Analytical and Numerical Approaches for the Computation of Aeroelastic Sensitivities Using the Direct and Adjoint Methods

Lukas Scheucher

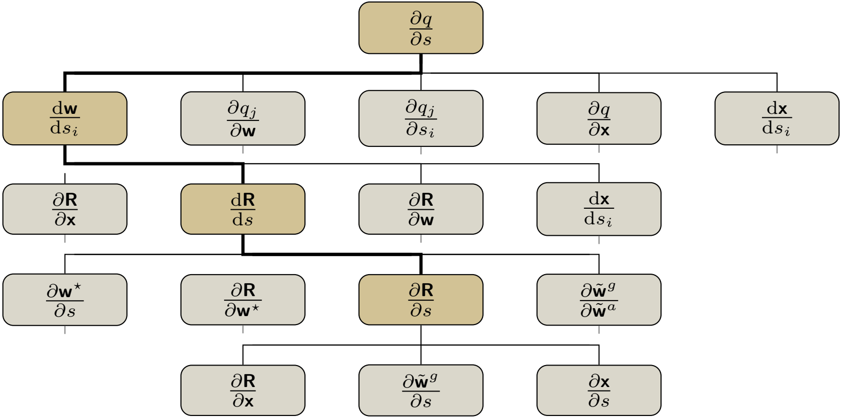
Stanford University

October 2016 - June 2017

Overview

Outline for section 1

AAA



$\frac{\partial q}{\partial s}$
$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i}$
$\frac{\partial q_j}{\partial \mathbf{w}}$
$\frac{\partial q_j}{\partial s_i}$
$\frac{\partial q}{\partial \mathbf{x}}$
$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}$
$\frac{\partial \mathbf{R}}{\partial \mathbf{x}}$
$\frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s}$
$\frac{\partial \mathbf{R}}{\partial \mathbf{w}}$
$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}$
$\frac{\partial \mathbf{w}^\star}{\partial s}$
$\frac{\partial \mathbf{R}}{\partial \mathbf{w}^\star}$
$\frac{\partial \mathbf{R}}{\partial s}$
$\frac{\partial \tilde{\mathbf{w}}^g}{\partial \tilde{\mathbf{w}}^a}$
$\frac{\partial \mathbf{R}}{\partial \mathbf{x}}$
$\frac{\partial \tilde{\mathbf{w}}^g}{\partial s}$
$\frac{\partial \mathbf{x}}{\partial s}$

Motivation
Basics

- Aerodynamic optimization
  - Gradient based
  - Take the human out of the loop
- Requirements on CFD
  - Complex flows (transonic, turbulent . . .) and high Reynolds numbers
  - Well-resolved boundary layers and flow features
  - Steady/unsteady flows
  - Numerical accuracy, solver robustness and short turn-around time
  - Moderate to highly complex geometries
- Requirements on design and optimization
  - Automatic framework
  - Efficient optimization algorithms
  - Large number of design variables
  - Multi-point design and multi-disciplinary design optimization
  - Geometrical/engineering constraints

Motivation
-> Why Sensitivity Analysis?

- Aerodynamic optimization
  - **Gradient based**
  - **Take the human out of the loop**
- Requirements on CFD
  - Complex flows (transonic, turbulent . . .) and high Reynolds numbers
  - Well-resolved boundary layers and flow features
  - Steady/unsteady flows
  - Numerical accuracy, solver robustness and short turn-around time
  - Moderate to highly complex geometries
- Requirements on design and optimization
  - Automatic framework
  - Efficient optimization algorithms
  - Large number of design variables
  - Multi-point design and multi-disciplinary design optimization
  - Geometrical/engineering constraints

Motivation
-> Why Embedded framework?

- Aerodynamic optimization
  - Gradient based
  - Take the human out of the loop
- Requirements on CFD
  - **Complex flows (transonic, turbulent . . .) and high Reynolds numbers**
  - Well-resolved boundary layers and flow features
  - Steady/unsteady flows
  - Numerical accuracy, solver robustness and short turn-around time
  - **Moderate to highly complex geometries**
- Requirements on design and optimization
  - Automatic framework
  - Efficient optimization algorithms
  - **Large number of design variables**
  - Multi-point design and multi-disciplinary design optimization
  - Geometrical/engineering constraints

→**Embedded framework**

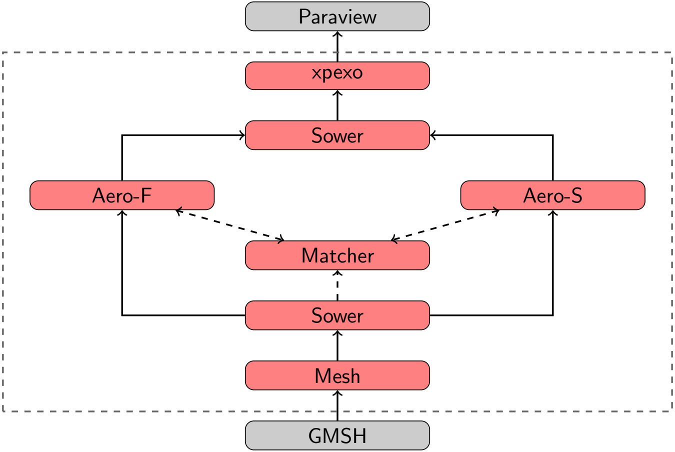
Motivation
-> Why analytic approach?

- Aerodynamic optimization
  - Gradient based
  - Take the human out of the loop
- Requirements on CFD
  - Complex flows (transonic, turbulent ...) and high Reynolds numbers
  - Well-resolved boundary layers and flow features
  - Steady/unsteady flows
  - **Numerical accuracy, solver robustness and short turn-around time**
  - Moderate to highly complex geometries
- Requirements on design and optimization
  - Automatic framework
  - Efficient optimization algorithms
  - Large number of design variables
  - Multi-point design and multi-disciplinary design optimization
  - Geometrical/engineering constraints

$\rightarrow$**Analytic Sensitivities**

The AERO-Suite[1,2]

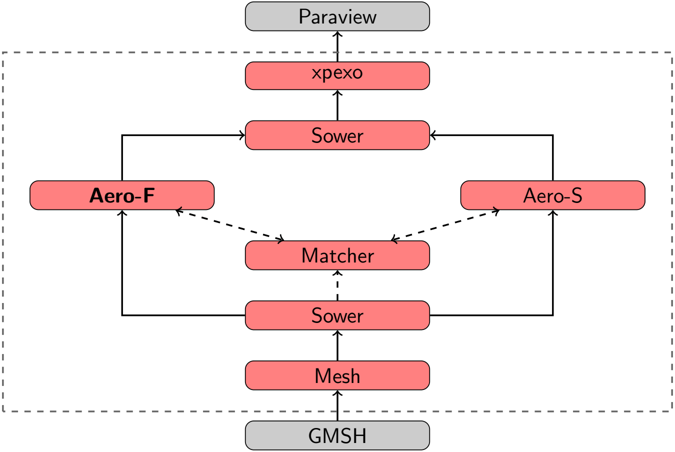
Workflow



[1] **Aerof**.
[2] **Aeros**.

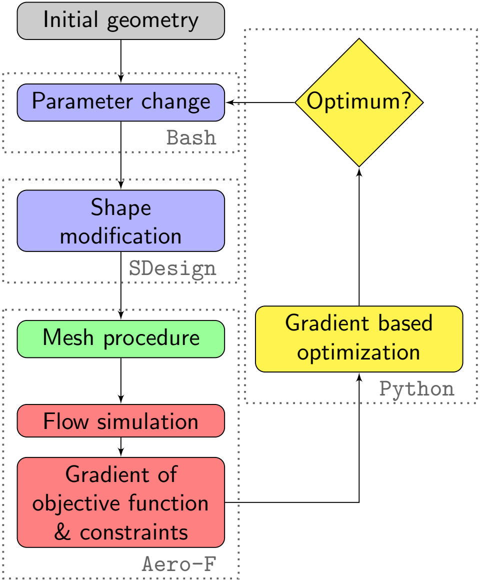
The AERO-Suite[1,2]

Workflow



[1] **Aerof**.
[2] **Aeros**.

Aerodynamic shape optimization
-> Overview



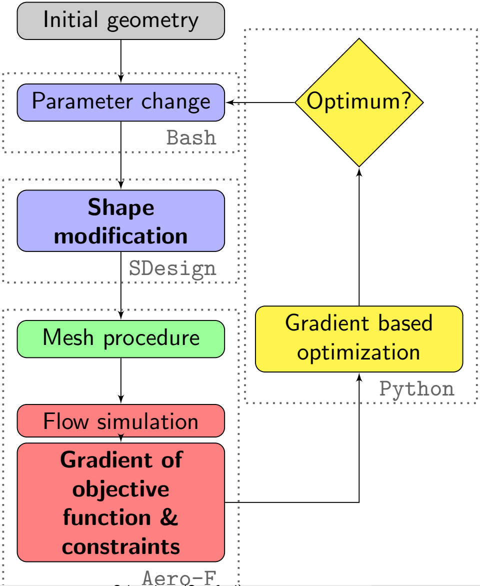
Gradient based optimization

- Requires objective function and constraints
- Gradient of objective function and constraints

How to compute the gradient

- Finite difference
- Direct approach
- Adjoint approach

Aerodynamic shape optimization
-> Thesis focus



Gradient based optimization

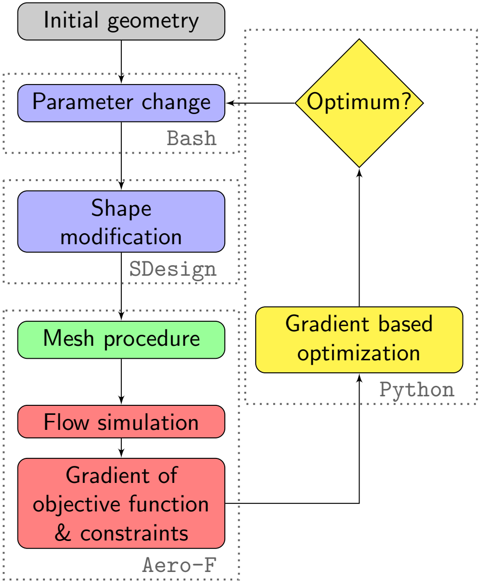
- Requires objective function and constraints
- Gradient of objective function and constraints

How to compute the gradient

- Finite difference
- **Direct approach**
- Adjoint approach

Outline for section 2

Aerodynamic shape optimization
-> Overview



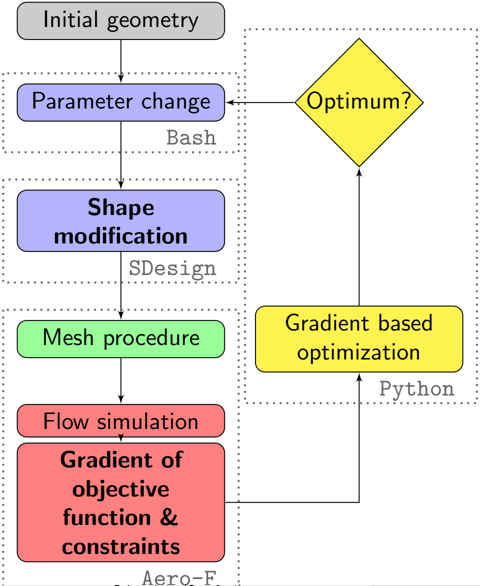
Gradient based optimization

- Requires objective function and constraints
- Gradient of objective function and constraints

How to compute the gradient

- Finite difference
- Direct approach
- Adjoint approach

Aerodynamic shape optimization
-> Thesis focus



Gradient based optimization

- Requires objective function and constraints
- Gradient of objective function and constraints

How to compute the gradient

- Finite difference
- **Direct approach**
- Adjoint approach

PDE-constrained optimization

- PDE-constrained optimization for steady problems

$$\begin{aligned} \underset{\mathbf{w}\in\mathbb{R}^{N_\mathbf{w}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad & q(\mathbf{w}, s) \\ \text{subject to} \quad & \mathbf{R}(\mathbf{w}, s) = 0 \\ & \mathbf{c}(\mathbf{w}, s) \leq 0 \end{aligned}$$

- Nested approach

$$\begin{aligned} \underset{\boldsymbol{\mu}\in\mathbb{R}^{N_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & q(\mathbf{w}(s), s) \\ \text{subject to} \quad & \mathbf{c}(\mathbf{w}(s), s) \leq 0. \end{aligned}$$

PDE-constrained optimization

- PDE-constrained optimization for steady problems

$$\begin{aligned} \underset{\mathbf{w}\in\mathbb{R}^{N_{\mathbf{w}}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad & q(\mathbf{w}, s) \rightarrow \textbf{e.g. Lift-Drag ratio} \\ \text{subject to} \quad & \mathbf{R}(\mathbf{w}, s) = 0 \\ & \mathbf{c}(\mathbf{w}, s) \leq 0 \end{aligned}$$

- Nested approach

$$\begin{aligned} \underset{\boldsymbol{\mu}\in\mathbb{R}^{N_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & q(\mathbf{w}(s), s) \\ \text{subject to} \quad & \mathbf{c}(\mathbf{w}(s), s) \leq 0. \end{aligned}$$

PDE-constrained optimization

- PDE-constrained optimization for steady problems

$$\begin{aligned} \underset{\mathbf{w}\in\mathbb{R}^{N_\mathbf{w}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad & q(\mathbf{w}, s) \rightarrow \textbf{e.g. Lift-Drag ratio} \\ \text{subject to} \quad & \mathbf{R}(\mathbf{w}, s) = 0 \rightarrow \textbf{e.g. geometry of engine mount} \\ & \mathbf{c}(\mathbf{w}, s) \leq 0 \end{aligned}$$

- Nested approach

$$\begin{aligned} \underset{\boldsymbol{\mu}\in\mathbb{R}^{N_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & q(\mathbf{w}(s), s) \\ \text{subject to} \quad & \mathbf{c}(\mathbf{w}(s), s) \leq 0. \end{aligned}$$

PDE-constrained optimization

- PDE-constrained optimization for steady problems

$$\begin{aligned} \underset{\mathbf{w}\in\mathbb{R}^{N_\mathbf{w}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad & q(\mathbf{w}, s) \rightarrow \textbf{e.g. Lift-Drag ratio} \\ \text{subject to} \quad & \mathbf{R}(\mathbf{w}, s) = 0 \rightarrow \textbf{e.g. geometry of engine mount} \\ & \mathbf{c}(\mathbf{w}, s) \leq 0 \rightarrow \textbf{e.g. minimum Lift} \end{aligned}$$

- Nested approach

$$\begin{aligned} \underset{\boldsymbol{\mu}\in\mathbb{R}^{N_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & q(\mathbf{w}(s), s) \\ \text{subject to} \quad & \mathbf{c}(\mathbf{w}(s), s) \leq 0. \end{aligned}$$

PDE-constrained optimization

- PDE-constrained optimization for steady problems

$$\begin{aligned} \underset{\mathbf{w}\in\mathbb{R}^{N_\mathbf{w}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad & q(\mathbf{w}, s) \rightarrow \textbf{e.g. Lift-Drag ratio} \\ \text{subject to} \quad & \mathbf{R}(\mathbf{w}, s) = 0 \rightarrow \textbf{e.g. geometry of engine mount} \\ & \mathbf{c}(\mathbf{w}, s) \leq 0 \rightarrow \textbf{e.g. minimmum Lift} \end{aligned}$$

- Nested approach

$$\begin{aligned} \underset{\boldsymbol{\mu}\in\mathbb{R}^{N_{\boldsymbol{\mu}}}}{\text{minimize}} \quad & q(\mathbf{w}(s), s) \\ \text{subject to} \quad & \mathbf{c}(\mathbf{w}(s), s) \leq 0. \end{aligned}$$

Gradient based optimization requires total derivatives(**Sensitivites**) !

Outline for section 3

Computation of the gradient

- Objective

$$\underset{\mathbf{w}\in\mathbb{R}^{N_{\mathbf{w}}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad q(s, \mathbf{w}, \boldsymbol{x})$$

- Gradients of the objective function

$$\left.\frac{dq_j}{ds_i}\right|_{\boldsymbol{w}_0} = \underbrace{\left.\frac{\partial q_j}{\partial s_i}\right|_{\mathbf{w}_0}}_{\substack{\text{directly derived from} \\ \text{the definition of } q}} +$$

Computation of the gradient

- Objective

$$\underset{\mathbf{w}\in\mathbb{R}^{N_\mathbf{w}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad q(s,\mathbf{w},\boldsymbol{x})$$

- Gradients of the objective function

$$\left.\frac{dq_j}{ds_i}\right|_{\boldsymbol{w}_0} = \underbrace{\left.\frac{\partial q_j}{\partial s_i}\right|_{\mathbf{w}_0}}_{\text{directly derived from the definition of } q} + \underbrace{\left.\frac{\partial q_j}{\partial \mathbf{w}}\right|_{\mathbf{w}_0}}_{\text{derived analytically or by FD}}$$

Computation of the gradient

- Objective

$$\underset{\mathbf{w}\in\mathbb{R}^{N_{\mathbf{w}}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad q(s, \mathbf{w}, \boldsymbol{x})$$

- Gradients of the objective function

$$\left.\frac{dq_j}{ds_i}\right|_{\boldsymbol{w}_0} = \underbrace{\left.\frac{\partial q_j}{\partial s_i}\right|_{\mathbf{w}_0}}_{\substack{\text{directly derived from} \\ \text{the definition of } q}} + \underbrace{\left.\frac{\partial q_j}{\partial \mathbf{w}}\right|_{\mathbf{w}_0}}_{\substack{\text{derived analytically or} \\ \text{by FD}}} \quad \underbrace{\left.\frac{d\mathbf{w}}{ds_i}\right|_{\mathbf{w}_0}}_{\substack{\text{derived from dynamic} \\ \text{fluid equilibrium}}} +$$

Computation of the gradient

- Objective

$$\underset{\mathbf{w}\in\mathbb{R}^{N_{\mathbf{w}}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad q(s, \mathbf{w}, \boldsymbol{x})$$

- Gradients of the objective function

$$\left.\frac{dq_j}{ds_i}\right|_{\boldsymbol{w}_0} = \underbrace{\left.\frac{\partial q_j}{\partial s_i}\right|_{\mathbf{w}_0}}_{\substack{\text{directly derived from}\\ \text{the definition of } q}} + \underbrace{\left.\frac{\partial q_j}{\partial \mathbf{w}}\right|_{\mathbf{w}_0}}_{\substack{\text{derived analytically or}\\ \text{by FD}}} \underbrace{\left.\frac{d\mathbf{w}}{ds_i}\right|_{\mathbf{w}_0}}_{\substack{\text{derived from dynamic}\\ \text{fluid equilibrium}}} + \underbrace{\underbrace{\left.\frac{\partial q}{\partial \mathbf{x}}\right|_{\mathbf{w}_0}}_{\substack{\text{derived analytically or}\\ \text{by FD}}}}_{=0 \text{ for Embedded}}$$

Computation of the gradient

- Objective

$$\underset{\mathbf{w}\in\mathbb{R}^{N_{\mathbf{w}}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad q(s,\mathbf{w},\boldsymbol{x})$$

- Gradients of the objective function

$$\left.\frac{dq_j}{ds_i}\right|_{\boldsymbol{w}_0} = \underbrace{\left.\frac{\partial q_j}{\partial s_i}\right|_{\mathbf{w}_0}}_{\text{directly derived from the definition of } q} + \underbrace{\left.\frac{\partial q_j}{\partial \mathbf{w}}\right|_{\mathbf{w}_0}}_{\text{derived analytically or by FD}} \underbrace{\left.\frac{d\mathbf{w}}{ds_i}\right|_{\mathbf{w}_0}}_{\text{derived from dynamic fluid equilibrium}} + \underbrace{\underbrace{\left.\frac{\partial q}{\partial \mathbf{x}}\right|_{\mathbf{w}_0}}_{\text{derived analytically or by FD}} \underbrace{\left.\frac{\partial \mathbf{x}}{\partial s_i}\right|_{\mathbf{w}_0}}_{\text{derived from SDESIGN}}}_{=0 \text{ for Embedded}}$$

Computation of the gradient

- Objective

$$\underset{\mathbf{w}\in\mathbb{R}^{N_\mathbf{w}},\ s\in\mathbb{R}^{N_s}}{\text{minimize}} \quad q(s,\mathbf{w},\boldsymbol{x})$$

- Gradients of the objective function

$$\left.\frac{dq_j}{ds_i}\right|_{\boldsymbol{w}_0} = \underbrace{\left.\frac{\partial q_j}{\partial s_i}\right|_{\mathbf{w}_0}}_{\substack{\text{directly derived from}\\ \text{the definition of } q}} + \underbrace{\left.\frac{\partial q_j}{\partial \mathbf{w}}\right|_{\mathbf{w}_0}}_{\substack{\text{derived analytically or}\\ \text{by FD}}} \underbrace{\boxed{\left.\frac{d\mathbf{w}}{ds_i}\right|_{\mathbf{w}_0}}}_{\substack{\text{derived from dynamic}\\ \text{fluid equilibrium}}} + \underbrace{\underbrace{\left.\frac{\partial q}{\partial \mathbf{x}}\right|_{\mathbf{w}_0}}_{\substack{\text{derived analytically or}\\ \text{by FD}}} \underbrace{\left.\frac{\partial \mathbf{x}}{\partial s_i}\right|_{\mathbf{w}_0}}_{\text{derived from SDESIGN}}}_{=0 \text{ for Embedded}}$$

→ **We only look into this term**!

DERIVATION OF $\frac{\partial \mathbf{w}}{\partial s}$

- The final system can be written as

$$\left.\frac{\mathrm{d}q_j}{\mathrm{d}s_i}\right|_{\mathbf{w}_0} = -\left.\frac{\mathrm{d}q_j}{\mathrm{d}\boldsymbol{w}}\right|_{\mathbf{w}_0} \left[\left.\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right|_{\boldsymbol{w}_0}\right]^{-1}$$

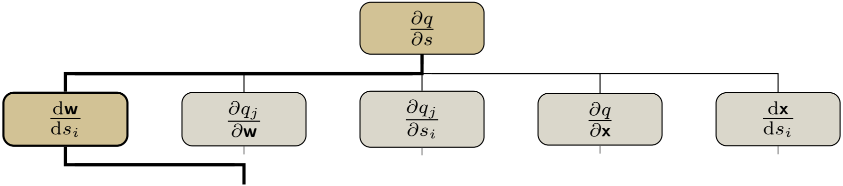
$$\left(\left.\frac{\partial \mathbf{R}}{\partial s_i}\right|_{\boldsymbol{w}_0} + \left[\alpha \left.\frac{\partial \mathbf{R}}{\partial \dot{\mathbf{x}}_\Omega}\right|_{\mathbf{w}_0} \quad \left.\frac{\partial \mathbf{R}}{\partial \dot{\mathbf{x}}_\Gamma}\right|_{\mathbf{w}_0} \right] \begin{bmatrix} \alpha \bar{\mathbf{K}}_{\Omega\Omega}^{-1} \bar{\mathbf{K}}_{\Omega\Gamma} \\ \mathbf{I} \end{bmatrix} \frac{\mathrm{d}\dot{\mathbf{x}}_\Gamma}{\mathrm{d}s_i} \right)$$

$$\alpha = \begin{cases} 1 \text{ in ALE framework} \\ 0 \text{ in Embedded framework} \end{cases}$$

$\rightarrow$ **We only look into these terms**

Sensitivity derivation - chain rule

$\frac{\partial q}{\partial s}$



Derivation of $\frac{\partial \mathbf{w}}{\partial s}$

- Consider the fluid equations at equilibrium

$$\cancelto{0}{\frac{\partial \bar{\mathbf{w}}_i}{\partial t}} + \underbrace{\underbrace{\sum_{j \in \kappa(i)} \boldsymbol{\phi}_{ij}(\mathbf{w}_{ij}, \mathbf{w}_{ji}, \boldsymbol{\nu}_{ij})}_{\mathbf{R}^i} - \underbrace{\sum_{T_i \in \lambda(i)} \int_{T_j} \mathbb{K} \nabla \mathbf{w} \nabla \phi_j dx}_{\mathbf{R}^v}}_{\mathbf{R}} = \mathbf{0}$$

Derivation of $\frac{\partial \mathbf{w}}{\partial s}$

- Consider the fluid equations at equilibrium

$$\cancelto{0}{\frac{\partial \bar{\mathbf{w}}_i}{\partial t}} + \underbrace{\underbrace{\sum_{j \in \kappa(i)} \boldsymbol{\phi}_{ij}(\mathbf{w}_{ij}, \mathbf{w}_{ji}, \boldsymbol{\nu}_{ij})}_{\mathbf{R}^i} - \underbrace{\sum_{T_i \in \lambda(i)} \int_{T_j} \mathbb{K} \nabla \mathbf{w} \nabla \phi_j dx}_{\mathbf{R}^v}}_{\mathbf{R}} = \mathbf{0}$$

- Therefore

$$\frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s_i} = \mathbf{0} = \frac{\partial \mathbf{R}}{\partial s_i} + \frac{\partial \mathbf{R}}{\partial \mathbf{w}} \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i} + \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}$$

Derivation of $\frac{\partial \mathbf{w}}{\partial s}$

- Consider the fluid equations at equilibrium

$$\cancelto{0}{\frac{\partial \bar{\mathbf{w}}_i}{\partial t}} + \underbrace{\underbrace{\sum_{j \in \kappa(i)} \phi_{ij}(\mathbf{w}_{ij}, \mathbf{w}_{ji}, \boldsymbol{\nu}_{ij})}_{\mathbf{R}^i} - \underbrace{\sum_{T_i \in \lambda(i)} \int_{T_j} \mathbb{K} \nabla \mathbf{w} \nabla \phi_j dx}_{\mathbf{R}^v}}_{\mathbf{R}} = \mathbf{0}$$

- Therefore

$$\frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s_i} = \mathbf{0} = \frac{\partial \mathbf{R}}{\partial s_i} + \frac{\partial \mathbf{R}}{\partial \mathbf{w}} \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i} + \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}$$

- Which leads to

$$\frac{\partial \mathbf{R}}{\partial \mathbf{w}} \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i} = -\frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s_i} - \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}$$

Derivation of $\frac{\partial \mathbf{w}}{\partial s}$
Direct and Adjoint method

- Matrix-Matrix-Matrix product

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i} = -\frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-1} \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s_i} - \frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-1} \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}$$

- Two different ways to do the product
  - Direct: **ABC** $\mathcal{O}(n_{eq}^2 n_{\boldsymbol{s}} + n_{\boldsymbol{q}} n_{eq} n_{\boldsymbol{s}})$
  - Adjoint: $\left[\mathbf{C}^T(\mathbf{AB})\right]^T$ $\mathcal{O}(n_{eq}^2 n_{\boldsymbol{q}} + n_{\boldsymbol{q}} n_{eq} n_{\boldsymbol{s}})$

$n_{eq} \rightarrow$ number of equations

$n_{\boldsymbol{s}} \rightarrow$ number of abstract variables

$n_{\boldsymbol{q}} \rightarrow$ number of optimization criteria

Derivation of $\frac{\partial \mathbf{w}}{\partial s}$
Direct and Adjoint method

- Matrix-Matrix-Matrix product

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i} = -\frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-1} \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s_i} - \underbrace{\frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-1} \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}}_{\mathbf{ABC}}$$

- Two different ways to do the product
  - Direct: **ABC** $\mathcal{O}(n_{eq}^2 n_{\boldsymbol{s}} + n_{\boldsymbol{q}} n_{eq} n_{\boldsymbol{s}})$
  - Adjoint: $\left[\mathbf{C}^T(\mathbf{AB})\right]^T$ $\mathcal{O}(n_{eq}^2 n_{\boldsymbol{q}} + n_{\boldsymbol{q}} n_{eq} n_{\boldsymbol{s}})$

$n_{eq} \rightarrow$ number of equations

$n_{\boldsymbol{s}} \rightarrow$ number of abstract variables

$n_{\boldsymbol{q}} \rightarrow$ number of optimization criteria

Derivation of $\frac{\partial \mathbf{w}}{\partial s}$
Direct and Adjoint method

- Matrix-Matrix-Matrix product

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i} = -\frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-1} \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s_i} - \underbrace{\frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-1} \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s_i}}_{\mathbf{ABC}}$$

- Two different ways to do the product
  - Direct: **ABC** $\mathcal{O}(n_{eq}^2 n_{\boldsymbol{s}} + n_{\boldsymbol{q}} n_{eq} n_{\boldsymbol{s}})$
  - Adjoint: $\left[\mathbf{C}^T(\mathbf{AB})\right]^T$ $\mathcal{O}(n_{eq}^2 n_{\boldsymbol{q}} + n_{\boldsymbol{q}} n_{eq} n_{\boldsymbol{s}})$

$n_{eq} \rightarrow$ number of equations

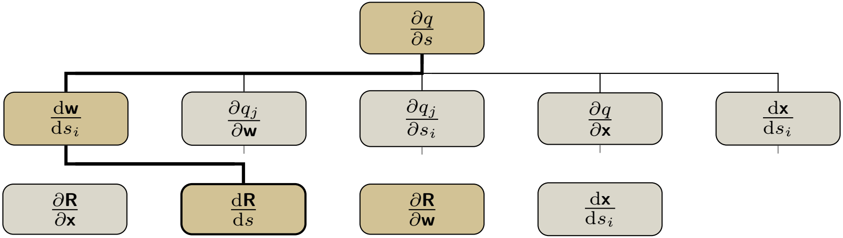
$n_{\boldsymbol{s}} \rightarrow$ number of abstract variables

$n_{\boldsymbol{q}} \rightarrow$ number of optimization criteria

We only look into these terms

Sensitivity derivation - chain rule

$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}s_i}$



Derivation of $\frac{\partial \mathbf{R}}{\partial \mathbf{w}}$

- Also known as fluid Jacobian

$$\frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \mathbf{w}_k} = \frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \tilde{\mathbf{w}}_{ij}^{\star}} \frac{\partial \tilde{\mathbf{w}}_{ij}^{\star}}{\partial \tilde{\mathbf{w}}_k} \frac{\partial \tilde{\mathbf{w}}_k}{\partial \mathbf{w}_k} + \frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \tilde{\mathbf{w}}_{ij}} \frac{\partial \tilde{\mathbf{w}}_{ij}}{\partial \tilde{\mathbf{w}}_k} \frac{\partial \tilde{\mathbf{w}}_k}{\partial \mathbf{w}_k} + \underbrace{\frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \boldsymbol{n}_{ij}}}_{=\mathbf{0} \text{ for embedded}}$$

  - Analytical Jacobian of the (Roe's) centering flux
  - Analytical derivative of the solution of the 1D half-Riemann problem
  - Analytical derivative of the MUSCL reconstruction and limitation

Derivation of $\frac{\partial \mathbf{R}}{\partial \mathbf{w}}$

- Also known as fluid Jacobian

$$\frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \mathbf{w}_k} = \frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \tilde{\mathbf{w}}_{ij}^{\star}} \frac{\partial \tilde{\mathbf{w}}_{ij}^{\star}}{\partial \tilde{\mathbf{w}}_k} \frac{\partial \tilde{\mathbf{w}}_k}{\partial \mathbf{w}_k} + \frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \tilde{\mathbf{w}}_{ij}} \frac{\partial \tilde{\mathbf{w}}_{ij}}{\partial \tilde{\mathbf{w}}_k} \frac{\partial \tilde{\mathbf{w}}_k}{\partial \mathbf{w}_k} + \underbrace{\frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \boldsymbol{n}_{ij}}}_{=\mathbf{0} \text{ for embedded}}$$

  - Analytical Jacobian of the (Roe's) centering flux
  - Analytical derivative of the solution of the 1D half-Riemann problem
  - Analytical derivative of the MUSCL reconstruction and limitation

→ **We don't go into any more detail here!**

Analytic derivatives

$\frac{\partial \mathbf{R}}{\partial s_i}$

- Remember

$$\mathbf{R} = \mathbf{R}^i + \mathbf{R}^v$$

- Approach
  - Separate treatment of inviscid and viscous contribution
  - Re-use information from the intersector, obtained by FIVER, whenever possible
- Inviscid part

$$\mathbf{R}_{ij}^{c,i} = \boldsymbol{\phi}_{ij}^{i}(\tilde{\mathbf{w}}_{ij}, \tilde{\mathbf{w}}_{ij}^{*}(s), \boldsymbol{n}_{ij})$$
$$\frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial s} = \frac{\partial \mathbf{R}_{ij}^{c,i}}{\partial \mathbf{w}_{ij}^{\star}} \frac{\partial \mathbf{w}_{ij}^{\star}}{\partial s}$$

Analytic derivatives

$\frac{\partial \mathbf{R}}{\partial s_i}$

- Diffusive part

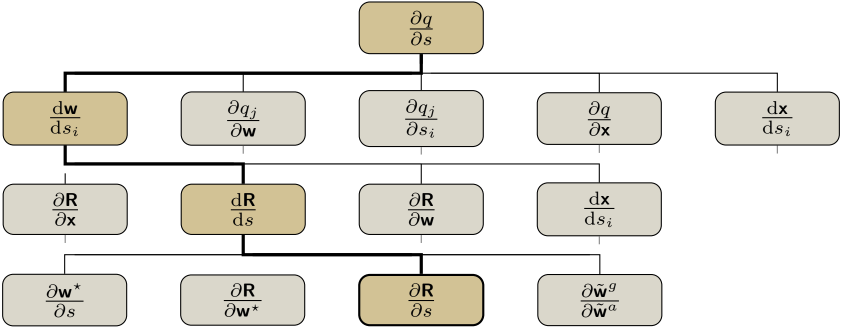
$$\mathbf{R}_i^v = -\sum_{T_i \in \lambda(i)} \sum_{i=1}^{n_g} w_i \tilde{\mathbb{K}} \nabla \tilde{\mathbf{w}}(\mathbf{x}_i) \nabla \phi_j(\mathbf{x}_i) dx$$

$$\frac{\partial \mathbf{R}^v(\boldsymbol{s}, \tilde{\mathbf{w}}^a(\boldsymbol{s}), \tilde{\mathbf{w}}^g(\tilde{\mathbf{w}}^a(\boldsymbol{s})), \boldsymbol{x}(\boldsymbol{s}))}{\partial \boldsymbol{s}} =$$

$$\underbrace{\frac{\partial \mathbf{R}^v}{\partial \tilde{\mathbf{w}}^a} \frac{\partial \tilde{\mathbf{w}}^a}{\partial \boldsymbol{s}}}_{\substack{\text{can be re-used from ALE} \\ \text{after the ghost-point population}}} + \frac{\partial \mathbf{R}^v}{\partial \tilde{\mathbf{w}}^g} \cdot \overbrace{\frac{\partial \tilde{\mathbf{w}}^g}{\partial \tilde{\mathbf{w}}^a}}^{\substack{\text{obtained during the} \\ \text{population process}}} \frac{\partial \tilde{\mathbf{w}}^a}{\partial \boldsymbol{s}} + \underbrace{\frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \boldsymbol{s}}}_{= \mathbf{0} \text{ for embedded}}$$

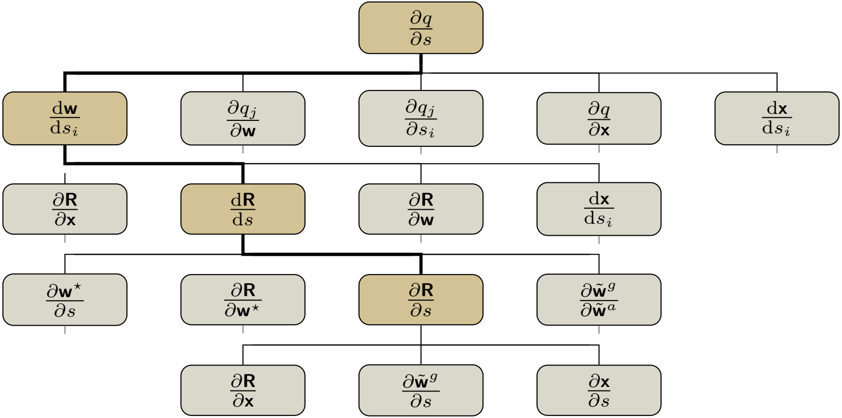
Sensitivity derivation - chain rule

$\frac{\mathrm{d}\mathbf{R}}{\mathrm{d}s}$



Sensitivity derivation - chain rule

$\frac{\partial \mathbf{R}}{\partial s}$



Outline for section 4

Compressible Navier Stokes equations

The compressible Navier Stokes equations in conservative form can be written as

$$\underbrace{\frac{\partial \bar{\boldsymbol{w}}}{\partial t}}_{\text{time derivative}} + \underbrace{\nabla \cdot \mathcal{F}(\bar{\boldsymbol{w}})}_{\text{inviscid}} + \underbrace{\nabla \cdot \mathcal{G}(\bar{\boldsymbol{w}})}_{\text{viscous}} = \underbrace{\mathsf{S}(\bar{\boldsymbol{w}}, \chi_1, \cdots, \chi_m)}_{\text{source term}}$$

Inviscid fluxes

$$\mathcal{F} = \boldsymbol{w}\boldsymbol{v}^T + p \begin{bmatrix} 0 \\ \mathbf{I} \\ \boldsymbol{v}^T \end{bmatrix}$$

Viscous fluxes

$$\mathcal{G} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau} \\ \boldsymbol{\tau}\boldsymbol{v} + \boldsymbol{q} \end{bmatrix}$$

Body-fitted vs. embedded framework

Mesh is structure specific

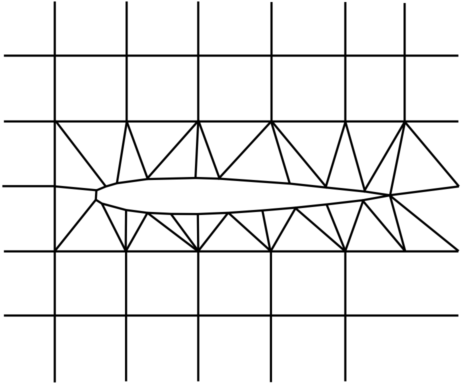


Figure: Body-fitted

Body-fitted vs. embedded framework

Mesh is structure specific

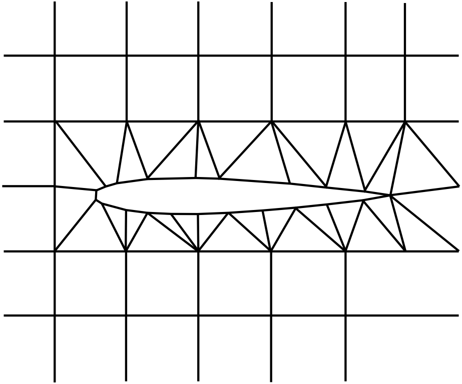


Figure: Body-fitted

Mesh independent of structure geometry

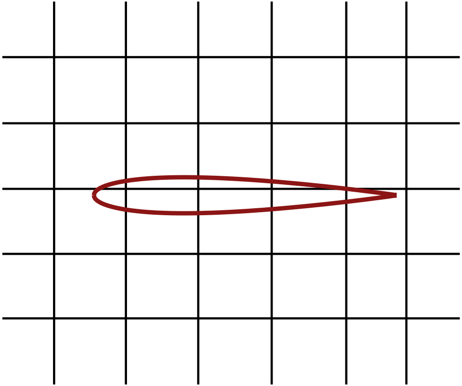
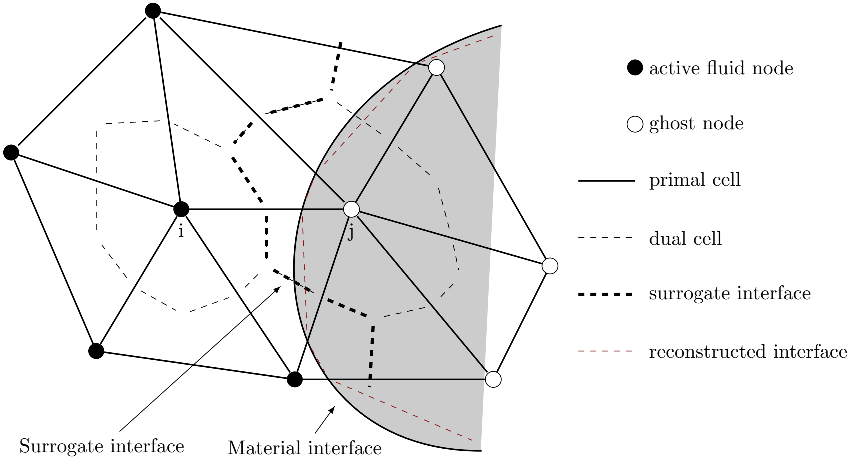


Figure: Embedded

Immersed Boundary Method

FIVER[3]



[3]Main, "Implicit and higer-order discretization methods for compressible multi-phase fluid and fluid-structure problems".

Discretization

- Body-fitted and Immersed boundaries (FIVER)
- FE-like treatment of the visocus term

$$\frac{\partial \boldsymbol{w}_i}{\partial t} + \int_{\partial \mathcal{C}_i} \mathcal{F}(\boldsymbol{w}) \cdot dS - \int_{\sum_{T_i}} \mathbb{K}\boldsymbol{w} \nabla \phi_i dx = \mathbf{0}$$

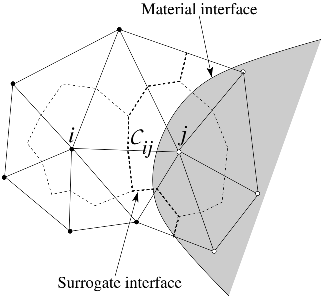
$$\int_{\partial \mathcal{C}_i} \mathcal{F}(\boldsymbol{w}) \cdot dS \approx \underbrace{\sum_{j \in \kappa(i)^a} \boldsymbol{\phi}_{ij}(\boldsymbol{w}_i, \boldsymbol{w}_j, \boldsymbol{\nu}_{ij})}_{\text{non-intersected elements}} + \underbrace{\sum_{j \in \kappa(i) \setminus \kappa(i)^a} \boldsymbol{\phi}_{ij}(\boldsymbol{w}_i, \boldsymbol{w}^*, \boldsymbol{\nu}_{ij})}_{\text{intersected elements treated with FIVER}}$$

ϕ ... flux function of Roe[4]

[4] Roe, "Approximate Riemann solvers, parameter vectors, and difference schemes".

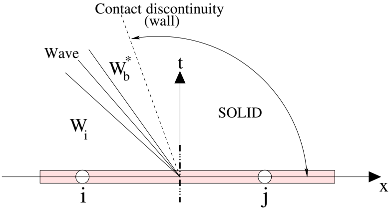
IB WITH THE FIVER APPROACH: ORIGINAL FORMULATION

- Identify immersed boundaries with control volume interfaces $\mathcal{C}_{ij}$



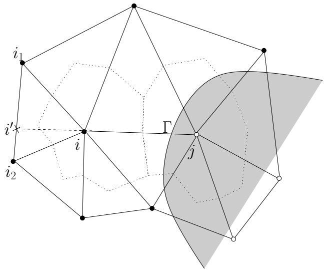
- Solve exactly local one-dimensional half-Riemann problems at $\mathcal{C}_{ij}$

$$\begin{cases} \dfrac{\partial \tilde{\mathbf{w}}^\star}{\partial \tau} + \dfrac{\partial \tilde{\mathbf{f}}(\tilde{\mathbf{w}}^\star)}{\partial \xi} = 0 & \\ \tilde{\mathbf{w}}^\star(\xi, 0) = \tilde{\mathbf{w}}_{ij}, & \xi \leq 0 \\ \boldsymbol{v}(0,\tau) \cdot \mathbf{n}_{\text{wall}} = \boldsymbol{v}_{\text{wall}} \cdot \mathbf{n}_{\text{wall}}, & 0 \leq \tau \leq \Delta t \end{cases}$$



- Evaluate numerical flux: $\mathbf{F}_{ij} = \mathbf{F}_{ij}(\mathbf{w}_{ij}, \mathbf{w}_b^\star, \boldsymbol{n}_{ij})$

IB with the FIVER approach: enhanced formulation

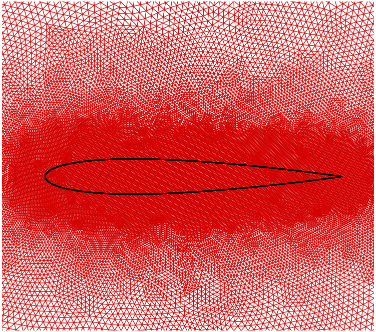


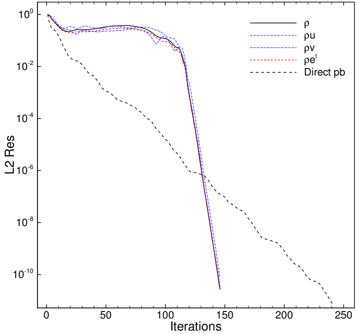
- The fluid state is extrapolated to the material interface Γ
$$\mathbf{w}_\Gamma = \mathbf{w}_i + \nabla \mathbf{w}_i \cdot (\mathbf{x}_\Gamma - \mathbf{x}_i)$$
- The one-dimensional half-Riemann problem is solved at material interface Γ
$$\tilde{\mathbf{w}}^\star_\Gamma = \tilde{\mathbf{w}}^\star(\tilde{\mathbf{w}}_\Gamma, \boldsymbol{v}_{\text{wall}}, \mathbf{n}_{\text{wall}})$$
- The fluid state is inter/extra-polated at control volume interface $\mathcal{C}_{ij}$
$$\mathbf{w}^\star_{ij} = \mathbf{w}^\star_{ij}(\mathbf{w}^\star_\Gamma, \mathbf{w}_{i'})$$
- Numerical flux at the control volume interface: $\mathbf{F}_{ij} = \mathbf{F}_{ij}(\mathbf{w}_{ij}, \mathbf{w}^\star_{ij}, \boldsymbol{n}_{ij})$
- Second-order convergence is recovered in the vicinity of the interface

Outline for section 5

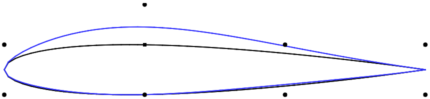
Verification of the analytical sensitivities

NACA-0012, Ma $= 0.5$, $\alpha = 2°$





- 3D Grid $\sim 200\,000$ nodes
- CFD CPU time ~ 10 min
- Direct problem CPU time: seconds



Outline for section 6

- Validating the fluid Jacobian via Finite Difference

$$\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\bigg|_{\boldsymbol{w}_0} \mathbf{u} = \frac{\mathbf{R}_i(\mathbf{w}_0 + \epsilon \mathbf{u}) - \mathbf{R}_i(\mathbf{w}_0 - \epsilon \mathbf{u})}{2\epsilon}$$

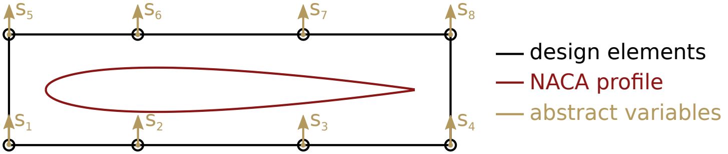
- Validate the fluid solution via the finite difference of two steady state simulations

$$\frac{\mathrm{d}\mathbf{w}(s)}{\mathrm{d}s}\Big|_{\boldsymbol{w}_0} = \frac{\mathbf{w}(s+\epsilon) - \mathbf{w}(s-\epsilon)}{2\epsilon}$$

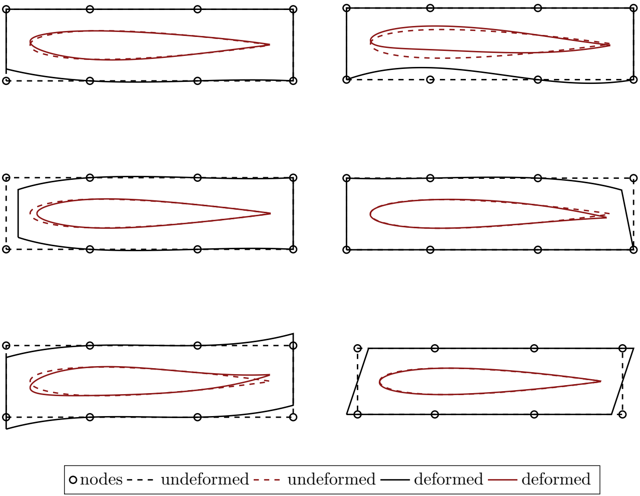
- Validating both Body-fitted and Embedded formulation

Simple NACA0012 profile

- $\alpha = 0.0°, 3.0, 6.0, 9.0$
- $M = 0.1,\ 0.3,\ 0.7,\ 0.9$
- Stiffened Gas



Shape-modification via design variables



Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Body-Fitted

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a body-fitted framework

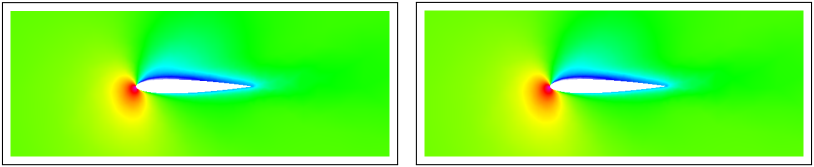


Figure: $\frac{\mathrm{d}\mathbf{w}_1}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$
Body-Fitted

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a body-fitted framework

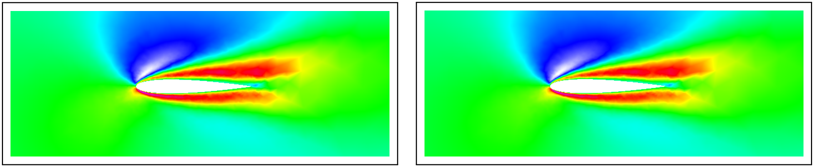


Figure: $\frac{\mathrm{d}\mathbf{w}_2}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Body-Fitted

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a body-fitted framework

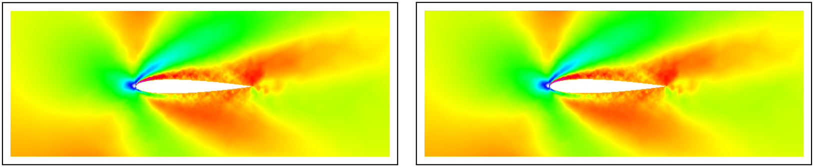


Figure: $\frac{\mathrm{d}\mathbf{w}_3}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

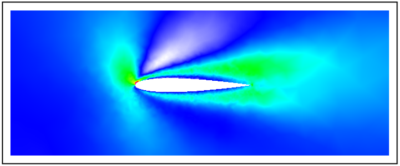
Body-Fitted

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a body-fitted framework

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Body-Fitted

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a body-fitted framework



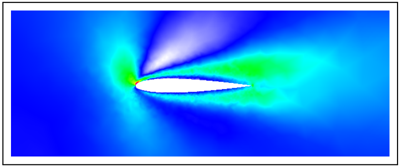


Figure: $\frac{\mathrm{d}\mathbf{w}_5}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$
Body-Fitted

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a body-fitted framework

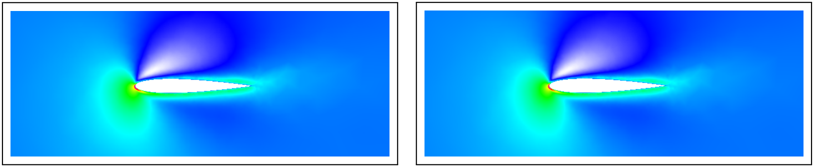
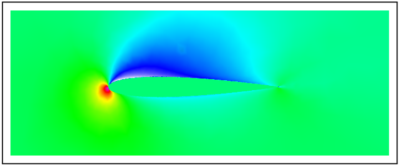


Figure: $\frac{\mathrm{d}\mathbf{w}_6}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Embedded

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a embedded framework



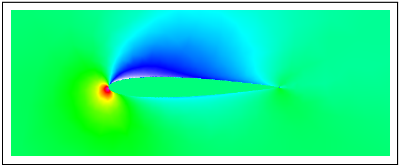


Figure: $\frac{\mathrm{d}\mathbf{w}_1}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Embedded

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a embedded framework

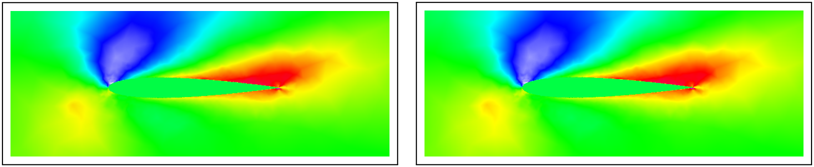


Figure: $\frac{\mathrm{d}\mathbf{w}_2}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Embedded

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a embedded framework

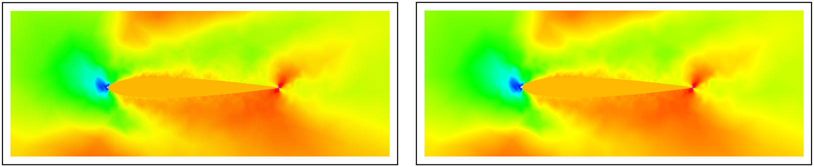


Figure: $\frac{\mathrm{d}\mathbf{w}_3}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

VERIFICATION OF $\frac{\partial \mathbf{w}}{\partial s}$

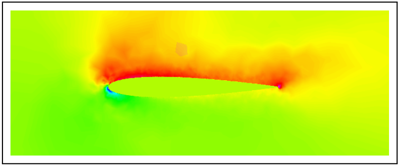
EMBEDDED

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a embedded framework

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Embedded

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a embedded framework



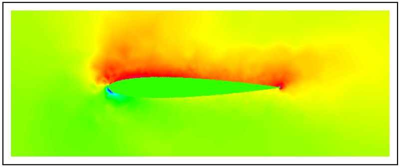
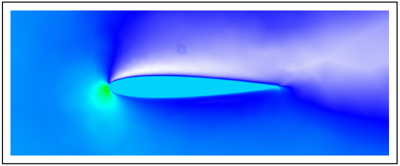


Figure: $\frac{\mathrm{d}\mathbf{w}_5}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Verification of $\frac{\partial \mathbf{w}}{\partial s}$

Embedded

Verification $\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}M_\infty}$ for RANS equations and a embedded framework



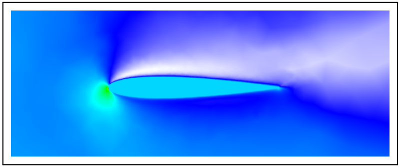


Figure: $\frac{\mathrm{d}\mathbf{w}_6}{\mathrm{d}M_\infty}$: FD-sensitivity on the left, analytic sensitivity

Convergence of force-sensitivties

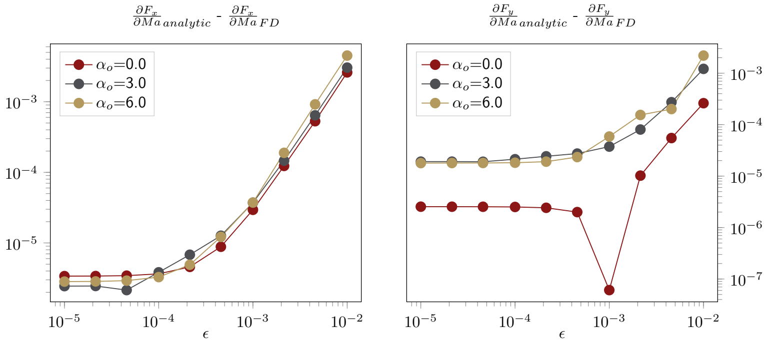


Figure: Convergence of the analytic results for Euler-equations

Convergence of force-sensitivties

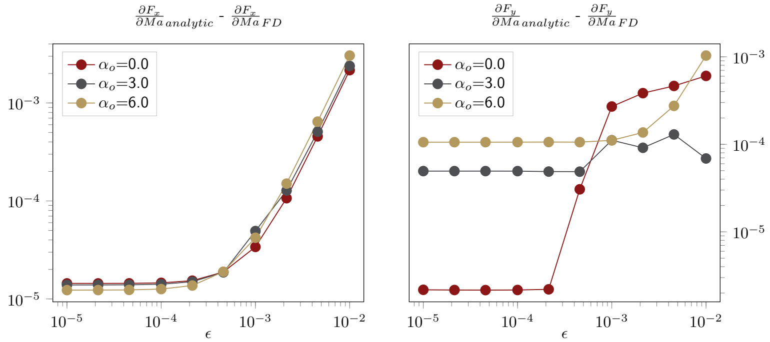


FIGURE: Convergence of the analytic results for Laminar-equations

Convergence of force-sensitivties

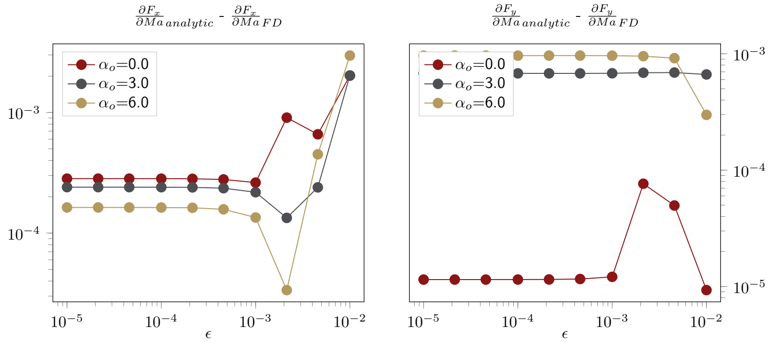


FIGURE: Convergence of the analytic results for RANS-equations

Outline for section 7

Multi-element airfoil with large kinematics

Prime example for embedded framework

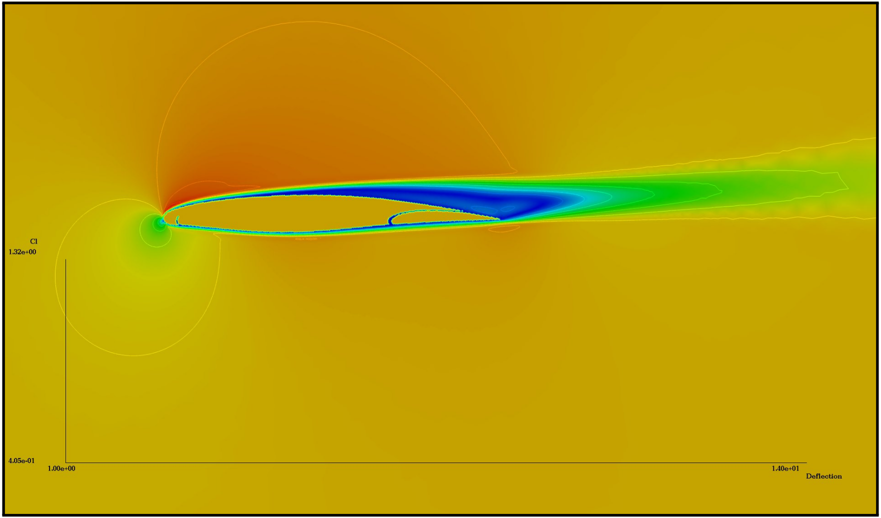


Figure: Optimization iteration 1

Multi-element airfoil with large kinematics

Prime example for embedded framework

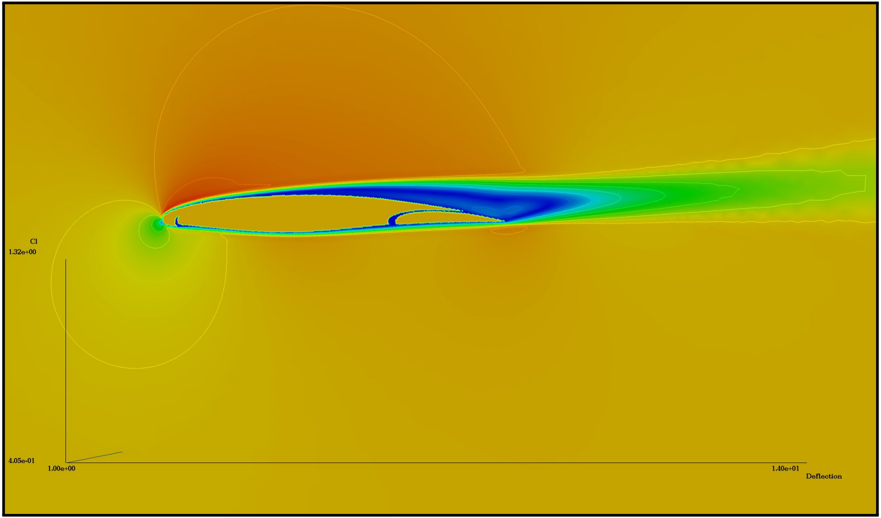


Figure: Optimization iteration 2

Multi-element airfoil with large kinematics

Prime example for embedded framework

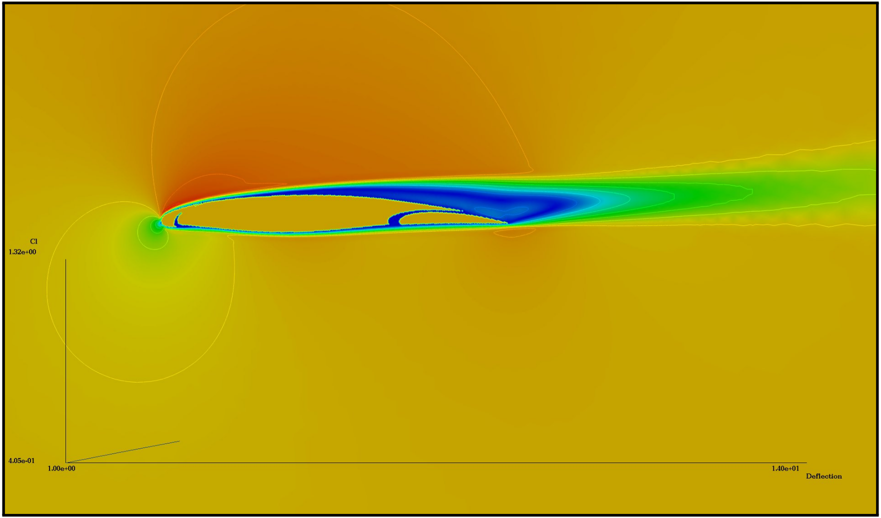


FIGURE: Optimization iteration 3

Multi-element airfoil with large kinematics

Prime example for embedded framework

Multi-element airfoil with large kinematics

Prime example for embedded framework

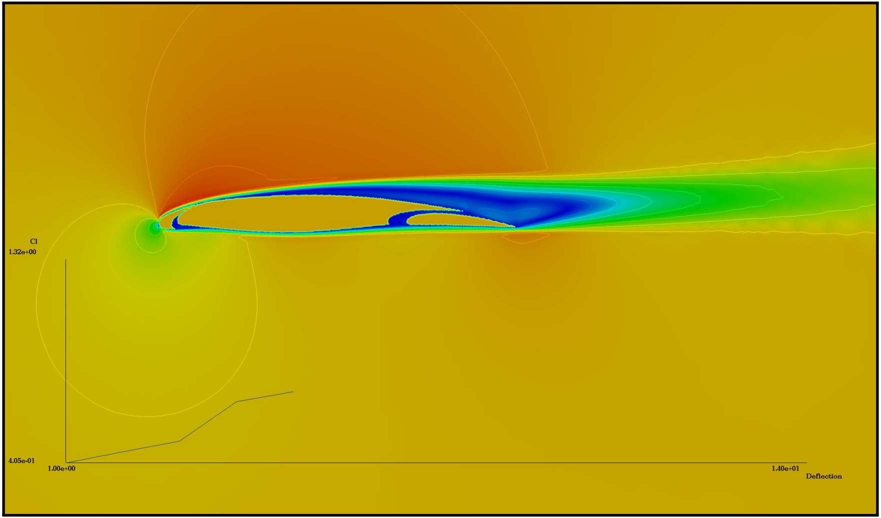


Figure: Optimization iteration 5

Multi-element airfoil with large kinematics

Prime example for embedded framework

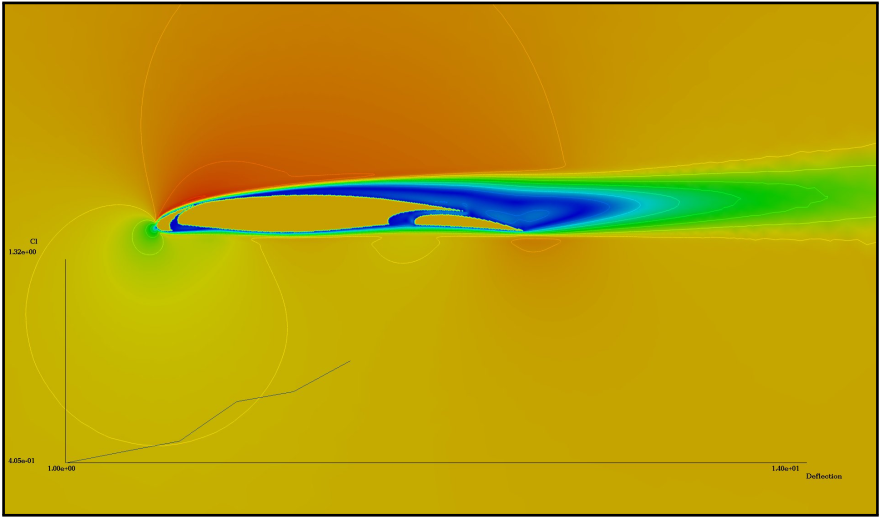


Figure: Optimization iteration 6

Multi-element airfoil with large kinematics

Prime example for embedded framework

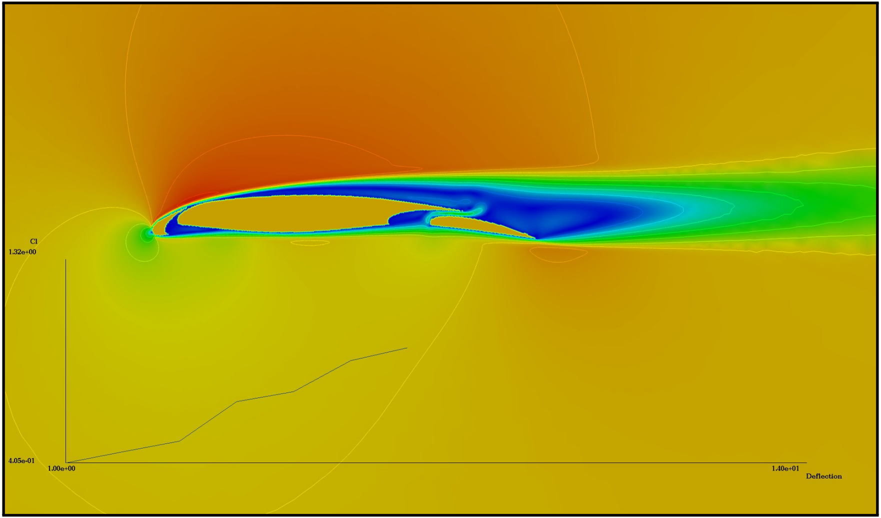


Figure: Optimization iteration 7

Multi-element airfoil with large kinematics

Prime example for embedded framework

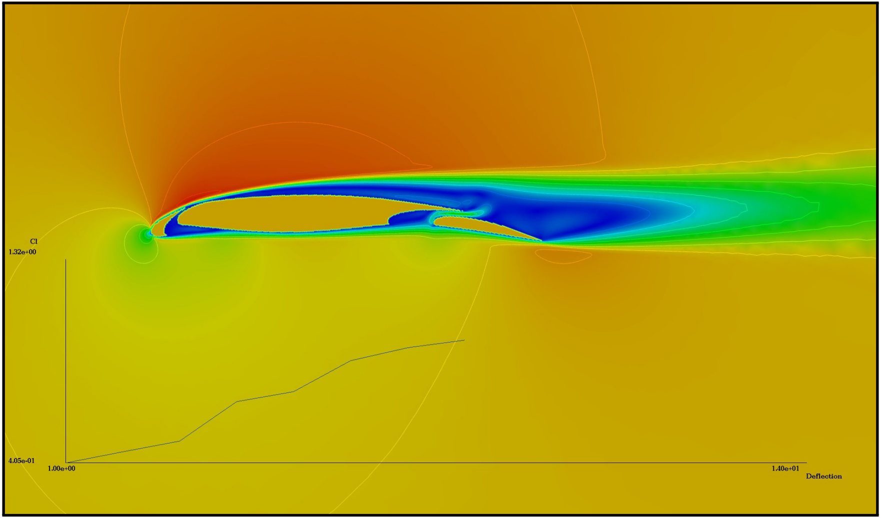


Figure: Optimization iteration 8

Multi-element airfoil with large kinematics

Prime example for embedded framework

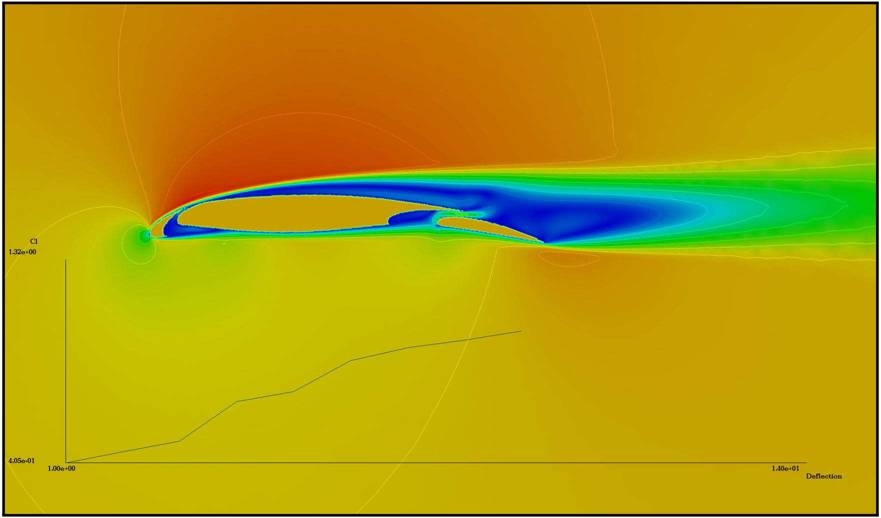


Figure: Optimization iteration 9

Multi-element airfoil with large kinematics

Prime example for embedded framework

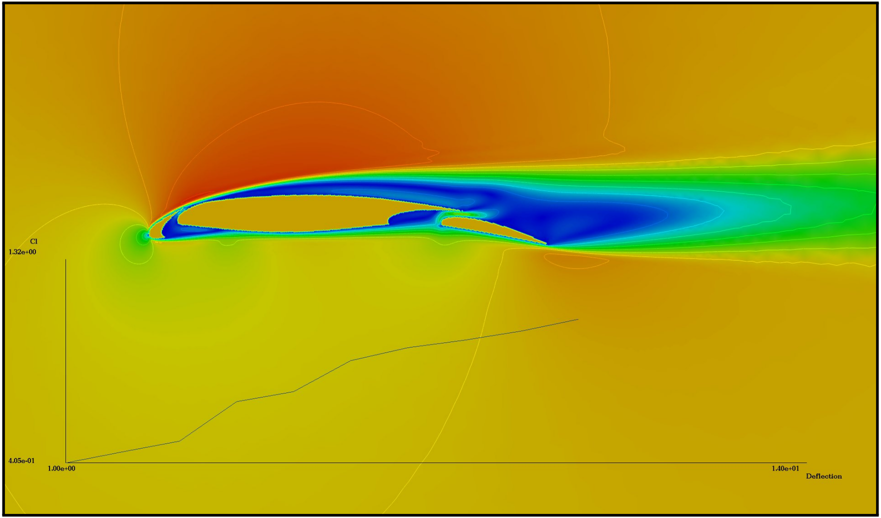


Figure: Optimization iteration 10

Multi-element airfoil with large kinematics
Prime example for embedded framework

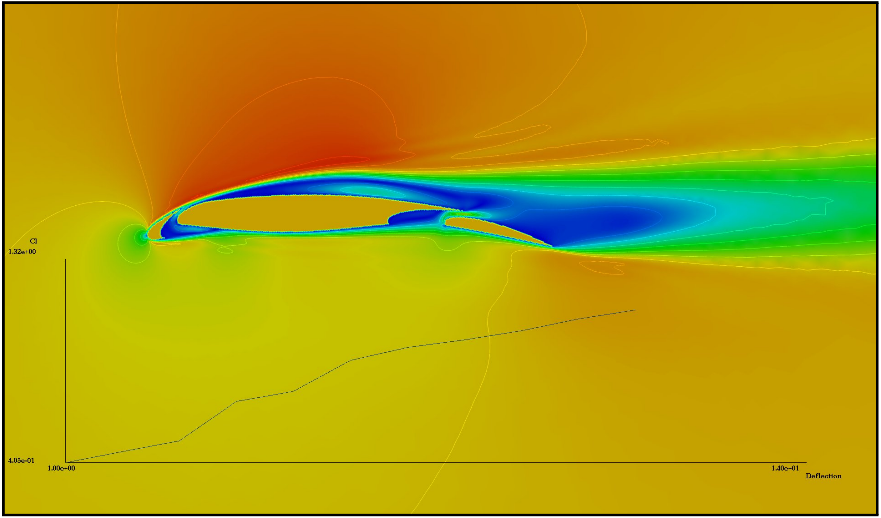


Figure: Optimization iteration 11

Multi-element airfoil with large kinematics

Prime example for embedded framework

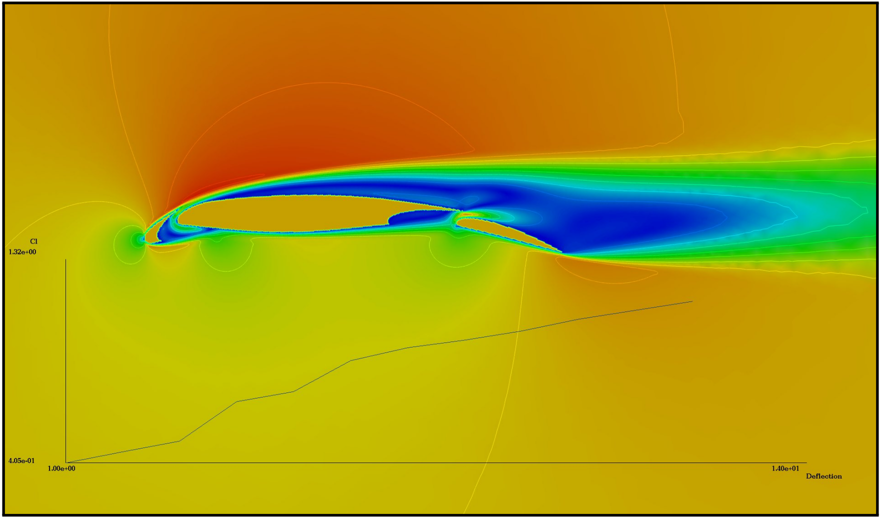


Figure: Optimization iteration 12

Multi-element airfoil with large kinematics

Prime example for embedded framework

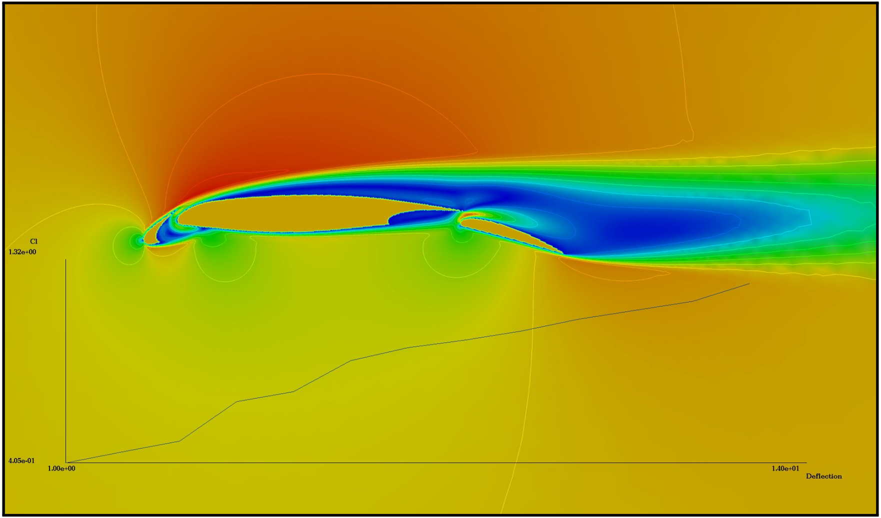


Figure: Optimization iteration 13

Multi-element airfoil with large kinematics

Prime example for embedded framework

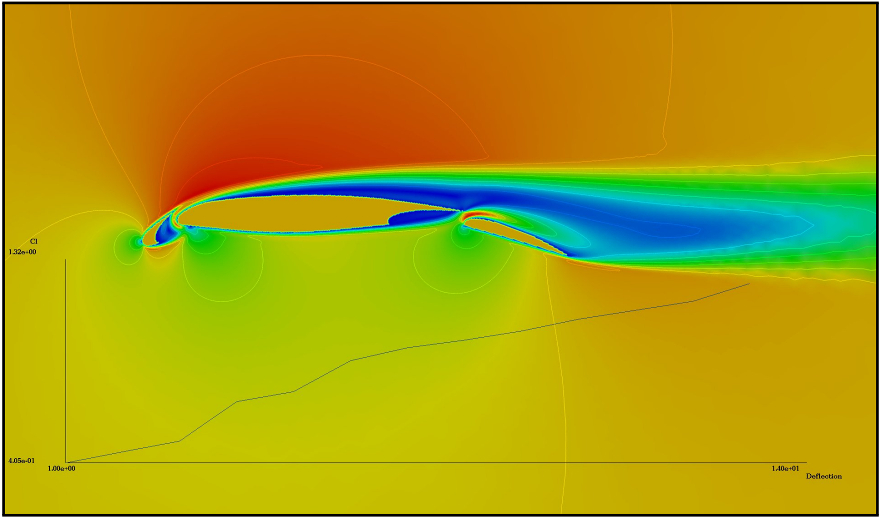
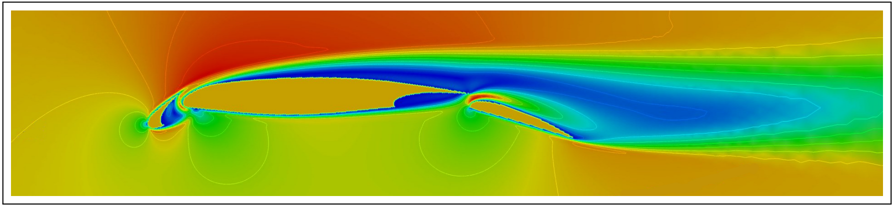


Figure: Optimization iteration 14

Multi-element airfoil with large kinematics

Prime example for embedded framework



- $M = 0.2$ and $\alpha = 10^\circ$
- Starting from closed configuration; let optimizer find the best relative positions of the airfoil elements
- 6 design variables: rotation, vertical and horizontal displacement of the elements
- Final value of the lift doubles after 6 optimization iterations

Main, Alexander. "Implicit and higer-order discretization methods for compressible multi-phase fluid and fluid-structure problems". PhD Thesis. Stanford University, 2014.

Roe, P. "Approximate Riemann solvers, parameter vectors, and difference schemes". In: *Journal of Computational Physics* 372 (1981), pp. 357–372. DOI: 10.1016/0021-9991(81)90128-5.