# Optimal Unsteady Flow Control and One-Shot Aerodynamic Shape Optimization using AD

Nicolas R. Gauger, Anil Nemili, Emre Özkaya

Computational Mathematics, CCES

RWTH Aachen University

(Recent) Work supported by

DFG Priority Program 1253 on Optimization with PDEs

DFG Project on Optimal Flow Control

BMBF Project DGHPOPT

# Ingredients

## for Efficient Optimization and Control in Aerodynamics

- Adjoint-based sensitivity evaluation (discrete, continuous, hybrid)
- Checkpointing for unsteady adjoint computation
- Non-loosely coupled (adjoint) PDEs for MDO
- One-shot methods (also called All-at-once, SAND, ...)
- Preconditioning of design equation
- Gradient smoothing
- Multilevel parameterization / Free node parameterization
- Shape derivatives and shape gradients
- Incorporation of adaptation by dual weighted residuals (DWR)
- Calculation of Pareto-fronts by “equality-constraint-based scans”
- …

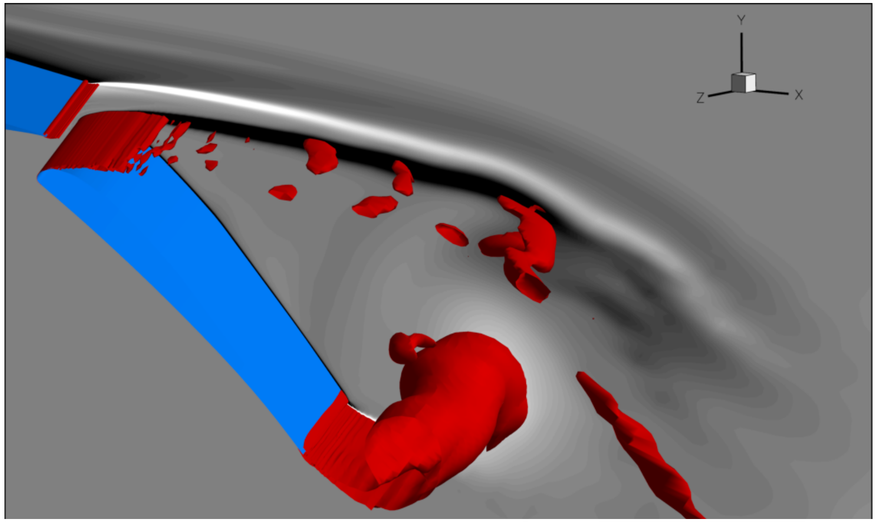
# Ingredients

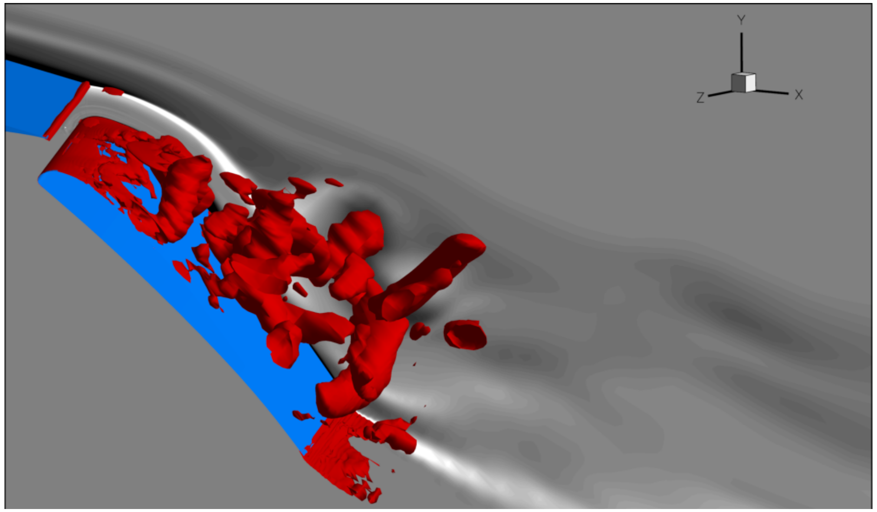
## for Efficient Optimization and Control in Aerodynamics

- Adjoint-based sensitivity evaluation (discrete, continuous, hybrid)
- Checkpointing for unsteady adjoint computation
- Non-loosely coupled (adjoint) PDEs for MDO
- One-shot methods (also called All-at-once, SAND, ...)
- Preconditioning of design equation
- Gradient smoothing
- Multilevel parameterization / Free node parameterization
- Shape derivatives and shape gradients
- Incorporation of adaptation by dual weighted residuals (DWR)
- Calculation of Pareto-fronts by “equality-constraint-based scans”
- …

# Optimal Active Flow Control - Motivation

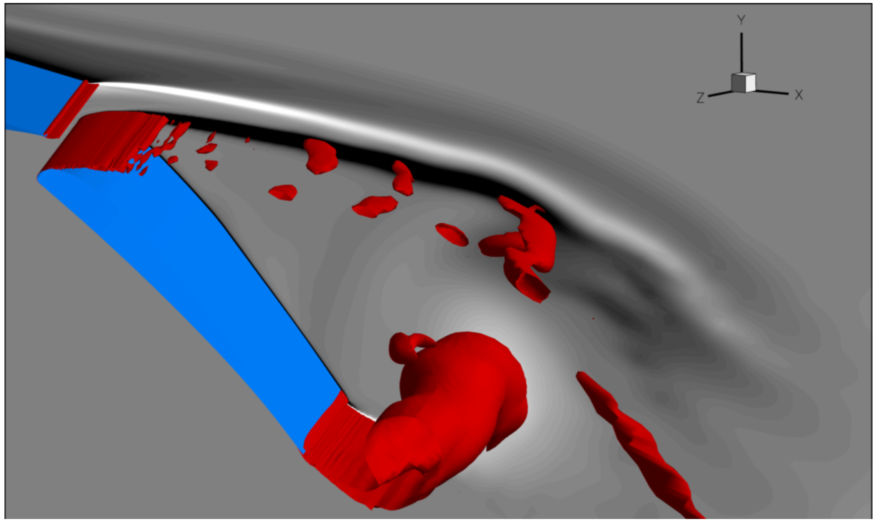
- ***Aerodynamic behaviour often influenced by separation***
  - *wings*
  - *vehicles*
- ***Active flow control is promising concept to manipulate separated flow***
  - *often realised by blowing and / or suction*
- ***How to choose effective excitation parameters?***
  - *often determined by open-loop in expensive experiments / numerical simulations*

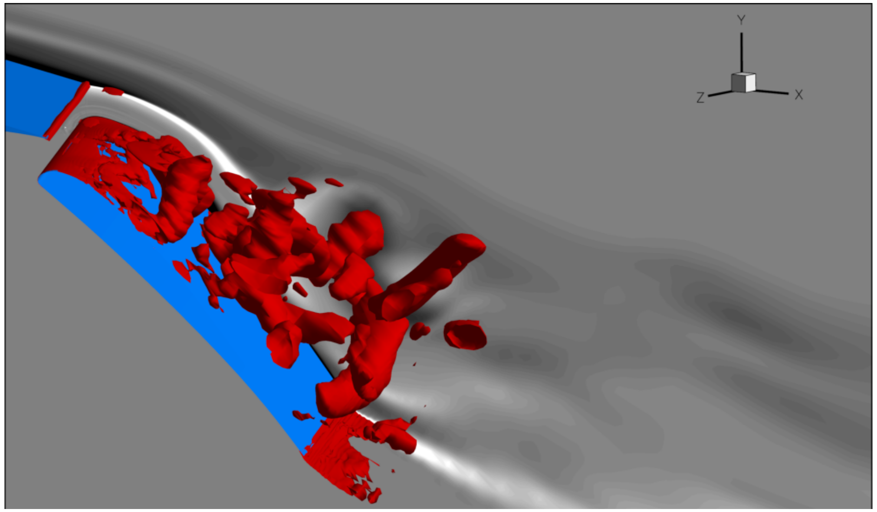




# Optimal Active Flow Control - Motivation

- *Aerodynamic behaviour often influenced by separation*
  - *wings*
  - *vehicles*
- *Active flow control is promising concept to manipulate separated flow*
  - *often realised by blowing and / or suction*
- *How to choose effective excitation parameters?*
  - *often determined by open-loop in expensive experiments / numerical simulations*





***⇒ Gradient based optimal control methods much more efficient!***

# Adjoints - Motivation

*Optimal control problem*

$$\min_{\phi \in \Phi} J(\phi, w) \quad s.t. \quad R(\phi, w) = 0$$

*Gradient-based optimal control, e.g.*

$$\phi_{k+1} = \phi_k - \alpha_k \nabla J\big|_{\phi_k}$$

*Calculation of the gradient*

$$\nabla J\big|_{\phi_k} = \left( \frac{dJ}{d\left(\phi_k\right)_n} \right)_{n=1,\ldots,M}$$

*Finite differences* | *Continuous adjoint* | *Discrete adjoint*

# Adjoints - Motivation

*Optimal control problem*

$$\min J(\phi, w) \quad s.t. \quad R(\phi, w) = 0$$

***Finite differences are inexact and too expensive to calculate***

***Gradients calculated by adjoints are accurate***

***Numerical effort for gradient computation by adjoints is independent from the number of design / control variables***

***⇒ Use of adjoints!***

*Finite differences* | *Continuous adjoint* | *Discrete adjoint*

# Different Adjoint Approaches

- ***Continuous Adjoint***

  ***- optimize then discretize***
  ***- hand coded adjoint solvers***
  ***- time consuming in implementation***
  ***- efficient in run and memory***

- ***Discrete Adjoint / Algorithmic Differentiation (AD)***

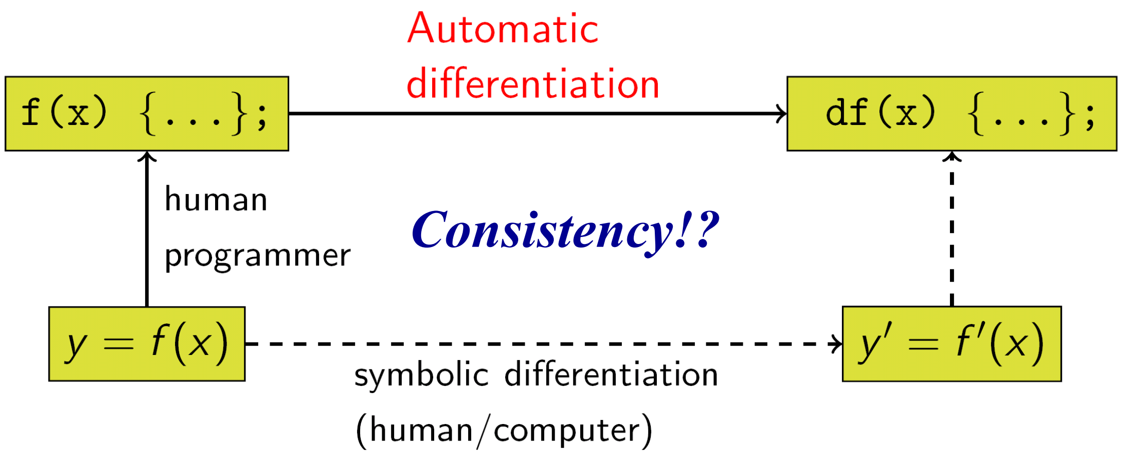
  ***- discretize then optimize***
  ***- hand coding of adjoint solvers or …***
  ***- … more or less automated generation***
  ***- memory effort increases (way out e.g. check-pointing)***

- ***Hybrid Adjoint***

  ***- merge "continuous and discrete" routines***
  ***- optimize differentiated code***

# Discrete Adjoint Method

- ***First discretise then optimise***
  - ***First discretise the primal system***
  - ***Then obtain the adjoint system based on the discretised primal equations***
- ***Possibility proposed here: Automatic or Algorithmic Differentiation (AD)***
- ***Basic principle***
  - ***Computer code is concatenation of basic operations (+,-,*,etc.)***
  - ***Apply differentiation rules to this concatenation by using chain rule***



# Governing Equations

***The incompressible RANS equations are given as***

$$\frac{\partial\left(\rho\bar{u}_i\right)}{\partial x_i}=0$$

$$\frac{\partial\left(\rho\bar{u}_i\right)}{\partial t}+\frac{\partial}{\partial x_i}\left(\rho\bar{u}_i\bar{u}_j+\rho\overline{u_i'u_j'}\right)=-\frac{\partial\bar{p}}{\partial x_i}+\frac{\partial\bar{\tau}_{ij}}{\partial x_i}$$

***The Reynolds stresses are modeled by the eddy viscosity model***

$$-\rho\overline{u_i'u_j'}=\mu_t\left(\frac{\partial\bar{u}_i}{\partial x_j}+\frac{\partial\bar{u}_j}{\partial x_i}\right)-\frac{2}{3}\rho\delta_{ij}k \quad \text{and} \quad \mu_t=0.31\rho k/\max\left(0.31\omega;\Omega F2\right)$$

***Two equation SST k-ω turbulence model***

$$\frac{\partial\left(\rho k\right)}{\partial t}+\frac{\partial\left(\rho k\bar{u}_i\right)}{\partial x_i}=\bar{\tau}_{ij}\frac{\partial\bar{u}_j}{\partial x_i}-\beta^*\rho\omega k+\frac{\partial}{\partial x_i}\left(\left(\mu+\sigma_k\mu_t\right)\frac{\partial k}{\partial x_i}\right)$$

$$\frac{\partial\left(\rho\omega\right)}{\partial t}+\frac{\partial\left(\rho\omega\bar{u}_i\right)}{\partial x_i}=\frac{\gamma}{\upsilon_t}\bar{\tau}_{ij}\frac{\partial\bar{u}_j}{\partial x_i}-\beta\rho\omega^2+\frac{\partial}{\partial x_i}\left(\left(\mu+\sigma_\omega\mu_t\right)\frac{\partial\omega}{\partial x_i}\right)+2\left(1-F_1\right)\rho\sigma_{w2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial\omega}{\partial x_i}$$

# Governing Equations

*The incompressible RANS equations are given as*

$$\frac{\partial\left(\rho\overline{u}_i\right)}{\partial x_i}=0$$

*The Re ... odel*

*Two eq...*

***But: ω → ∞ at wall!***

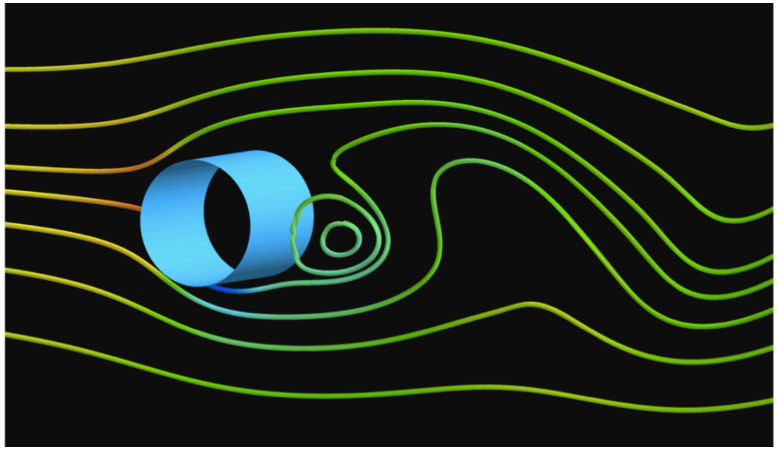
***⇒ SST k-ω turbulence model is non-differentiable!***

***⇒ Frozen turbulence assumption for continuous adjoint approach necessary!***

***⇒ Leads to extra inconsistency!***

# Test Case and Approaches

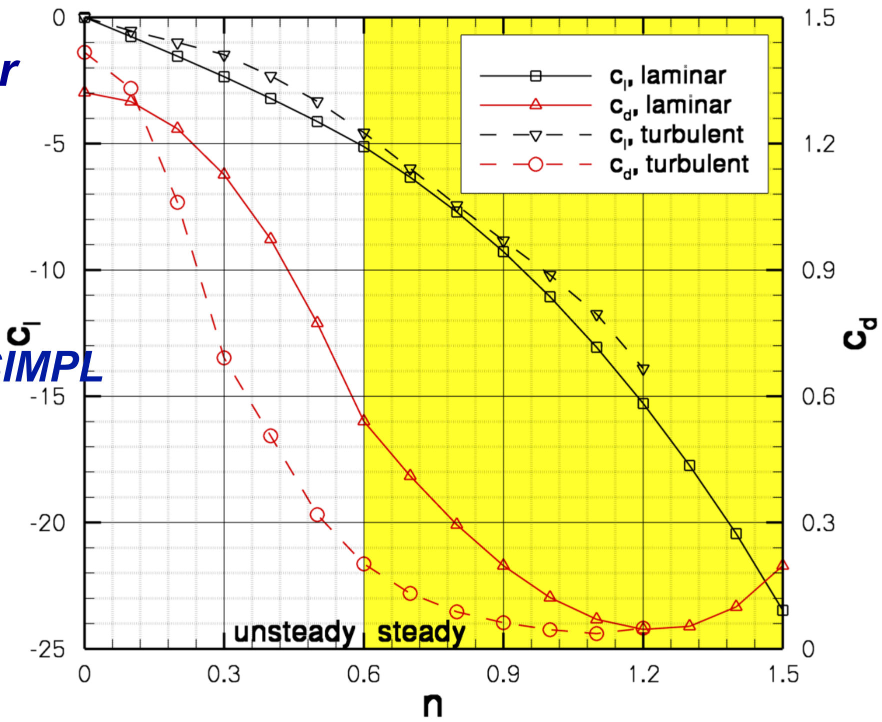
- *Drag minimisation of a rotating cylinder*
- *Control variable: Rotational speed*
- *RANS flow solver: ELAN (TU Berlin)*
  - *Block-structured, FVM, incompressible, SIMPLE*
  - *Fully implicit, MPI based parallelisation*
  - *Turbulence model : SST k-ω*
  - *Coded in Fortran*



- *AD tool for adjoint : TAPENADE (INRIA Sophia – Antipolis)*
- *Reverse Accumulation for SIMPL loops* [Christianson]
- *Checkpointing by REVOLVE* [Griewank, Walther]
  - *Usable for Fortran and C*

# Test Case and Approaches

- ***Drag minimisation of a rotating cylinder***
- ***Control variable: Rotational speed***
- ***RANS flow solver: ELAN (TU Berlin)***
  - ***Block-structured, FVM, incompressible, SIMPL***
  - ***Fully implicit, MPI based parallelisation***
  - ***Turbulence model : SST k-ω***
  - ***Coded in Fortran***

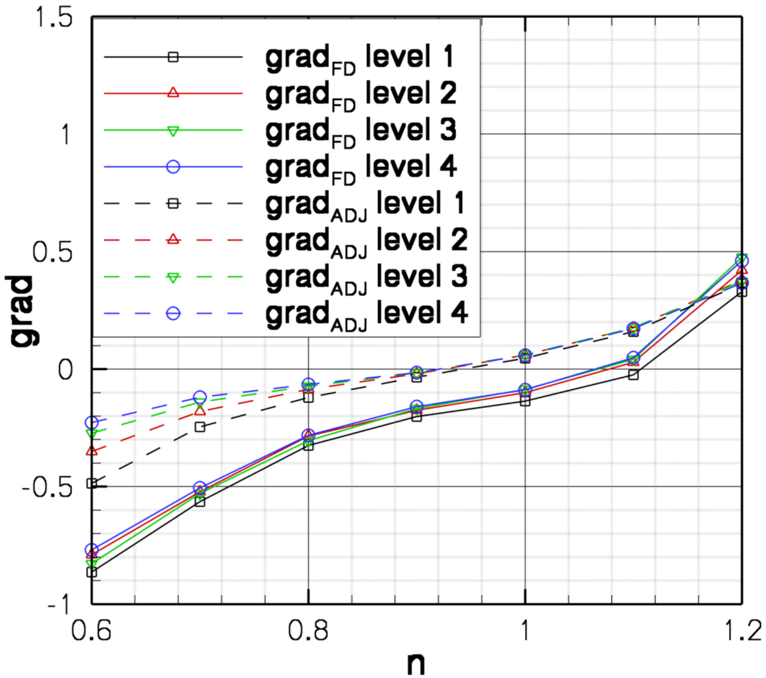



- ***AD tool for adjoint : TAPENADE (INRIA Sophia – Antipolis)***
- ***Reverse Accumulation for SIMPL loops*** **[Christianson]**
- ***Checkpointing by REVOLVE*** **[Griewank, Walther]**
  - ***Usable for Fortran and C***

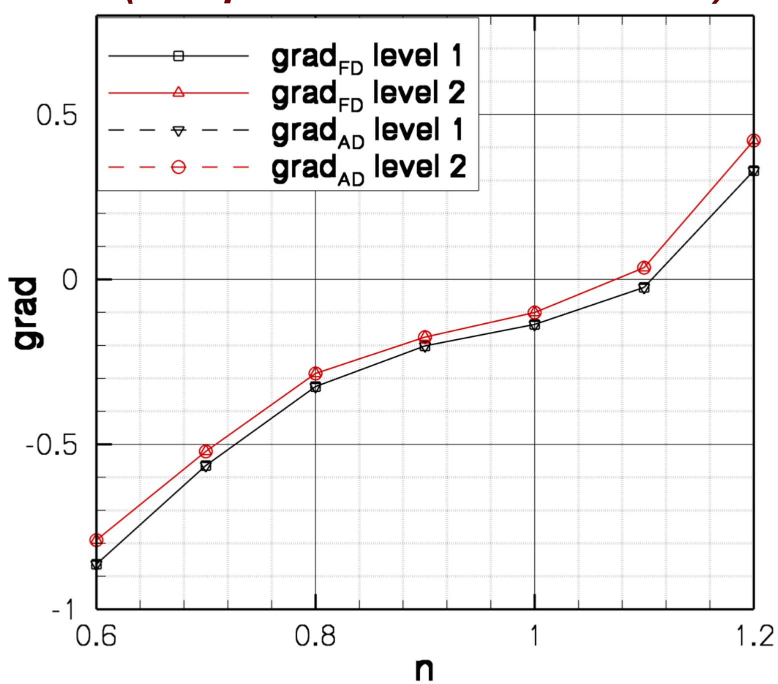
# Validation of the discrete adjoint solver

- *Pre-study: Continuous vs Discrete adjoints*
- *Steady turbulent flow with Re = 5000*

*Continous approach vs FD (frozen eddy viscosity)*




*Discrete approach vs FD (complete turbulence model)*



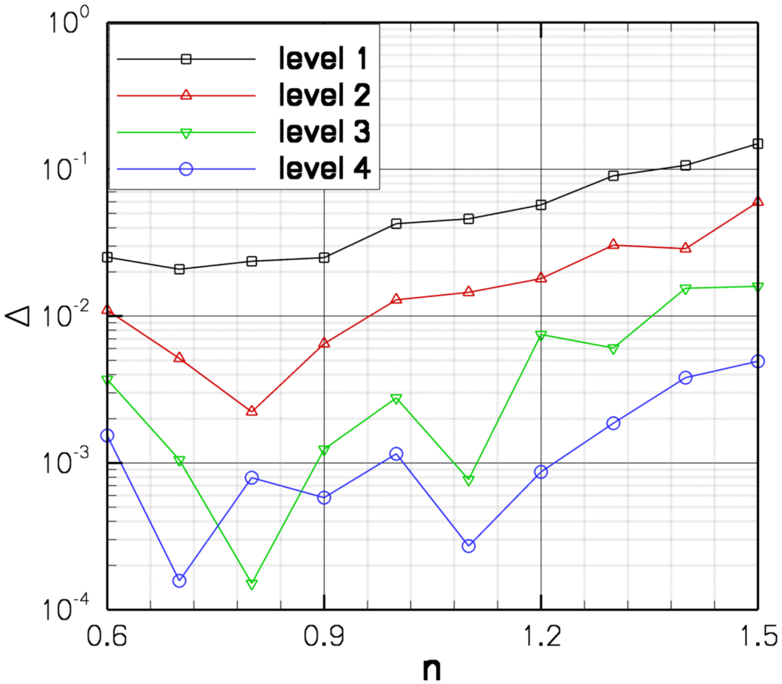

*Discrete adjoints : Grid independent, more accurate and consistent*

*[Carnarius, Thiele, Özkaya, Gauger, AIAA-2010-5088-435, 2010]*

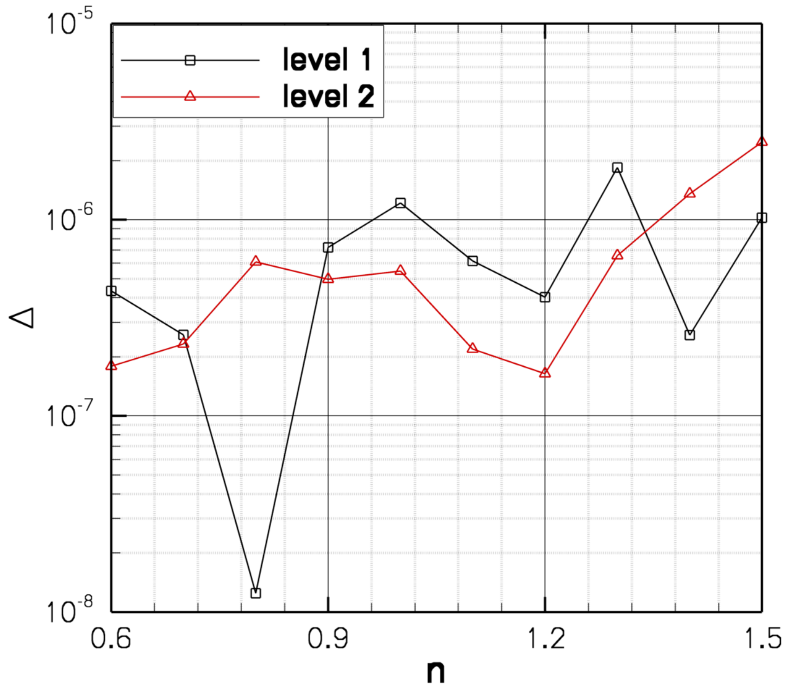
# Validation of the discrete adjoint solver

- ***Pre-study: Continuous vs Discrete adjoints***
- ***Steady laminar flow with Re = 100***

***Continous approach vs FD***




***Discrete approach vs FD***




***Discrete adjoints : Grid independent, more accurate and consistent***

***[Carnarius, Thiele, Özkaya, Gauger, AIAA-2010-5088-435, 2010]***

# Development of discrete adjoint methods for unsteady optimal flow control

- Unsteady adjoint-based optimal flow control is very challenging
- Up to now not used for complex practical aerodynamic applications
- One of the main reasons is the prohibitive storage requirement as one has to store the entire flow history
- For example, the storage cost of the primal solution for a 2D URANS incompressible solver with $10^5$ grid points and 1000 unsteady time iterations is O(10) Gb
- Obviously, for practical complex configurations in 3D with more number of grid points and time iterations, the storage requirements may become prohibitively large

# Exact and approximate approaches

**Several strategies have been proposed to circumvent the memory requirements**

***Exact methods***

- ***Store all in hard disk approach***
- ***Use of checkpoints***
- ***…***

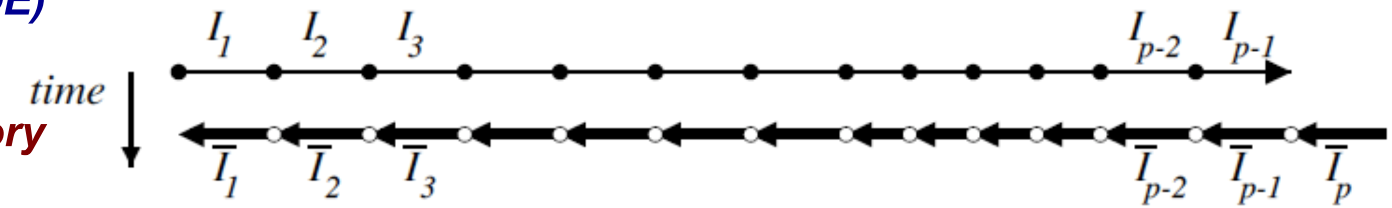
***Approximate methods***

- ***Reduced order methods (e.g. POD models)***
- ***Non-linear frequency domain methods***
- ***…***

# Checkpointing

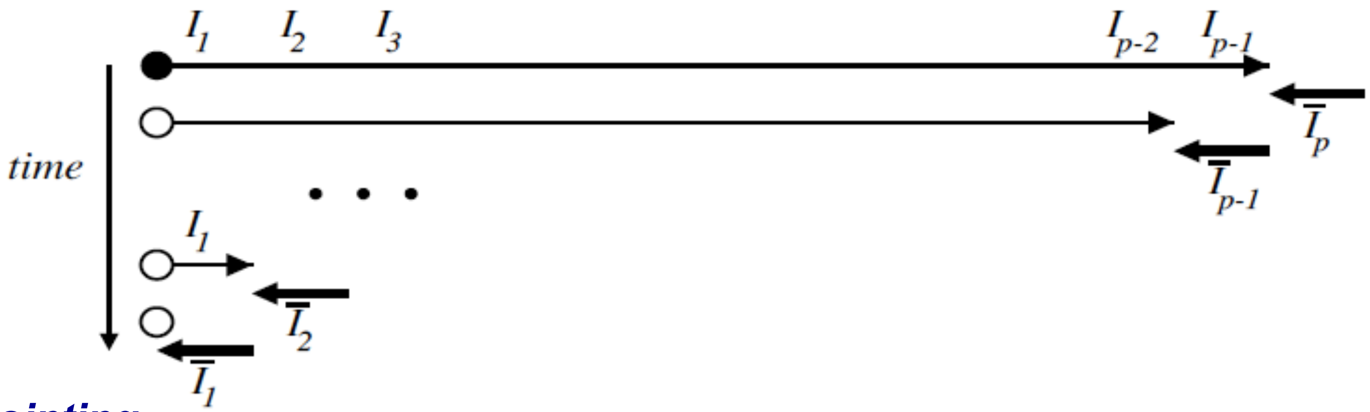
***Store-All (TAPENADE)***

***Excessive memory requirements***

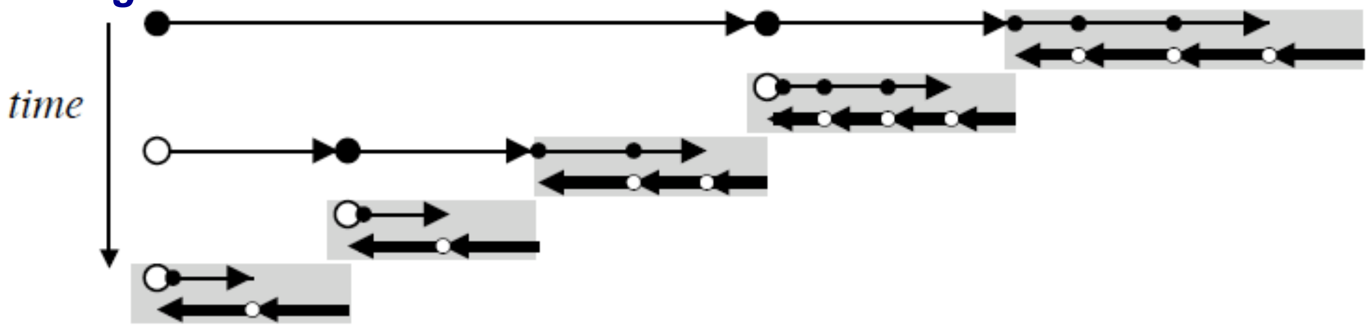



***Recompute-All (TAF)***

***Excessive CPU requirements***



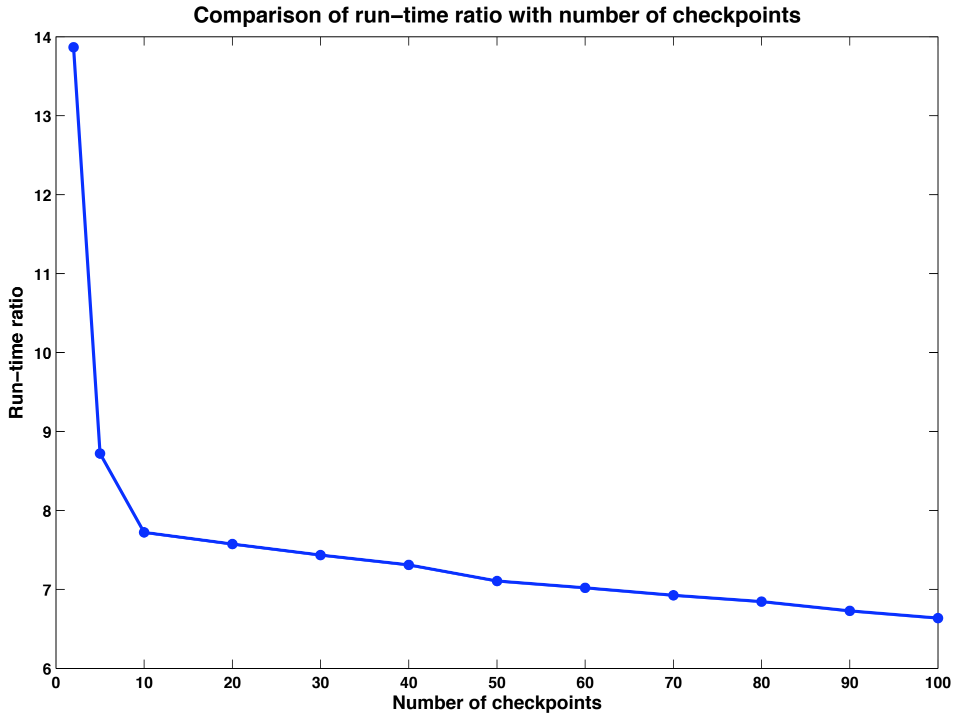

***Compromise: Checkpointing***



# Variation of run time ratio with checkpoints

*Unsteady time steps = 100*

*Run time ratio =* $\dfrac{T(adjoint)}{T(primal)}$




*Comments on checkpointing approach :*

- *Significantly reduces the memory*
- *Increases the run time due to extra flow calculations*
- *Binomial checkpointing ensures optimal extra flow computations*

# Validation of the discrete adjoint solver

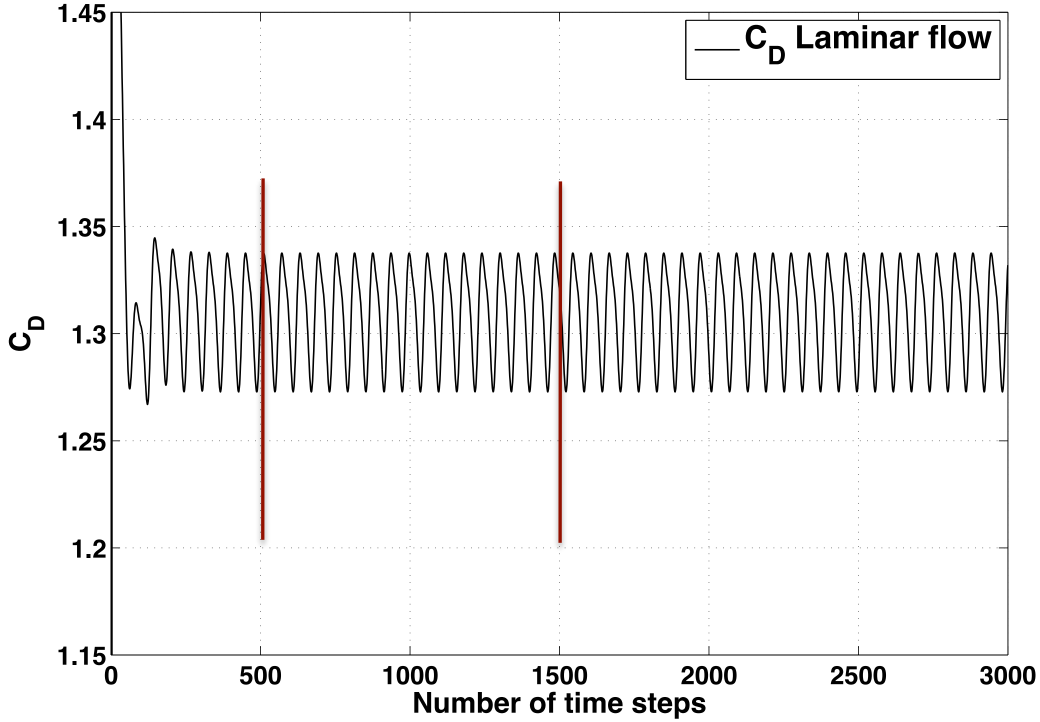
**Laminar flow, *Re* = 100**

***Cost function***

$$J = \bar{C}_D = \frac{1}{N - N'} \sum_{n=N'+1}^{N} C_D^n$$

$$N = 1500, \quad N' = 500$$

$$\Delta t = 0.1, \quad \omega = 0.1$$




***Number of Checkpoints = 150***

***The run time cost for recomputations is a factor of 1.8987***

***Sensitivity gradients:***

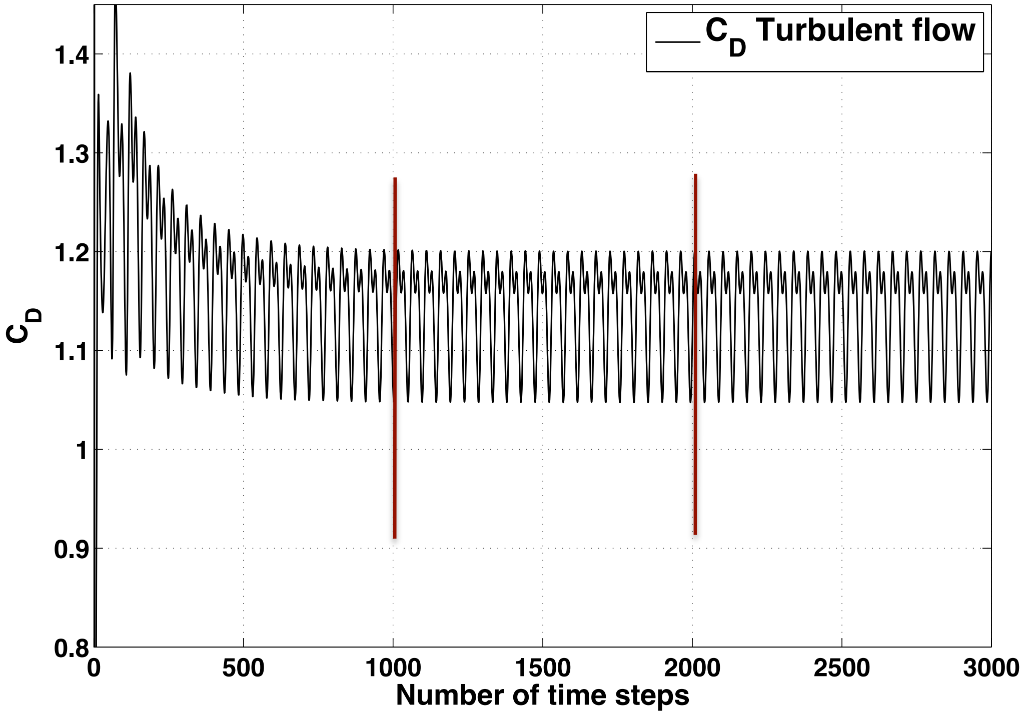
| 2nd order FDM : $\Delta\omega = 10^{-4}$ | Adjoint mode AD code | Forward mode AD code |
|---|---|---|
| -0.430024408 | -0.429528263715753 | -0.429528222994623 |

***Difference:*** $O(10^{-4})$ $O(10^{-8})$

[Nemili, Özkaya, Gauger, 2010]

# Validation of the discrete adjoint solver

## Turbulent flow, *Re* = 5000




***Cost function***

$$J = \bar{C}_D = \frac{1}{N - N'} \sum_{n=N'+1}^{N} C_D^n$$

$$N = 2000, \quad N' = 1000$$

$$\Delta t = 0.1, \quad \omega = 0.1$$

***Number of Checkpoints = 150***

***The run time cost for recomputations is a factor of 1.9240***

***Sensitivity gradients:***

| 2nd order FDM : $\Delta\omega = 10^{-4}$ | Adjoint mode AD code | Forward mode AD code |
|---|---|---|
| -0.56466950 | -0.564139519316557 | -0.564124863073591 |

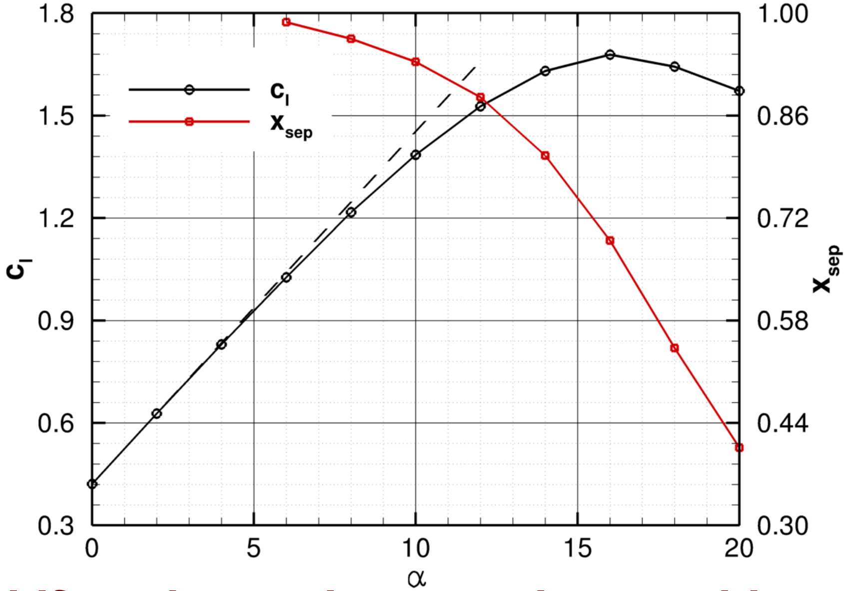
***Difference:*** $O(10^{-4})$ $O(10^{-5})$

[Nemili, Özkaya, Gauger, 2010]

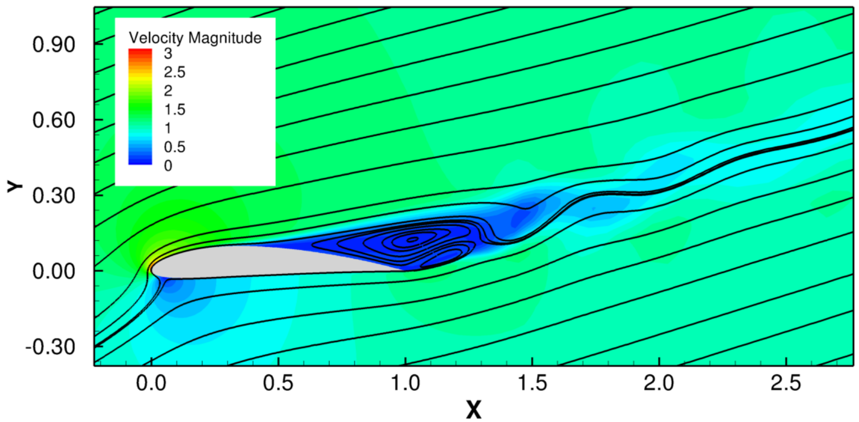
# NACA4412 Airfoil Test Case

## Turbulent flow, $Re=10^6$, $\alpha=20^o$

### Primal flow solutions:




Lift polar and separation position




Contours of velocity magnitude

# NACA4412 Airfoil Test Case

## Turbulent flow, $Re=10^6$, $\alpha=20^o$

### Lift Maximisation using active flow control:

- A cascade of 4 actuation slots is installed on the suction side
- Sinusoidal blowing and suction with zero net mass flux is applied at each slot using

$$\begin{pmatrix} u^k \\ v^k \end{pmatrix} = A^k \begin{pmatrix} \cos\beta^k \\ \sin\beta^k \end{pmatrix} \sin\left[2\pi f\left(t - t_0^k\right)\right], \quad k = 1,\ldots,4.$$

# NACA4412 Airfoil Test Case

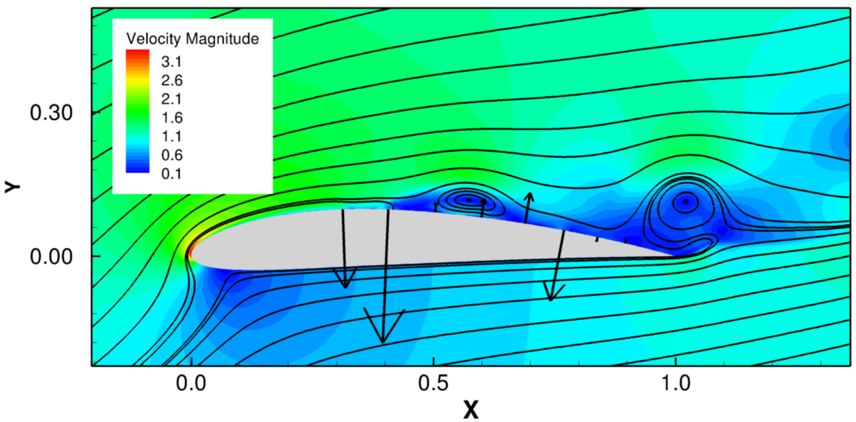
## Turbulent flow, Re=$10^6$, α=$20^o$

### Lift Maximisation using active flow control:

- A cascade of 4 actuation slots is installed on the suction side
- Sinusoidal blowing and suction with zero net mass flux is applied at each slot using

$$\begin{pmatrix} u^k \\ v^k \end{pmatrix} = A^k \begin{pmatrix} \cos\beta^k \\ \sin\beta^k \end{pmatrix} \sin\left[2\pi f\left(t - t_0^k\right)\right], \quad k = 1,\dots,4.$$

- Actuated flow from preliminary simulations



# NACA4412 Airfoil Test Case

## Turbulent flow, Re=$10^6$, α=$20^o$

### Optimal active flow control:

**Cost function:** Mean lift coefficient

$$J = \bar{C}_l = \frac{1}{N}\sum_{n=1}^{N} C_l^n \qquad N = 100,\ \Delta t = 0.005$$

**Control variables:** $A^k$ – Amplitude, $t_0^k$ – Phase shift, $\beta^k$ – Angle ($k = 1$ to 4)

**Turbulence model:** Wilcox k-ω

# NACA4412 Airfoil Test Case

## Turbulent flow, $Re=10^6$, $\alpha=20^o$

### Optimal active flow control:

| Control Parameter | Adjoint mode AD code | Forward mode AD code |
|---|---|---|
| Amplitude - 01 | 0.132869414677547 | 0.132843488475446 |
| Amplitude - 02 | 0.167070460770784 | 0.167065662623720 |
| Amplitude - 03 | 0.181247271988999 | 0.181252126166289 |
| Amplitude - 04 | 0.155844489164031 | 0.155843813170431 |

*Comparison of the sensitivity gradients with respect to Amplitude at 1st, 2nd, 3rd and 4th actuation slots*

[Nemili, Özkaya, Gauger, 2011]

# *NACA4412 Airfoil Test Case*

*Turbulent flow, Re=$10^6$, α=$20^o$*

## *Optimal active flow control:*

| Control Parameter | Adjoint mode AD code | Forward mode AD code |
|---|---|---|
| Phase shift - 01 | 0.204246051913213 | 0.204178681978258 |
| Phase shift - 02 | 0.244791572866917 | 0.244693324906123 |
| Phase shift - 03 | 0.244817849004966 | 0.244819168327026 |
| Phase shift - 04 | 0.125957535080716 | 0.125955476539906 |

*Comparison of the sensitivity gradients with respect to Phase shift at $1^{st}$, $2^{nd}$, $3^{rd}$ and $4^{th}$ actuation slots*

[Nemili, Özkaya, Gauger, 2011]

# *NACA4412 Airfoil Test Case*

***Turbulent flow, Re=$10^6$, α=$20^o$***

## *Optimal active flow control:*

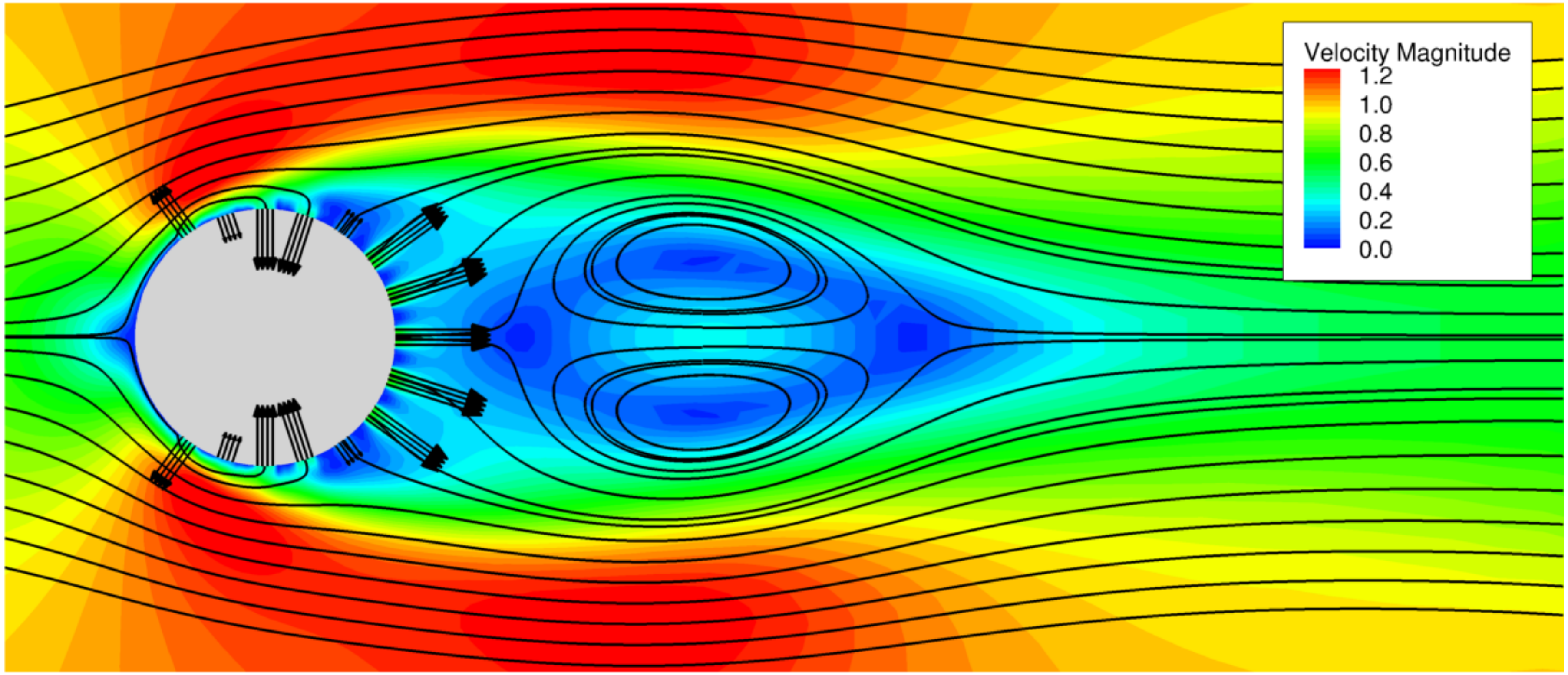
| Control Parameter | Adjoint mode AD code | Forward mode AD code |
|---|---|---|
| Angle - 01 | -0.005212791392634 | -0.005209720130677 |
| Angle - 02 | -0.006697219503597 | -0.006705398122871 |
| Angle - 03 | -0.006841492789784 | -0.006841527973356 |
| Angle - 04 | -0.007246751418587 | -0.007246750396135 |

***Comparison of the sensitivity gradients with respect to Angle at $1^{st}$, $2^{nd}$, $3^{rd}$ and $4^{th}$ actuation slots***

[Nemili, Özkaya, Gauger, 2011]

# Optimal Control of Cylinder Flow

- Flow around cylinder at Re=100
  - 15 actuation slots for pulsed blowing/suction
  - Actuation parameters: Amplitude at each slot
  - Goal: Minimize drag

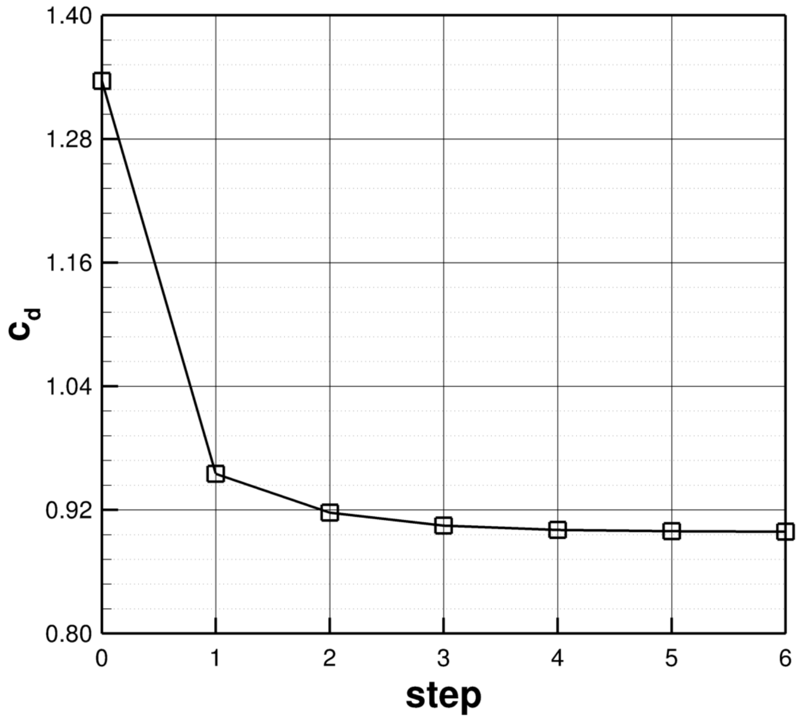


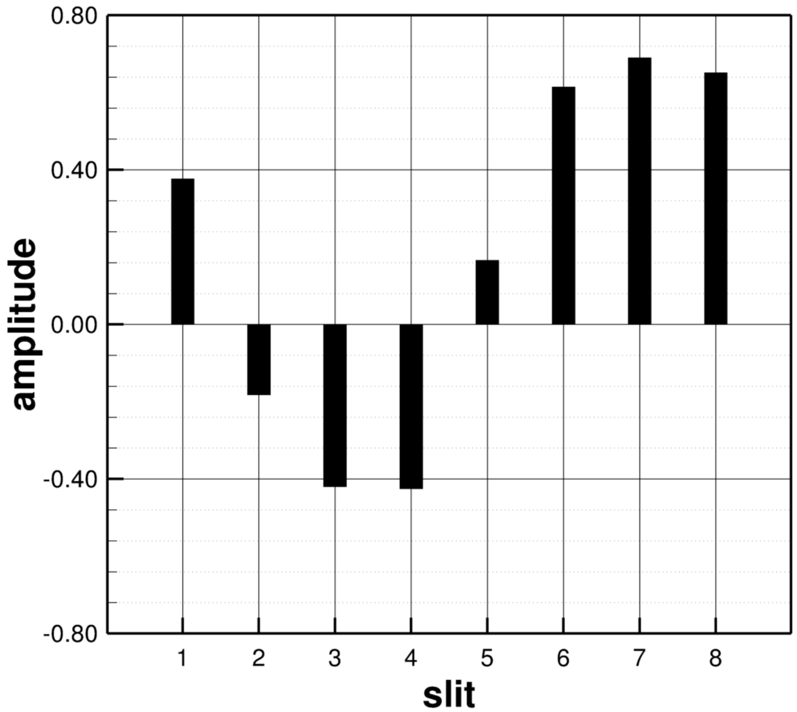

[Carnarius, Nemili, Özkaya, Gauger, Thiele, 2012]

# Optimal Control of Cylinder Flow

- Optimal distribution of amplitudes
  - Blowing/suction on upper/lower side generates strong vortices (slit 1-4)
  - Blowing at rear pushes vortices away, increasing base pressure (slit 5-8)

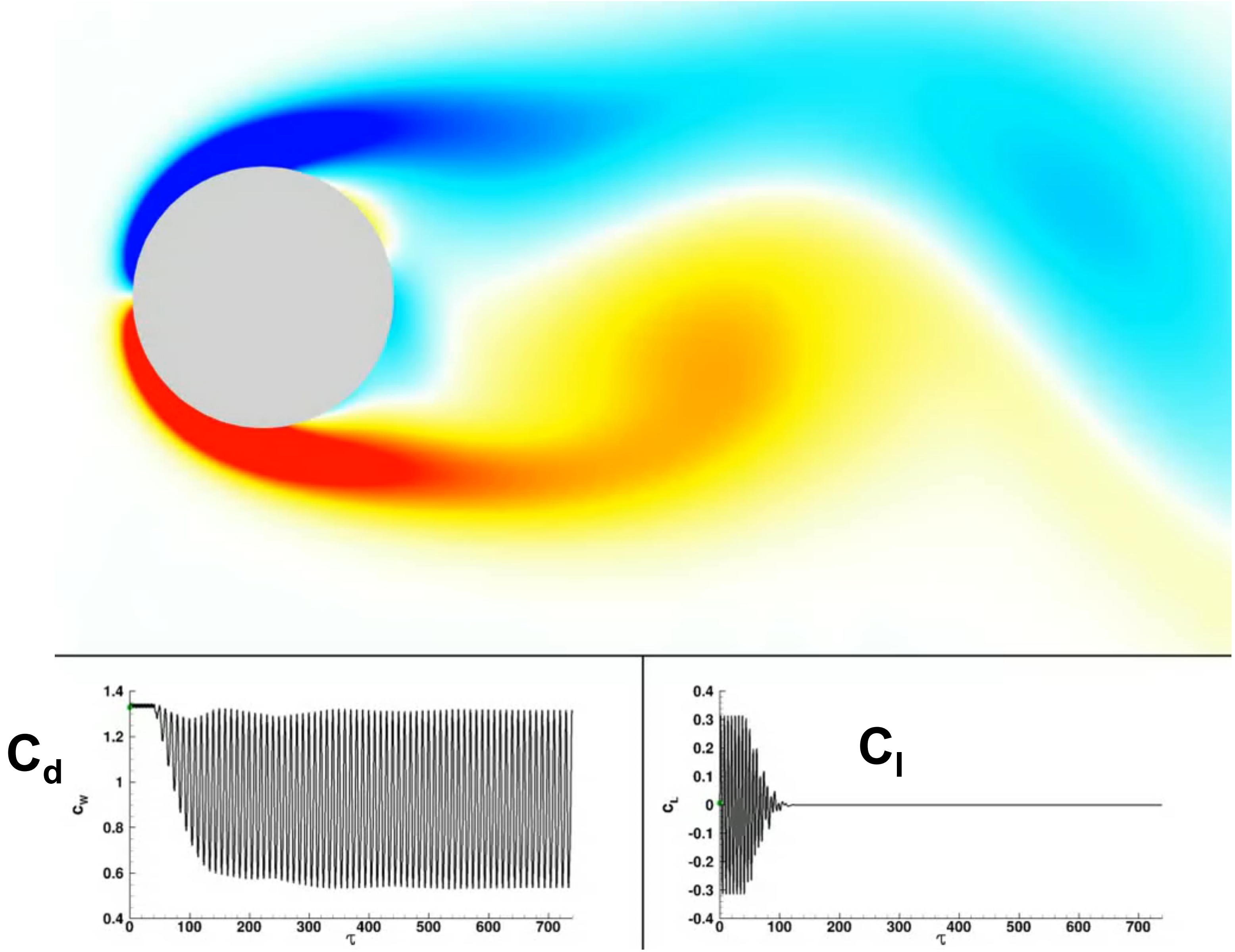
    → Drag is reduced from 1.34 to 0.90




[Carnarius, Nemili, Özkaya, Gauger, Thiele, 2012]

# Optimal Control of Cylinder Flow

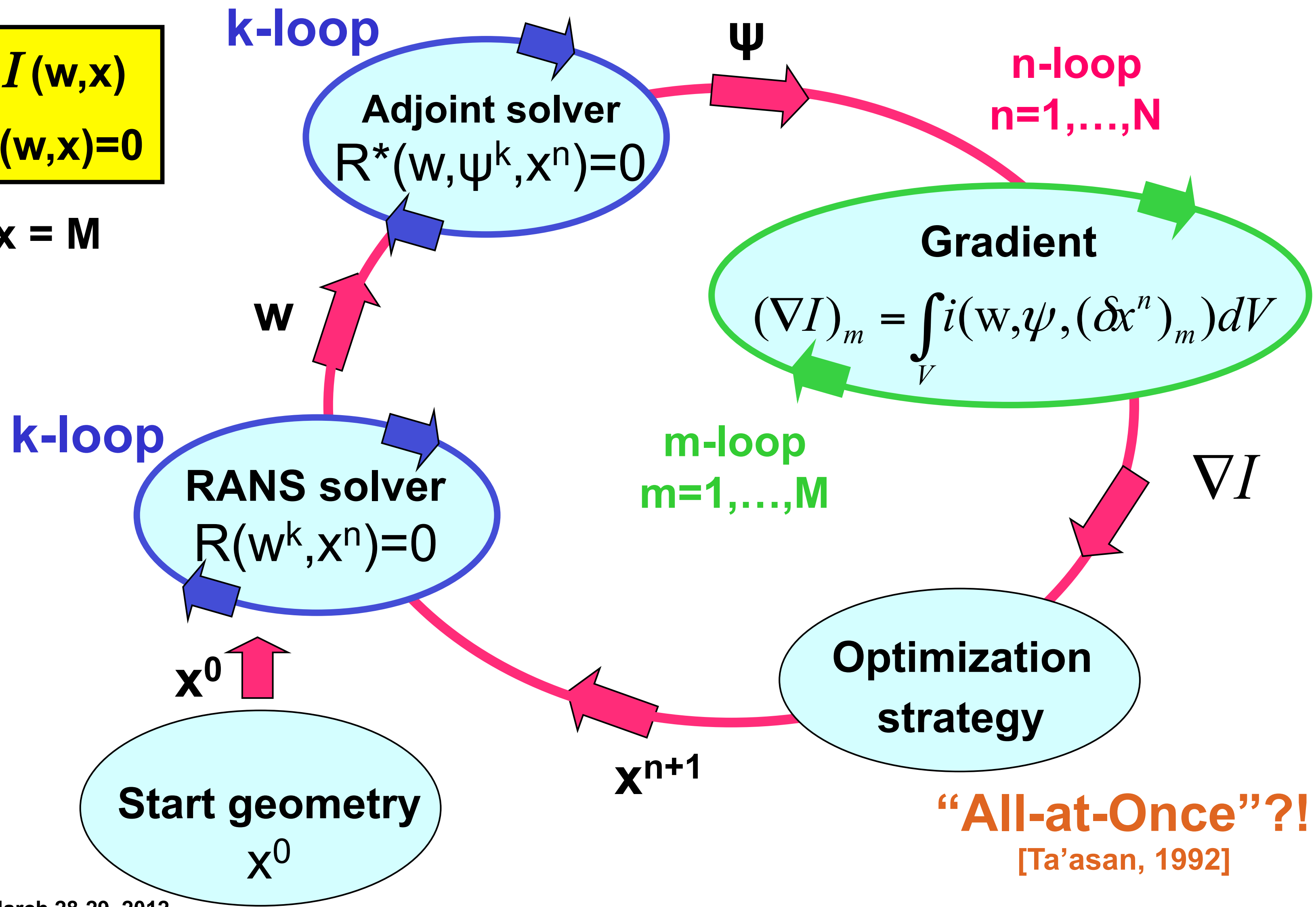



[Carnarius, Nemili, Özkaya, Gauger, Thiele, 2012]

# Nested approach

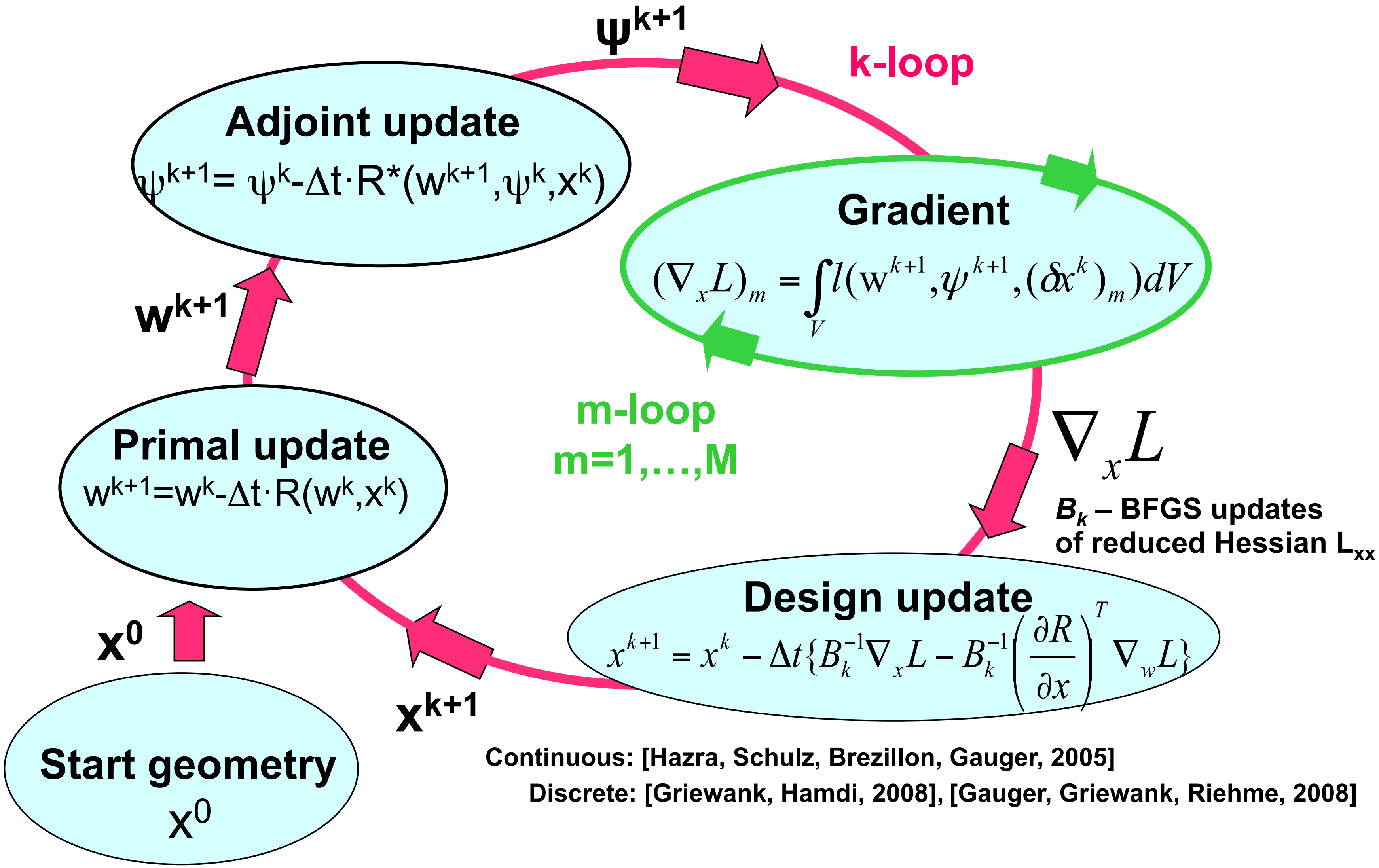
$\min I(w,x)$
s.t. $R(w,x)=0$

dim x = M




"All-at-Once"?!
[Ta'asan, 1992]

# One-Shot approach




Continuous: [Hazra, Schulz, Brezillon, Gauger, 2005]

Discrete: [Griewank, Hamdi, 2008], [Gauger, Griewank, Riehme, 2008]

# Problem Statement

**Goal:** $\min_u f(y,u)$ **s.t.** $c(y,u)=0,$

**where $y$ and $u$ are the state and design variables.**

**Given fixed point iteration $y_{k+1}=G(y_k,u)$ (e.g. pseudo-time stepping) to solve PDE $c(y,u)=0.$**

**Assumptions:**

- $\frac{\partial c}{\partial y}$ **always invertible. IFT $\Rightarrow$ given $u, \exists!\, y$ s.t. $c(y,u)=0.$**
- $G, f \in C^{2,1}.$
- $G$ **contractive:** $\left\|G_y(y,u)\right\| = \left\|G_y^T(y,u)\right\| \leq \rho < 1$

# One-Shot approach

$$L(y,\bar{y},u) = f(y,u) + (G(y,u) - y)^T \bar{y}$$
$$= \underbrace{N(y,\bar{y},u)}_{\text{shifted Lagragian}} - y^T \bar{y}$$

**Stationary point:**
$$\begin{cases} L_{\bar{y}} = G(y,u) - y = 0 \\ L_y = N_y(y,\bar{y},u)^T - \bar{y} = 0 \\ L_u = N_u(y,\bar{y},u)^T = 0 \end{cases}$$

**One-step one-shot** (step *k+1*):

$$(OS)\begin{cases} y_{k+1} = G(y_k, u_k) & \text{primal update} \\ \bar{y}_{k+1} = N_y(y_k, \bar{y}_k, u_k)^T & \text{dual update} \\ u_{k+1} = u_k - B_k^{-1} N_u(y_k, \bar{y}_k, u_k)^T & \text{design update} \end{cases}$$

**Aims:** **Choose *B* such that:**
- **Convergence of *(OS)*.**
- **Bounded retardation.**

# Bounded retardation

**Jacobian of the extended iteration:**

$$J_* = \left.\frac{\partial(y_{k+1}, \bar{y}_{k+1}, u_{k+1})}{\partial(y_k, \bar{y}_k, u_k)}\right|_{(y^*, \bar{y}^*, u^*)} = \begin{pmatrix} G_y & 0 & G_u \\ N_{yy} & G_y^T & N_{yu} \\ -B^{-1}N_{uy} & -B^{-1}G_u^T & I - B^{-1}N_{uu} \end{pmatrix}$$

**Whenever we can define $B$ such that**

$$\frac{1-\rho(G_y)}{1-\hat{\rho}(J_*)} < const, \quad \textit{i.e.} \quad O(opt)\,/\,O(sim) < const$$

**we have bounded retardation.**

# Exact penalty function: $L^a$

## Remark:

**Deriving (sufficient) conditions on $B$ for $J_*$ to have a spectral radius smaller than 1 has proven difficult.**

**Instead, we look for descent on the augmented Lagrangian**

$$L^a(y,\bar{y},u) := \frac{\alpha}{2}\underbrace{\left\|G(y,u)-y\right\|^2}_{\text{primal residual}} + \frac{\beta}{2}\underbrace{\left\|N_y(y,\bar{y},u)^T-\bar{y}\right\|^2}_{\text{dual residual}} + \underbrace{N-\bar{y}^T y}_{\text{Lagrangian}},$$

**where $\alpha > 0$ and $\beta > 0$ .**

# Descent condition

**Theorem (Descent condition):**

$$s(y,\bar{y},u) = \begin{bmatrix} G(y,u) - y \\ N_y(y,\bar{y},u)^T - \bar{y} \\ -B^{-1}N_u(y,\bar{y},u)^T \end{bmatrix}$$ **is a descent direction for**

**all large positive $B$ if and only if**

$$\alpha\beta(I - \frac{1}{2}(G_y + G_y{}^T)) > (I + \frac{\beta}{2}N_{yy})(I - \frac{1}{2}(G_y + G_y{}^T))^{-1}(I + \frac{\beta}{2}N_{yy}),$$

**which is implied by** $\sqrt{\alpha\beta}(1-\rho) > 1 + \frac{\beta}{2}\|N_{yy}\|.$

- **Satisfied for** $\beta = \frac{2}{c}, \quad \alpha = \frac{2c}{(1-\rho)^2}$ **with** $c = \|N_{yy}\|$ **.**

**Theorem:** **A suitable $B$ is given by:**

$$B = \alpha G_u^T G_u + \beta N_{yu}^T N_{yu} + N_{uu}.$$

**[Hamdi, Griewank, 2008]**

# One-step one-shot

## Aerodynamic shape design

**Descent for** $\beta = \frac{2}{c}, \quad \alpha = \frac{2c}{(1-\rho)^2}$ **with** $c = \|N_{yy}\|$ **.**

**(In practice choose** $c = 1, \;\Rightarrow\; \beta = 2, \;\alpha >> 1.$**)**

**A suitable** $B$ **is given by** $B = \alpha G_u^T G_u + \beta N_{yu}^T N_{yu} + N_{uu}.$

**Instead BFGS updates for the Hessian**

$$\nabla_u^2 L^a = \underbrace{\alpha G_u^T G_u + \beta N_{yu}^T N_{yu} + N_{uu}}_{B} + \alpha \underbrace{(G - y)}_{\to_* 0}{}^T G_{uu} + \beta \underbrace{(N_y^T - \bar{y})}_{\to_* 0}{}^T N_{yuu}.$$

**The gradient** $\nabla_u L^a = \alpha (G-y)^T G_u + \beta (N_y - \bar{y})^T N_{yu} + N_u$

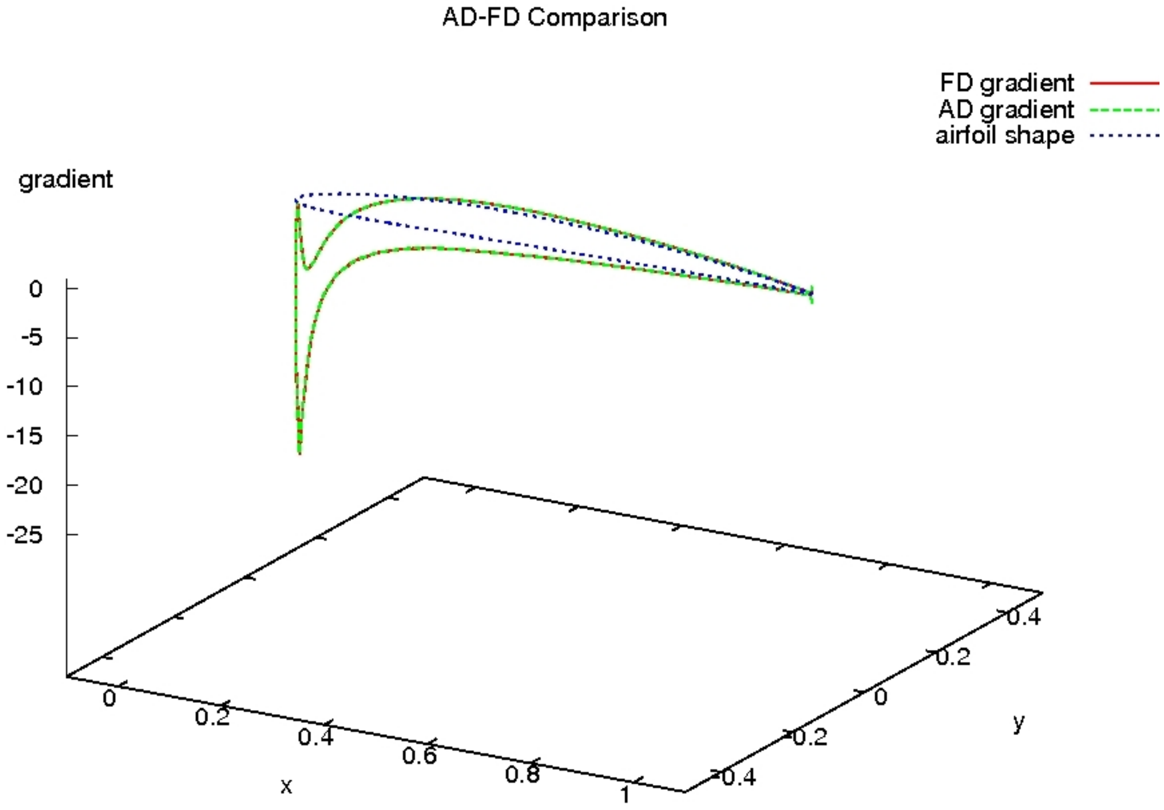
**is evaluated by Algorithmic Differentiation (AD).**

**[Özkaya, Gauger, 2008]**

# Navier-Stokes (ELAN Code)

**Flow Solver: ELAN (TU Berlin)**
- 3D Navier-Stokes (RANS)
- incompressible with pressure correction
- multiblock
- k-ω (Wilcox) turbulence model (and others)
- Fortran (20.000 lines)

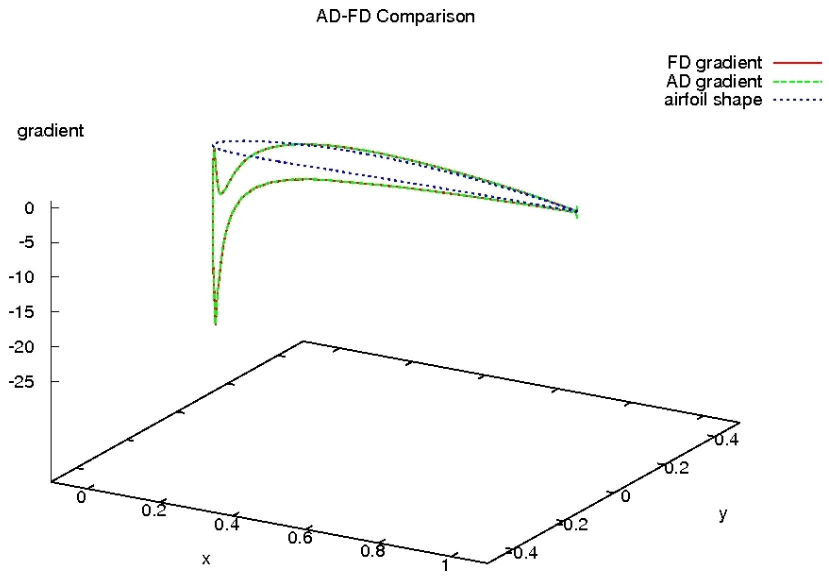
**AD Tool: TAPENADE (INRIA)**
- source to source
- reverse for **first derivatives**
- tangent on reverse for **second derivatives**

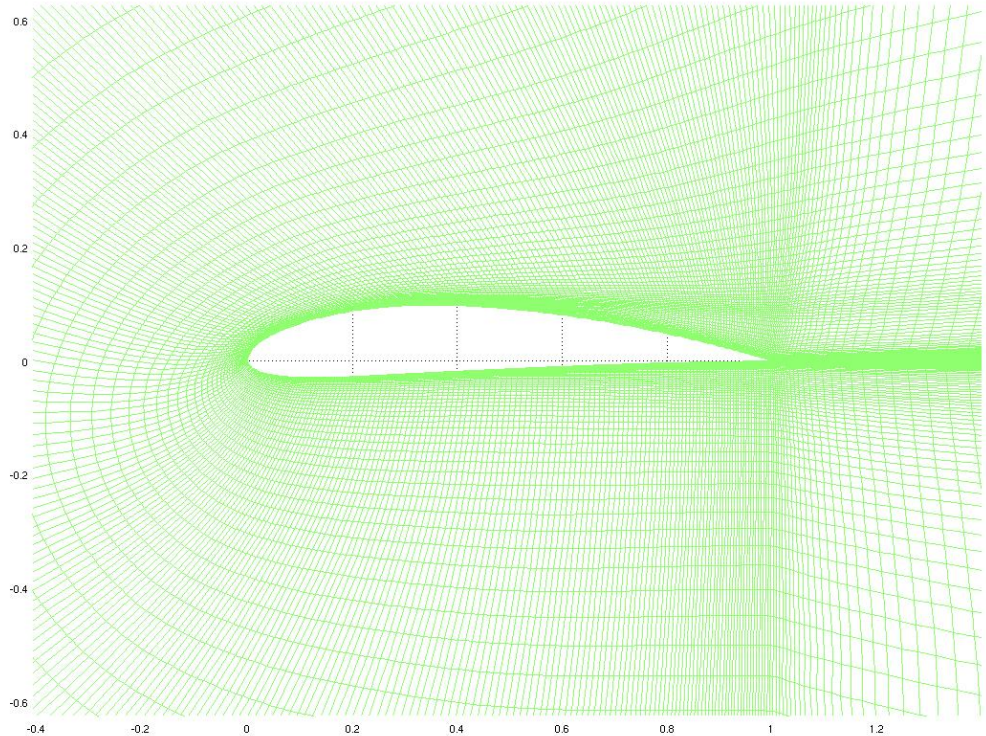


# Navier-Stokes (ELAN Code)

**Drag reduction with lift constraint**
- NACA 4412
- Re = 1.000.000,   $\alpha$=5.1°
- RANS
- k-ω (Wilcox) turbulence model
- 300 surface mesh points

**Approaches for Optimization**
- one-shot method
- entire design chain differentiated
- gradient smoothing
- penalty multiplier method



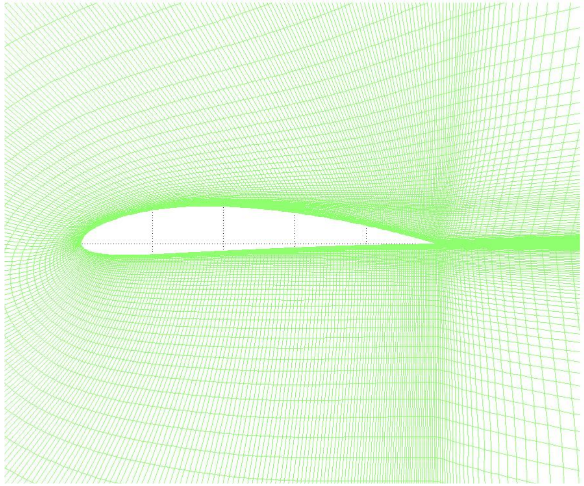
# Navier-Stokes (ELAN Code)

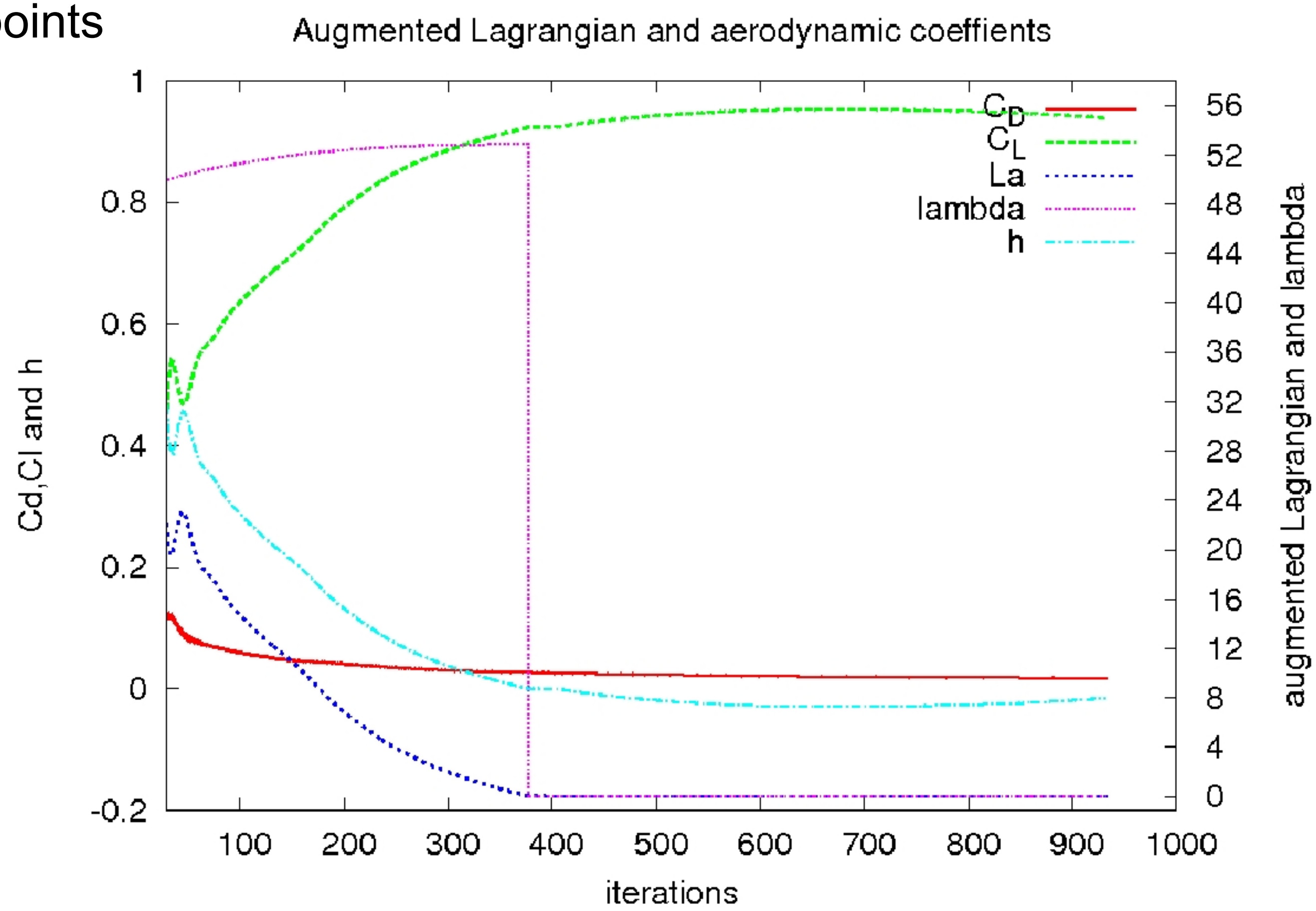
**Drag reduction with lift constraint**

- NACA 4412
- Re = 1.000.000, $\alpha$=5.1°
- RANS
- k-ω (Wilcox) turbulence model
- 300 surface mesh points

**Approaches for Optimization**

- one-shot method
- entire design chain differentiated
- gradient smoothing
- penalty multiplier method





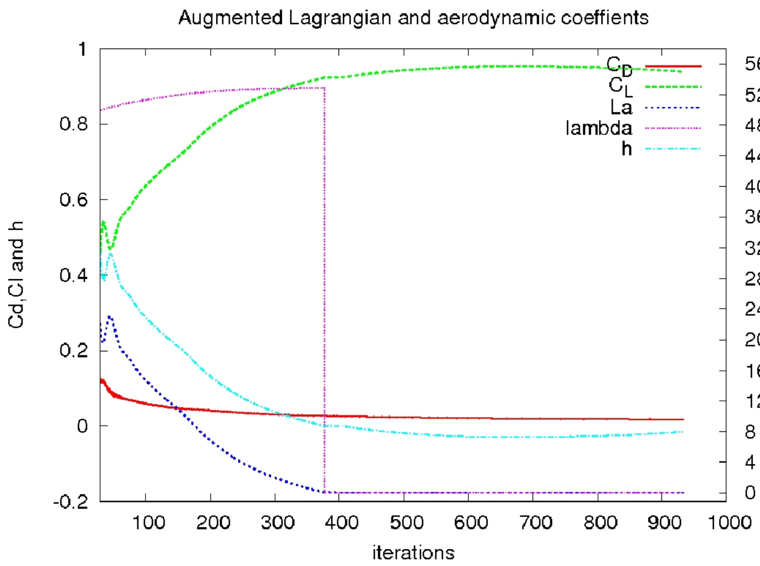
# Navier-Stokes (ELAN Code)

RWTH AACHEN UNIVERSITY

**Drag reduction with lift constraint**

- NACA 4412
- Re = 1.000.000, $\alpha$=5.1°
- RANS
- k-ω (Wilcox) turbulence model
- 300 surface mesh points

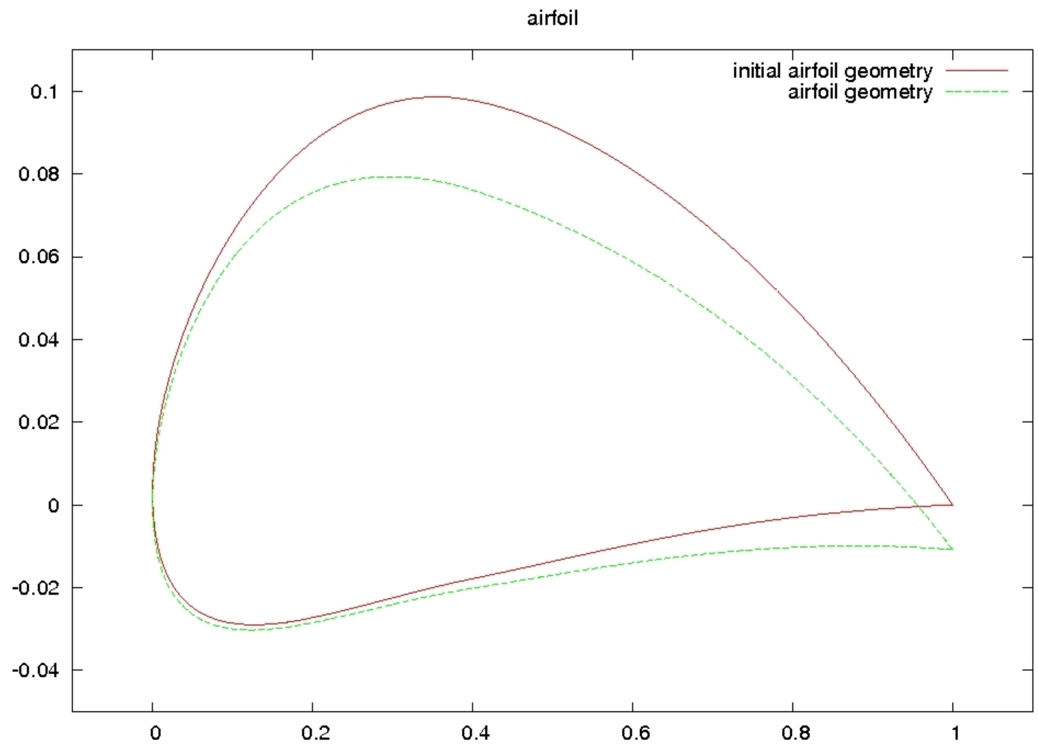
**5% drag reduction**




**Retardation-Factor = 3**

**Approaches for Optimization**

- one-shot method
- entire design chain differentiated
- gradient smoothing
- penalty multiplier method




**[Özkaya, Gauger, 2009]**

# Ingredients

## for Efficient Optimization and Control in Aerodynamics

- Adjoint-based sensitivity evaluation (discrete, continuous, hybrid)
- Checkpointing for unsteady adjoint computation
- Non-loosely coupled (adjoint) PDEs for MDO
- One-shot methods (also called All-at-once, SAND, ...)
- Preconditioning of design equation
- Gradient smoothing
- Multilevel parameterization / Free node parameterization
- Shape derivatives and shape gradients
- Incorporation of adaptation by dual weighted residuals (DWR)
- Calculation of Pareto-fronts by "equality-constraint-based scans"
- ...

**Thanks!**