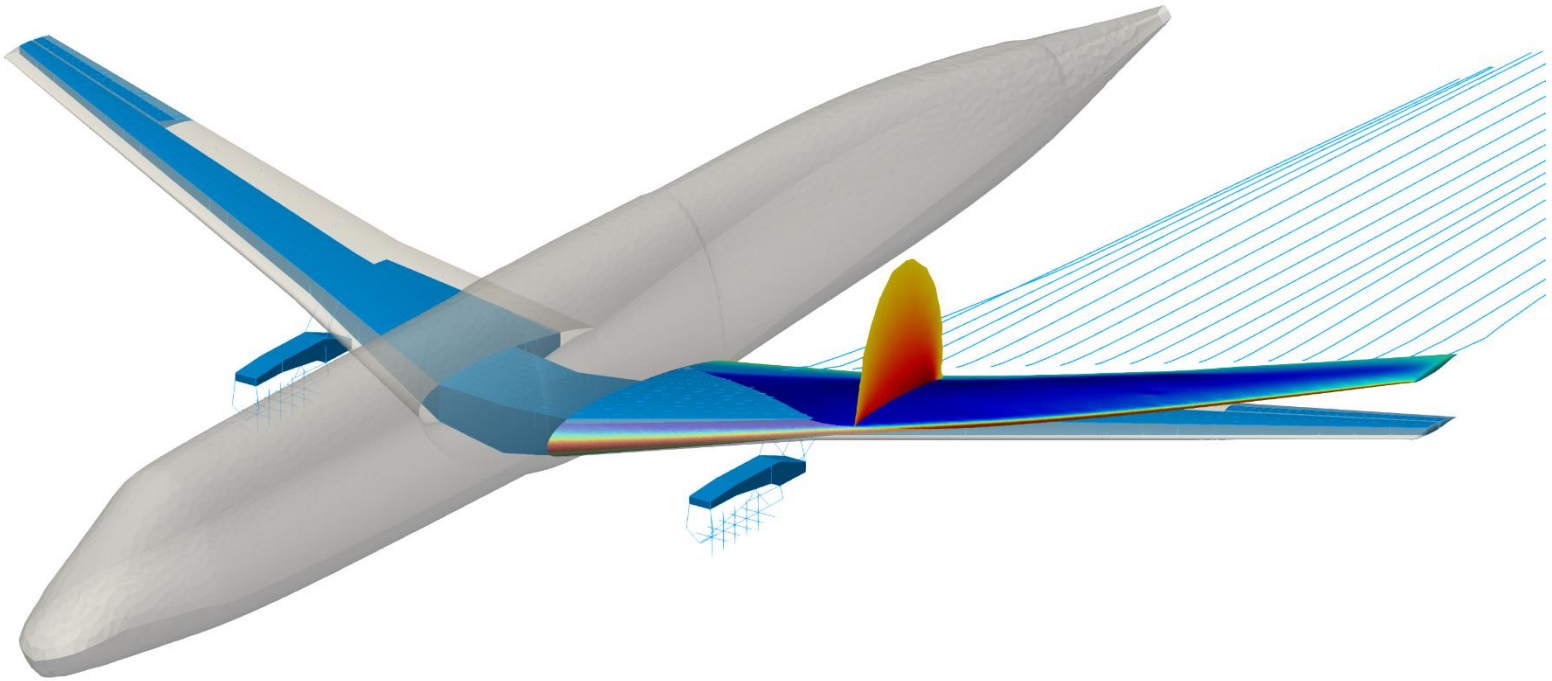
Aerostructural modeling for preliminary aircraft design

Adrien Crovato



EMBRAER

Châtillon, June 2024

LIÈGE université

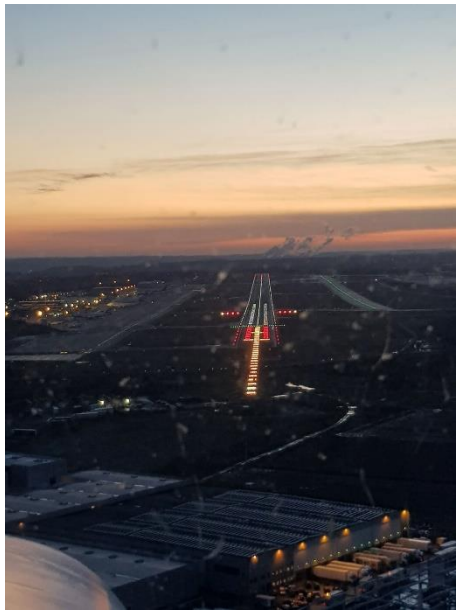
About me

Professional

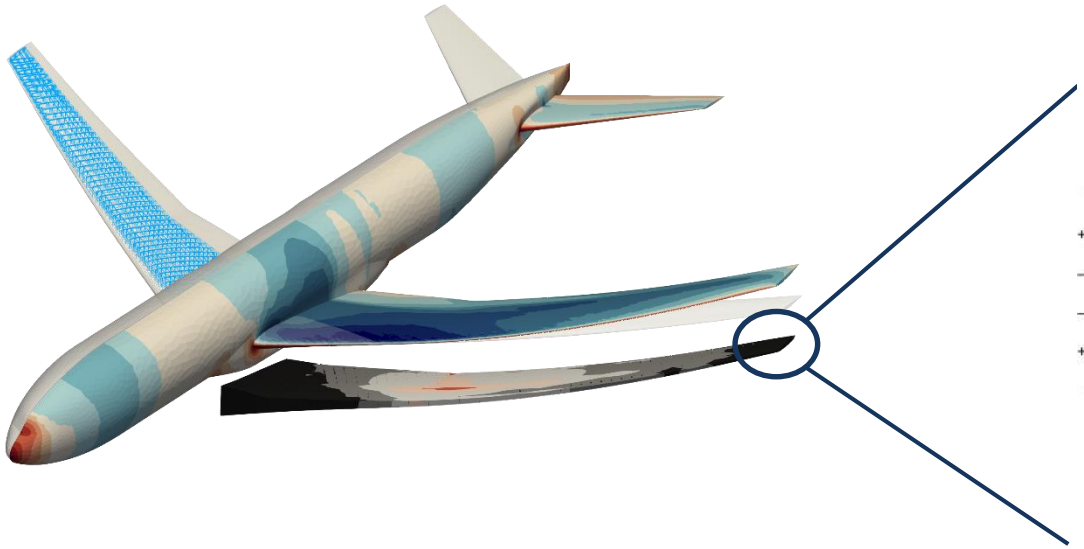
- MSc in Aerospace Eng., 2015
- PhD in Aerospace Eng., 2020
- Post-doc in Aerospace Eng., since 2020
- Teaching activities, since 2015
- ULiège with Embraer

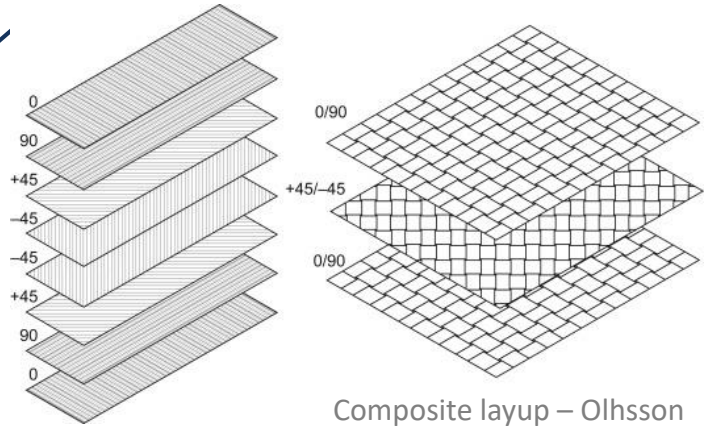
Personal

- Martial artist
- Private pilot



Aerostructural optimization





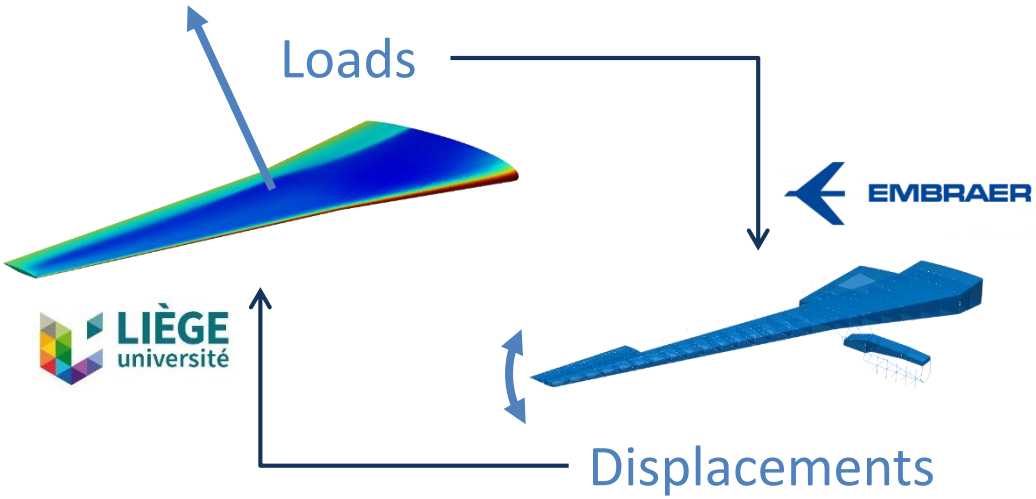
Composite layup – Olhsson

Optimize shape and laminates

- Decrease fuel burn

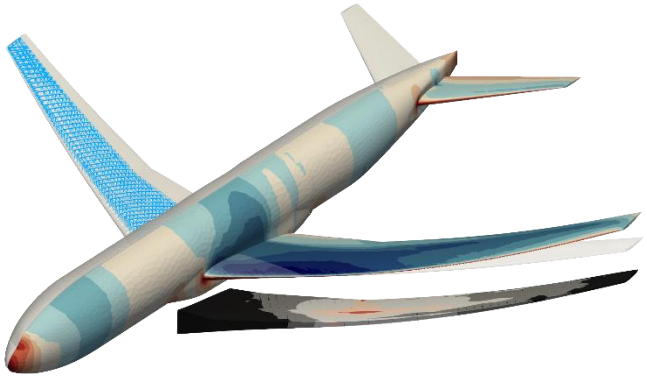
Such that

- No failure
- No flutter



Preliminary aircraft design

Conceptual → **Preliminary** → Detail



Numerical model (9%)

- Global design
- Optimization
- Performance

Results must be obtained **quickly**
Adequate models must be chosen

Outline

Optimization

- Mathematical formulation
- Gradients calculation

Static aeroelasticity

- Steady aerodynamic modeling
- DART

Dynamic aeroelasticity

- Unsteady aerodynamic and flutter modeling
- SDPM and PyPk

Benchmark case

- Problem description
- Optimization results

Optimization formulation

Gradient-based approach

$$d_x F(u;\ x) \to 0$$

$$\text{s.t.}\quad \begin{matrix} R(u;x) = 0 \\ C(u;x) = 0 \end{matrix}$$

"perturbation"

$$\begin{cases} R\big(u(x+\delta x)\big) = 0 \\ d_x F = \Delta\left\{\dfrac{F\big(u(x+\delta x)\big)}{\delta x}\right\} \end{cases}$$

$$\begin{matrix} d_x F(u;x) \to 0 \\ R(u;x) = 0 \end{matrix}$$

"chain rule"

$$\begin{cases} R\big(u(x)\big) = 0 \\ d_x F = \partial_x F - \partial_u F \partial_u R^{-1} \partial_x R \end{cases}$$

Computational cost analysis

Finite differences

$$\begin{cases} R\big(u(x)\big) = 0 \\ R\big(u^+(x+\delta x)\big) = 0 \\ d_x F = \dfrac{F(u^+) - F(u)}{\delta x} + O(\delta x) \end{cases}$$

Complex step

$$\begin{cases} R\big(u(x)\big) = 0 \\ R\big(u^+(x+i\delta x)\big) = 0 \\ d_x F = \mathrm{Im}\left\{\dfrac{F(u^+)}{\delta x}\right\} + O(\delta x^2) \end{cases}$$

→ Cost ~ n. o. design variables × time to solve nonlinear equations

Direct and adjoint

$$\begin{cases} R\big(u(x)\big) = 0 \\ d_x F = \partial_x F - \boxed{\partial_u F}\boxed{\partial_u R^{-1}}\boxed{\partial_x R} \end{cases}$$

$\partial_u R^{\mathrm{T}}\,\lambda = \partial_u F^{\mathrm{T}}$ — **Adjoint**

$\partial_u R\,\lambda = \partial_x R$ — Direct

→ Cost (adjoint) ~ n. o. functionals × time to solve linear equations

Calculation of the derivatives

Hand differentiation

✓ Most effective

× Difficult, sometimes not feasible

Finite differences

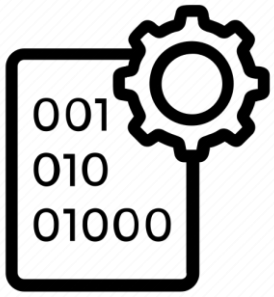
✓ Very easy

× Inaccurate

Complex step

✓ Accurate

× Complex arithmetic



Automatic differentiation

✓ Straightforward

× Increased memory usage

Outline

Optimization

- Mathematical formulation
- Gradients calculation

Static aeroelasticity

- Steady aerodynamic modeling
- DART

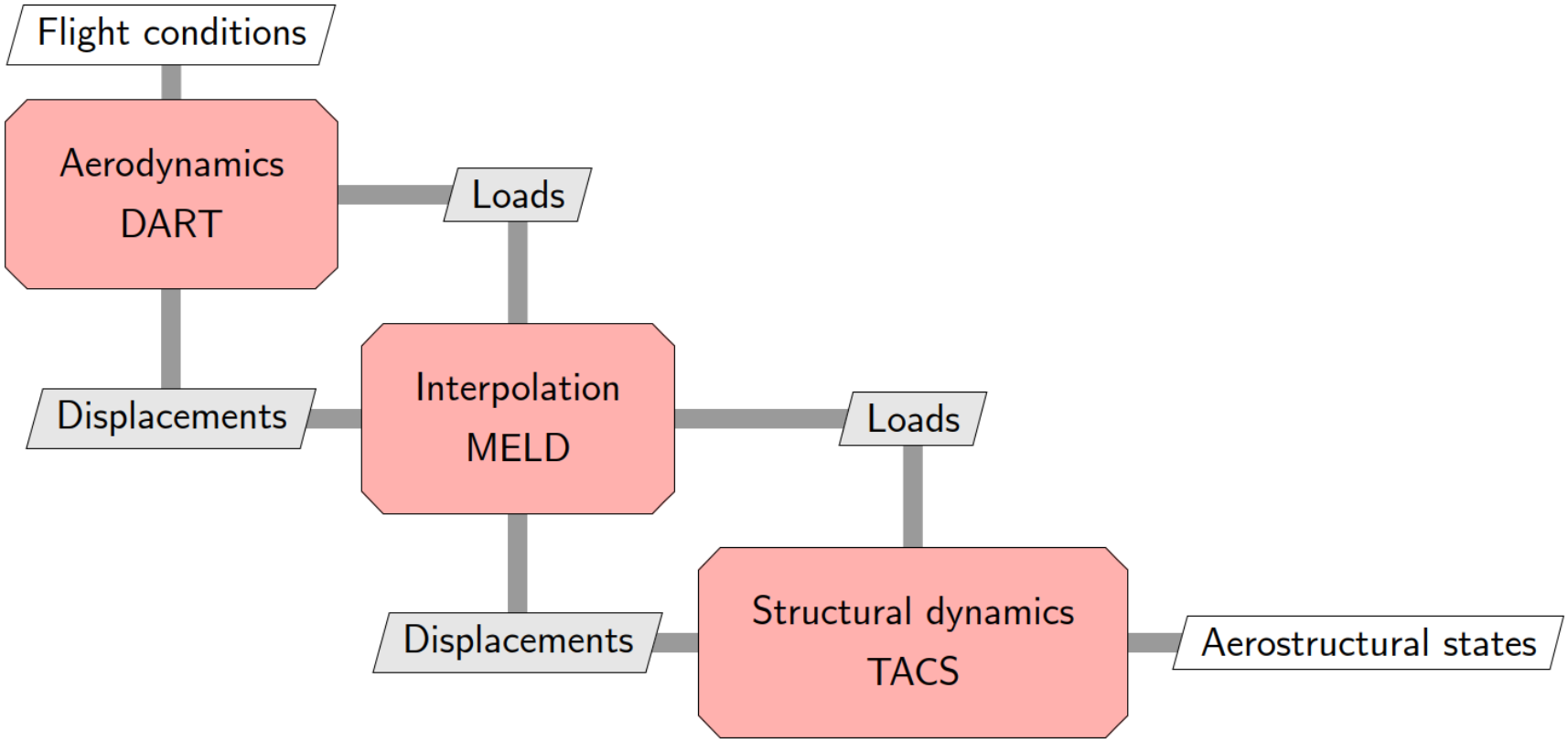
Dynamic aeroelasticity

- Unsteady aerodynamic and flutter modeling
- SDPM and PyPk

Benchmark case

- Problem description
- Optimization results

Analysis process



https://openmdao.org

https://github.com/openmdao/mphys

Aerodynamic models for aircraft design

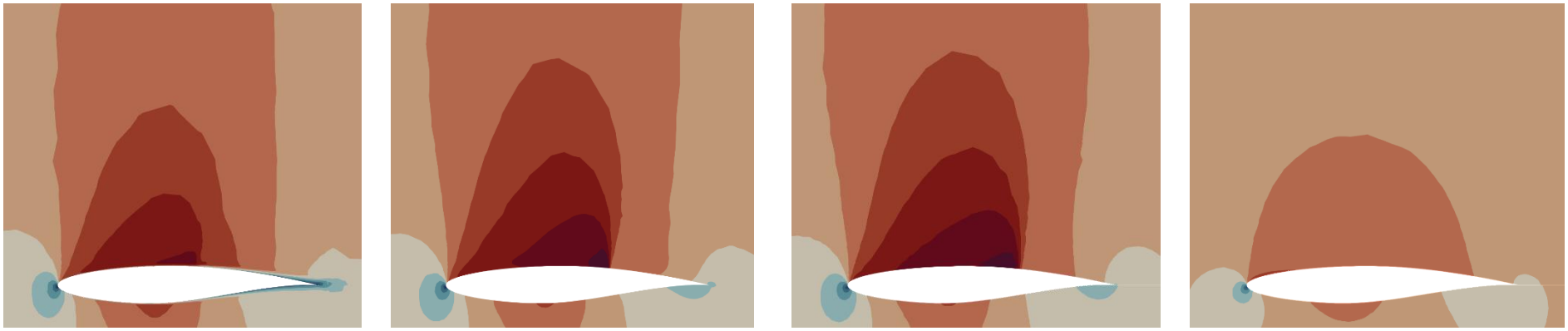
RANS
equations
- Subsonic
- Supersonic
- Transonic
- Viscous

Euler
equations
- Subsonic
- Supersonic
- Transonic
- **Inviscid**

Full potential
equation
- Subsonic
- Supersonic
- **~Transonic**
- Inviscid

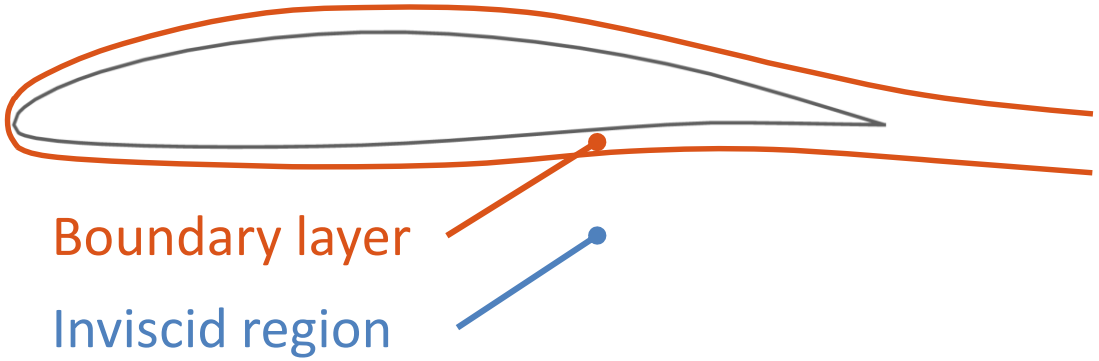
Linear potential
equation
- **~Subsonic**
- **~Supersonic**
- ~~**Transonic**~~
- Inviscid

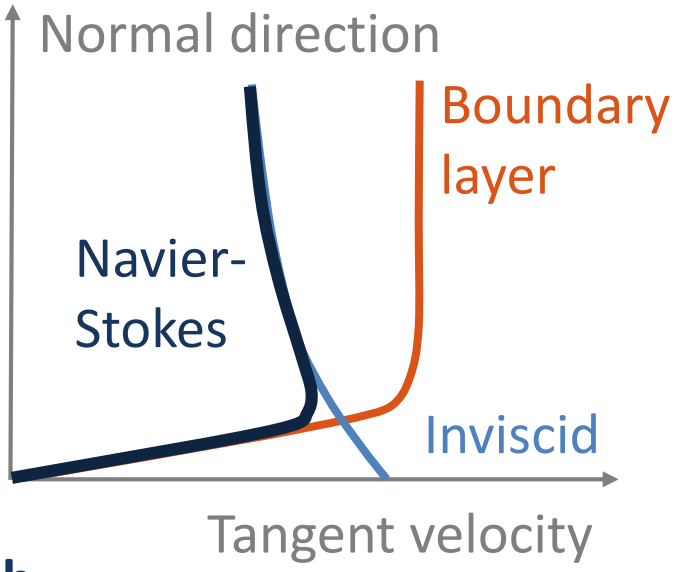
Inviscid → Isentropic → Linear



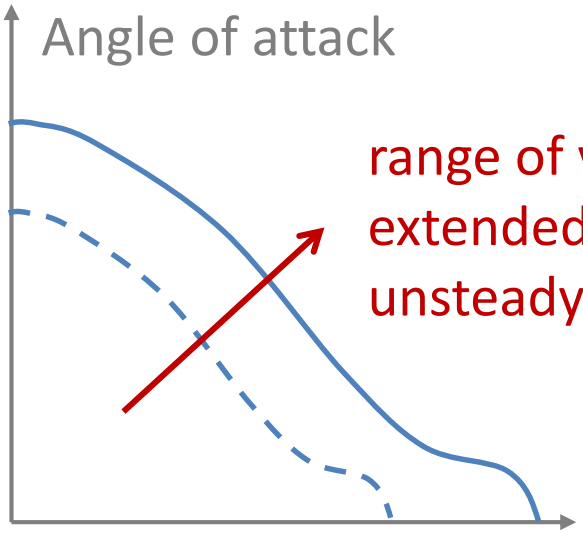
Mach number

Viscous-inviscid interaction

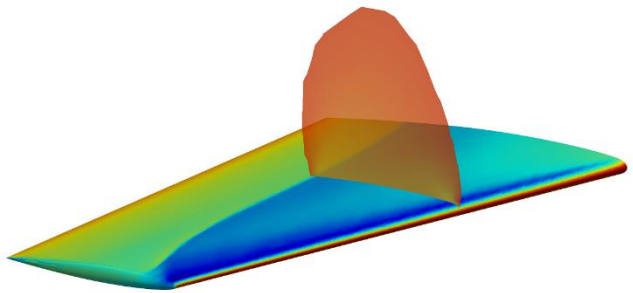


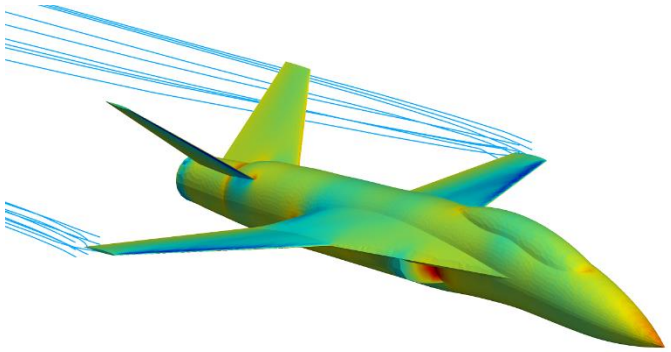


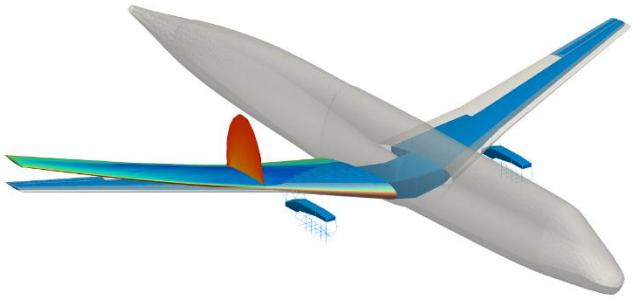
Quasi-simultaneous pseudo-unsteady approach



DART







Discrete Adjoint for Rapid Transonic Flows

- Steady full potential formulation
- Finite element discretization
- Unstructured tetrahedral grid
- Analytical discrete adjoint
- Mesh morphing
- Viscous-inviscid interaction
- C++ with Python API

Performance (712k elements @ 3.4GHz)

- Solution – 100 s
- Morphing – 25 s
- Gradient – 45 s

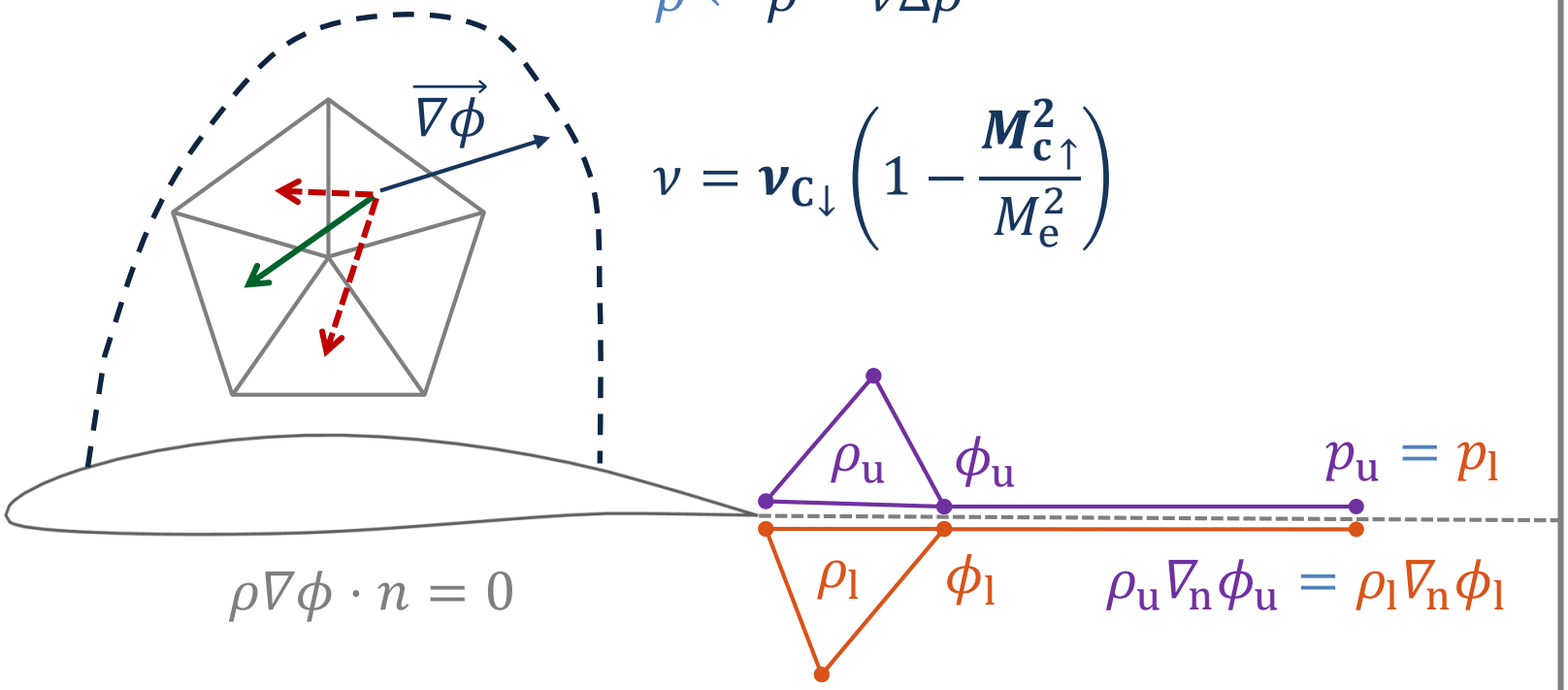
https://gitlab.uliege.be/am-dept/dartflo

Formulation and implementation

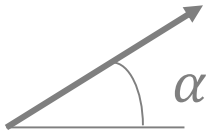
$$\nabla \cdot (\rho \nabla \phi) = 0$$

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

$$\nu = \boldsymbol{\nu_{c_\downarrow}} \left(1 - \frac{\boldsymbol{M^2_{c_\uparrow}}}{M_e^2}\right)$$



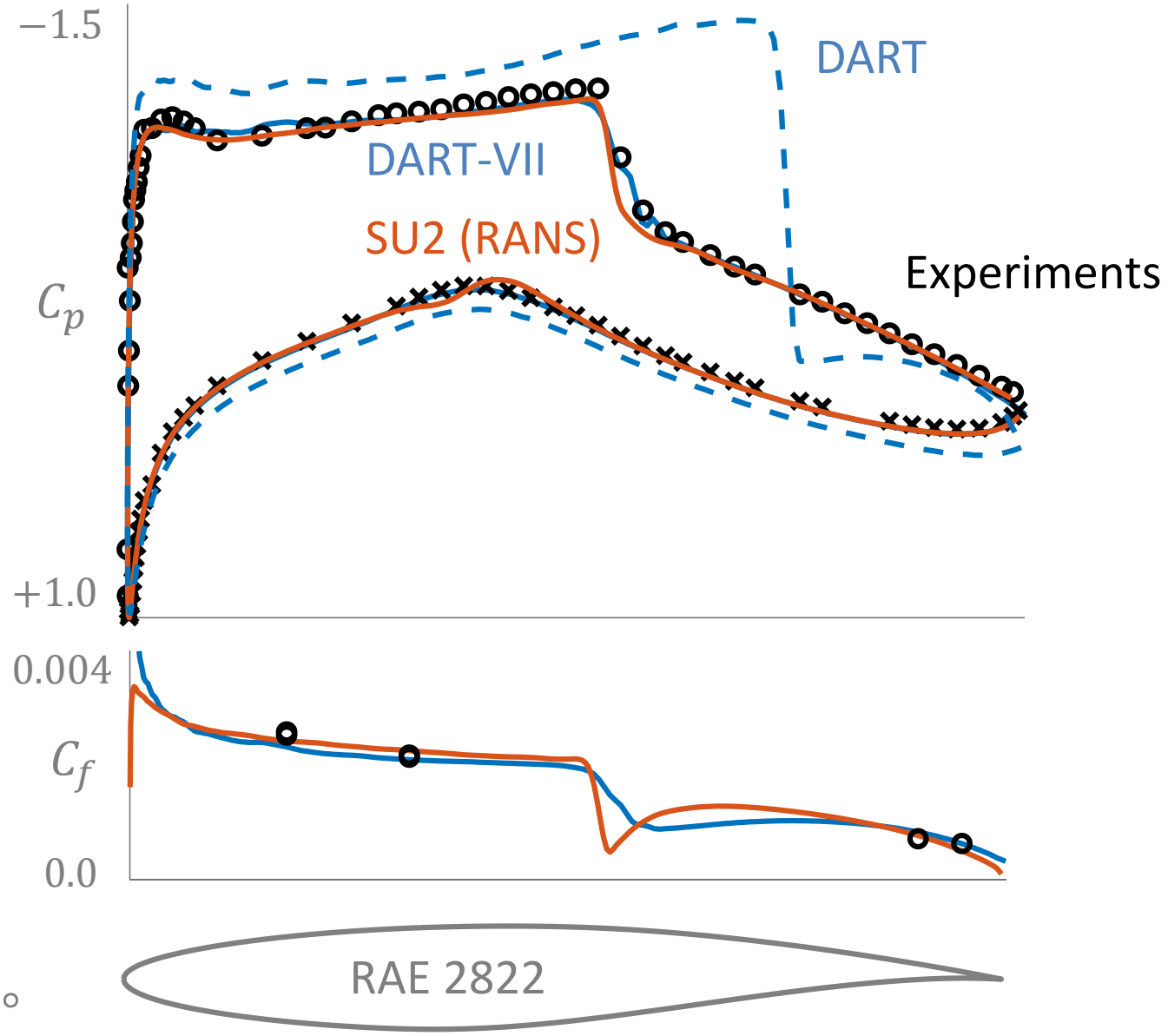
$$\rho \nabla \phi \cdot n = 0$$



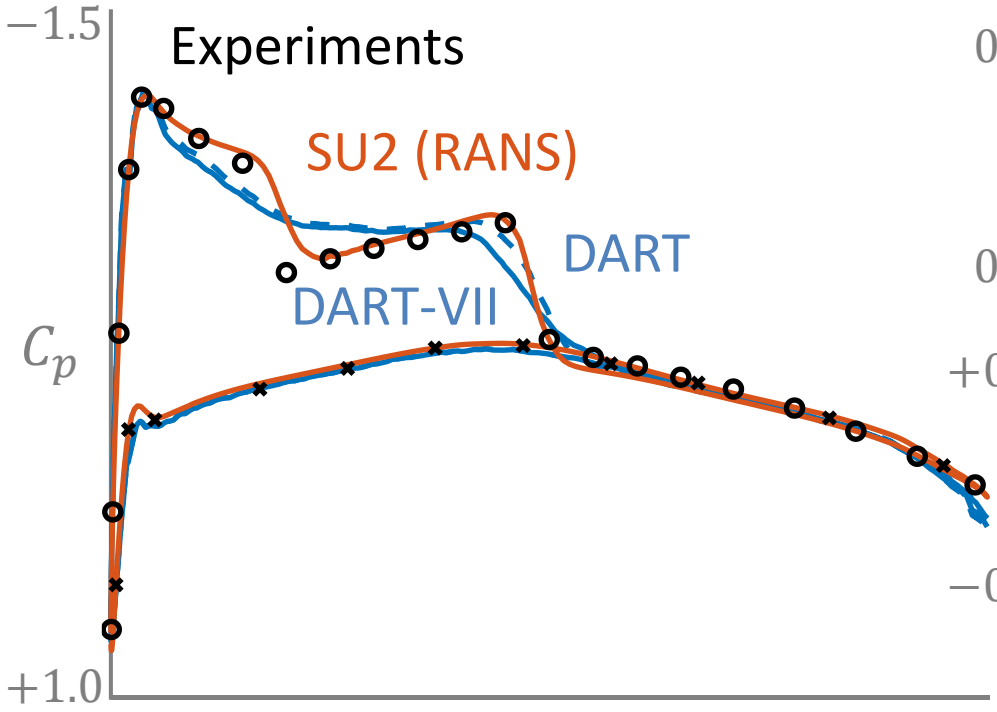
$$\rho_\infty U_\infty = [\cos\alpha, \sin\alpha]$$

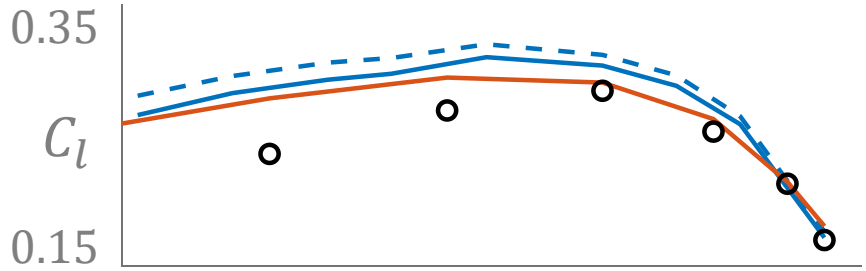
$$\nabla \phi \cdot n = \rho_\infty U_\infty \cdot n$$

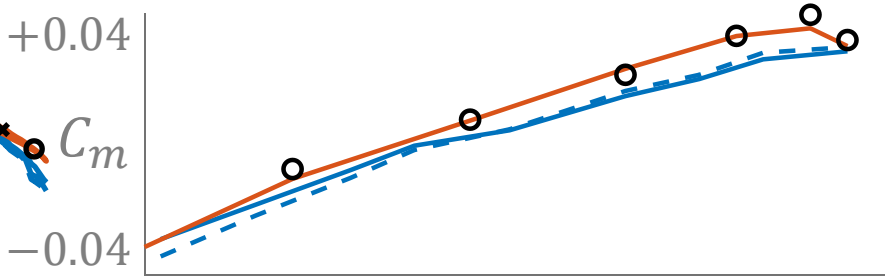
Two–dimensional viscous analysis

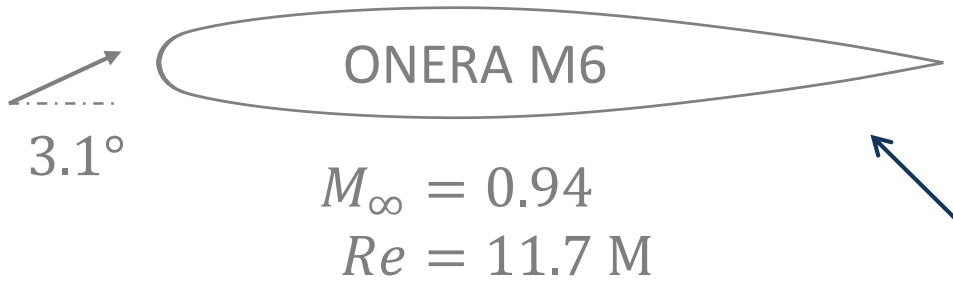


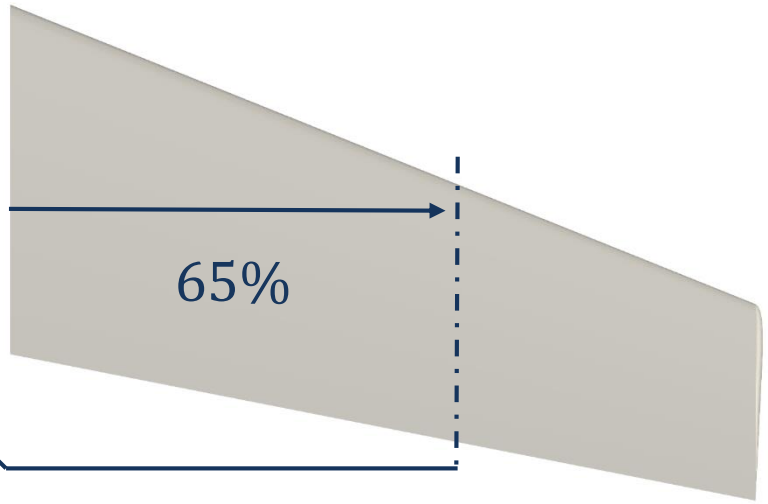
Three–dimensional viscous analysis











Outline

Optimization

- Mathematical formulation
- Gradients calculation

Static aeroelasticity

- Steady aerodynamic modeling
- DART

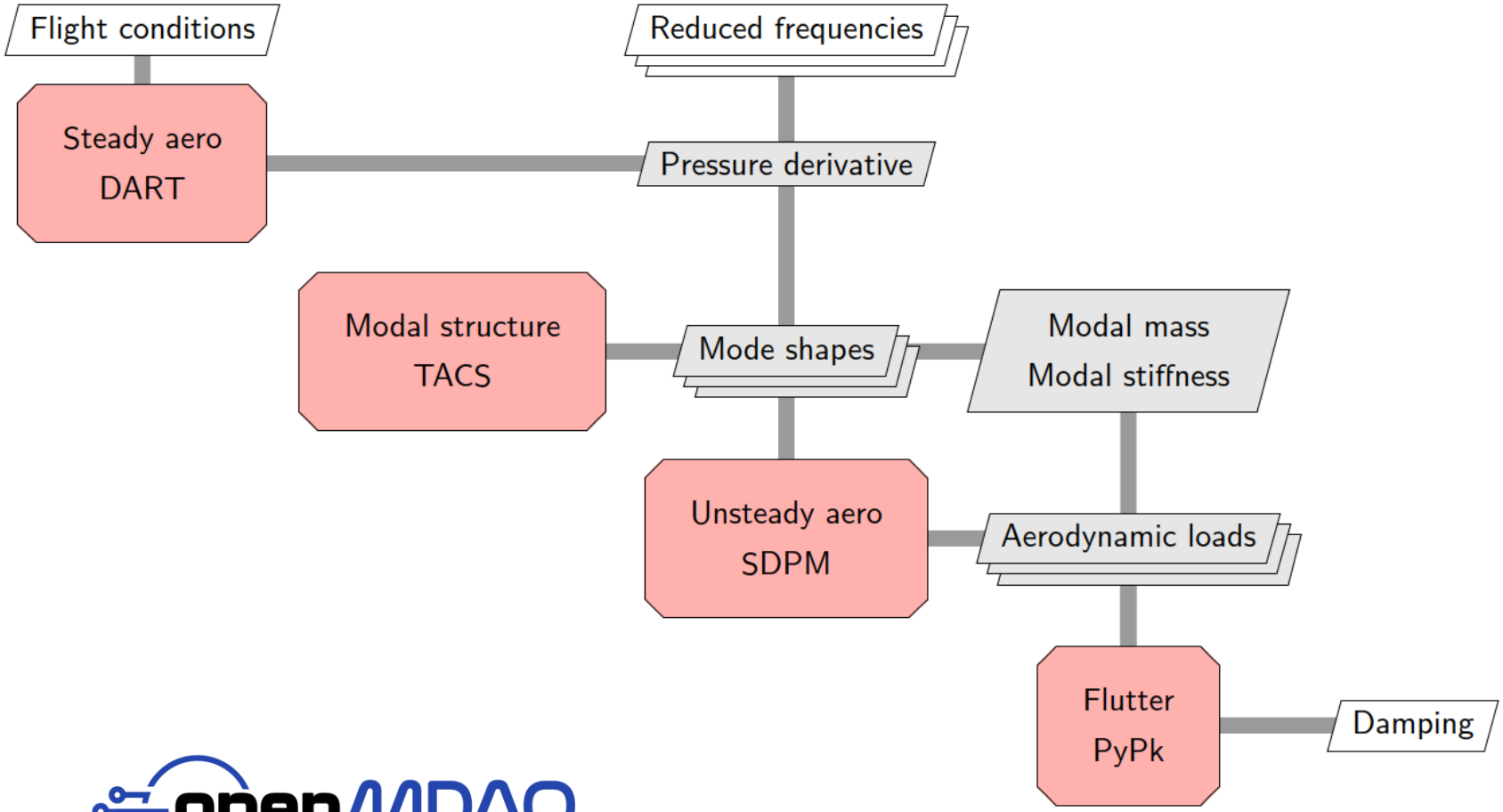
Dynamic aeroelasticity

- Unsteady aerodynamic and flutter modeling
- SDPM and PyPk

Benchmark case

- Problem description
- Optimization results

Analysis process



https://openmdao.org

https://gitlab.uliege.be/am-dept/omflut

Unsteady aerodynamics for flutter

Flutter equation

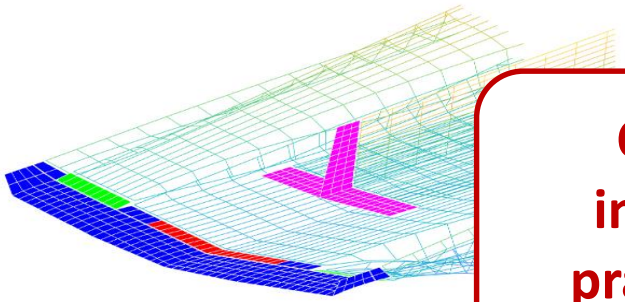
$$M\ddot{q}(t) + C\dot{q}(t) + Kq(t) = F(t) \quad \rightarrow \quad \left(p^2(\omega)M_r + K_r - Q_r(\omega)\right)q_r = 0$$

Boundary element method

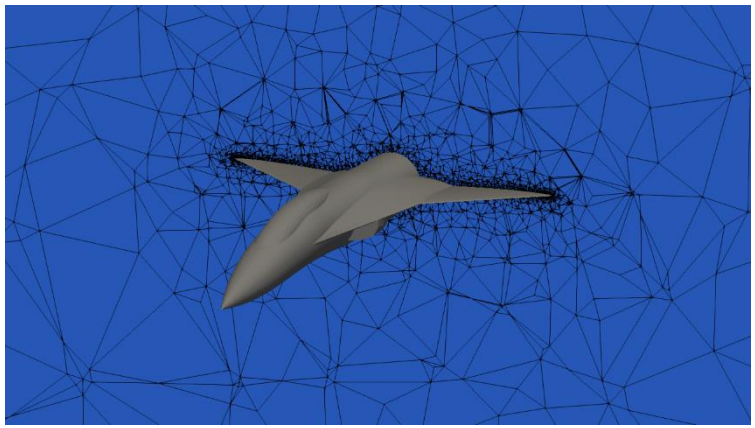
- Only boundary is discretized
- Linear equations only
- Panel/lattice methods

Field method

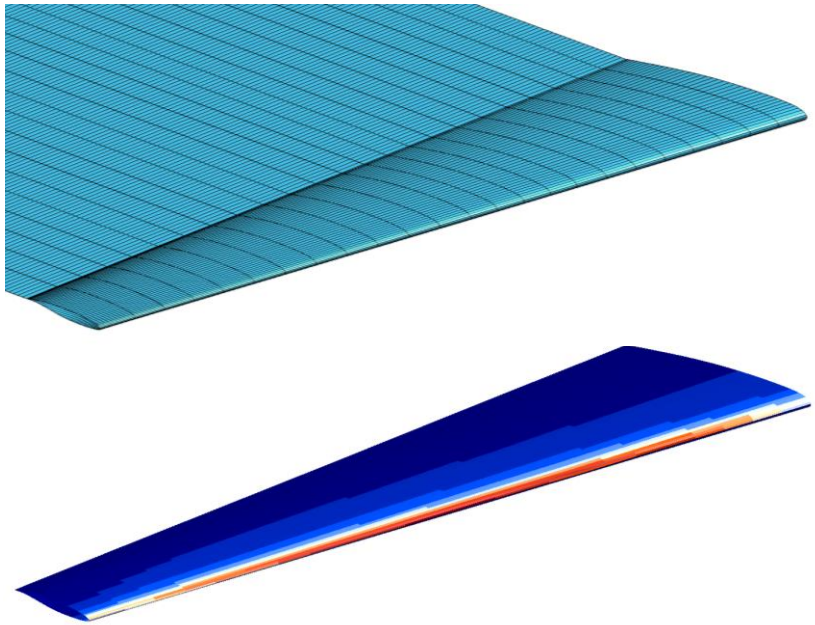
- Whole field is discretized
- Linear and nonlinear equations
- Finite volume/element methods

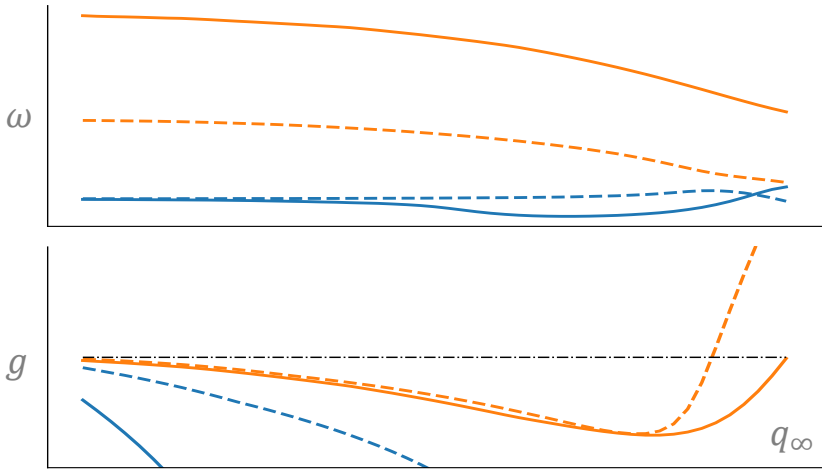


Current industrial practice for aeroelastic computations



SDPM and PyPk





Source and Doublet Panel Method

- Unsteady potential formulation
- Panel discretization
- Unstructured quadrangular grid
- Transonic correction
- Reverse automatic differentiation
- C++ with Python API

https://gitlab.uliege.be/am-dept/sdpm

Python p-k flutter methods

- Standard and non-iterative p-k
- Mode tracking
- Analytical gradients
- Python

https://gitlab.uliege.be/am-dept/pypk

Unsteady source and doublet panel method

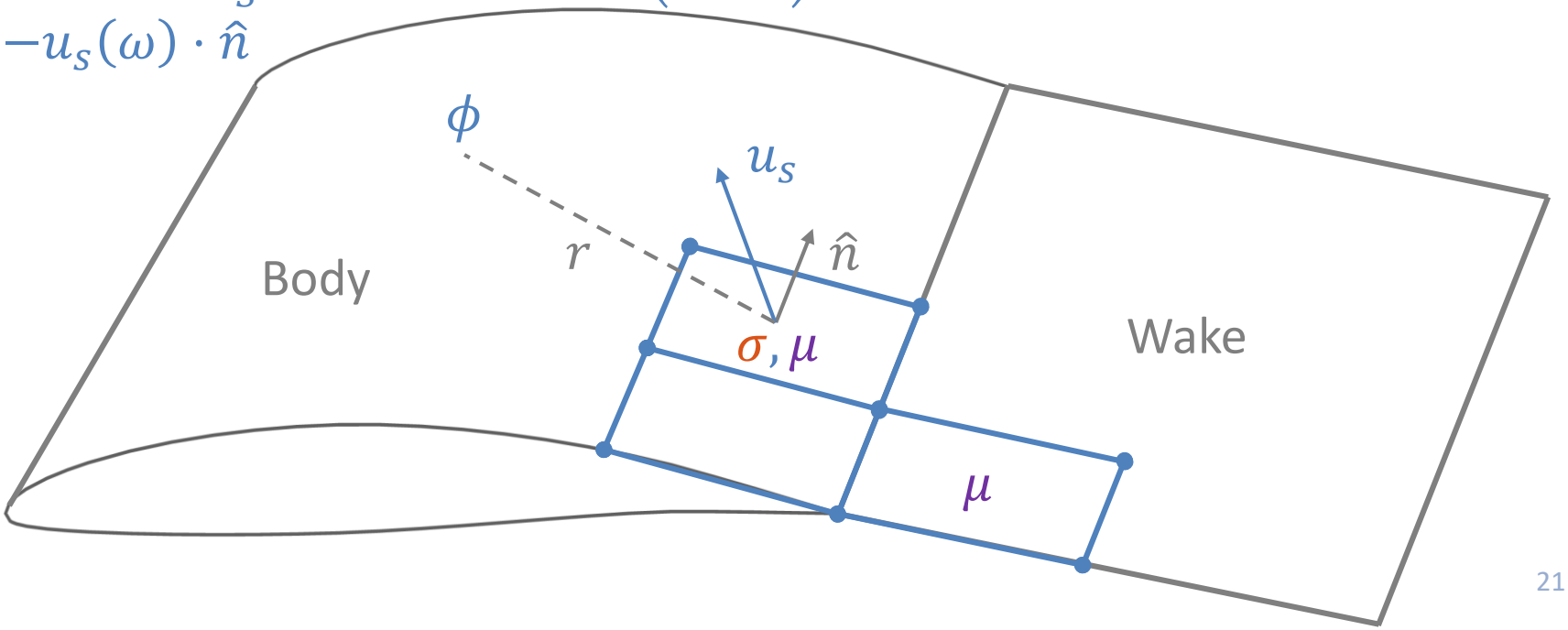
Formulation

$$\phi(\omega) = -\frac{1}{4\pi}\int_S \hat{n}\cdot\left(\nabla\phi\frac{E(\omega)}{r} - \phi\nabla\left(\frac{E(\omega)}{r}\right)\right) dS$$

Panel discretization

$$\phi(\omega) = -\frac{1}{4\pi}\int_S \sigma\frac{E(\omega)}{r} - \mu\hat{n}\cdot\nabla\left(\frac{E(\omega)}{r}\right) dS$$

$$\sigma = -u_s(\omega)\cdot\hat{n}$$



Transonic correction

Linearized pressure coefficient derivative

$$c_p(0) \simeq \frac{2}{\beta}\left(\partial_x^S \mu(0) + \hat{n}_x \sigma(0)\right)$$

$$\partial_\alpha c_p(0) \simeq \frac{2}{\beta}\left(\partial N_x^S A^{-1} B \hat{n}_z + \hat{n}_x \hat{n}_z\right)$$

$$\partial_\alpha c_p^{\text{ref}}(0) \simeq \frac{2}{\beta}\left(\partial N_x^S D^{\text{corr}} A^{-1} B \hat{n}_z + \hat{n}_x \hat{n}_z\right)$$

Procedure

1. Compute pressure derivative $\partial_\alpha c_p^{\text{ref}}(0)$ from steady CFD
2. Solve for correction: $\frac{2}{\beta}\partial N_x^S D^{\text{corr}} A^{-1} B \hat{n}_z = \partial_\alpha c_p^{\text{ref}}(0) - \hat{n}_x \hat{n}_z$
3. Compute doublets: $\mu(\omega) = A^{-1} B \sigma\left(\omega, u_{s_{x,y}}\right) + D^{\text{corr}} A^{-1} B \sigma\left(\omega, u_{s_z}\right)$

Flutter solution

Flutter equation

$$\left(\frac{u_\infty^2}{l_{\text{ref}}^2}p^2M + K - \frac{1}{2}\rho_\infty u_\infty^2 Q(k)\right)q = 0$$

$$p = gk + ik$$

Frequency matching (p-k)

1. Guess $k = \omega_N \frac{l_{\text{ref}}}{u_\infty}$
2. Compute $Q(k)$
3. Solve eigenvalue problem for p
4. Compute $k = \Im(p)$
5. Repeat 2-4 until k has converged

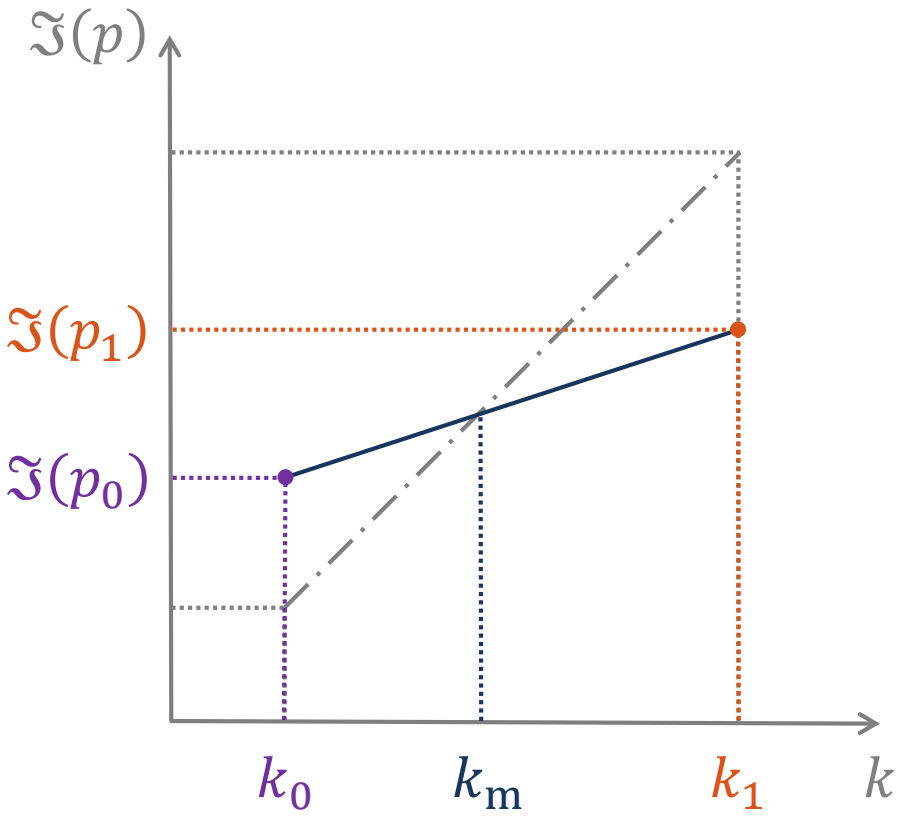
Computation of Q is costly

Interpolate Q or k

Non-iterative p-k method

Algorithm

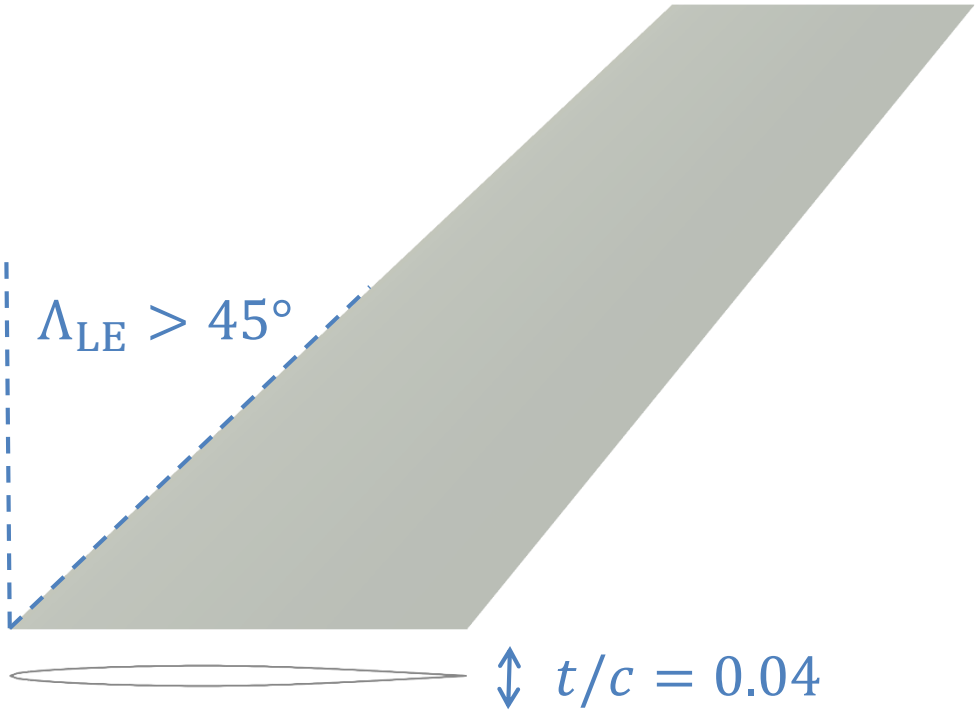
1. Compute $Q_i(k_i)$ for a set of k_i
2. Solve eigenvalue problem for p_i
3. Interpolate k_m such that $\Im(p_\mathrm{m}) - k_\mathrm{m} = 0$



AGARD 445.6 wing

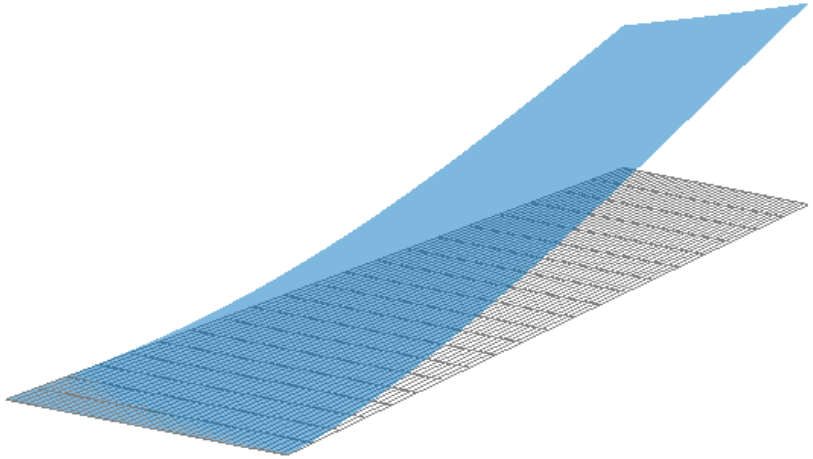
$M_{\infty,1} = 0.50$
$M_{\infty,2} = 0.68$
$M_{\infty,3} = 0.90$
$M_{\infty,4} = 0.96$

$\alpha = 0°$

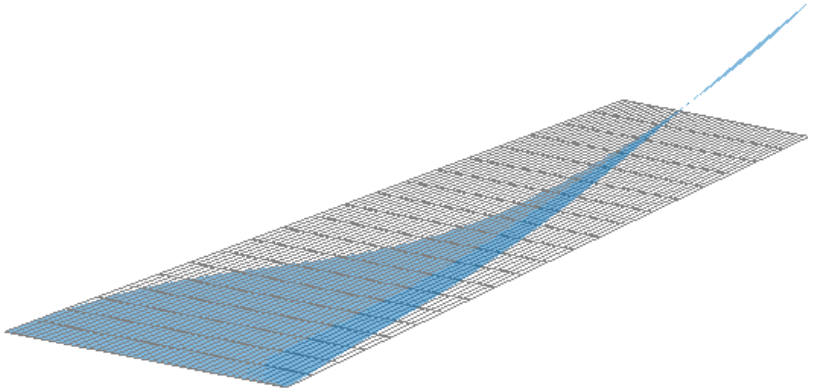


Mode shapes

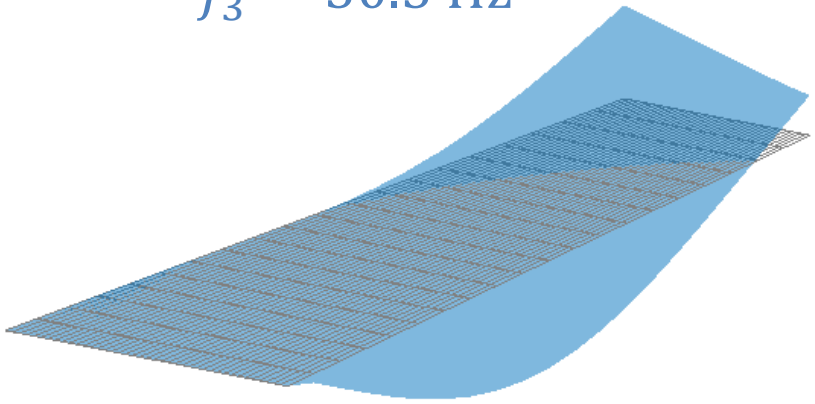
$f_1 = 9.7$ Hz



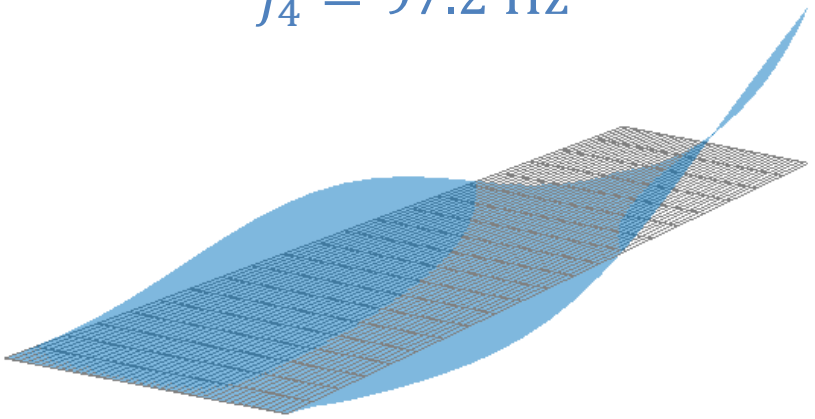
$f_2 = 40.2$ Hz



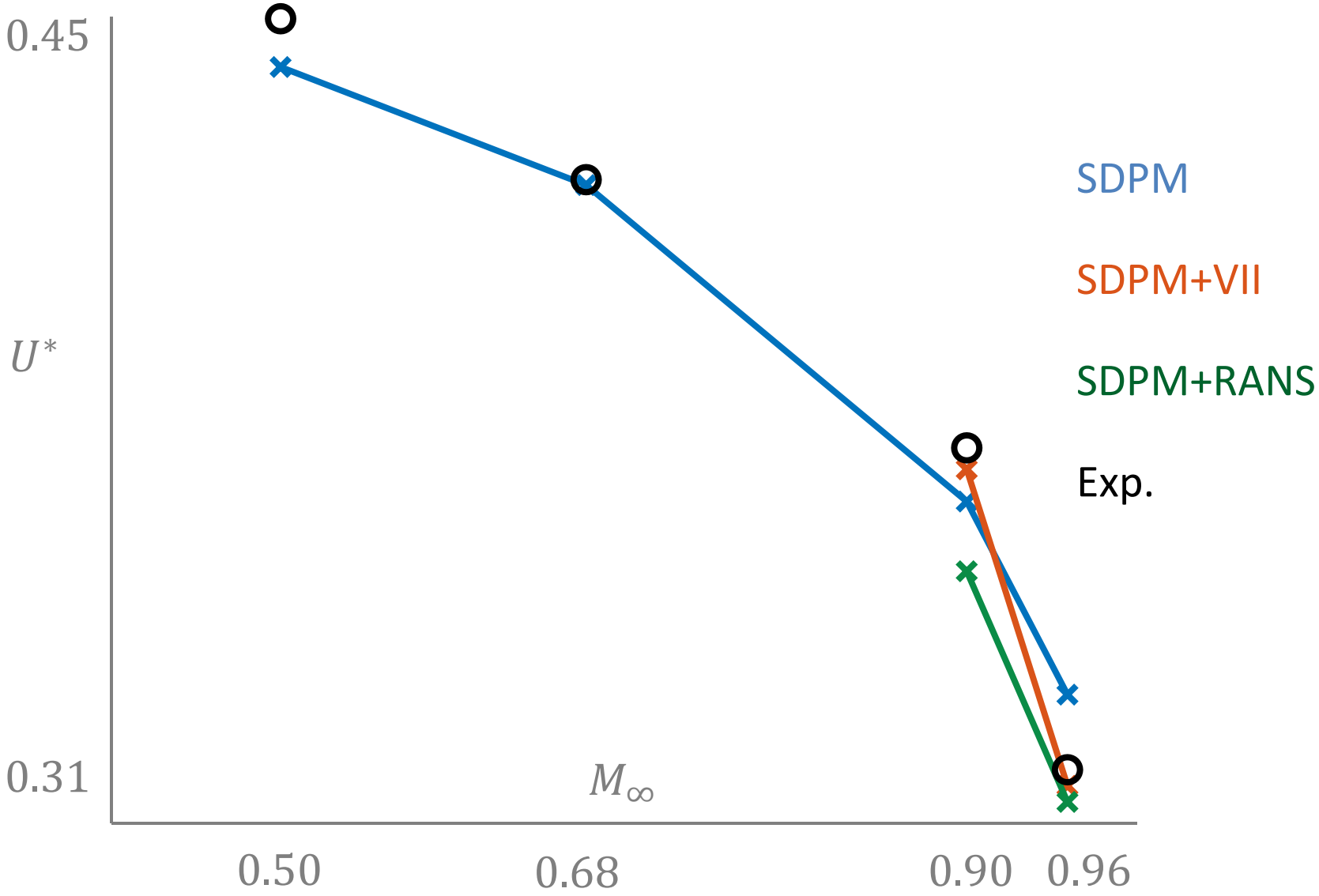
$f_3 = 50.5$ Hz



$f_4 = 97.2$ Hz



Flutter boundary



Outline

Optimization

- Mathematical formulation
- Gradients calculation

Static aeroelasticity

- Steady aerodynamic modeling
- DART

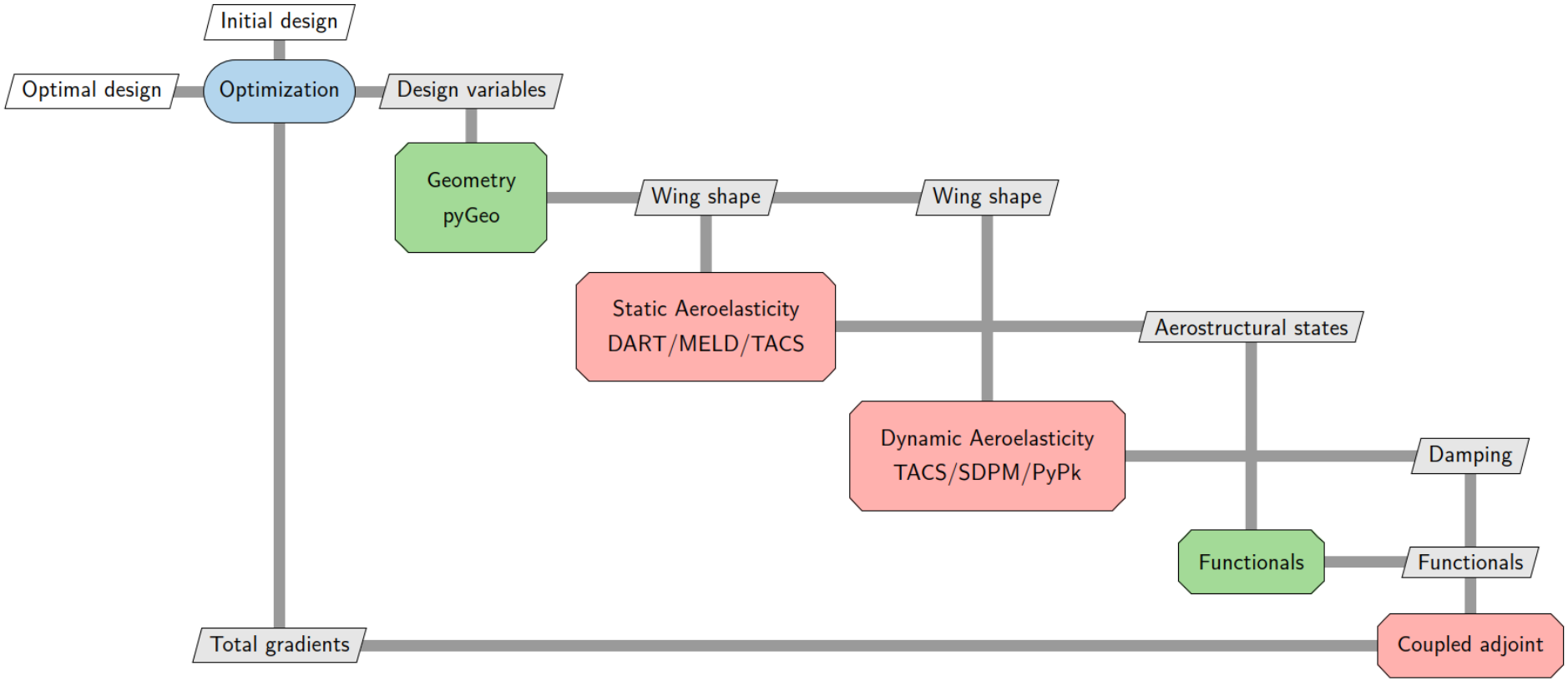
Dynamic aeroelasticity

- Unsteady aerodynamic and flutter modeling
- SDPM and PyPk

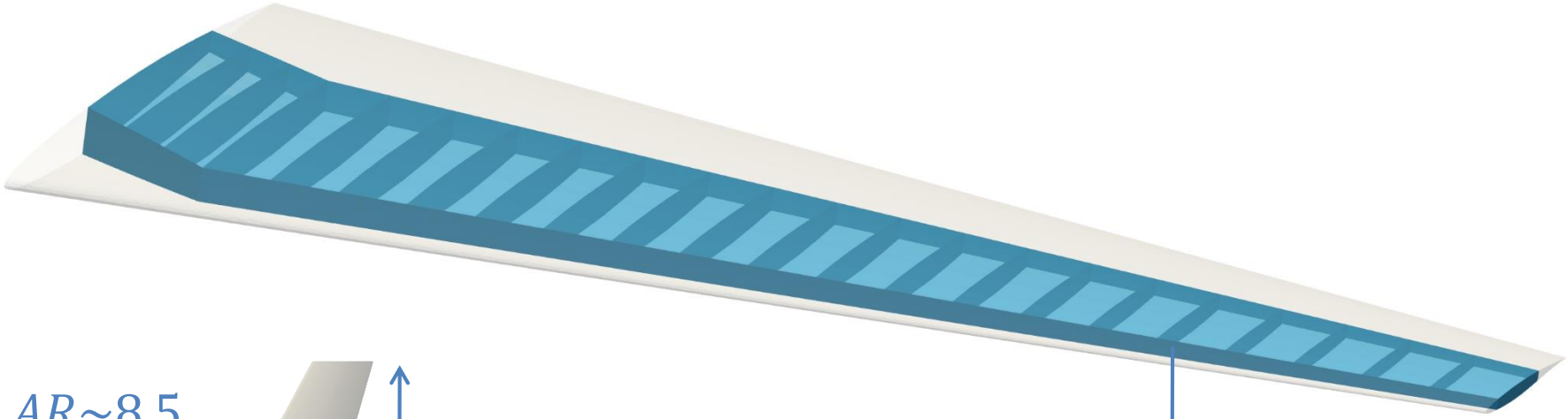
Benchmark case

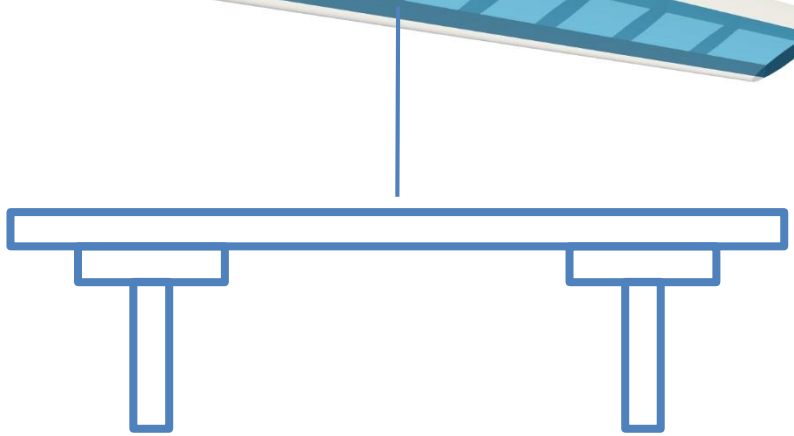
- Problem description
- Optimization results

Optimization framework



RAE wing





$AR \sim 8.5$
$\lambda \sim 0.33$

$\Lambda_{LE} \sim 28°$

$s = 14$ m

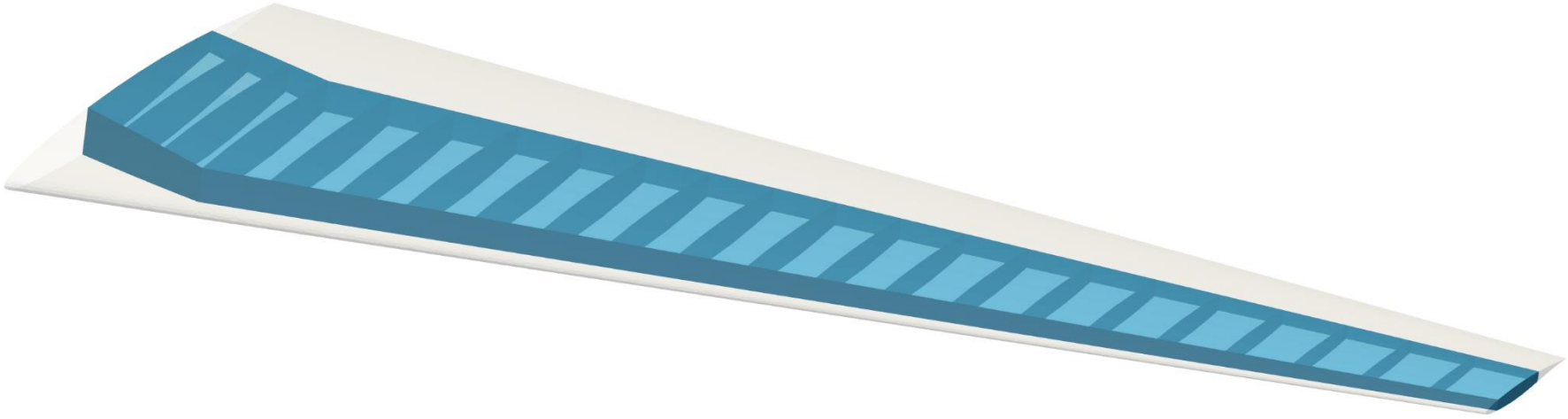
RAE 2822

$t/c = 0.12$

Skin/stiff: $[0_{0.45}/-45_{0.22}/+45_{0.22}/90_{0.11}]$

Spar/rib: $[0_{0.10}/-45_{0.35}/+45_{0.35}/90_{0.20}]$

Flight conditions



Cruise

$\mathrm{M}_\infty = 0.82 - \mathrm{FL}\,350$

$n = 1.0$

Maneuver

$\mathrm{M}_\infty = 0.78 - \mathrm{FL}\,200$

$n = 2.5$

Flutter (low-speed)

$\mathrm{M}_\infty = 0.50$

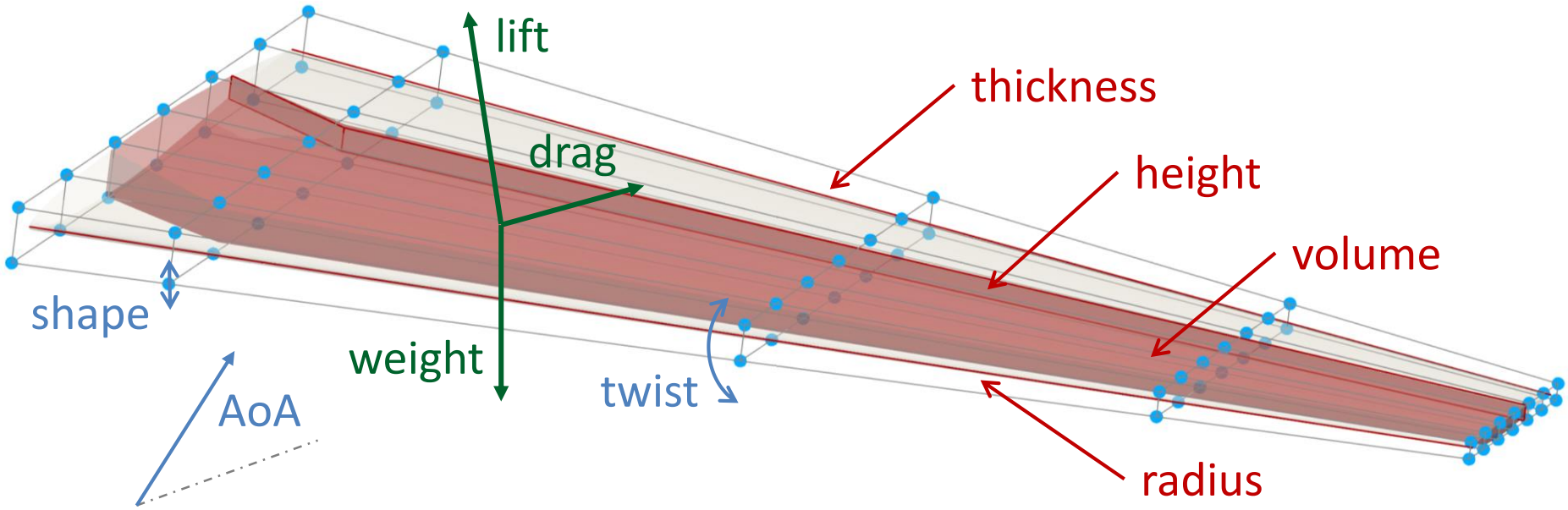
$\mathrm{FL} = [-200, +400]$

Flutter (high-speed)

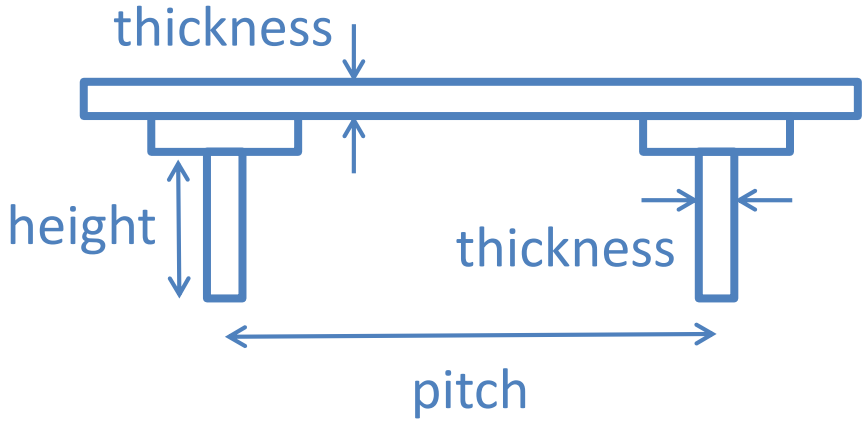
$\mathrm{M}_\infty = 0.95$

$\mathrm{FL} = [-200, +400]$

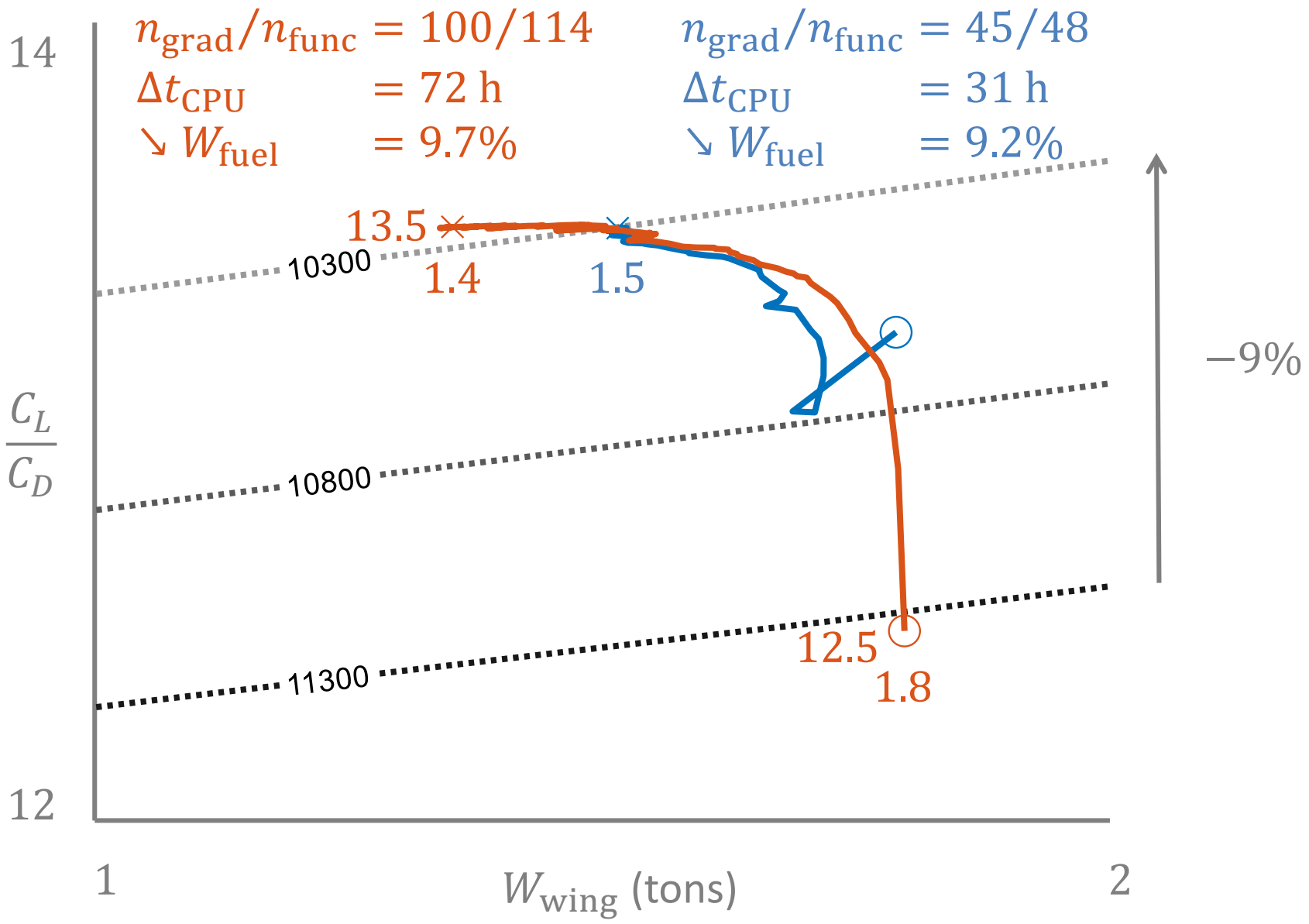
Optimization problem formulation



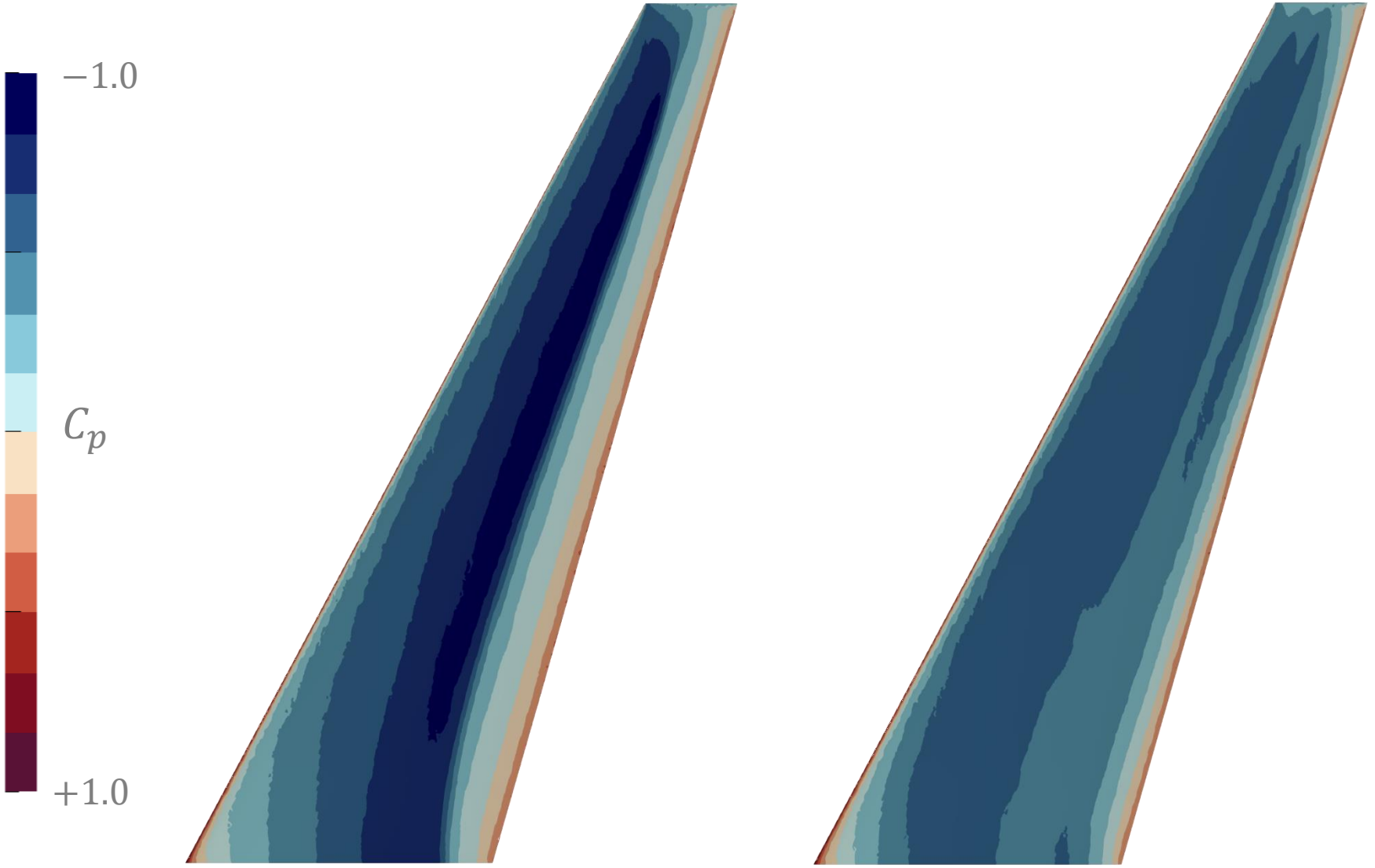
$$\begin{aligned} \min \quad & \text{fuel} = \text{Breguet}(\text{lift}, \text{drag}, \text{weight}) \\ \text{w. r. t.} \quad & \text{AoA}, \text{shape}, \text{twist}, \text{structure} \\ \text{s. t.} \quad & \text{load factor} \\ & \text{structural failure} \\ & \sim\text{flutter} \\ & \text{geometry} \end{aligned}$$



Fuel burn


$n_{\text{grad}}/n_{\text{func}} = 100/114$
$\Delta t_{\text{CPU}} = 72$ h
$\searrow W_{\text{fuel}} = 9.7\%$
$n_{\text{grad}}/n_{\text{func}} = 45/48$
$\Delta t_{\text{CPU}} = 31$ h
$\searrow W_{\text{fuel}} = 9.2\%$
14
12
$\frac{C_L}{C_D}$
10300
10800
11300
13.5
1.4
1.5
12.5
1.8
−9%
1
2
W_{wing} (tons)

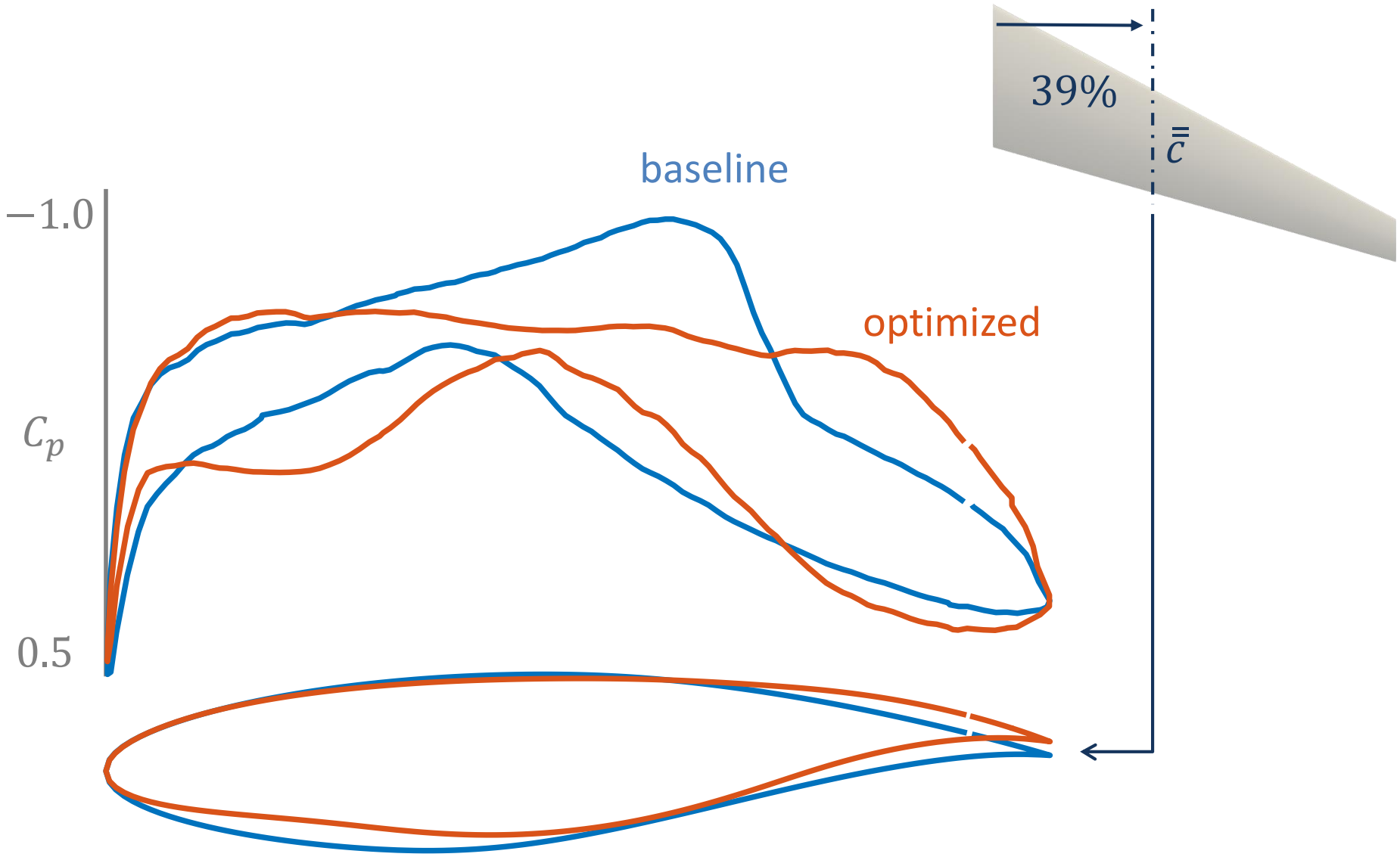
Pressure contour – cruise



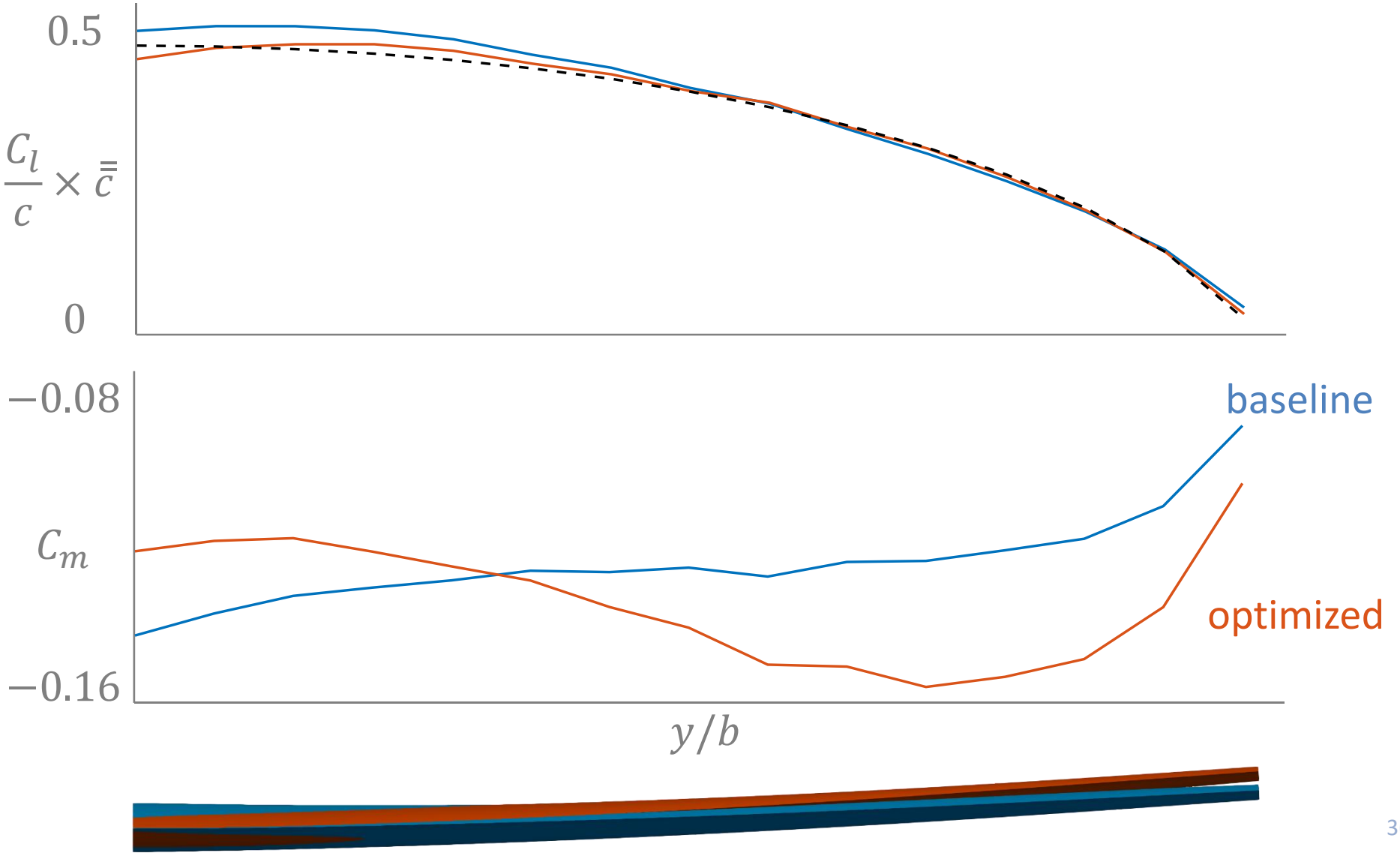
baseline

optimized

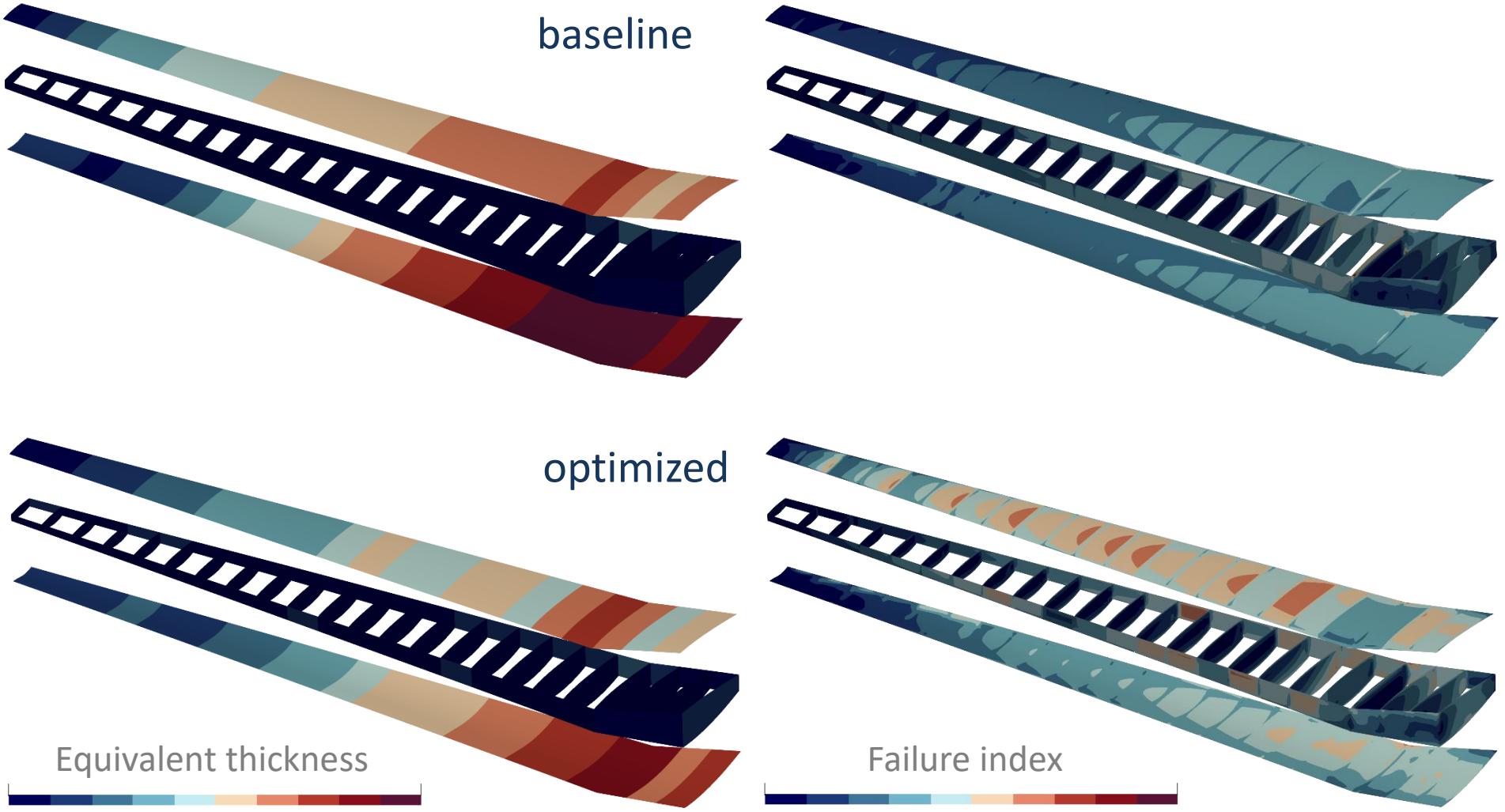
Pressure distribution – cruise



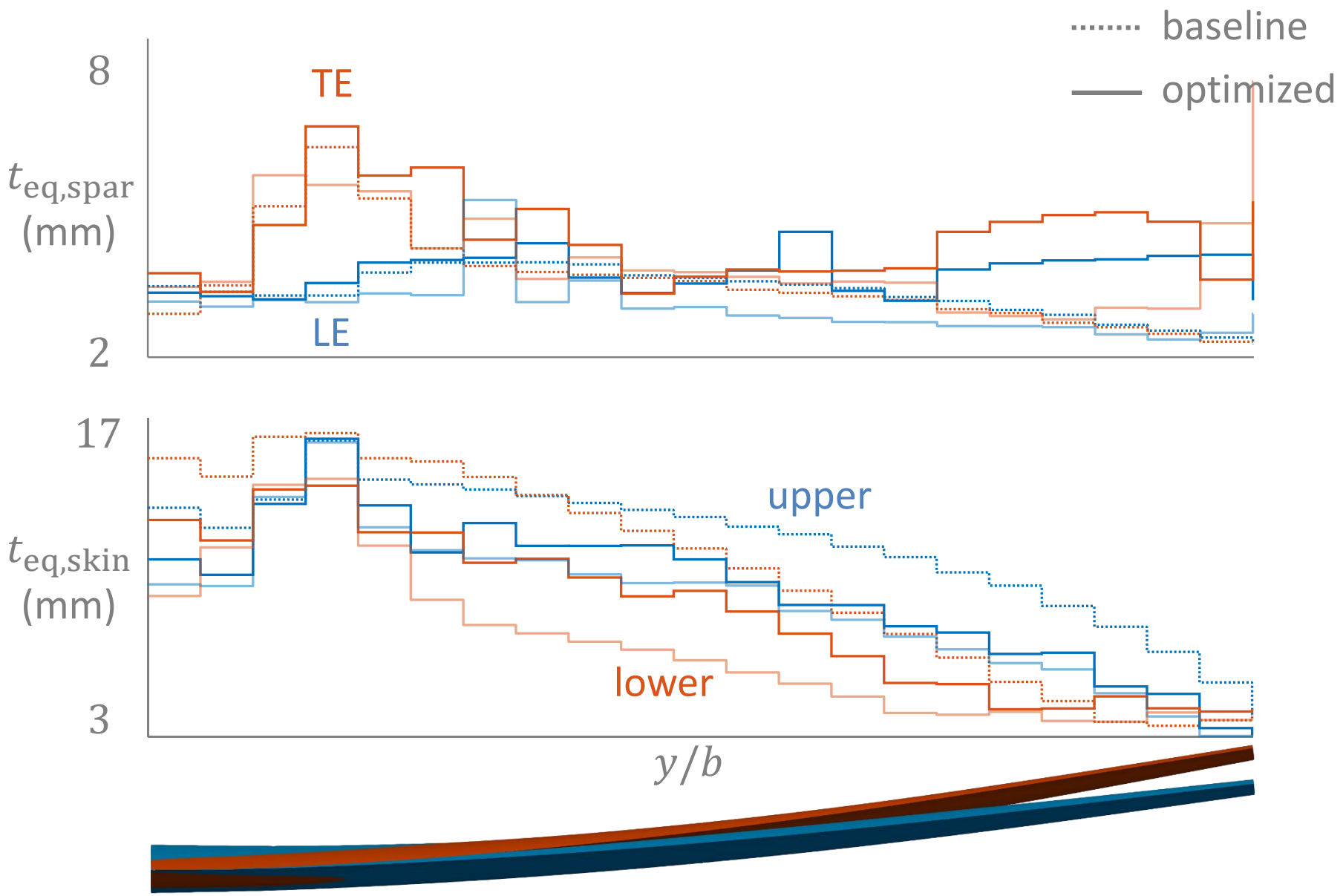
Loads distribution – cruise



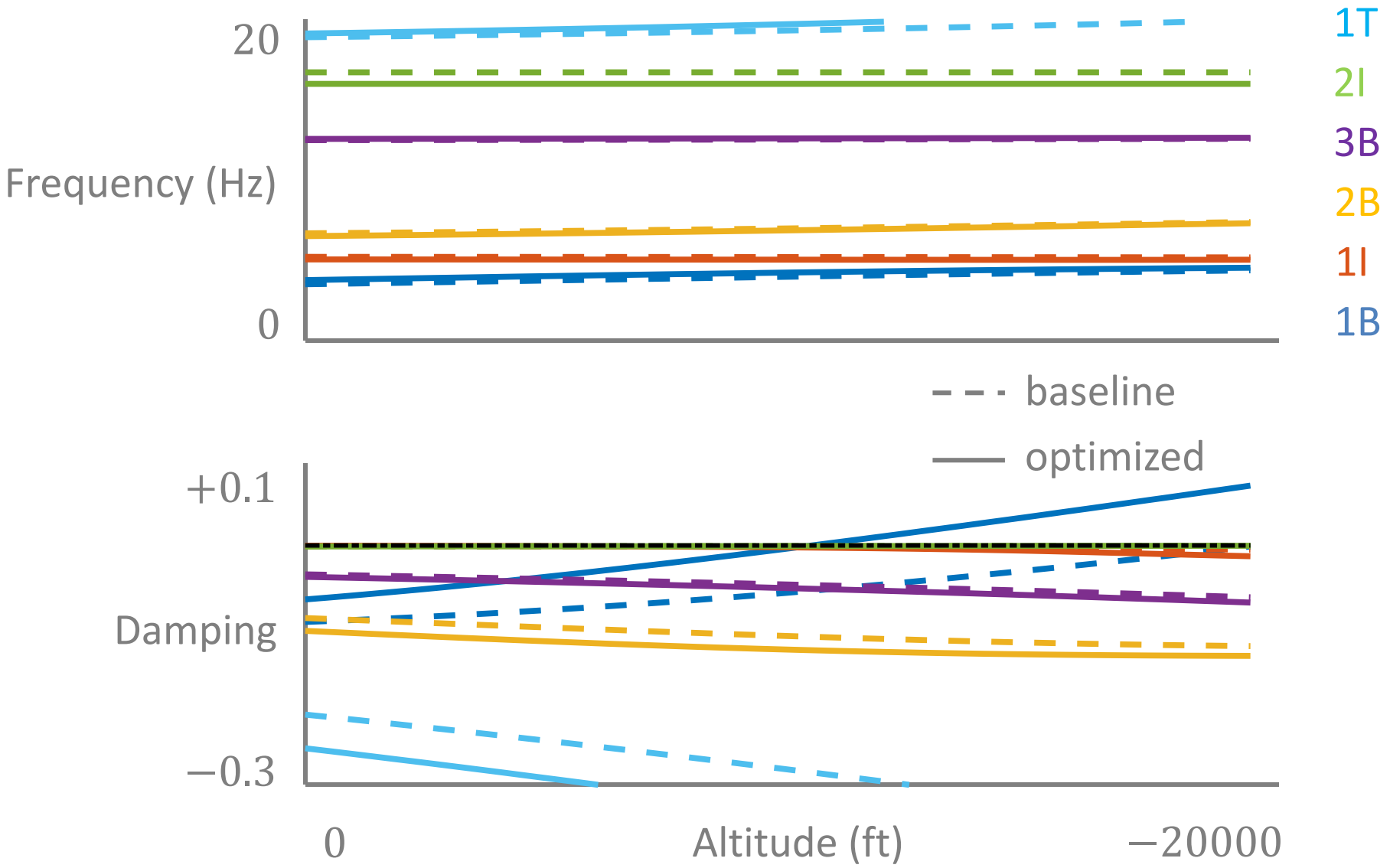
Thickness and failure index – maneuver



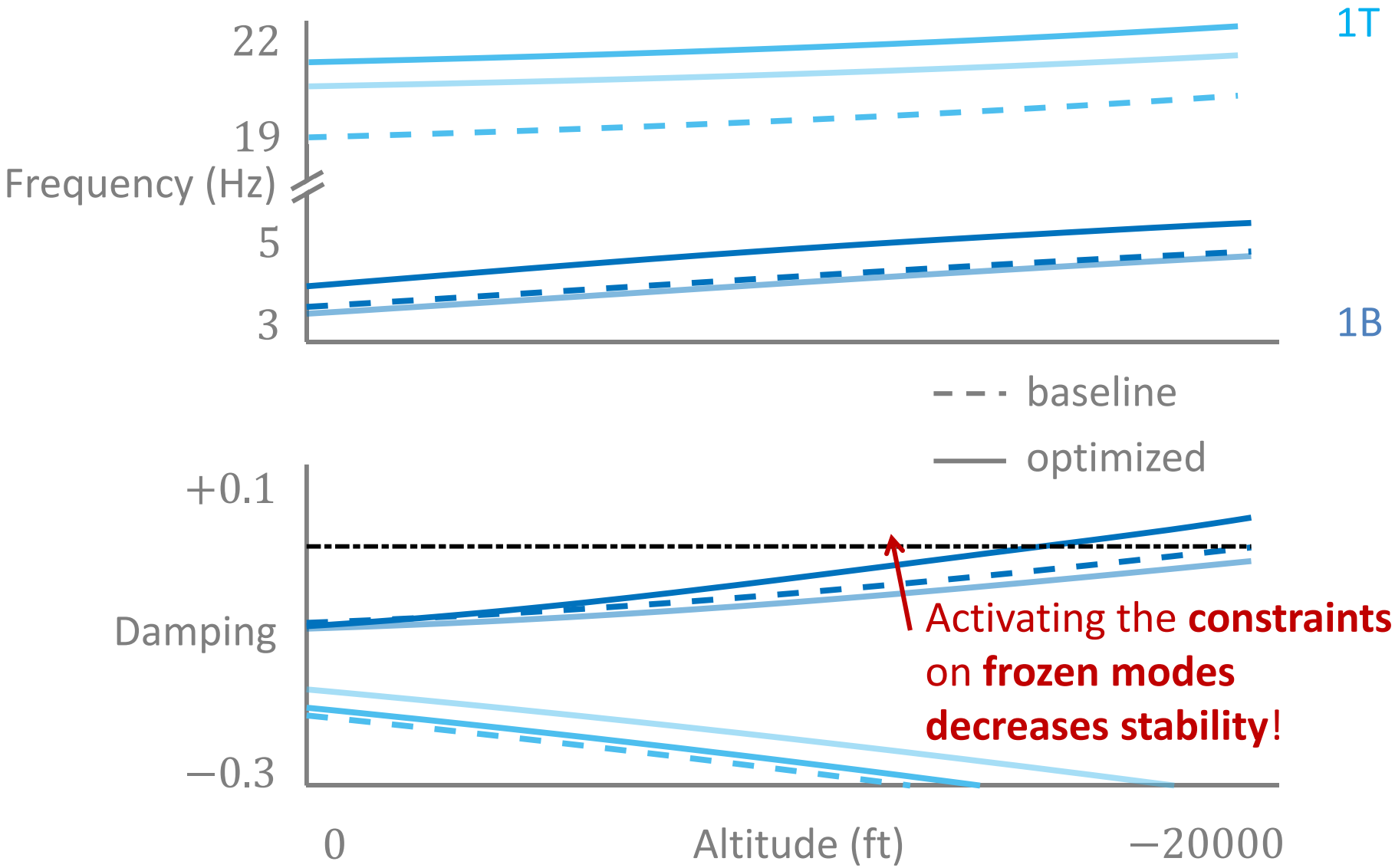
Equivalent thickness – maneuver



Modes migration – high-speed



Effect of flutter constraints – high-speed



Conclusion

Main points

- **Aerostructural optimization** is performed in **preliminary aircraft design**; choosing the **appropriate numerical models** and **methods** is of paramount importance
- **Developed DART** to **quickly** calculate **steady transonic** flows
- **Developed SDPM** to **quickly** calculate **unsteady** (transonic) flows
- **Implemented NIPK** with mode tracking to calculate **flutter**
- **Interfaced** all codes with **OpenMDAO**

Main results

- **Relevant** results for **static aerostructural** calculations can be obtained within a **few days**
- **Aeroelastic stability** can be **calculated** but **not constrained** yet

Conclusion

Challenges

- Some **structural** and **aerodynamic partial derivatives** required to calculate **total damping derivative** are **missing**; optimization **may not converge**
- **Flutter** modeling is **simplified**; **structural states** and **transonic aerodynamics** are **not taken** into **account**
- **Flutter** modeling is **costly;** accounts for **a third** of **computational time**

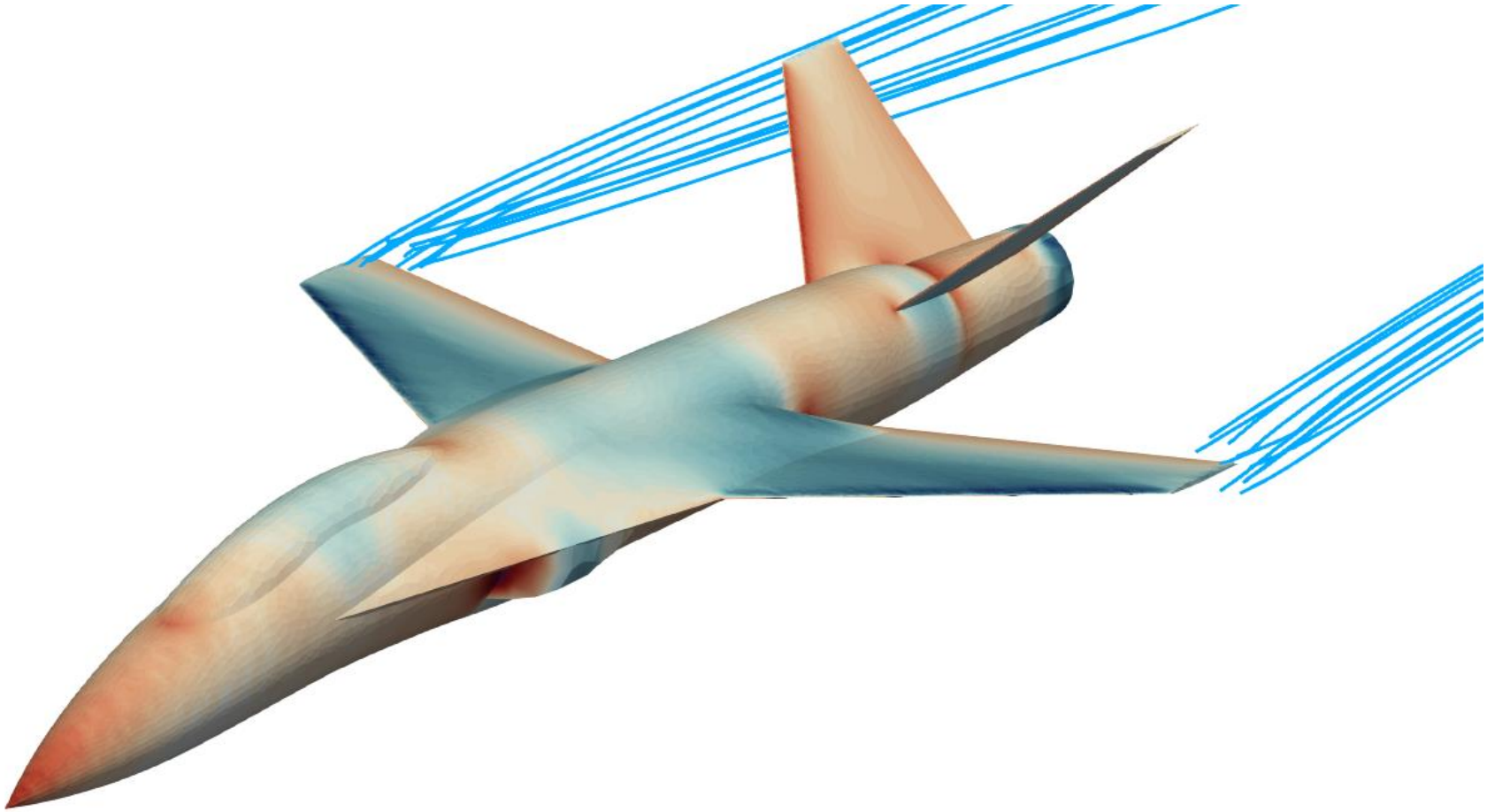
Next steps

- **Integrate viscous-inviscid interaction** in static optimization
- **Implement** missing **damping derivatives** for flutter constraint
- **Improve flutter** modeling and **reduce** computational **cost**
- **Improve benchmark** (optimize composite material, add push-down case, use full aircraft configuration, etc.)

ONERA seminar

Aerostructural modeling for preliminary aircraft design

Adrien Crovato – Châtillon, June 2024



EMBRAER

https://acrovato.github.io

LIÈGE université