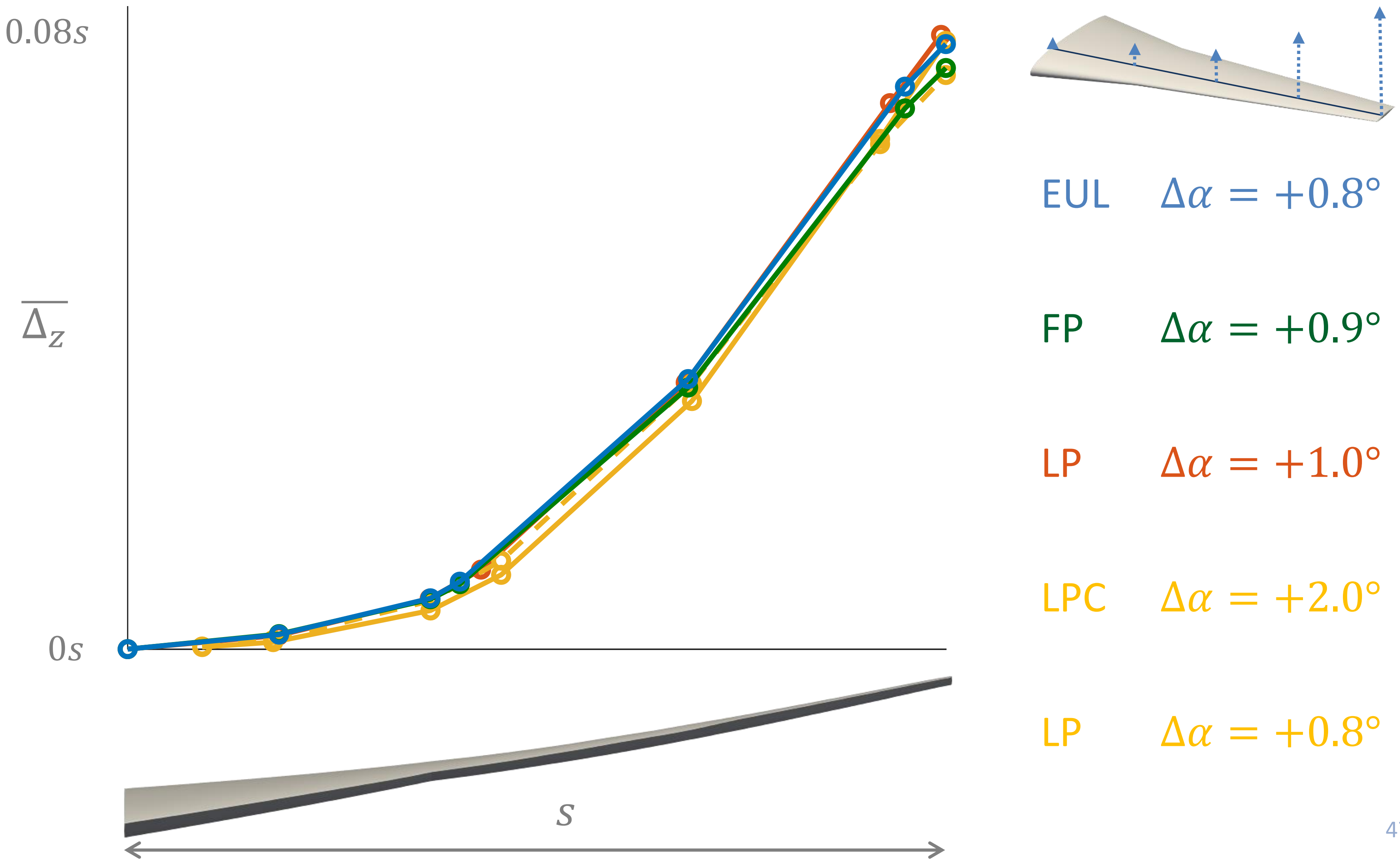
Aeroelastic computations – lift



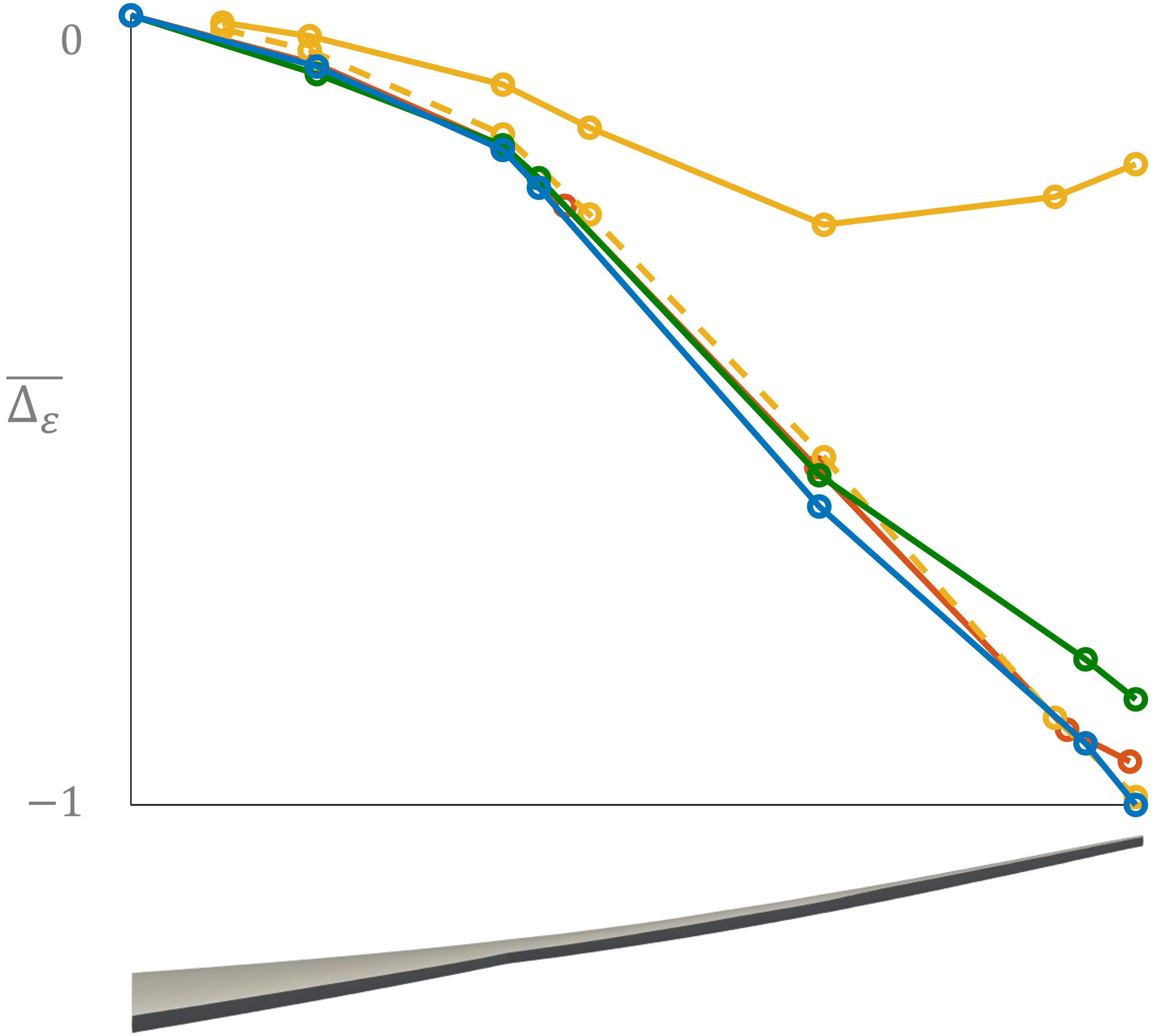
Aeroelastic computations – moment

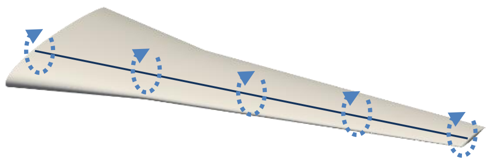


Aeroelastic computations – bending



Aeroelastic computations – torsion





EUL $\Delta\alpha = +0.8°$

FP $\Delta\alpha = +0.9°$

LP $\Delta\alpha = +1.0°$

LPC $\Delta\alpha = +2.0°$

LP $\Delta\alpha = +0.8°$

Aeroelastic computations – cost

Intel i7
2.8 GHz

EUL	$n_{\text{CPU}} = 1$	$t_{\text{real}} = 12\ \text{h}$
FP	$n_{\text{CPU}} = 1$	$t_{\text{real}} = 25\ \text{min}$
LP	$n_{\text{CPU}} = 1$	$t_{\text{real}} = 90\ \text{s}$
LPC	$n_{\text{CPU}} = 1$	$t_{\text{real}} = 75\ \text{s} + \text{t}_{\text{hf}}$
LP	$n_{\text{CPU}} = 1$	$t_{\text{real}} = 75\ \text{s}$

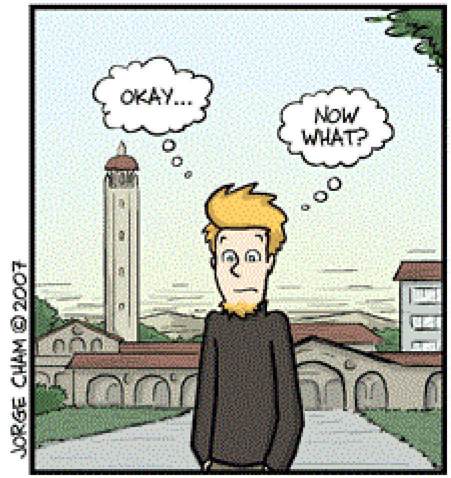
Outline – Conclusions

Conclusion

- Benchmark
- Development

Perspectives

- Aeroelasticity
- Flow code



Conclusions

Benchmark

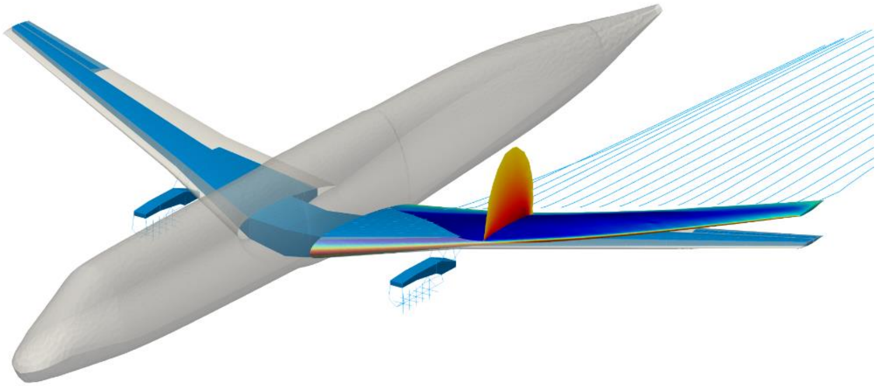
- **Linear potential** is **fast** but **not adapted** for transonic flows while **higher-fidelity Euler** and **RANS** are **slow**, **full potential** offers a **good tradeoff**
- **Lattice method** predictions are accurate only if **corrected**
- **Linear potential** yields static aeroelastic **deflections similar** to **nonlinear** models

Development

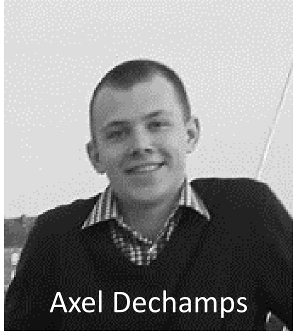
- A **field panel** code has been **developed** but does **not** yield **satisfying** results
- **Flow** – a **finite element** code has been **developed** and yields **similar** results to **commercial** code
- **Flow** has been **integrated** into a fluid-structure interaction framework **CUPyDO**

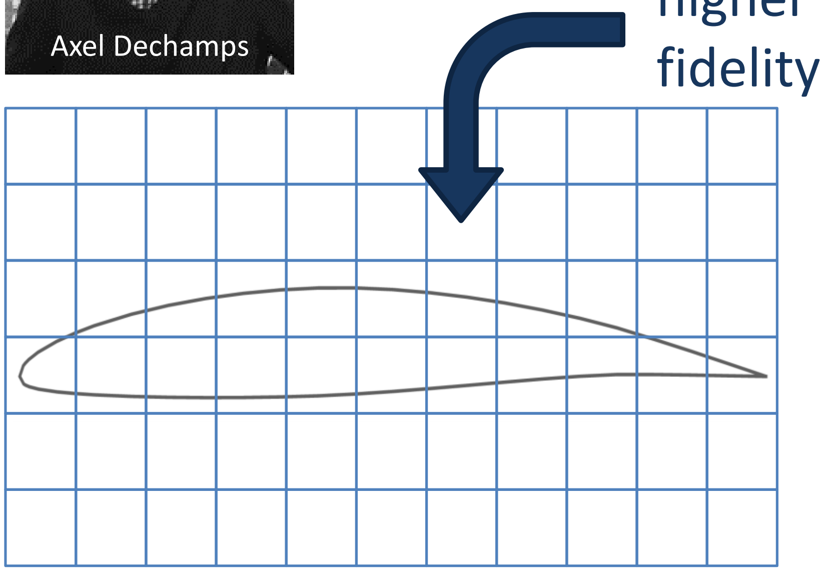
Perspectives – aeroelastic modeling

Improved aeroelasticity



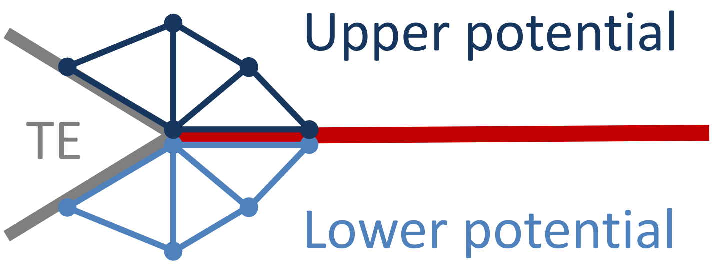
Multi-fidelity aeroelasticity



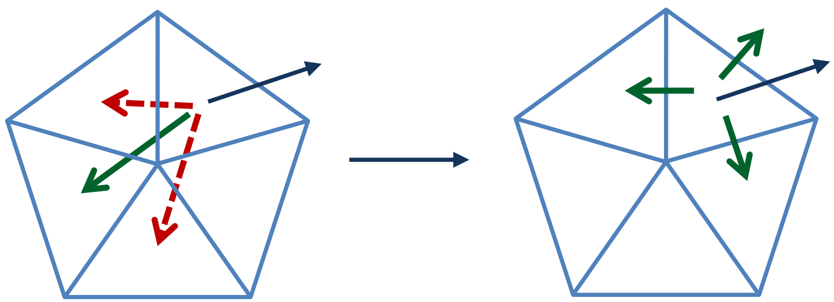


Perspectives – finite element code

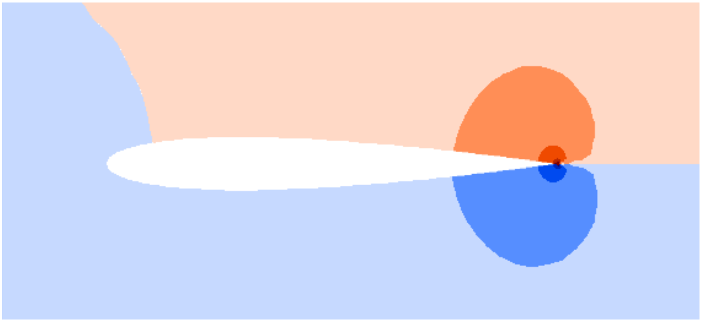
Wake model



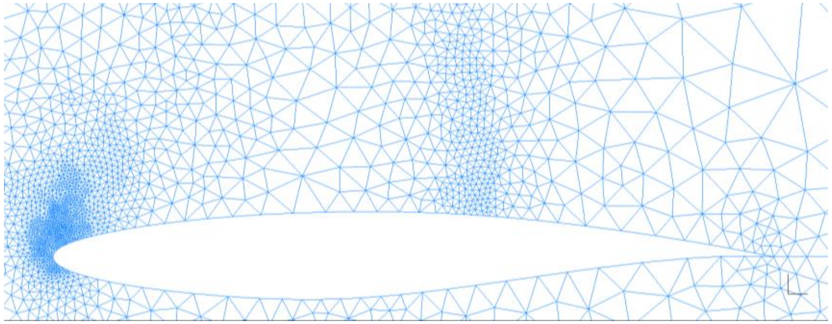
Numerical scheme



Adjoint computations

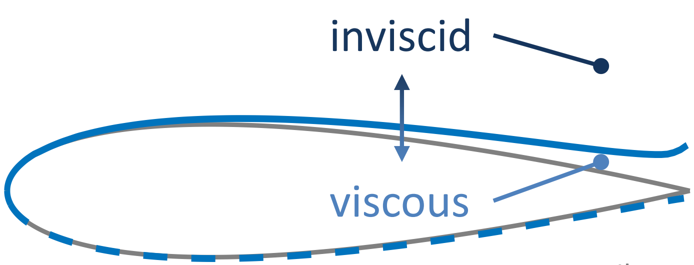


Solution adaptive grid



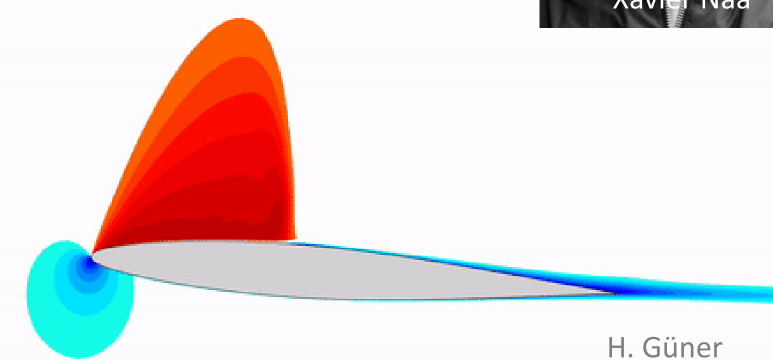
Perspectives – finite element code

Viscous-inviscid interaction



A. Bilocq

Unsteady modeling

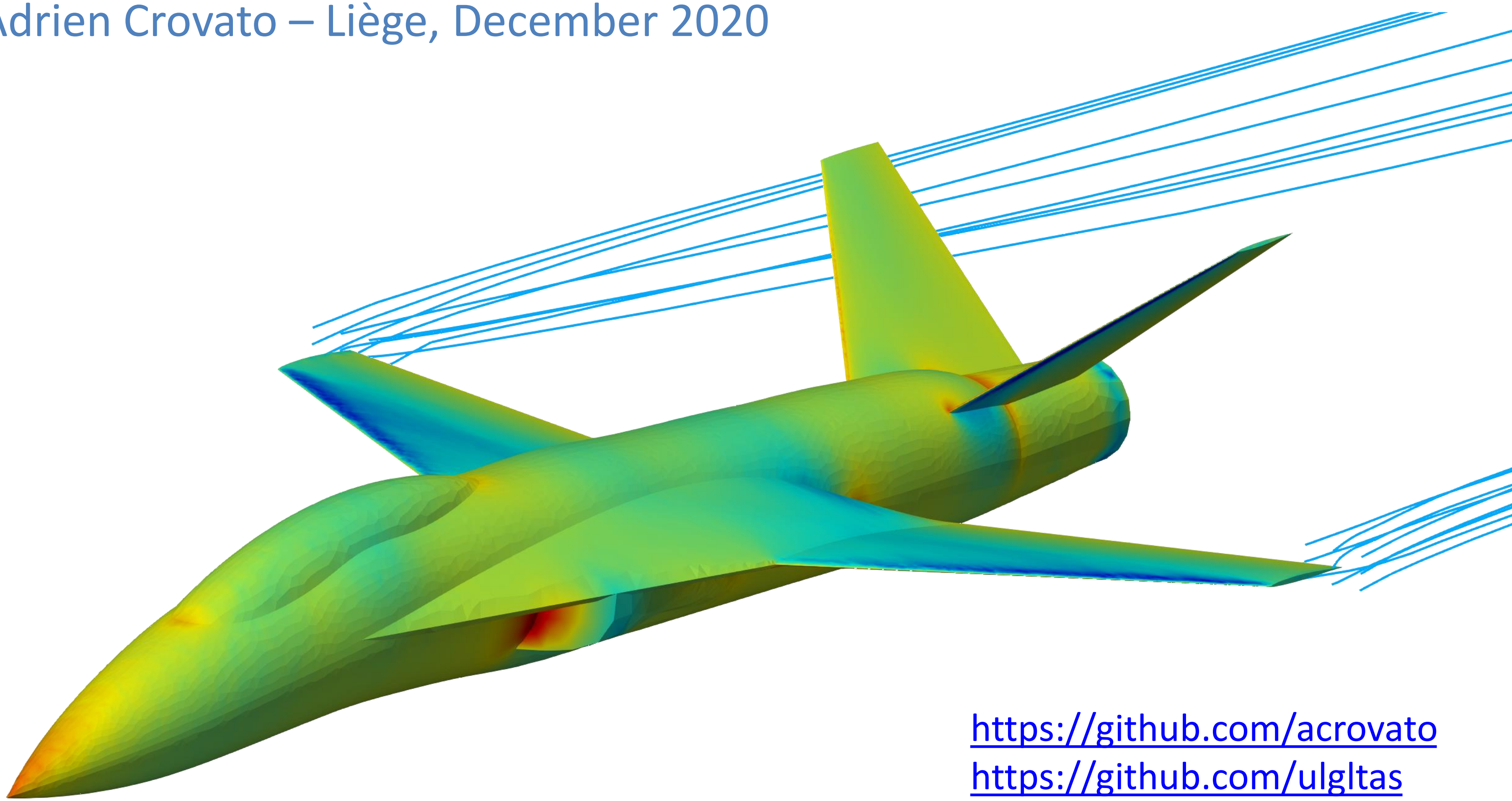


H. Güner

Thesis defense

Transonic aeroelastic modeling

Adrien Crovato – Liège, December 2020



https://github.com/acrovato
https://github.com/ulgltas
https://gitlab.uliege.be/am-dept

EMBRAER

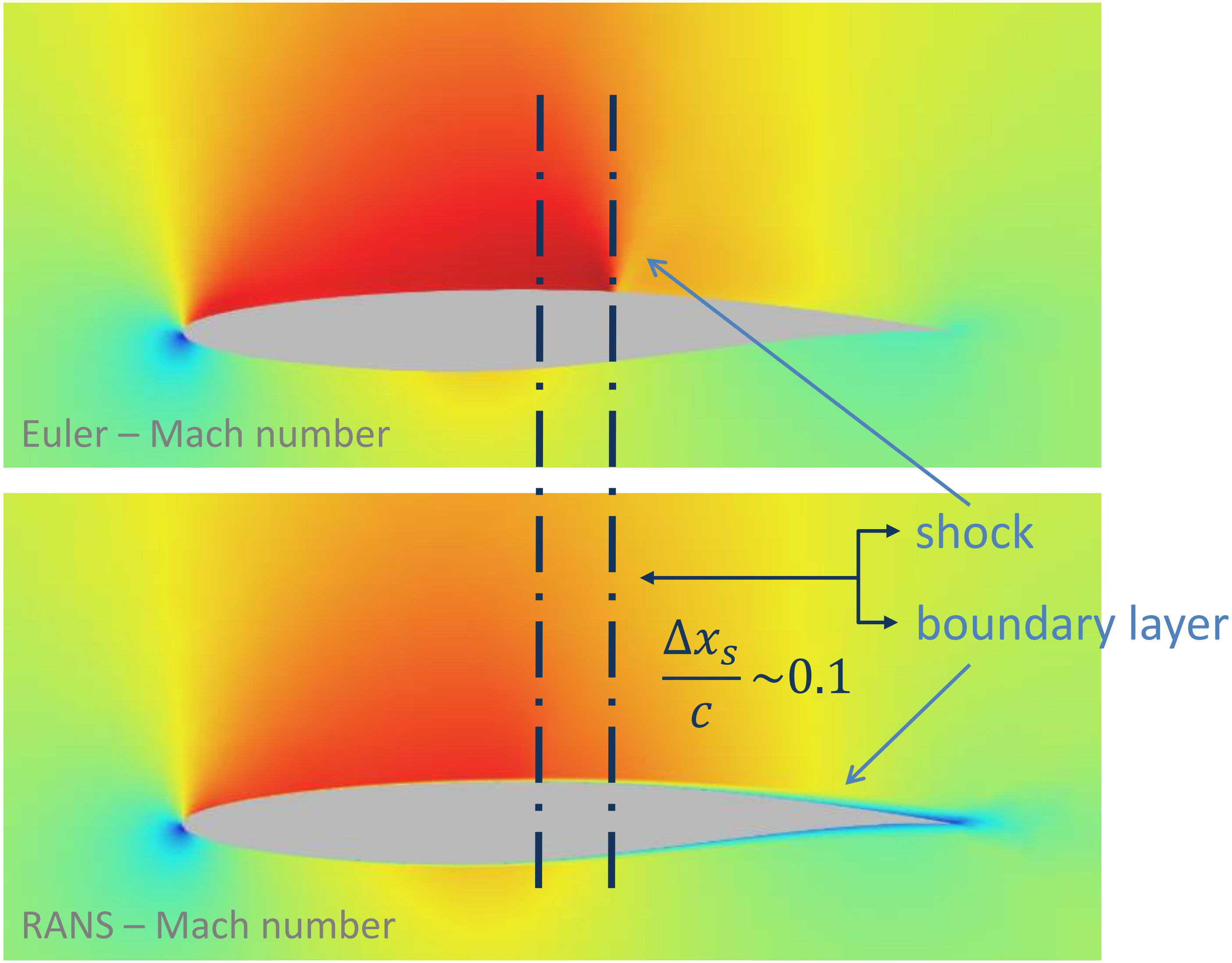
fnrs
LA LIBERTÉ DE CHERCHER

LIÈGE université

Flow physics – shock and boundary layer

RAE 2822

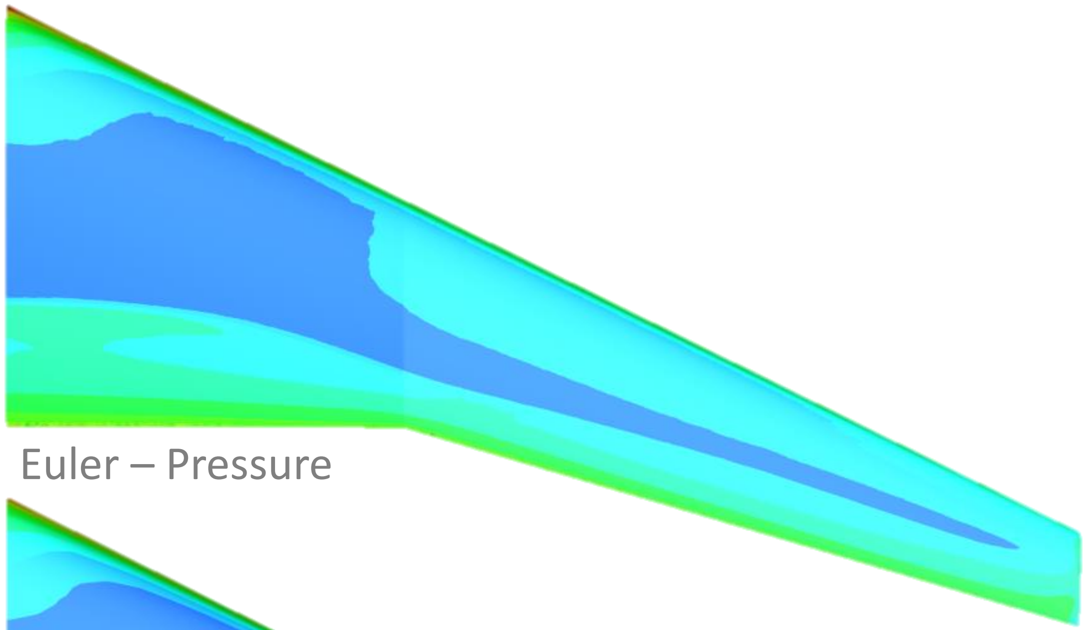
$$M = 0.715$$
$$\alpha = 2°$$



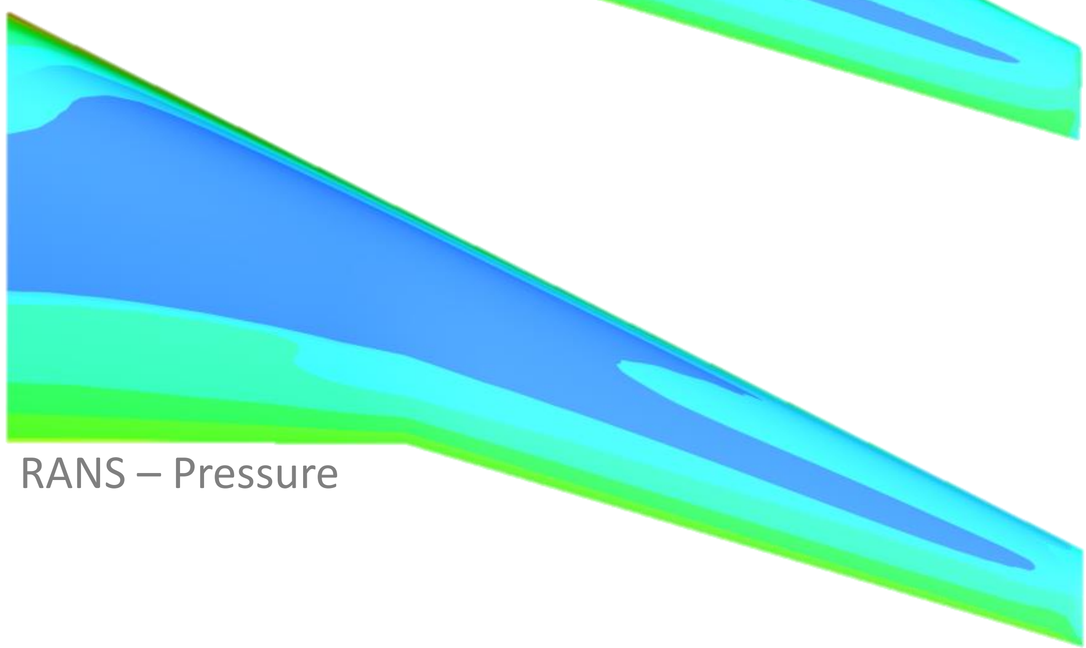
Flow physics – shock and boundary layer

EBW2

$M = 0.78$
$C_L = 0.60$



Euler – Pressure

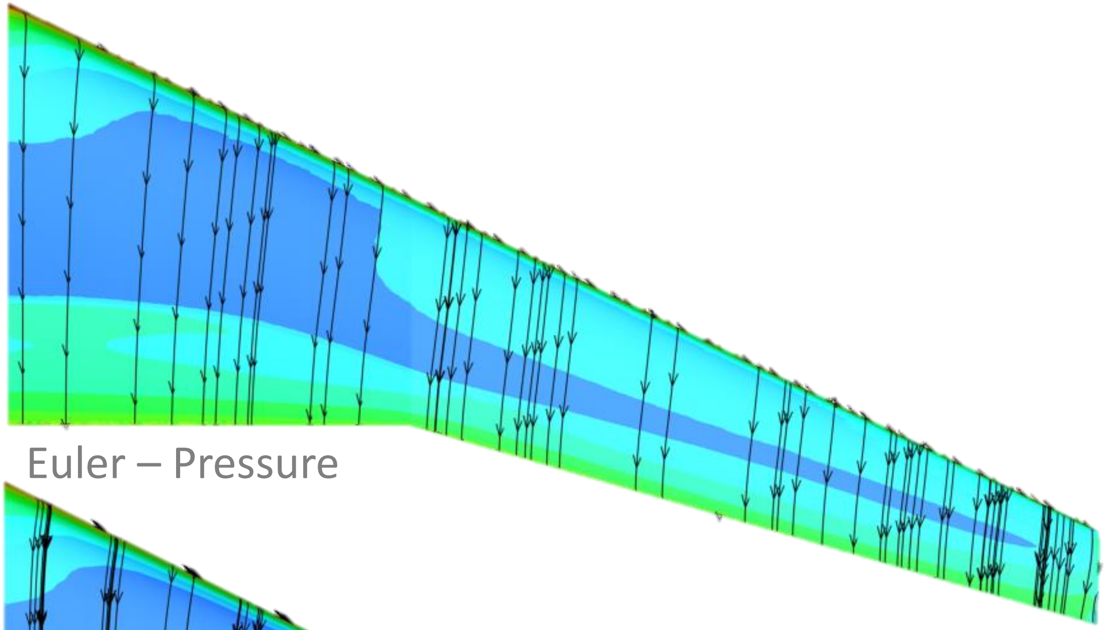


RANS – Pressure

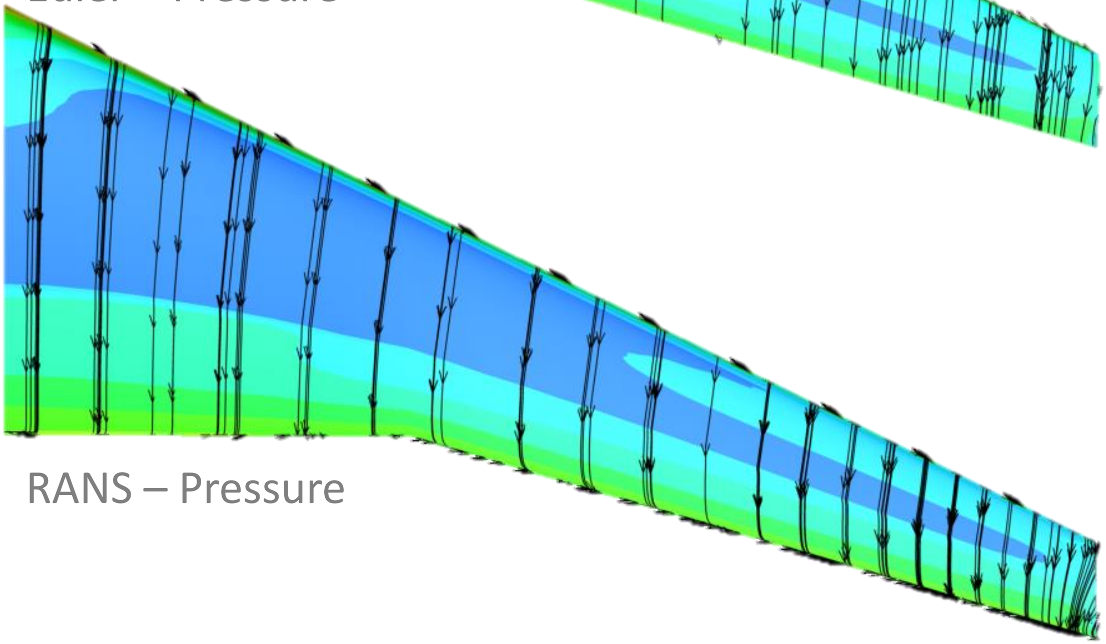
Flow physics – flow pattern

EBW2

$M = 0.78$
$C_L = 0.60$



Euler – Pressure

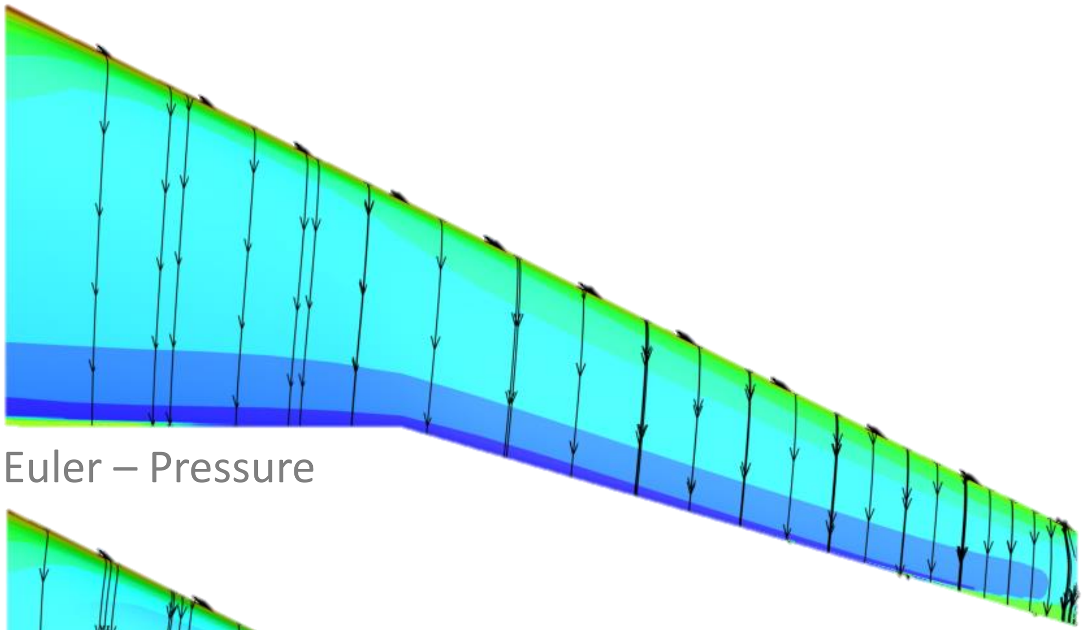


RANS – Pressure

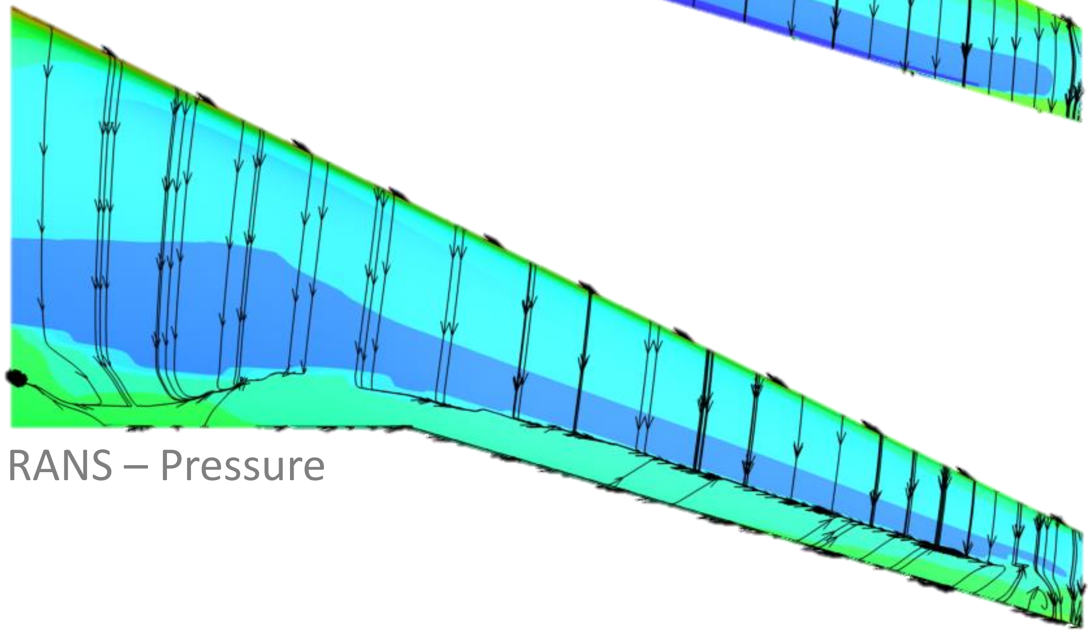
Flow physics – flow pattern (high-speed)

EBW2

$M = 0.89$
$C_L = 0.36$



Euler – Pressure



RANS – Pressure

Flow physics – drag

EBW2

$M = 0.78$
$C_L = 0.60$

FPA: $C_{D_f} = \frac{S}{S_{\text{ref}}} C_f \sim 2.1 \frac{0.027}{Re^{1/7}} \sim 0.0050$

Lock: $C_{D_w} = 20 \times (M - M_{\text{cr}})^4 \sim 0.0008$

wave →

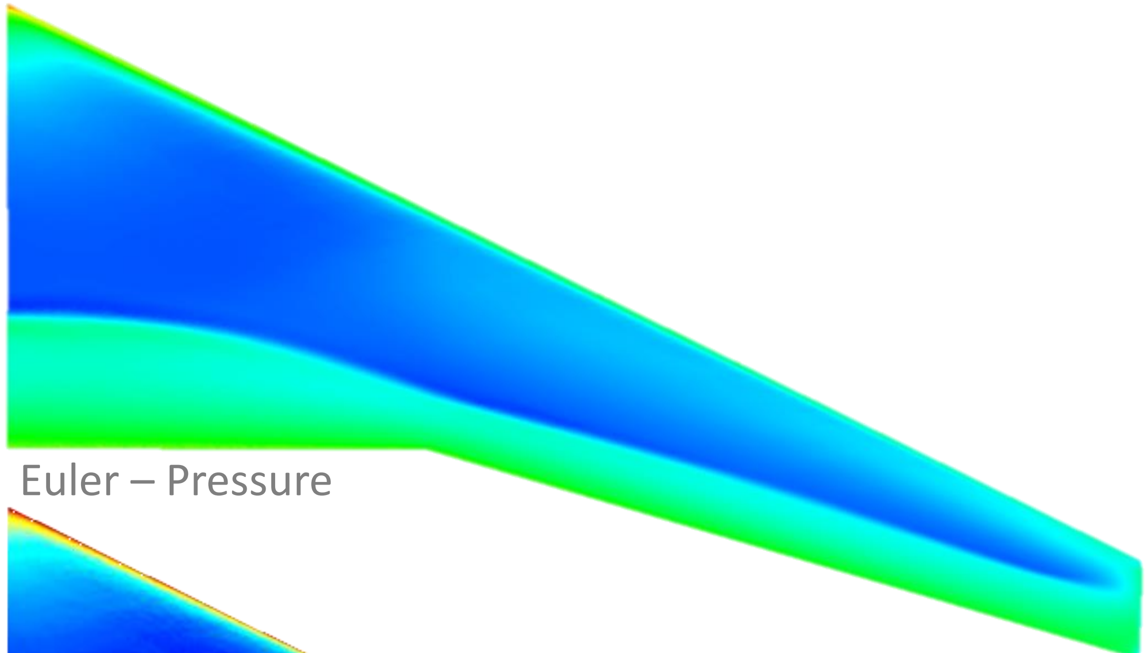
friction
form →

C_D	Linear inviscid	Nonlinear inviscid	Nonlinear viscous
C_{D_f}			0.0052 0.0050
C_{D_p}	0.0136	0.0167 0.0159	0.0192 0.0191

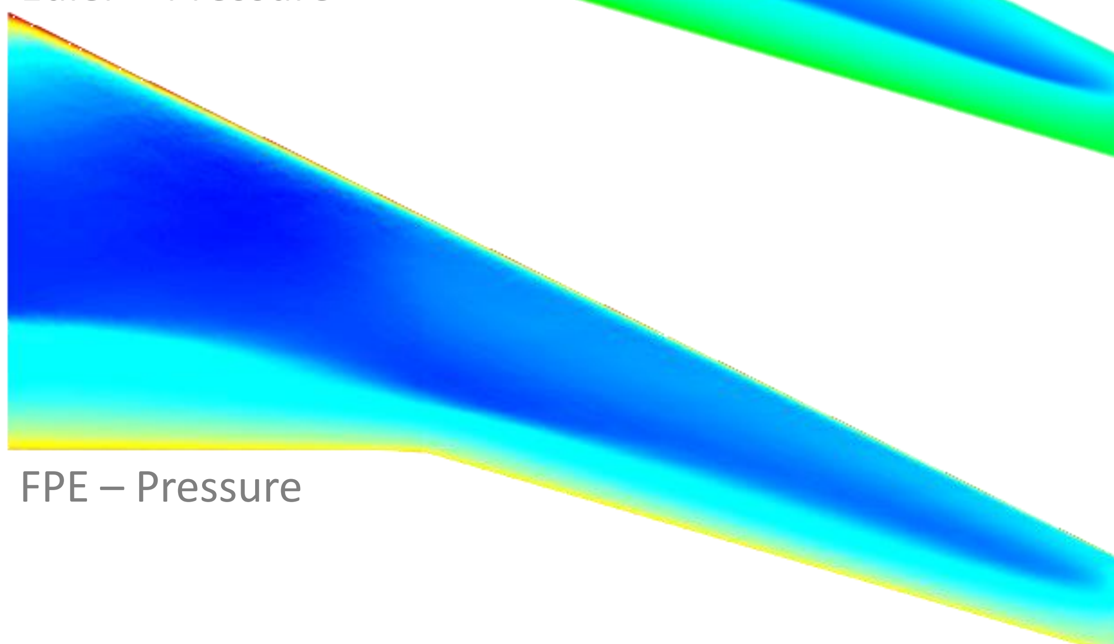
Numerical modeling – wingtip flow

EBW2

$$M = 0.78$$
$$C_L = 0.60$$



Euler – Pressure

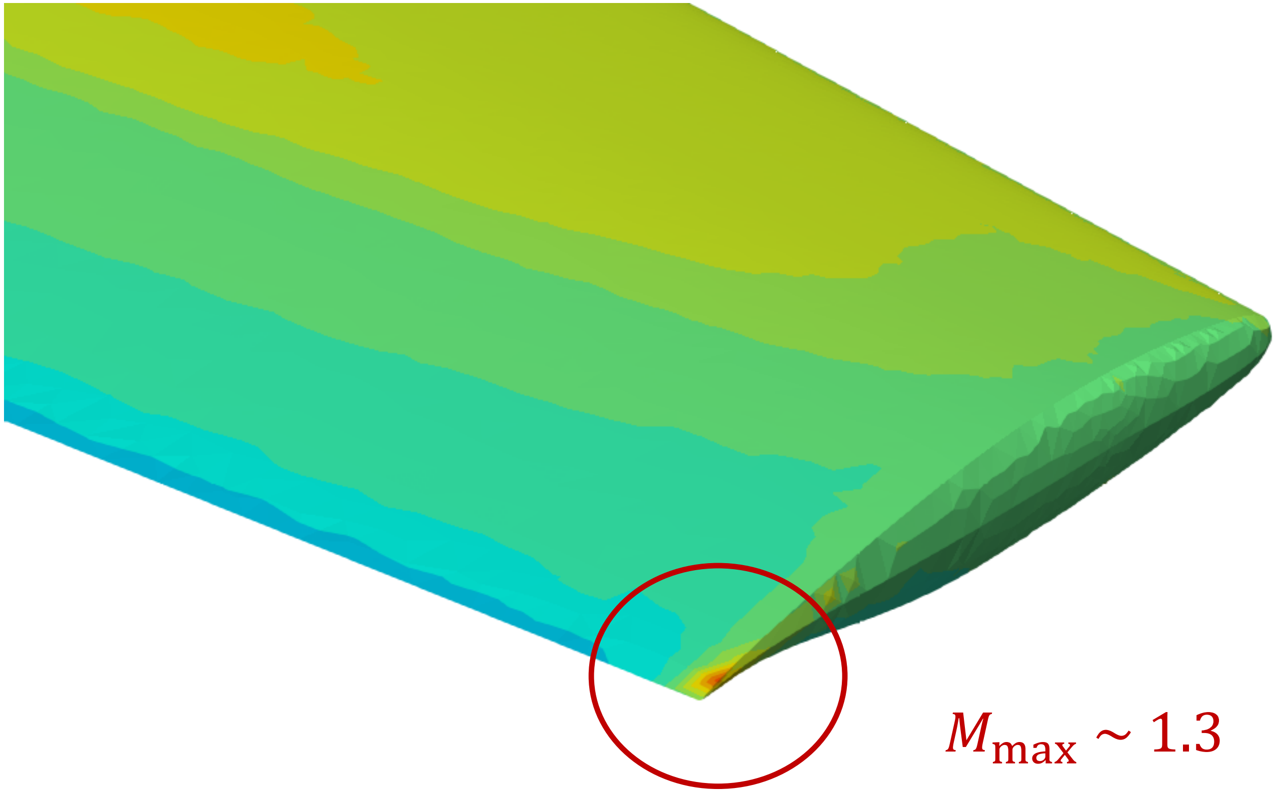


FPE – Pressure

Finite element code – wingtip vortex

EBW2

$M = 0.78$
$C_L = 0.60$



$M_{\max} \sim 1.3$
$M_{\max,\text{tip}} \sim 1.9$

Framework – python interface

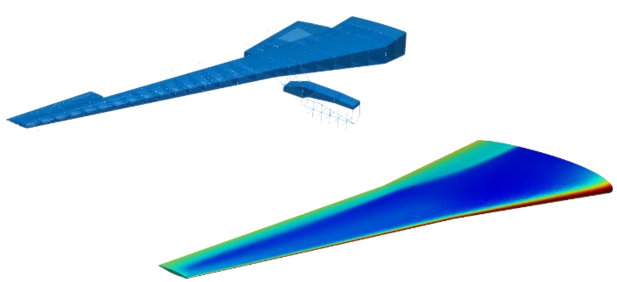
Core code (C/C++)

```
// Mesh data structure
class Mesh {
//… };

// Solver
class Solver {
//…
void run(); };
```

SWIG

Interface (python)

```
# Build mesh
msh = Mesh()

# Run solver
sol = Solver(msh)
sol.run()
```

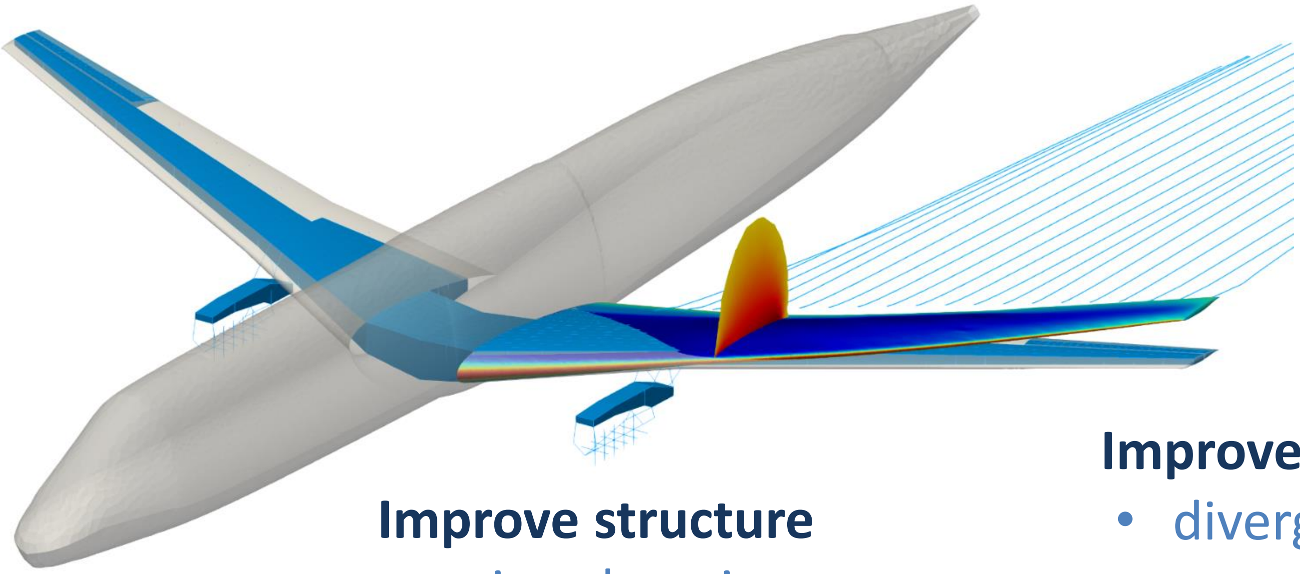


- ✓ Modular
- ✓ Efficient
- ✓ User-friendly

Perspectives – aeroelastic computations

Improve model
- full aircraft
- gravity load

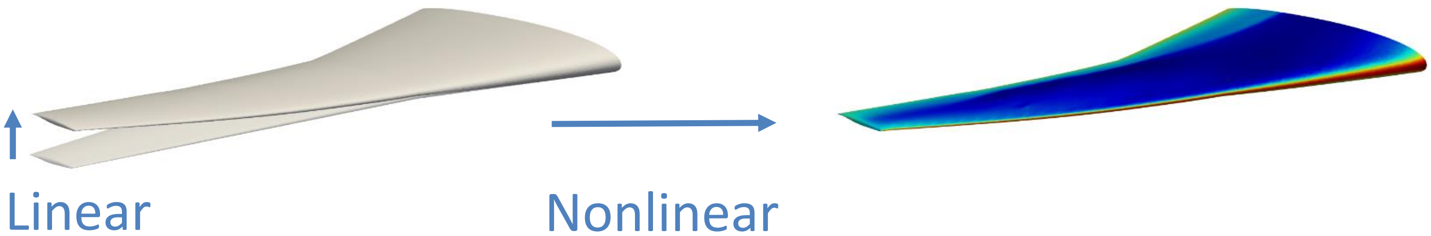
Improve flow
- RANS
- potential viscous-inviscid
- drag
- transition

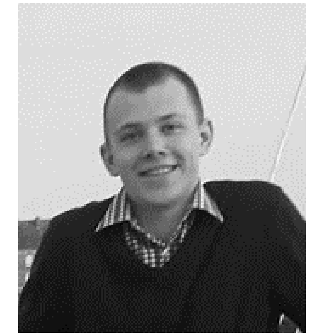


Improve structure
- wing clamping

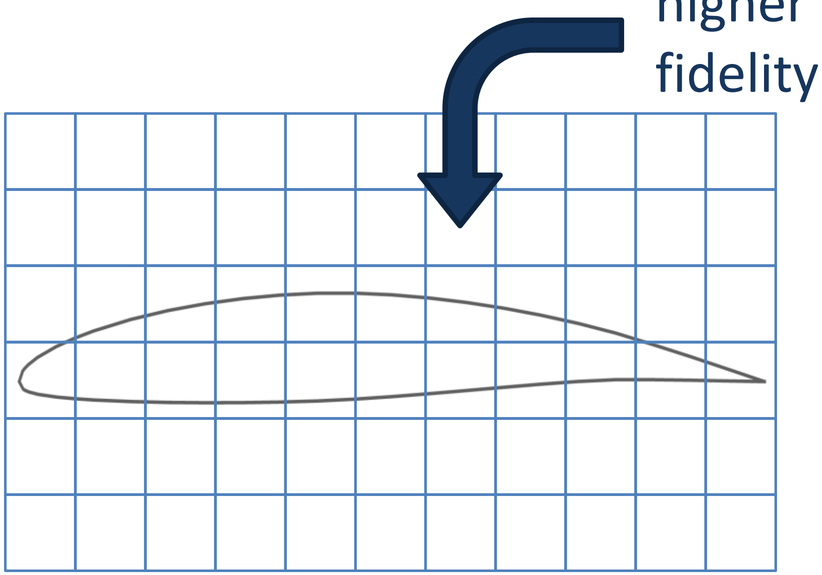
Improve output
- divergence speed

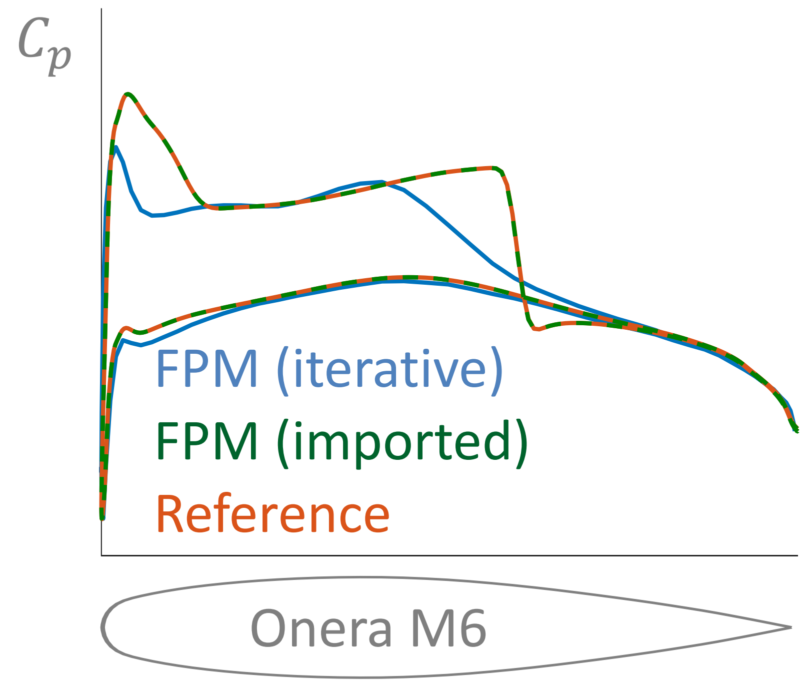
Perspectives – multi-fidelity aeroelasticity



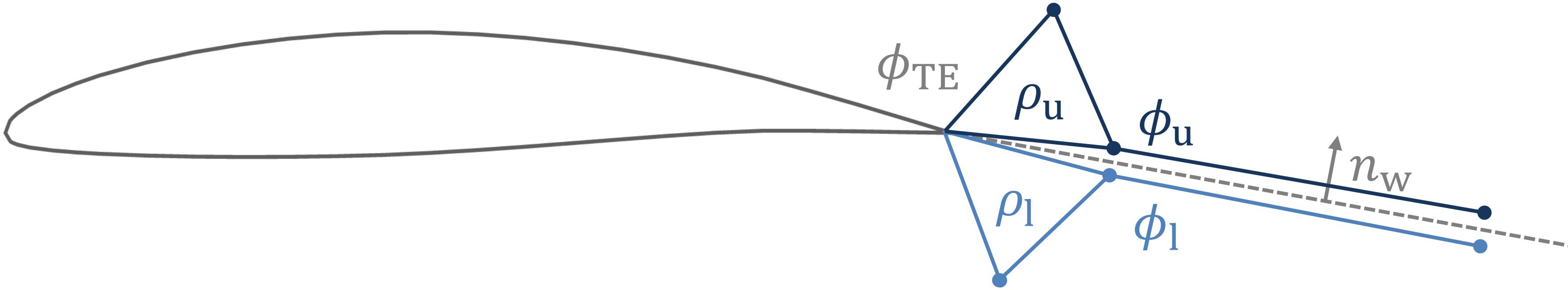


Axel Dechamps





Perspectives – wake model



3D results depend on wake inclination!

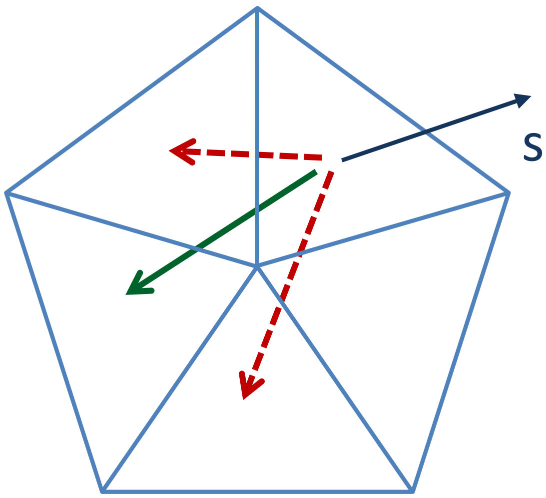
2D flows

- add contribution on TE

3D flows

- unstable!
- find suitable inclination…

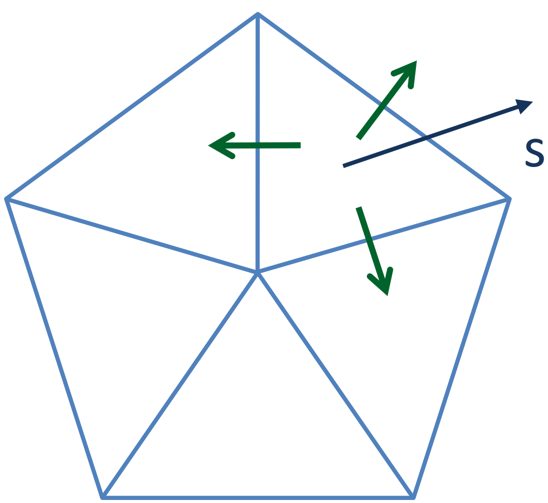
Perspectives – numerical scheme



Density upwinding

$$\rho \leftarrow \rho - \nu \frac{\partial \rho}{\partial s} \Delta s$$

$$\rho \leftarrow \rho - \nu \overleftarrow{\Delta} \rho$$

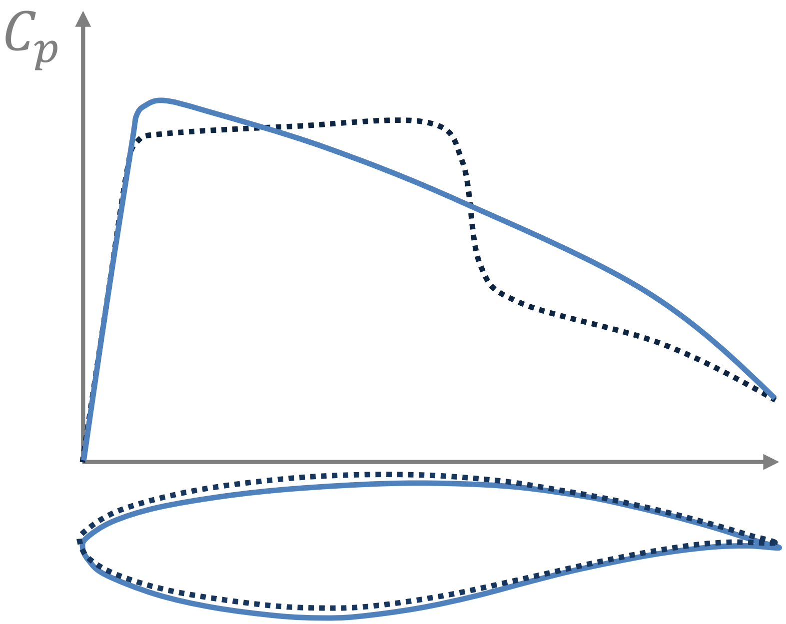


Mass flux upwinding

$$\rho \boldsymbol{u} \leftarrow \rho \boldsymbol{u} - \nu \frac{\partial \rho \boldsymbol{u}}{\partial s} \Delta s$$

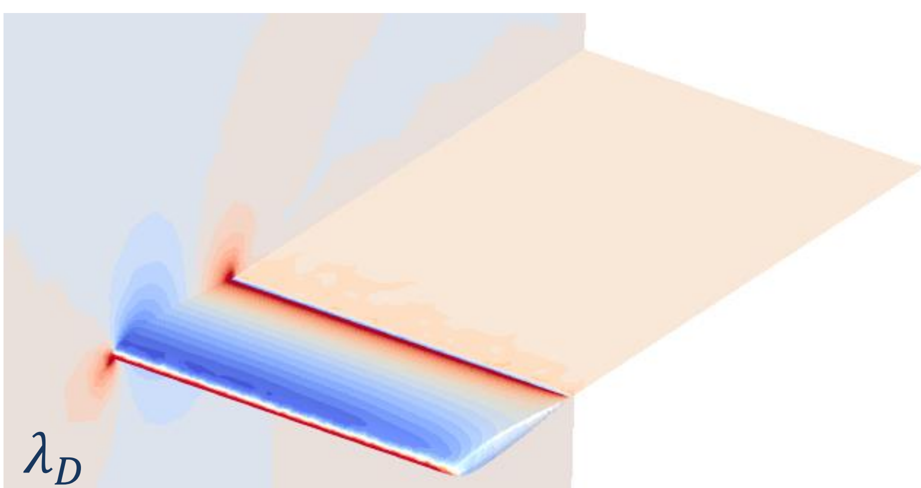
$$\rho \boldsymbol{u} \leftarrow \rho \boldsymbol{u} - \oint \rho \boldsymbol{u} \cdot \boldsymbol{n}$$

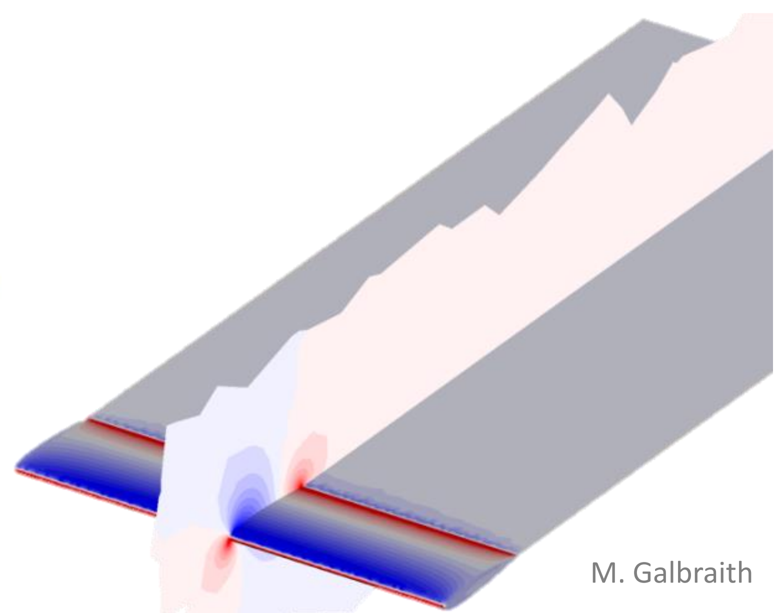
Perspectives – adjoint computations



$$\min J(\phi, v)$$
$$\text{s.t. } R(\phi, v) = 0$$

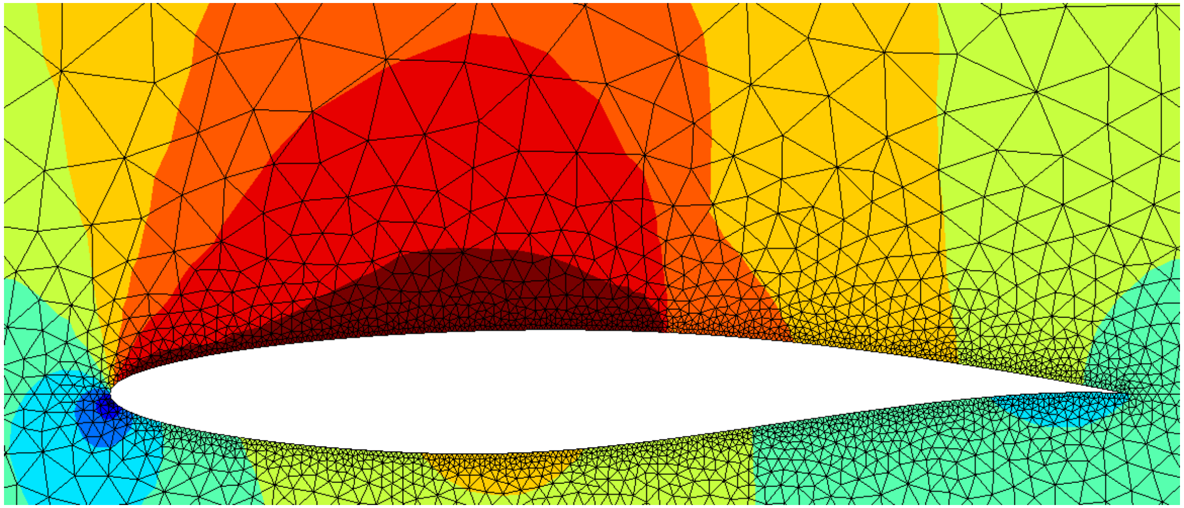
$$\mathcal{L} = J - \lambda \mathrm{R}$$





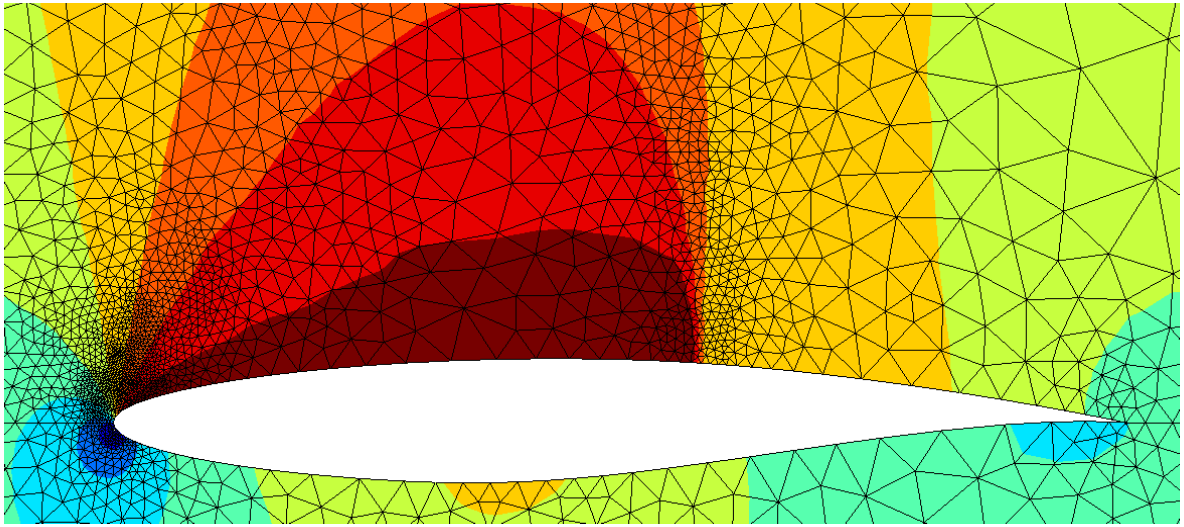
M. Galbraith

Perspectives – solution adaptive grid



***A-priori* imposed grid**

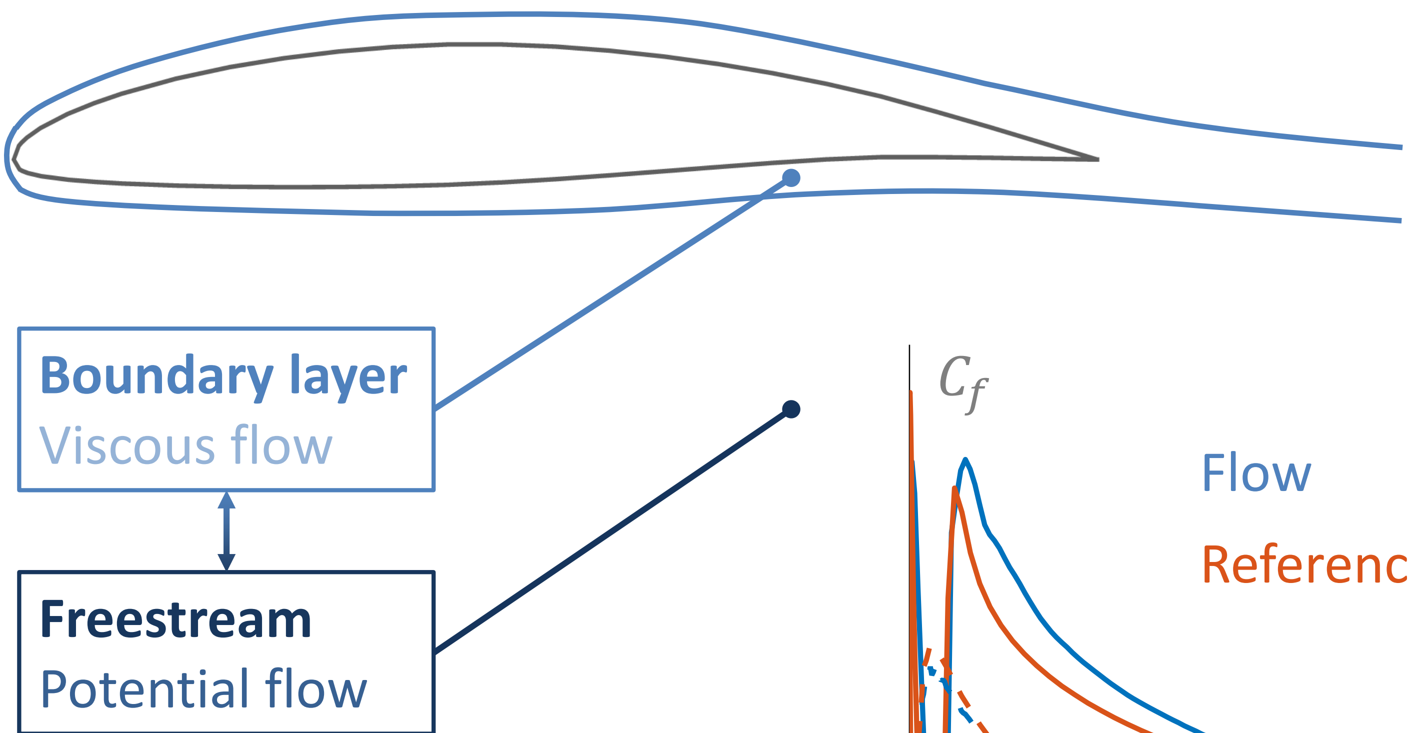
- Fine enough to capture physics
- Coarse enough to ensure quickness and robustness



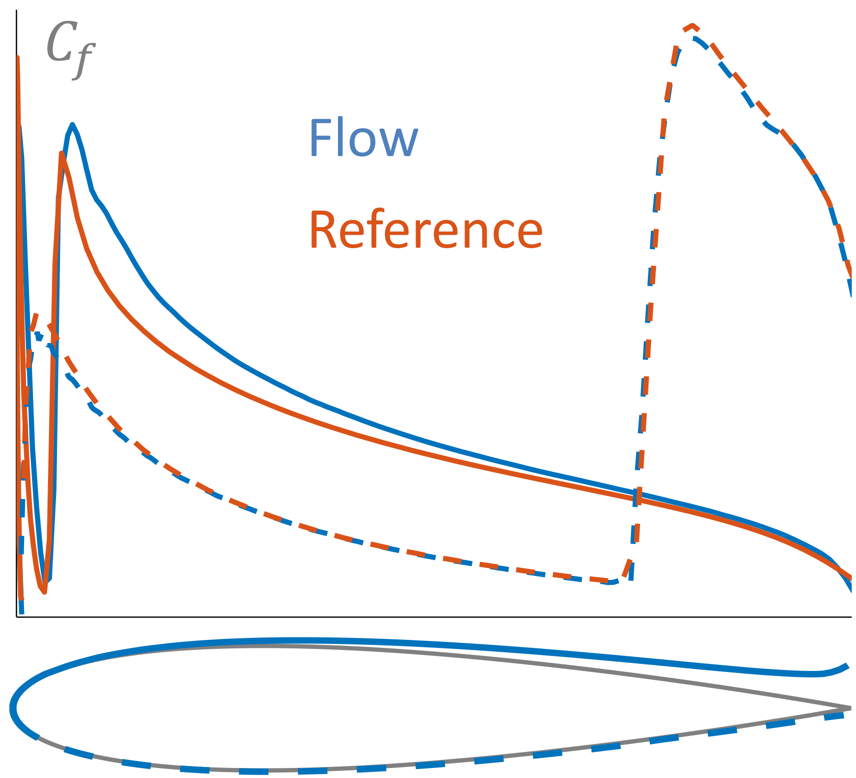
Adapted grid

- Refinement sensor
- Refinement technique
- Termination criterion

Perspectives – viscous-inviscid interaction



Amaury Bilocq



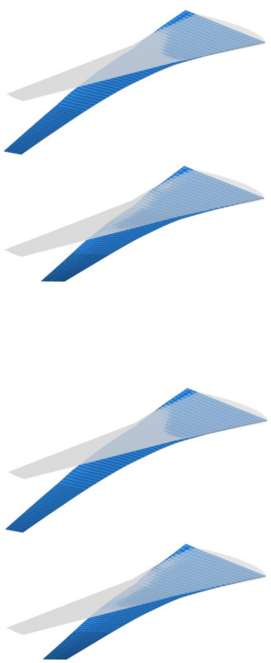
Perspectives – flutter computation


Tail flutter on PA-38 – NASA


Hüseyin Güner

Aeroelastic equation

$$\left(-\boldsymbol{M}_{\mathrm{q}} + \boldsymbol{K}_{\mathrm{q}}\right)\boldsymbol{q}(\omega) = -\boldsymbol{f}_{\mathrm{q}}(\boldsymbol{q}, \omega)$$

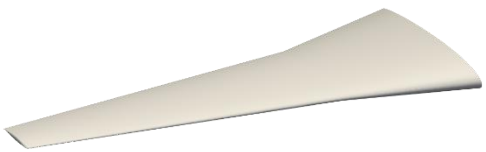


ω_1

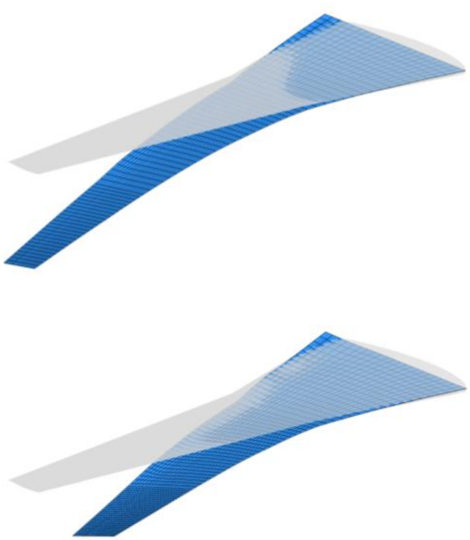
ω_2

interpolation → $\boldsymbol{f}_{\mathrm{q}}(\omega)$ p-k method → $\boldsymbol{u}_{\mathrm{f}}(\omega_{\mathrm{f}})$

Perspectives – unsteady modeling



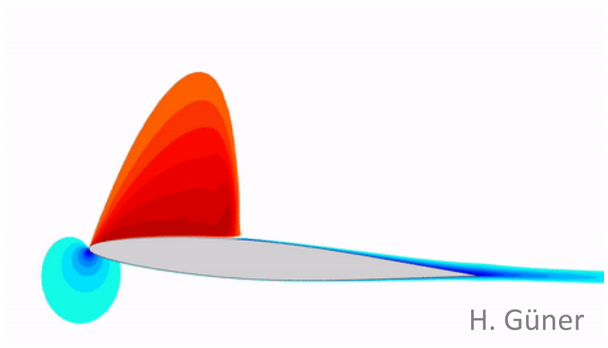
nonlinear
steady → Φ



linear
unsteady → φ_k

$$\phi_k \sim \Phi + \varphi_k \, \mathrm{e}^{\mathrm{i}\omega \mathrm{t}}$$

Xavier Naa



H. Güner