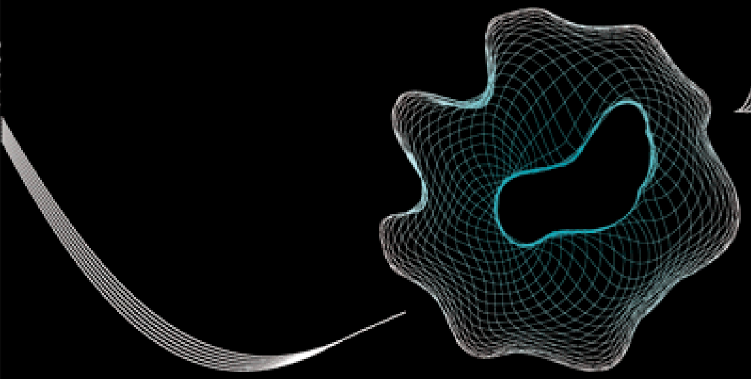
UNIVERSITY OF TWENTE.

# TOWARDS GRADIENT BASED AERODYNAMIC OPTIMIZATION OF WIND TURBINE BLADES USING OVERSET GRIDS



S. H. Jongsma
E. T. A. van de Weide
H. W. M. Hoeijmakers

Department of mechanical engineering

Overset symposium 10-18-2012

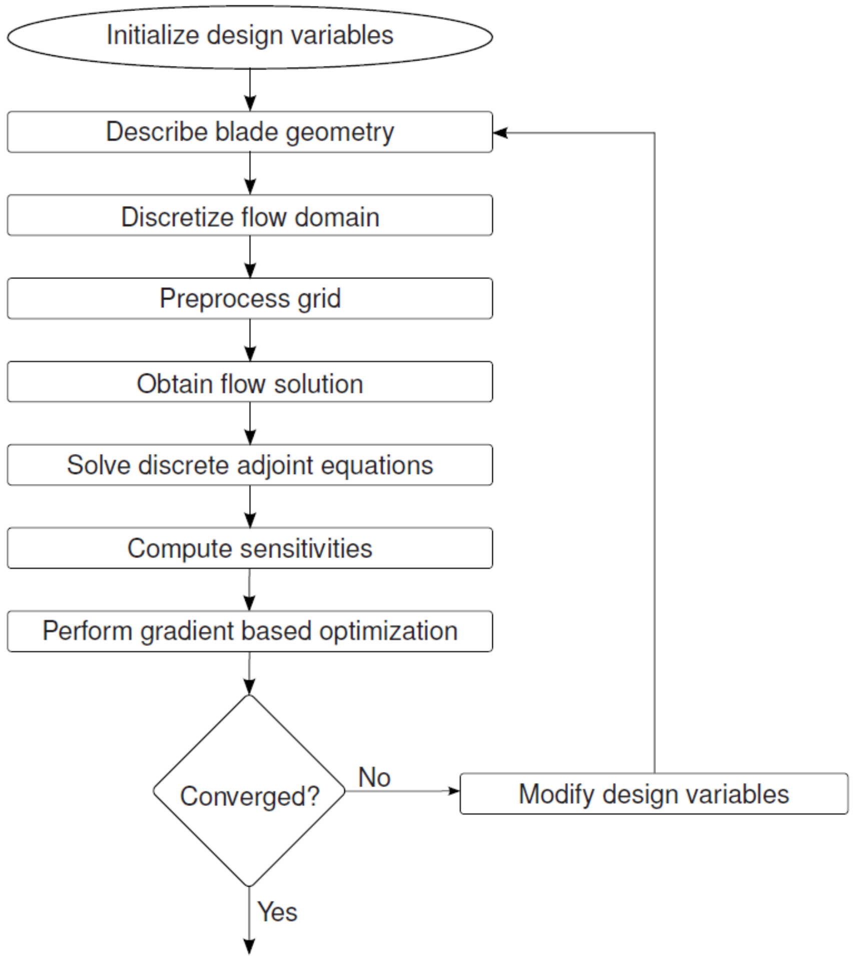
## OUTLINE

## MOTIVATION (1)

- Conventional approach in design of blade:
    - Design 2D aerofoil sections
    - Combine sections to obtain blade geometry
    - Incorporate effects 3D flow features
        - ➢ very difficult even for experienced designers

- Current approach:
    - Simulation based design of blade shape in 3D
    - 3D flow features taken into account
    - Design expertise required for defining objective function and constraints for optimization method

## GRADIENT BASED OPTIMIZATION PROCEDURE



## FLOW MODEL (1)

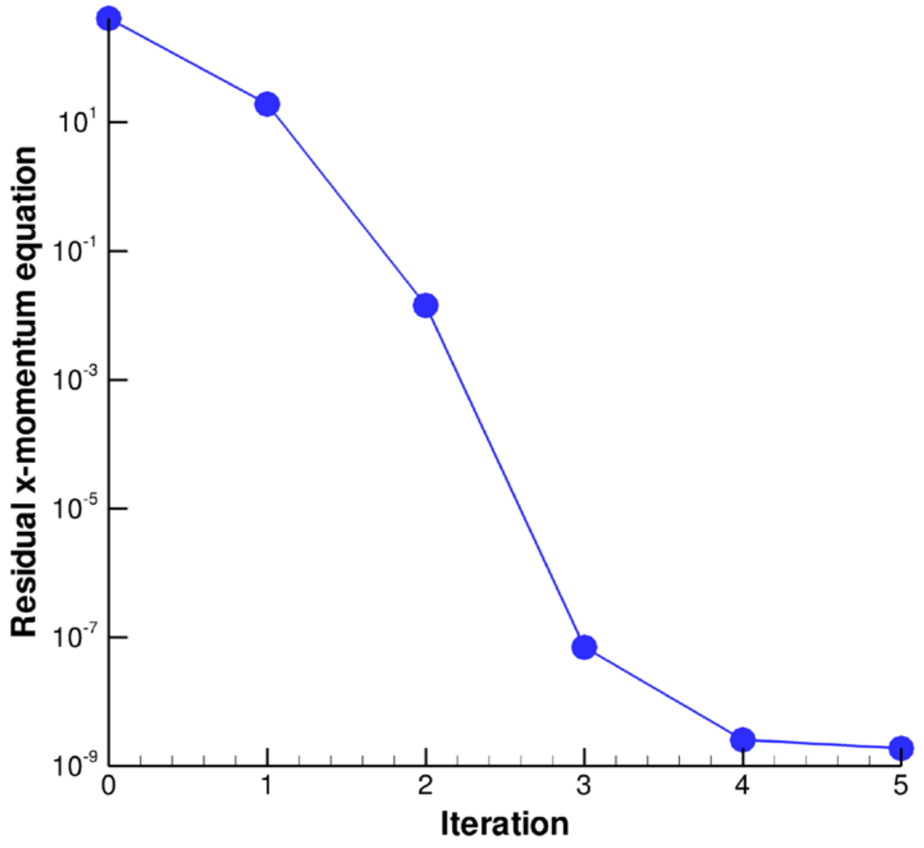
- Consider rigid isolated rotor:
  - i.e. effects of presence of tower and ground plane are neglected
  - effect of blade deformation is neglected

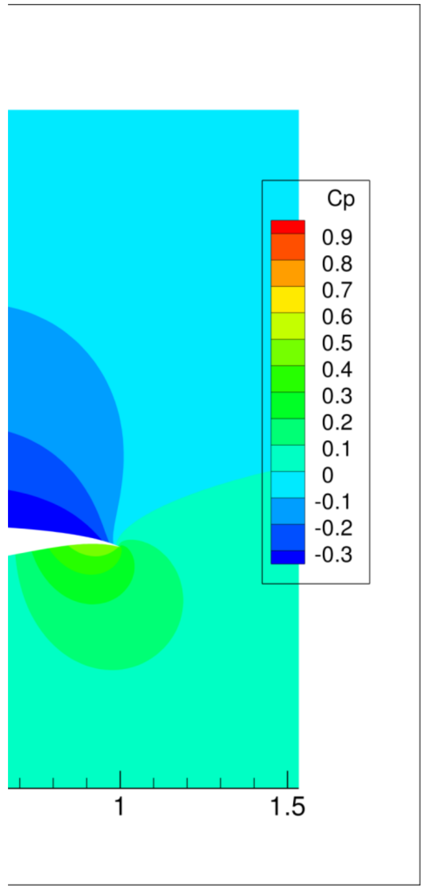
- Assume rotor plane perpendicular to steady wind:
  - ➢ flow is periodic → Steady flow around single blade in a rotating frame
  - i.e. effect of unsteady inflow and yaw is not considered

- Currently solve Euler equations
  - ➢ Later: RANS

## FLOW MODEL (2)

- Cell centered fi
- 2nd Order spati
- Integration to st
  - Newton's me
  - Runge-Kutta
- Use PETSc to s system of linea

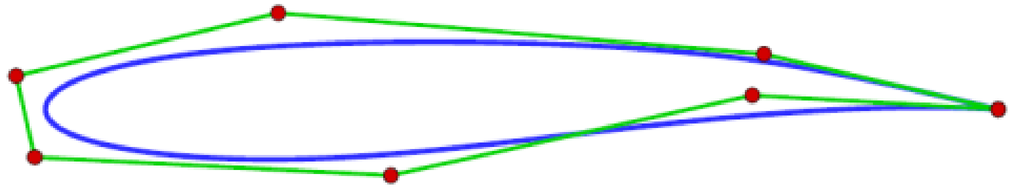




$Re_c = \infty$ $M_\infty = 0.2$ $a = 2.3^o$

O-grid and Cartesian background grid

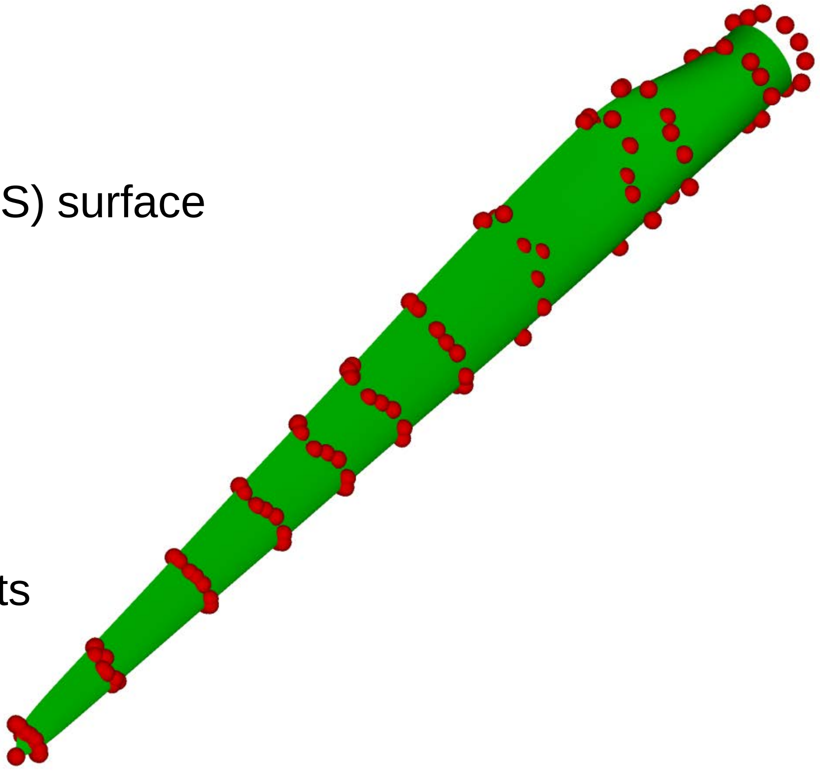
## PARAMETERIZATION METHOD



- Requirements:
  - Span complete design space
  - Limit number of design variables

- Choice:
  - Non-uniform rational basis spline (NURBS) surface
    - Flexible
    - Compatible with CAD

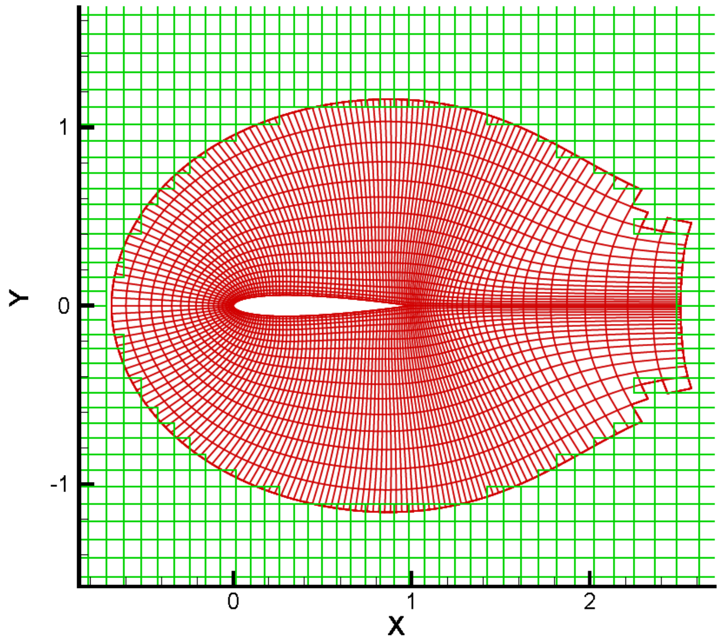
- Design variables:
  - (Selection of) coordinates of control points
  - Weights (if desired)

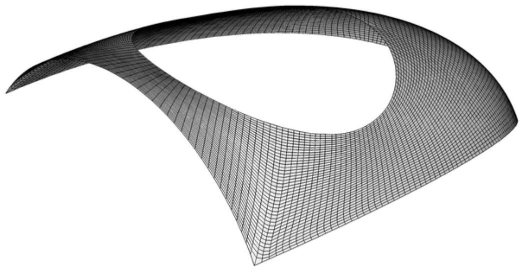


# FLOW DOMAIN DISCRETIZATION

- Requirements:
  - Account for change of shape after each design iteration
  - Not computationally intensive
  - Fully automatic

- Choice:
  - Multi-block overset grids
  - Use hyperbolic grid generation for field grids



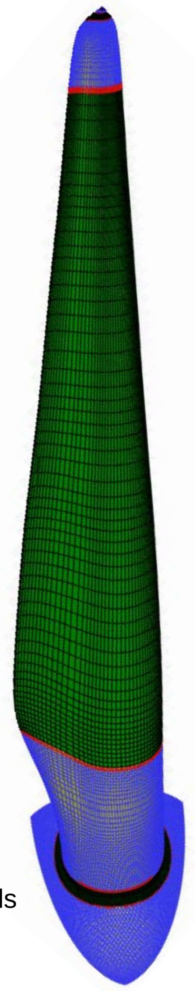


UNIVERSITY OF TWENTE.

## GRID PREPROCESSING (1)

Approach

- Evaluation of surface integrals
  - Zipper grid (Chan 2009)
- Elimination cells inside geometrical entities
  - Ray casting
- Block connectivity
  - Implicit hole cutting (Lee 2008)
- No sequential bottleneck
  - Reasonably efficient parallel implementation

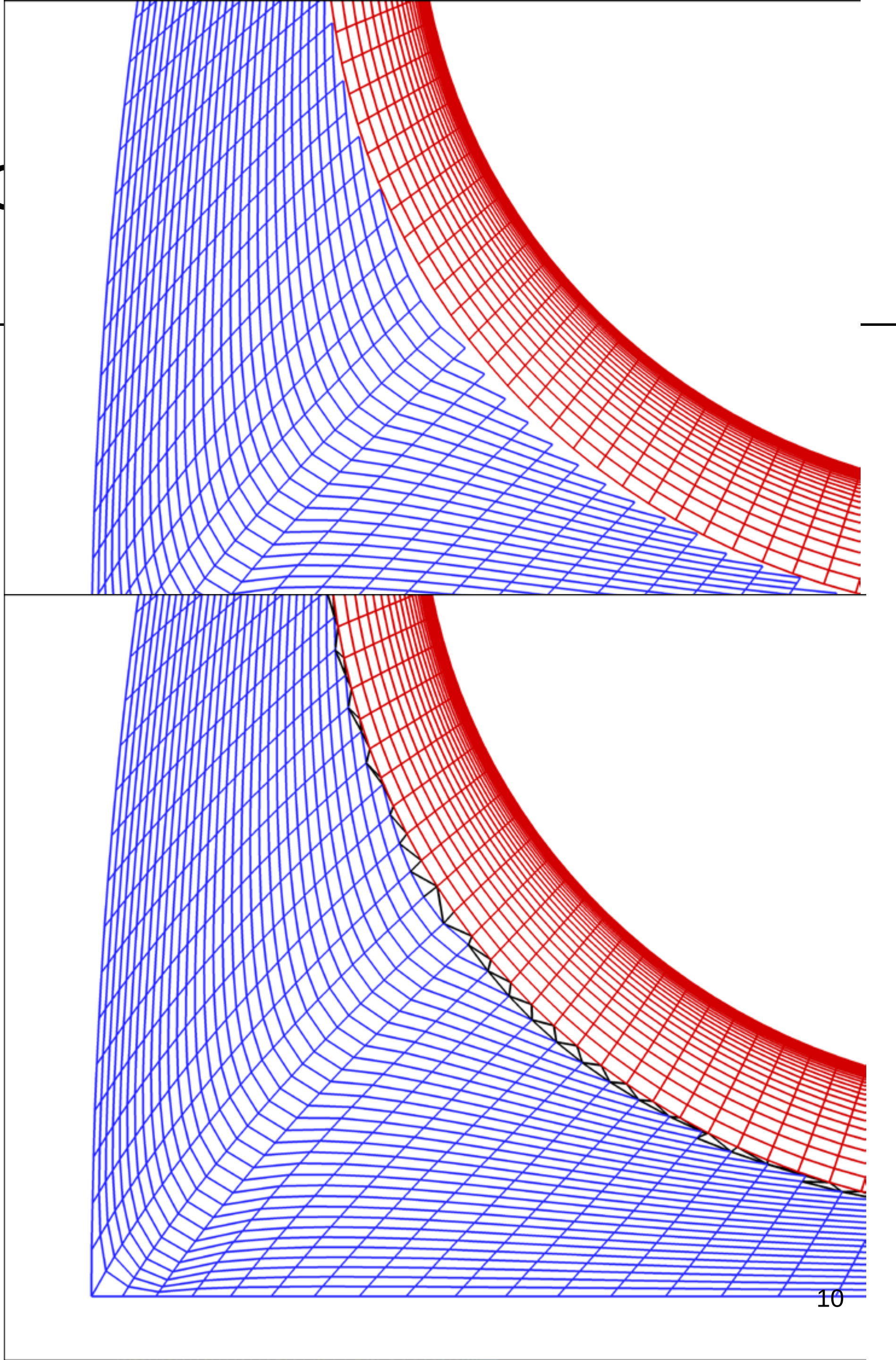


Chan (2009), Enhancements to the Hybrid Mesh Approach to Surface Loads Integration on Overset Structured Grids
Lee (2008), On Overset Grids Connectivity and Vortex Tracking in Rotorcraft CFD (*PhD thesis)*

# GRID PREPROCESSIN

## ZIPPER GRID (1)

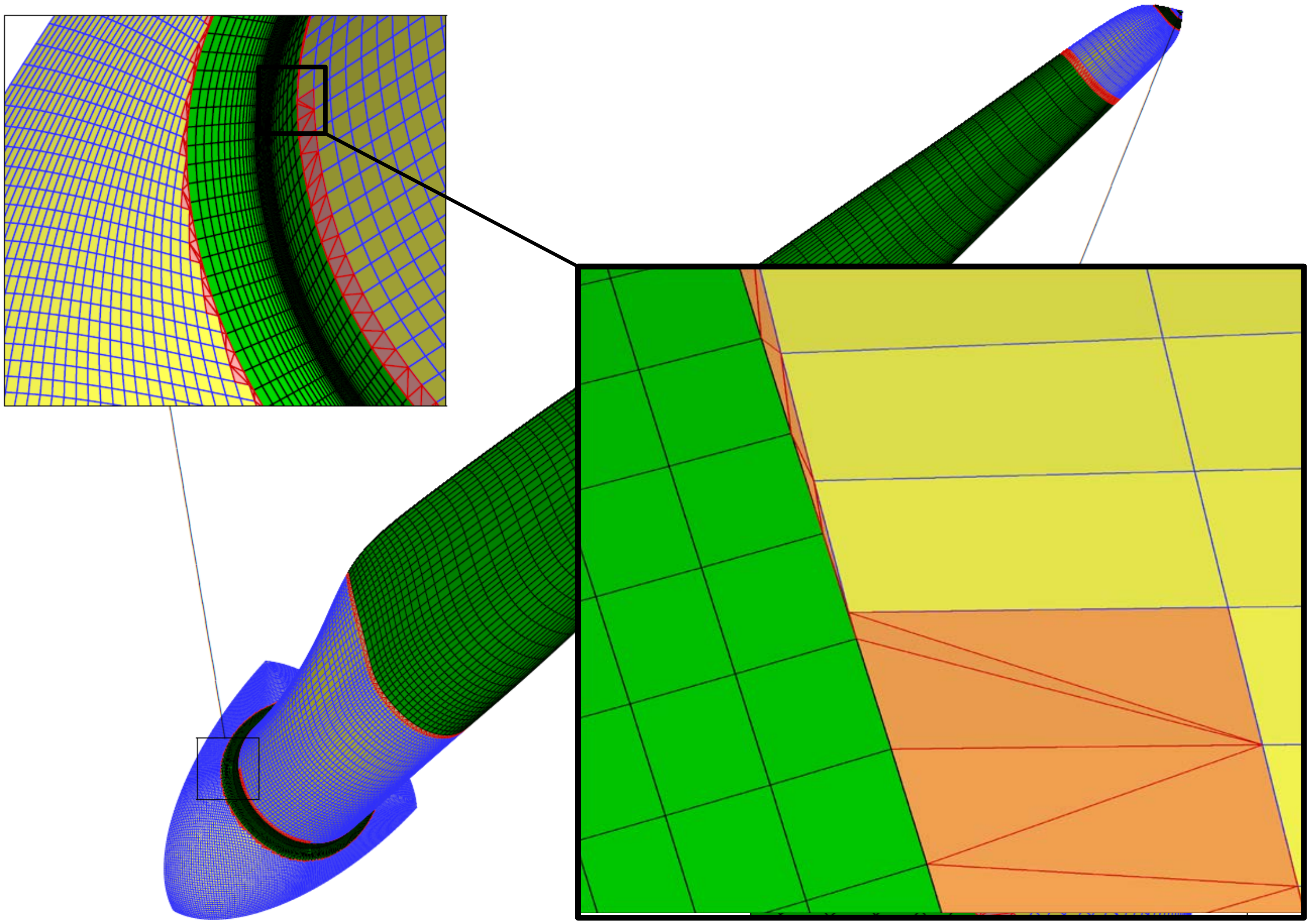
- Closed non-overlapping surface grid required for:
  - Evaluation of surface integrals
  - Ray-casting procedure
- Achieved by application of zipper grids:
  - remove overlapping part of surface grids
  - connect neighboring grids by creating triangles in between
- Works for problems considered so far



UNIVERSITY OF TWENTE.

## GRID PREPROCESSING (3)

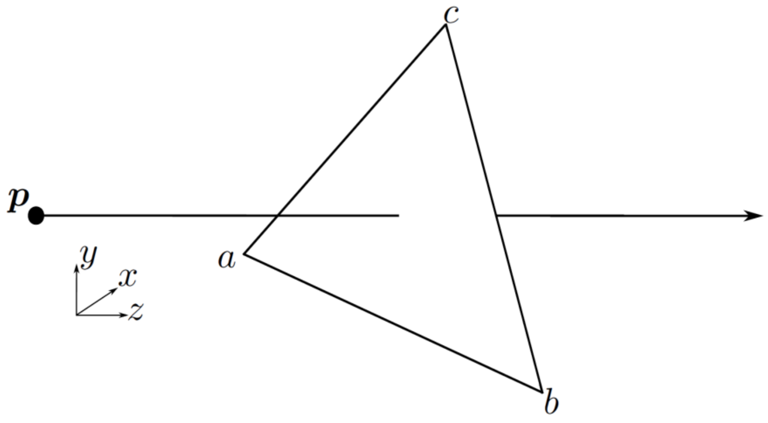
### ZIPPER GRID (2)



# GRID PREPROCESSING (4)

## RAY-CASTING

- Approach:
  - Choose $\boldsymbol{p}$ and casting direction
  - Find possible matches using tree search
  - Check possible matches with accurate check



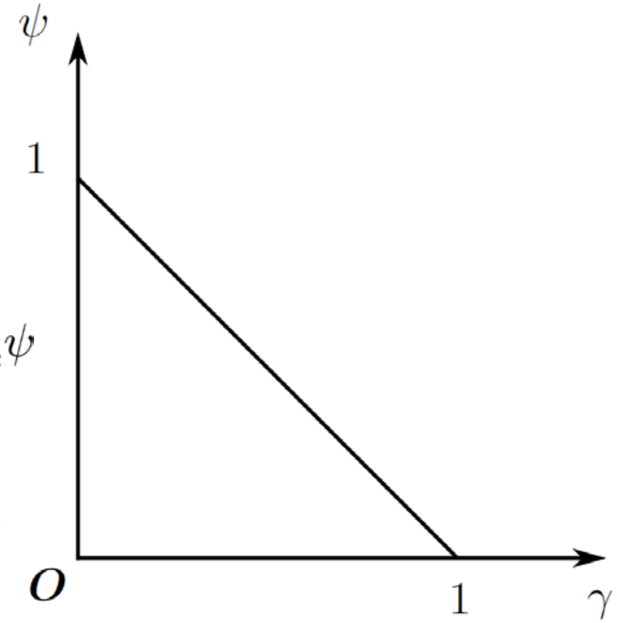
- Accurate check:
  - Split quadrilateral in 2 triangles
  - Translate triangle to origin:

$$\boldsymbol{r}'_i = \boldsymbol{r}_i - \boldsymbol{r}_a \quad \forall\, i \in \{a, b, c\}$$
$$\boldsymbol{p}' = \boldsymbol{p} - \boldsymbol{r}_a$$

  - Project $\boldsymbol{p}'$ and triangle on face perpendicular to casting direction
  - Use transformation to barycentric coordinates:

$$\boldsymbol{r}'(\gamma, \psi) = \boldsymbol{r}'_b\gamma + \boldsymbol{r}'_c\psi$$

  - Solve: $\begin{bmatrix} x'_b & x'_c \\ y'_b & y'_c \end{bmatrix} \begin{pmatrix} \gamma \\ \psi \end{pmatrix} = \begin{pmatrix} x'_p \\ y'_p \end{pmatrix}$
  - Ray crosses triangle if: $\gamma \geq 0 \ \wedge \ \psi \geq 0 \ \wedge \ (\gamma + \psi) \leq 1$



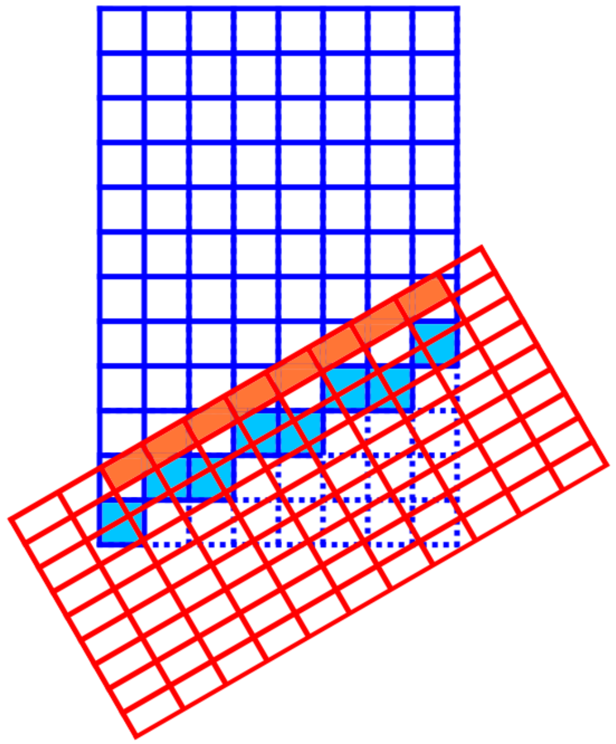
UNIVERSITY OF TWENTE.

# GRID PREPROCESSING (5)

## IMPLICIT HOLE CUTTING (1)

<u>Concept</u>

- Use dual grid for hole cutting
- Identify cells that reside in region covered by multiple blocks
- In region of overlap:
  - Consider user defined priority of blocks
  - Consider index distance to wall of cells
  - Consider cell volume
    - Use cells with the smallest volume to solve governing equations (field cell)
    - Cell with larger volume gets status: hole cell
  - Specify fringe cells at interface between hole and field cells (stencil dependent)

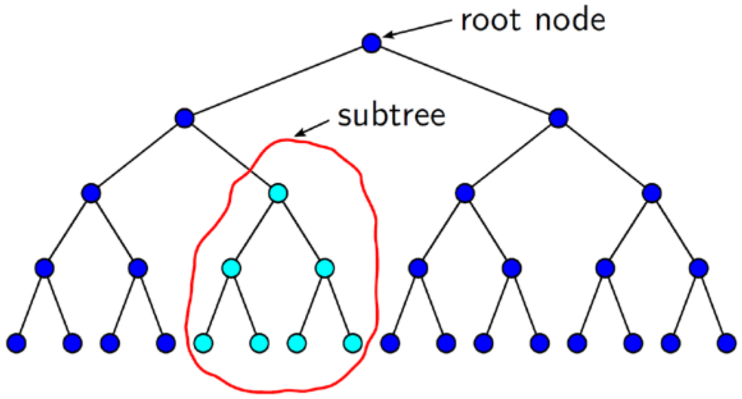


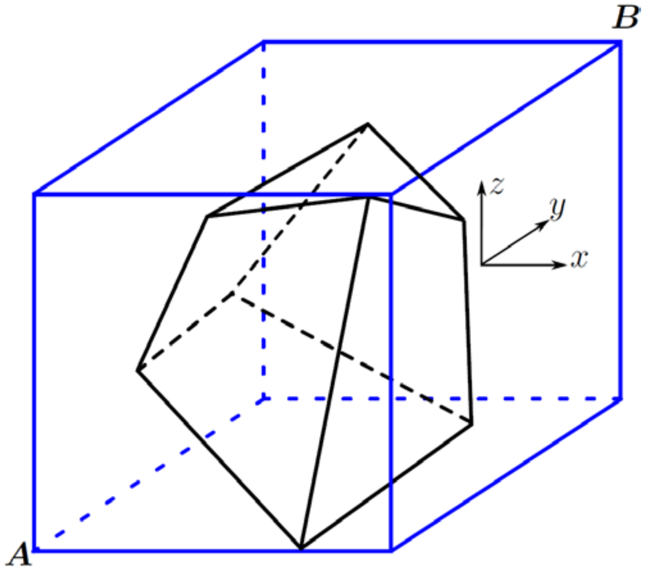
## GRID PREPROCESSING (6)

### IMPLICIT HOLE CUTTING (2)

<u>Approach</u>

- Construct alternating digital tree for all cells in each block
- Perform tree search to look for cells for which the bounding box overlaps a particular cell
  - Use tri-linear transformation to identify overlap for potential candidates from tree search
- Use criteria on block priority, index distance and cell volume to identify field and hole cells
- Identify fringe cells





## GRADIENT COMPUTATION (1)

### INTRODUCTION (1)

- Requirements:
  - Accurate
  - Obtained efficiently

- Methods:
  - Finite difference
  - Complex step finite difference
  - Discrete adjoint equation method

## GRADIENT COMPUTATION (2)

### DISCRETE ADJOINT EQUATION METHOD (1)

| | |
|---|---|
| ▪ Design variable | $\alpha$ |
| ▪ Computational mesh | $\boldsymbol{X} = \boldsymbol{X}(\alpha)$ |
| ▪ Flow variables | $\boldsymbol{u} = \boldsymbol{u}(\alpha, \boldsymbol{X})$ |
| ▪ Objective function | $I = I(\alpha, \boldsymbol{u}, \boldsymbol{X})$ |
| ▪ Residual of flow solution | $\boldsymbol{R} = \boldsymbol{R}(\alpha, \boldsymbol{u}, \boldsymbol{X})$ |

- Total derivative of objective function:

$$\frac{\mathrm{d}I}{\mathrm{d}\alpha} = \frac{\partial I}{\partial \alpha} + \frac{\partial I}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\alpha} + \frac{\partial I}{\partial \boldsymbol{u}}\boxed{\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\alpha}}$$

## GRADIENT COMPUTATION (3)

### DISCRETE ADJOINT EQUATION METHOD (2)

- Total derivative of residual:

$$\frac{\mathrm{d}\boldsymbol{R}}{\mathrm{d}\alpha} = \frac{\partial \boldsymbol{R}}{\partial \alpha} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\alpha} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\alpha}$$

- Rewrite to:

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\alpha} = -\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\right)^{-1}\left(\frac{\partial \boldsymbol{R}}{\partial \alpha} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\alpha}\right)$$

- Substitute:

$$\frac{\mathrm{d}I}{\mathrm{d}\alpha} = \frac{\partial I}{\partial \alpha} + \frac{\partial I}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\alpha} + \frac{\partial I}{\partial \boldsymbol{u}}\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\alpha}\frac{\boldsymbol{R}}{\boldsymbol{u}}\Bigg)^{-1}\left(\frac{\partial \boldsymbol{R}}{\partial \alpha} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\alpha}\right)$$

## GRADIENT COMPUTATION (4)

### DISCRETE ADJOINT EQUATION METHOD (3)

- Define adjoint vector: $\boldsymbol{\psi} := -\left(\frac{\partial I}{\partial \boldsymbol{u}}\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\right)^{-1}\right)^{T}$

- Substitute to obtain new expression for total derivative:

$$\frac{\mathrm{d}I}{\mathrm{d}\alpha} = \frac{\partial I}{\partial \alpha} + \frac{\partial I}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\alpha} - \frac{\partial I}{\partial \boldsymbol{u}}\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\right)^{-1}\left(\frac{\partial \boldsymbol{R}}{\partial \alpha} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\alpha}\right)$$

- Advantage:
  - Only one flow solution needed for all derivatives, once adjoint vector is found by solving:

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\right)^{T} \psi = -\left(\frac{\partial I}{\partial \boldsymbol{u}}\right)^{T}$$

## GRADIENT COMPUTATION (5)

### DUAL NUMBERS (1) (Fike et al. 2011)

$$\mathbb{D} := \left\{ a + \varepsilon b \; : \quad a, b \in \mathbb{R}, \quad \boxed{\varepsilon^2 \equiv 0} \right\}$$

$$f(x + \varepsilon h) = f(x) + \varepsilon h \frac{\mathrm{d} f(x)}{\mathrm{d} x} + \xcancel{\frac{\varepsilon^2 h^2}{2!} \frac{\mathrm{d}^2 f(x)}{\mathrm{d} x^2}} + \xcancel{\mathcal{O}\left(\varepsilon^2 [\varepsilon h^3]\right)}$$

$$\frac{\mathrm{d} f(x)}{\mathrm{d} x} \equiv \frac{\mathfrak{D}\left[f(x + \varepsilon h)\right]}{h}, \quad h \in \mathbb{R}.$$

Choose $h = 1$

Fike & Alonso (2011), The Development of Hyper-Dual Numbers for Exact Second-Derivative Calculations, AIAA paper 2011-886

## GRADIENT COMPUTATION (6)

### DUAL NUMBERS (2)

<u>Computational rules</u>

$$\text{Addition:}\qquad (a+\varepsilon b)+(c+\varepsilon d)=(a+c)+\varepsilon(b+d)$$

$$\text{Multiplication:}\quad (a+\varepsilon b)(c+\varepsilon d)=ac+\varepsilon(ad+bc)$$

$$\text{Division:}\qquad 1/(\mathrm{c}+\varepsilon \mathrm{d})=1/c-\varepsilon d/c^2$$

$$(a+\varepsilon b)/(c+\varepsilon d)=(a+\varepsilon b)\cdot 1/(c+\varepsilon d)=a/c+\varepsilon(b/c-ad/c^2)$$

$$\text{Math functions:}\ \cos(a+\varepsilon b)=\cos(a)-\varepsilon b\sin(a)$$

$$\sin(a+\varepsilon b)=\sin(a)+\varepsilon b\cos(a)$$

$$\tan(a+\varepsilon b)=\tan(a)+\varepsilon b[\tan^2(a)+1]$$

$$\exp(a+\varepsilon b)=\exp(a)+\varepsilon b\exp(a)$$

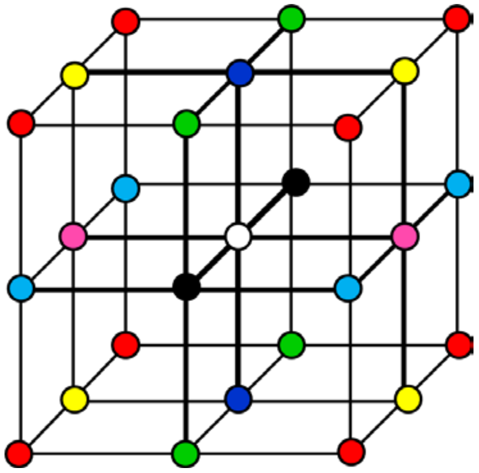
$$\ln(a+\varepsilon b)=\ln(a)+\varepsilon b/a$$

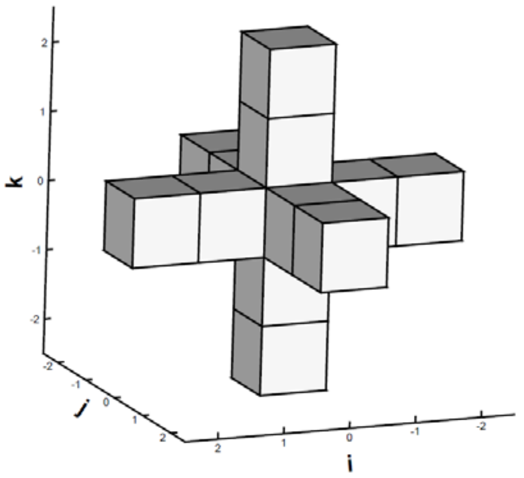
$$(\mathrm{a}+\varepsilon \mathrm{b})^{(\mathrm{c}+\varepsilon \mathrm{d})}=\exp[(c+\varepsilon d)\ln(a+\varepsilon b)]$$

## GRADIENT COMPUTATION (7)

### CONSIDERATIONS

- Limit implementation time by reusing existing functions
  - Use of C++ template functions
  - Use graph coloring to limit number of function evaluations
- Existence of fringe cells and halo cells requires special attention
  - Take effect of boundary conditions into account explicitly
  - Determine corresponding matrix index of donors
- Computation adjoint vector
  - Use first order Jacobian for construction ILU(k) preconditioning matrix
  - Use PETSc to solve system of linear equations by means of GMRES iterations



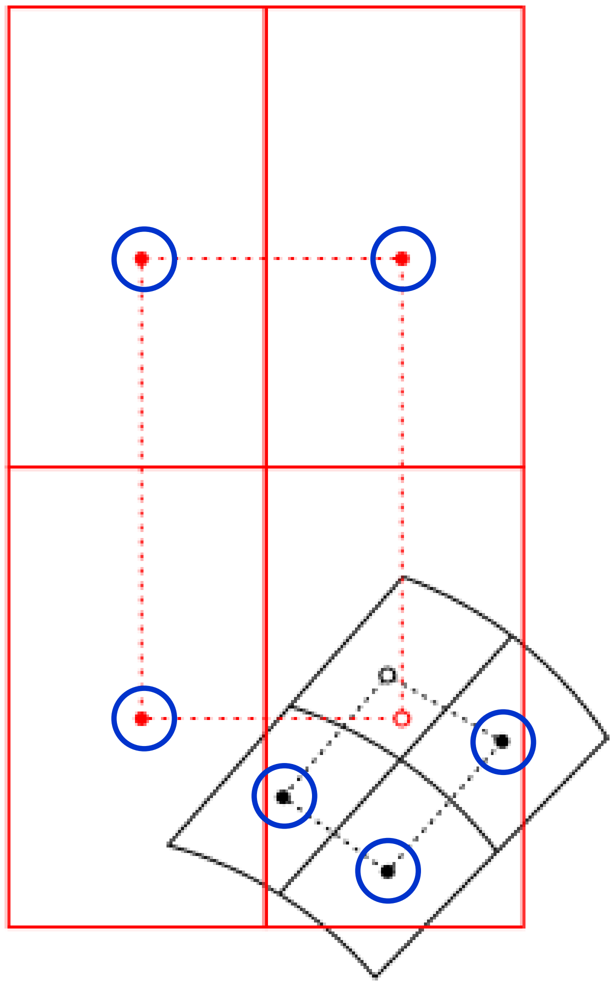


## GRADIENT COMPUTATION (8)

### HANDLING FRINGE CELLS

Construction Jacobian matrix:

- Fringe cells are represented by row in the matrix
  - Make sure each fringe cell has explicit representation by field cells
- Required to facilitate implicit solution method
  - Update solution after partial convergence GMRES iterations
- For consistency, fringe cells are updated using solution of field cells



## OPTIMIZATION (1)

### APPROACH

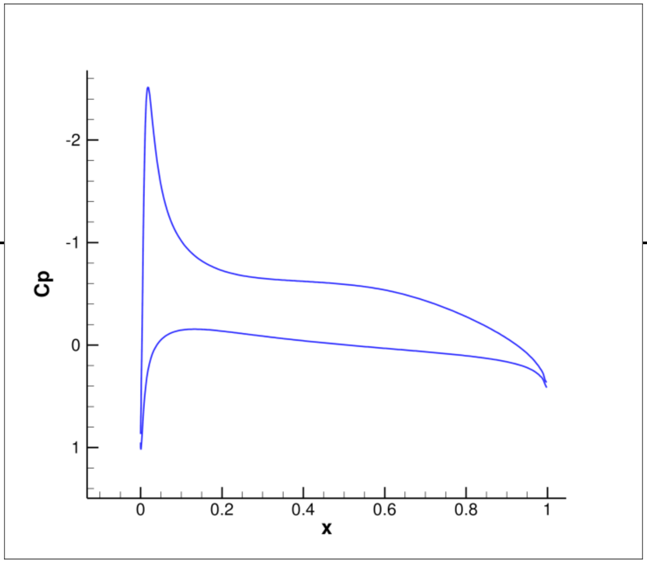
- Choose design variables
- Specify objective function and constraint functions
- Specify bounds on design variables
- Evaluate objective function and constraint functions
- Compute gradients
- Provide value of objective function, constraint functions and gradients to gradient based optimization algorithm:
  - SNOPT (sparse non-linear optimizer) (Stanford University (Gill et al. 2005))
    - BFGS quasi Newton method
    - ➢ Provides new design variables

o Repeat procedure

Gill, Murray & Saunders (2005), SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization, SIAM Review, Vol. 47, No. 1.

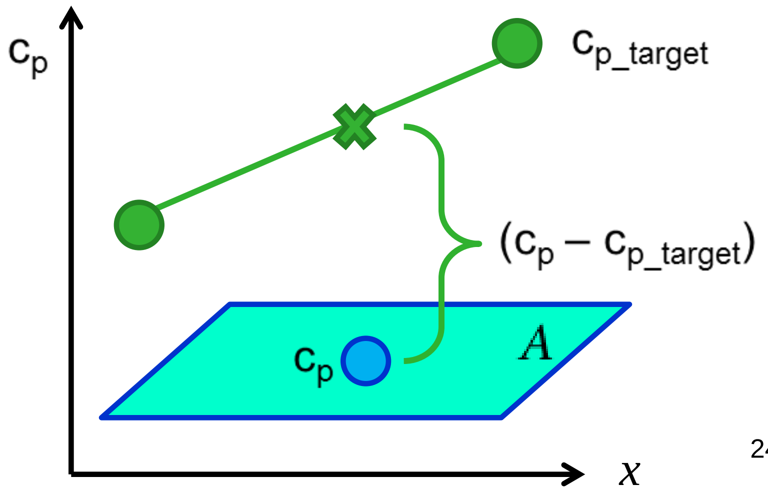
# OPTIMIZATION (2)

## PROBLEM DEFINITION (1)



Inverse(/reverse) design:

- Objective:
  - Find airfoil geometry which minimizes $(c_p - c_{p\_target})^2$
- Approach:
  - Read input file with $c_p$ data and corresponding $x$-coordinate
  - Interpolate input data for comparison of $c_p$ current design for different $x$-coordinate center of face
  - Sum square of difference
  - Minimize resulting sum

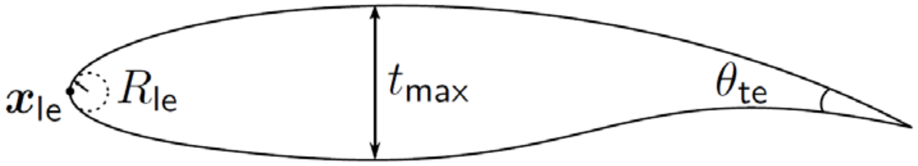


## OPTIMIZATION (3)

### PROBLEM DEFINITION (2)

- Constraint functions:

| Constraint | Value |
|---|---|
| Nose radius, $R_{\mathsf{le}}$ | $\geq 6.67 \cdot 10^{-3} c$ |
| Maximum thickness, $t_{\mathsf{max}}$ | $\geq 0.12c$ |
| Trailing edge angle, $\theta_{\mathsf{te}}$ | $\geq 10°$ |
| Coordinate leading edge, $x_{\mathsf{le}}$ | $= (0, 0, z)^T$ |
| Distance between control points | $\geq 0.1c$ |
| Angle between control points | $\leq 135°$ |



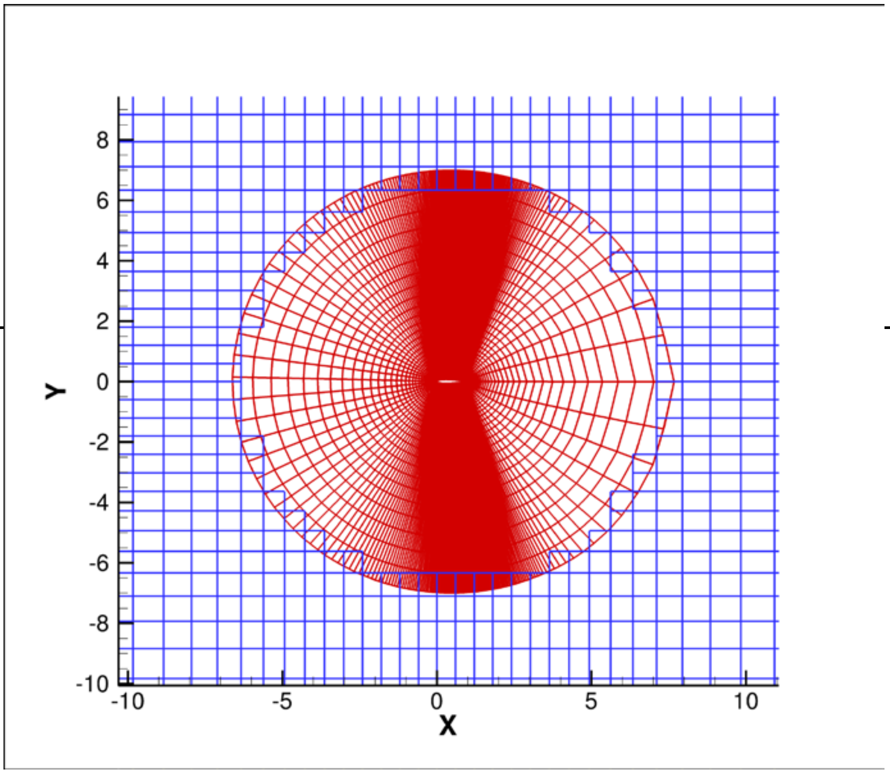
## OPTIMIZATION (4)

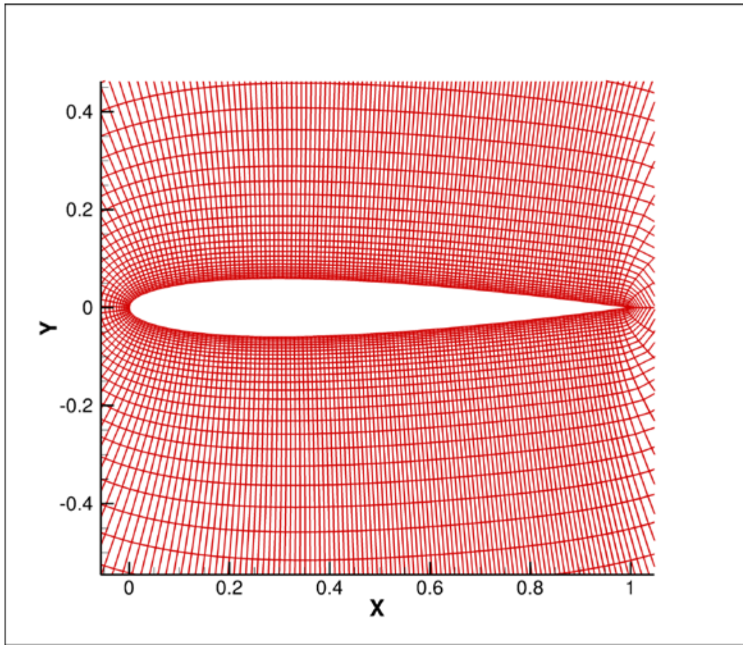
### SETUP

- Initial design:
  - NACA 0012 at 0° angle of attack
  - 13 control points → 36 design variables
- Flow conditions:
  - $M_\infty = 0.3$
- Grid dimensions:

| Grid | $n_i$ | $n_j$ |
|---|---|---|
| O-grid | 257 | 49 |
| Background | 65 | 65 |

- Far field:
  - At 50 chord lengths

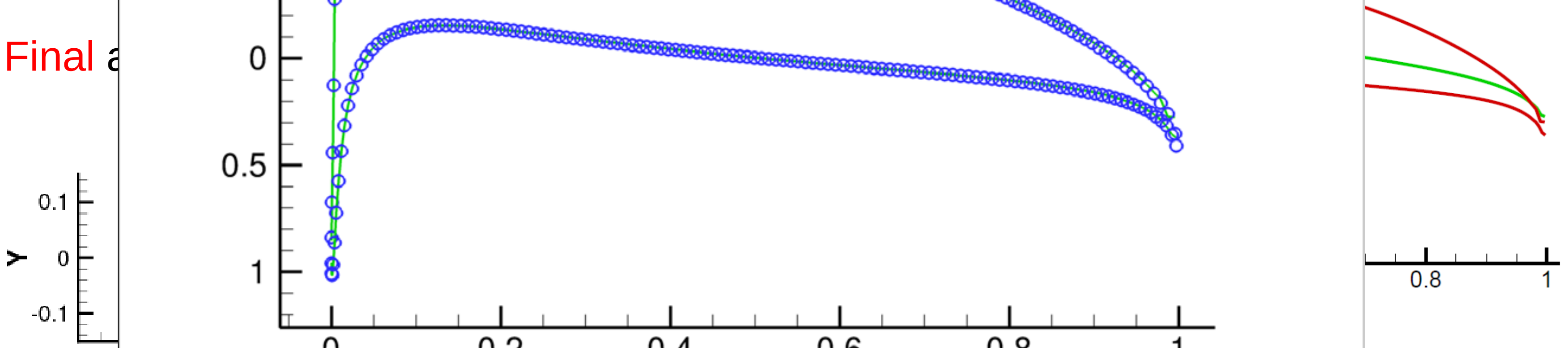


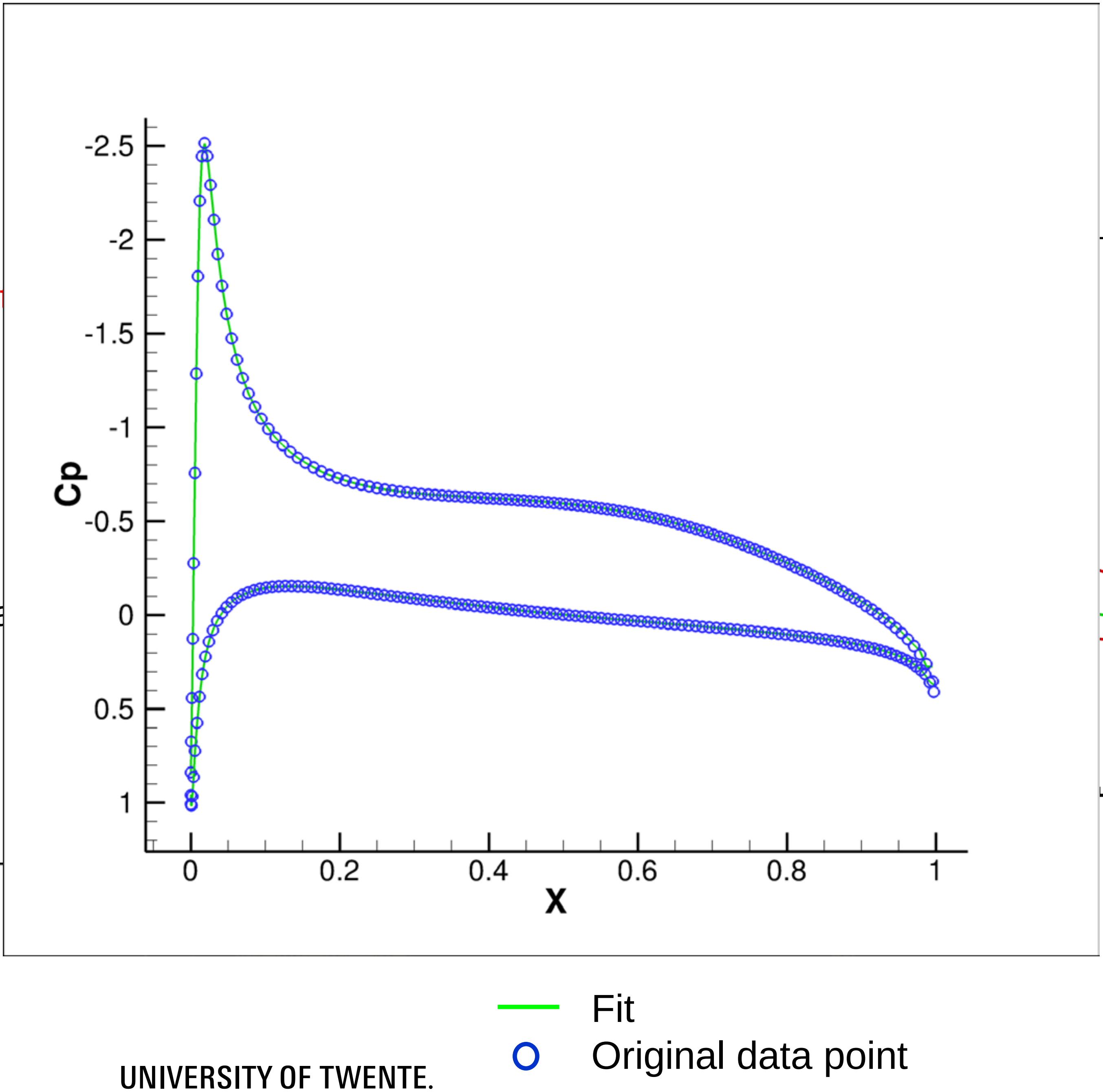


UNIVERSITY OF TWENTE.

- Targe

- Final a





## CONCLUDING REMARKS

- Parameterization using NURBS surface
- Discretization using hyperbolic grid generation method
- Fully automatic hole cutting procedure
- Derivatives computed using:
  - Discrete adjoint equation method
  - Dual number arithmetic
- Method successfully applied for solving 2D optimization problems on overset grids
  - Fit of $c_p$ distribution
  - Drag minimization in transonic flow

## FURTURE WORK

- Testing the method for simple 3D optimization problem
- Applying the method for full 3D wind turbine blade in co-rotating frame of reference
- Use RANS equations
- Extension to time-periodic optimization via time-spectral method

UNIVERSITY OF TWENTE.