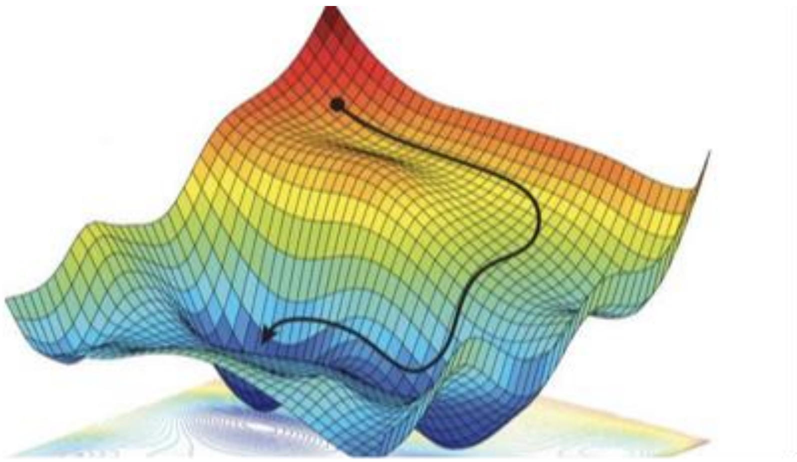
SHAPE OPTIMIZATION BASED ON DISCRETE ADJOINT

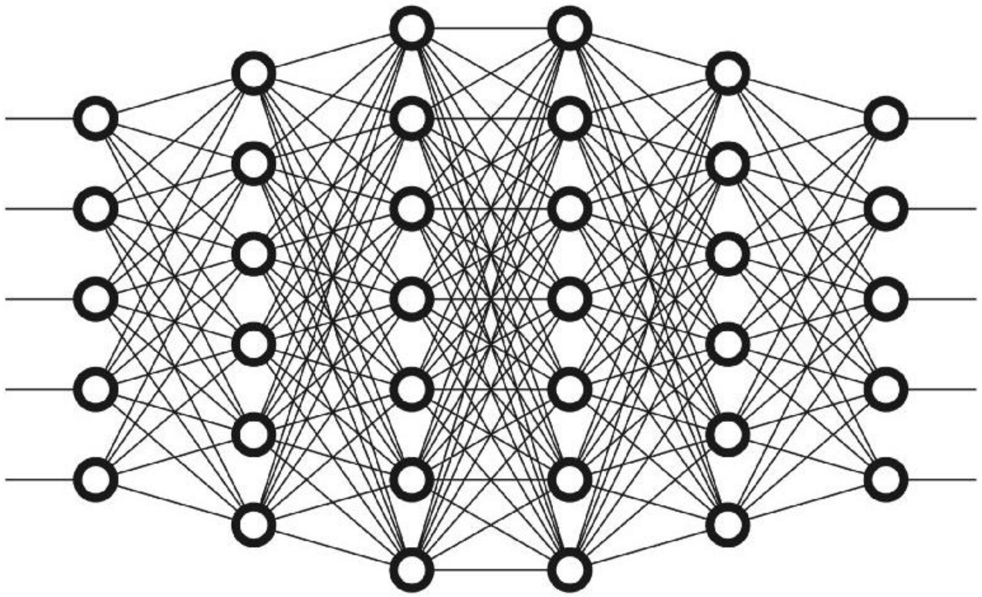
CFD Course, 27 November 2023

Luca Abergo, Prof. Barbara Re

Target:

Automatic Shape Optimization/Design of Aero Surfaces

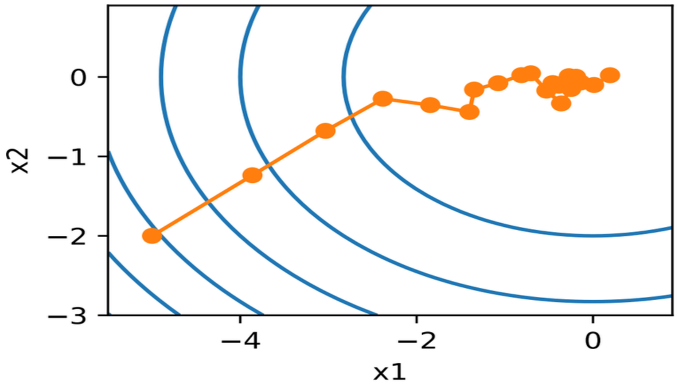




TWO MAJOR CATEGORIES
CAN YOU NAME THEM?

INTRODUCTION

GRADIENT BASED



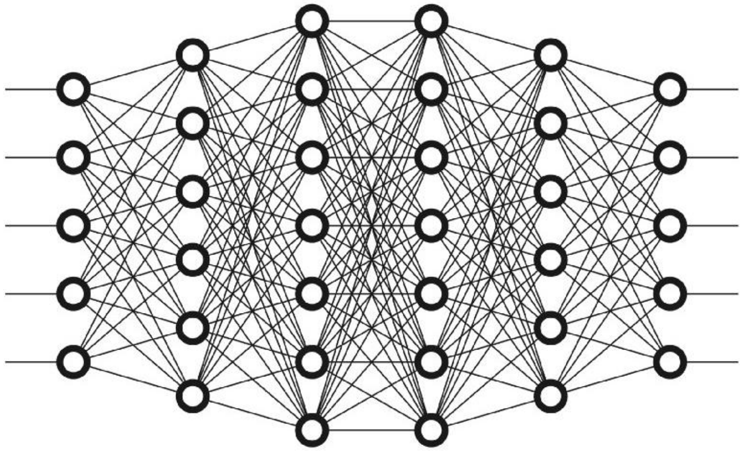
$$\alpha_{n+1} = \alpha_n - \frac{J(n)}{Jacobian(n)}$$

- **Sequential Quadratic Programming**
- **Stochastic Gradient Descent**

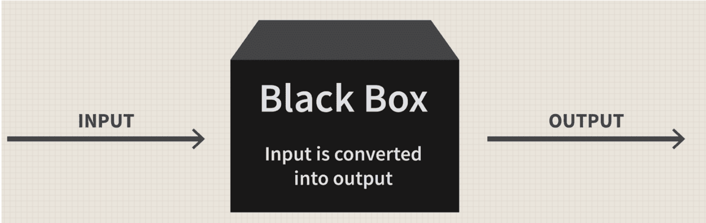
The derivative of the Objective Function is needed

There is more physics

BLACK BOX APPROACH

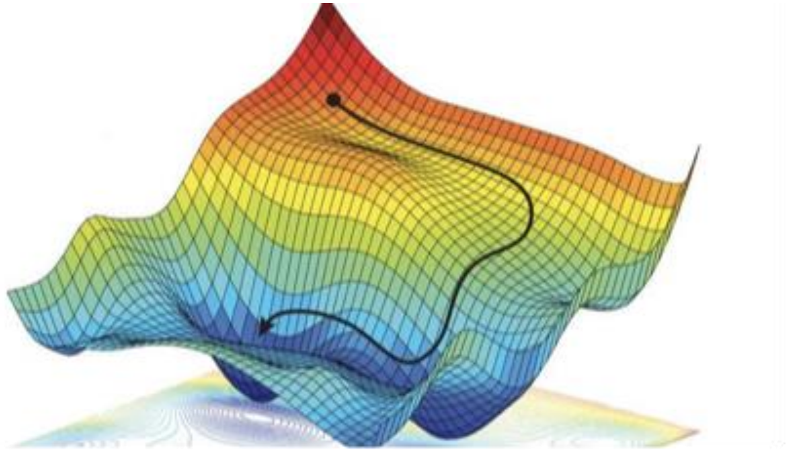


- **Deep Learning**
- **Bayesian-RBF**
- **Genetic Algo**
- **Physics informed ML**



INTRODUCTION

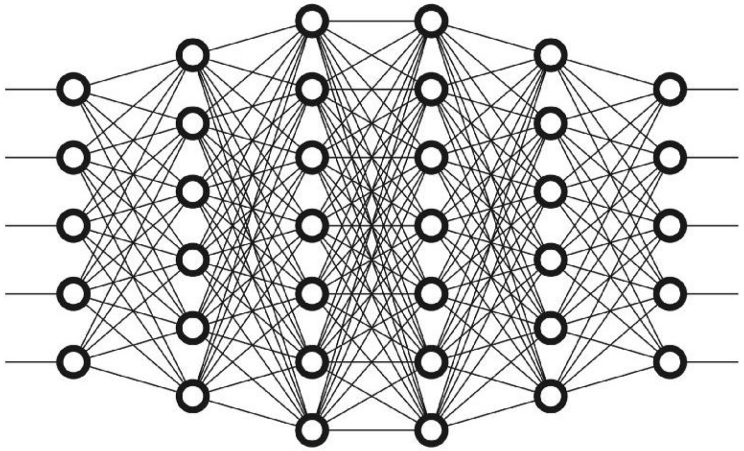
GRADIENT BASED



- Scales better with the number of DV
- We need to obtain first (second) derivative
- Get stuck in local minimum

- Used in the last stages of a design

BLACK BOX APPROACH



- Easier to implement
- Less DV but Higher Range
- Difficult to insert physics constraints

- Can be used in first stages of a design

OPTIMIZATION PROBLEM

$$
\begin{aligned}
\min_{\alpha} \quad & J(U(\alpha), X(\alpha)) \\
\text{subject to} \quad & U(\alpha) = G(U(\alpha), X(\alpha)) \\
& X(\alpha) = M(\alpha)\,.
\end{aligned}
$$

+ Geometrical Constraints

Where:

- J = Objective Function
- α = Design Variables: ffd-box
- U= Flow Variables
- X = Mesh

J can be anything that we can calculate from the flow solution: drag, lift, thrust, noise, pressure drop….

OPTIMIZATION CHAIN

Mesh Deformation

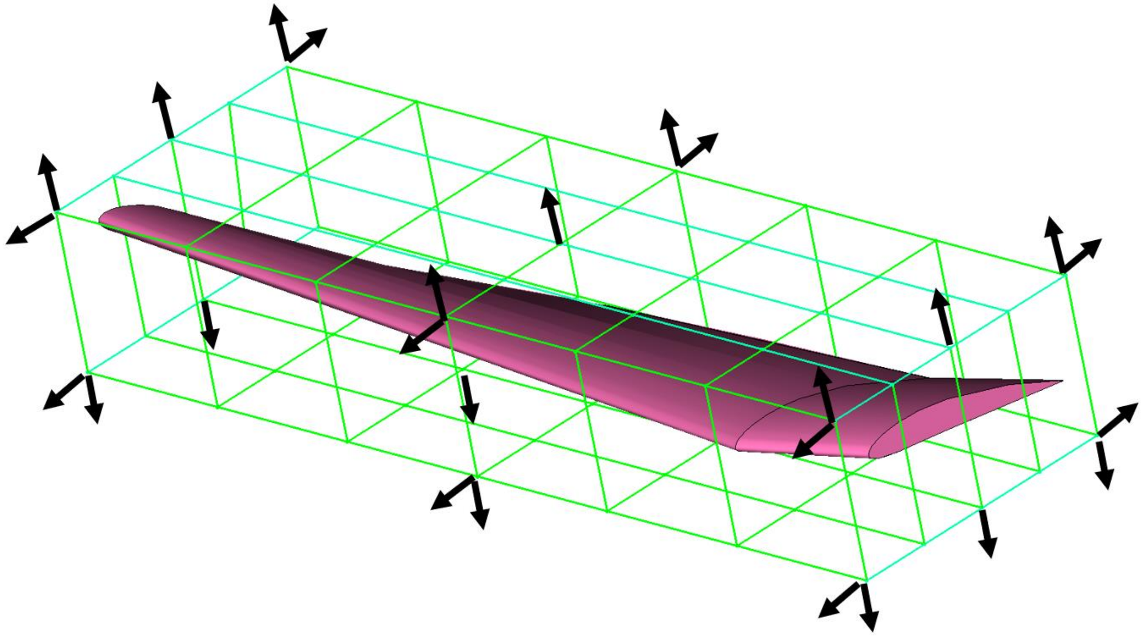
Geometry Parametrization

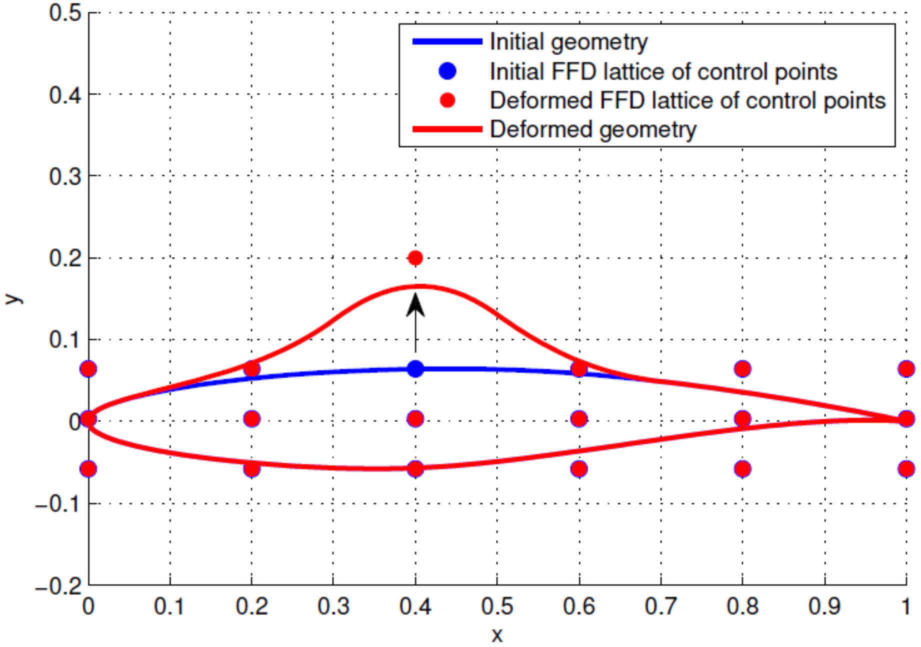
J Computation: RANS Simulation

Surface Sensitivity

- Finite Difference
- Continuous Adjoint
- DISCRETE ADJOINT

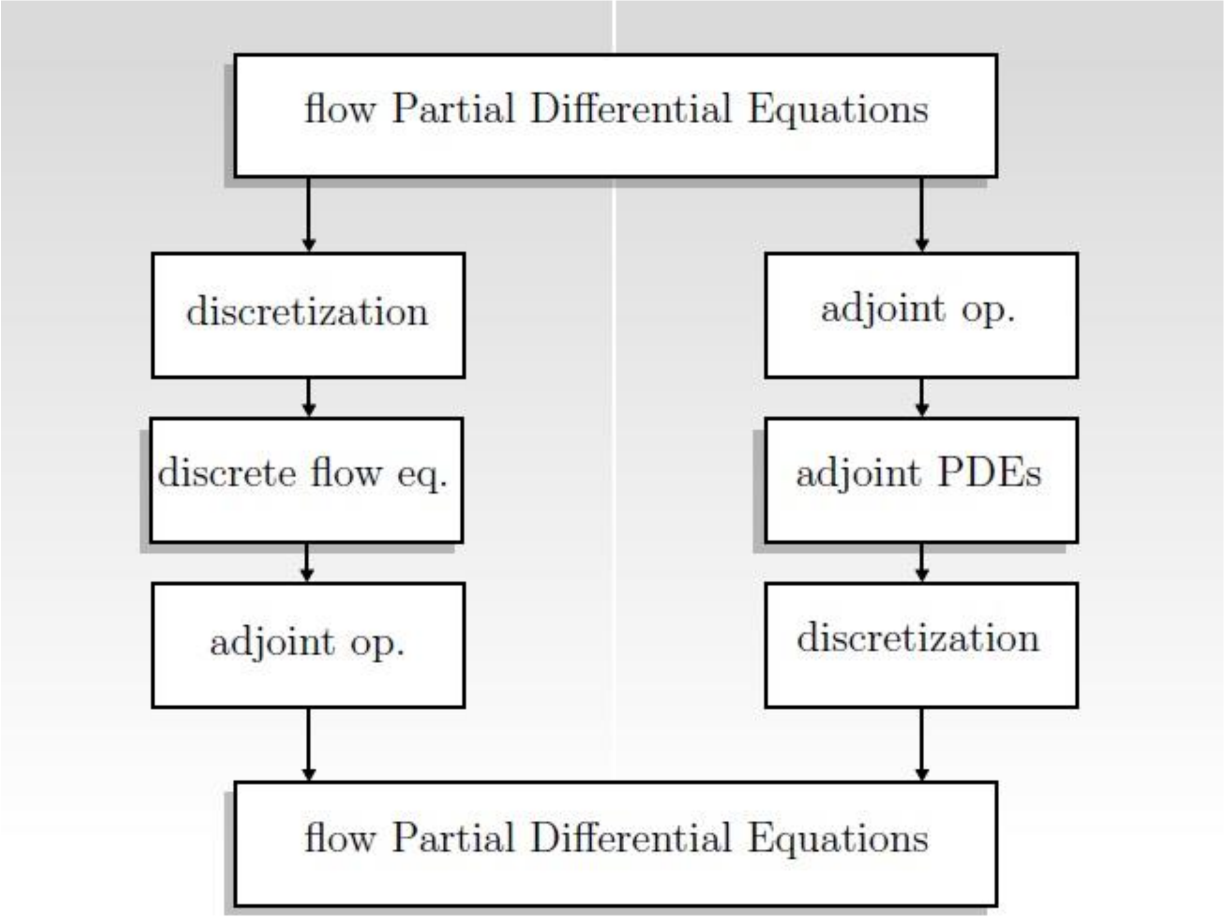
FREE FORM DEFORMATION
Plastic Box
Interpolation method
L x M x K sub-control volume





THE POSITION OF THE VERTICES OF THE FFD BOX ARE THE DESIGN VARIABLES

CONTINUOS VS DISCRETE ADJOINT



CONTINUOS ADJOINT

C. The Continuous Adjoint Equations

Eqn. 20 can now be introduced into Eqn. 12 to produce

$$\delta\mathcal{J} = \delta J + \frac{1}{\mathbb{T}}\int_{t_o}^{t_f}\int_{\Omega}\Psi^{\mathsf{T}}\frac{\partial}{\partial t}(\delta U)\,d\Omega\,dt + \frac{1}{\mathbb{T}}\int_{t_o}^{t_f}\int_{\Omega}\Psi^{\mathsf{T}}\nabla\cdot\left(\vec{A}^c - \bar{\bar{I}}\vec{u}_{\Omega} - \mu_{tot}^k\vec{A}^{vk}\right)\delta U\,d\Omega\,dt$$
$$-\frac{1}{\mathbb{T}}\int_{t_o}^{t_f}\int_{\Omega}\Psi^{\mathsf{T}}\nabla\cdot\mu_{tot}^k\bar{\bar{D}}^{vk}\delta(\nabla U)\,d\Omega\,dt - \frac{1}{\mathbb{T}}\int_{t_o}^{t_f}\int_{\Omega}\Psi^{\mathsf{T}}\frac{\partial Q}{\partial U}\delta U\,d\Omega\,dt. \qquad (21)$$

FROZEN TURBULENCE ASSUMPTION

Consider a perturbation to the flow equations while assuming constant, or frozen, viscosity, $(\delta\mu_{tot}^k = 0)$:

$$\begin{aligned}
\delta\mathcal{R}(U,\nabla U) &= \delta\left[\frac{\partial U}{\partial t} + \nabla\cdot\vec{F}^c_{ale} - \nabla\cdot\mu_{tot}^k\vec{F}^{vk} - Q\right] \\
&= \delta\left[\frac{\partial U}{\partial t} + \nabla\cdot\vec{F}^c - \nabla\cdot(U\otimes\vec{u}_{\Omega}) - \nabla\cdot\mu_{tot}^k\vec{F}^{vk} - Q\right] \\
&= \frac{\partial}{\partial t}(\delta U) + \nabla\cdot\left(\frac{\partial\vec{F}^c}{\partial U}\delta U\right) - \nabla\cdot\left[\frac{\partial(U\otimes\vec{u}_{\Omega})}{\partial U}\delta U\right] - \nabla\cdot\mu_{tot}^k\left[\frac{\partial\vec{F}^{vk}}{\partial U}\delta U + \frac{\partial\vec{F}^{vk}}{\partial(\nabla U)}\delta(\nabla U)\right] - \frac{\partial Q}{\partial U}\delta U \\
&= \frac{\partial}{\partial t}(\delta U) + \nabla\cdot\left(\vec{A}^c - \bar{\bar{I}}\vec{u}_{\Omega} - \mu_{tot}^k\vec{A}^{vk}\right)\delta U - \nabla\cdot\mu_{tot}^k\bar{\bar{D}}^{vk}\delta(\nabla U) - \frac{\partial Q}{\partial U}\delta U, \qquad (18)
\end{aligned}$$

where

$$\left.\begin{aligned}
\vec{A}^c &= \left(A_x^c, A_y^c, A_z^c\right), & A_i^c &= \left.\frac{\partial\vec{F}_i^c}{\partial U}\right|_{U(x,y,z)} \\
\vec{A}^{vk} &= \left(A_x^{vk}, A_y^{vk}, A_z^{vk}\right), & A_i^{vk} &= \left.\frac{\partial\vec{F}_i^{vk}}{\partial U}\right|_{U(x,y,z)} \\
\bar{\bar{D}}^{vk} &= \begin{pmatrix} D_{xx}^{vk} & D_{xy}^{vk} & D_{xz}^{vk} \\ D_{yx}^{vk} & D_{yy}^{vk} & D_{yz}^{vk} \\ D_{zx}^{vk} & D_{zy}^{vk} & D_{zz}^{vk}\end{pmatrix}, & D_{ij}^{vk} &= \left.\frac{\partial\vec{F}_i^{vk}}{\partial(\partial_j U)}\right|_{U(x,y,z)}
\end{aligned}\right\}\quad i,j = 1\ldots3,\quad k = 1,2, \qquad (19)$$

DISCRETE ADJOINT

$$\begin{aligned} \min_{\alpha} \quad & J(U(\alpha), X(\alpha)) \\ \text{subject to} \quad & U(\alpha) = G(U(\alpha), X(\alpha)) \\ & X(\alpha) = M(\alpha)\,. \end{aligned}$$

LAGRANGE EXPRESSION

$$L(\alpha, U, X, \Delta_f, \Lambda_g) = J(U, X, \alpha) + [G(U, X) - U]^T \Lambda_f + [M(\alpha) - X]^T \Lambda_g$$

DIFFERENTIATION

$$\frac{\mathrm{d}L}{\mathrm{d}\alpha} = \frac{\mathrm{d}J}{\mathrm{d}\alpha} + \cancel{\left[\frac{\partial U}{\partial \alpha}\right]^T} \left\{ \frac{\partial J^T}{\partial U} + \frac{\partial G^T}{\partial U}\Lambda_f - \Lambda_f \right\} + \cancel{\left[\frac{\partial X}{\partial \alpha}\right]^T} \left\{ \frac{\partial J^T}{\partial X} + \frac{\partial G^T}{\partial X}\Lambda_f - \Lambda_g \right\} + \frac{\mathrm{d}M^T}{\mathrm{d}\alpha}\Lambda_g$$

ADJOINT SYSTEM

$$\Lambda_f = \frac{\partial J^T}{\partial U} + \frac{\partial G^T}{\partial U}\Lambda_f$$

$$\Lambda_g = \frac{\partial J^T}{\partial X} + \frac{\partial G^T}{\partial X}\Lambda_f$$

$$\frac{\mathrm{d}L}{\mathrm{d}\alpha}^T = \frac{\mathrm{d}J}{\mathrm{d}\alpha}^T = \frac{\mathrm{d}M(\alpha)^T}{\mathrm{d}\alpha}\Lambda_g .$$

LET'S GIVE A NAME TO EACH TERM

$$w = ((a+b) * (c-d))^2$$

$$\begin{aligned} t_1 &= a + b \\ t_2 &= c - d \\ t_3 &= t_1 * t_2 \\ w &= t_3^2. \end{aligned}$$

Every complex code in the end is just a long sequence of simple operations

$$f(x) = Q_m \circ \Phi_l \circ \Phi_{l-1} \circ \cdots \circ \Phi_2 \circ \Phi_1 \circ P_n^T(x),$$

The sequence of simple operations is registered,
We know the first order derivative of each of them
The chain rule is applied

$$\frac{d}{dx}\left[(f(x))^n\right] = n(f(x))^{n-1} \cdot f'(x)$$

$$\frac{d}{dx}\left[f(g(x))\right] = f'(g(x))g'(x)$$

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = Q_m A_l A_{l-1} \ldots A_2 A_1 P_n^T.$$

TRANSONIC AIRFOIL OPTIMIZATION

Freestream conditions:

- M=0.796
- P= 101325
- Re= 12.56 E6
- $\alpha_0 = 3°$

Numerical methods:

- JST
- SA turbulence model

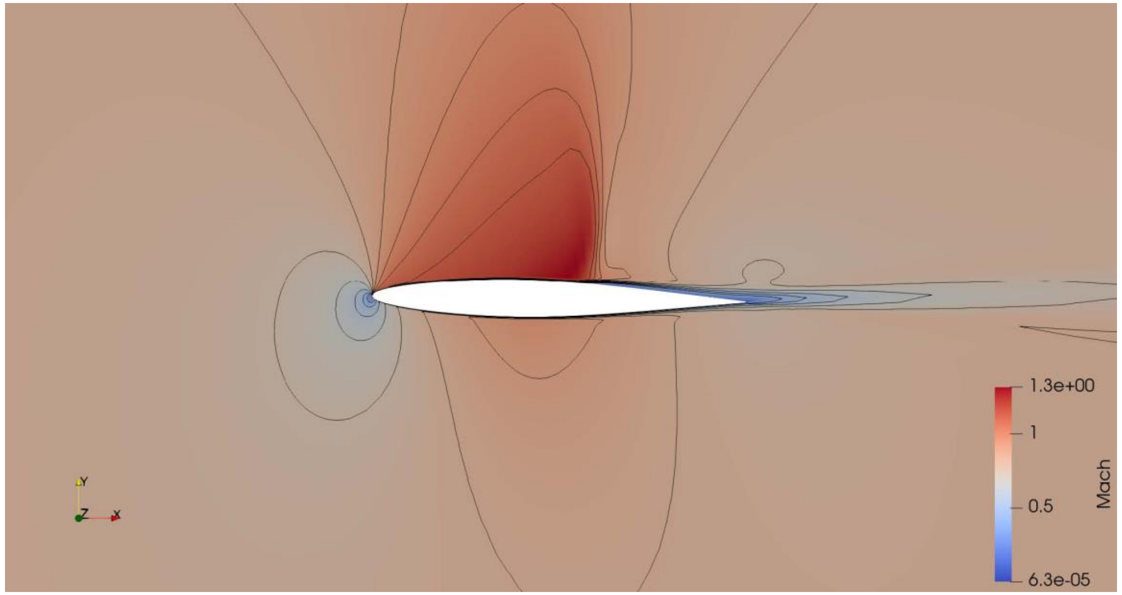


Fig.1 Transonic NACA64A010

AIRFOIL: Optimization

Adjoint Optimization:

- Reduce the Drag
- Preserve the Lift
- Fixed α_0
- Single FFD box
- 20 DVs free in y direction
- RBF with Wendland C2

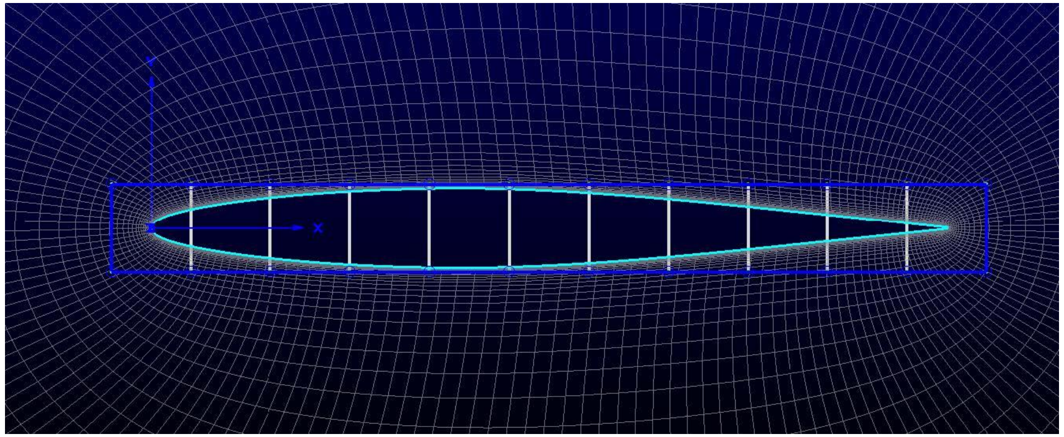


Fig.2 Pitching airfoil: FFD-Box

AIRFOIL: Results

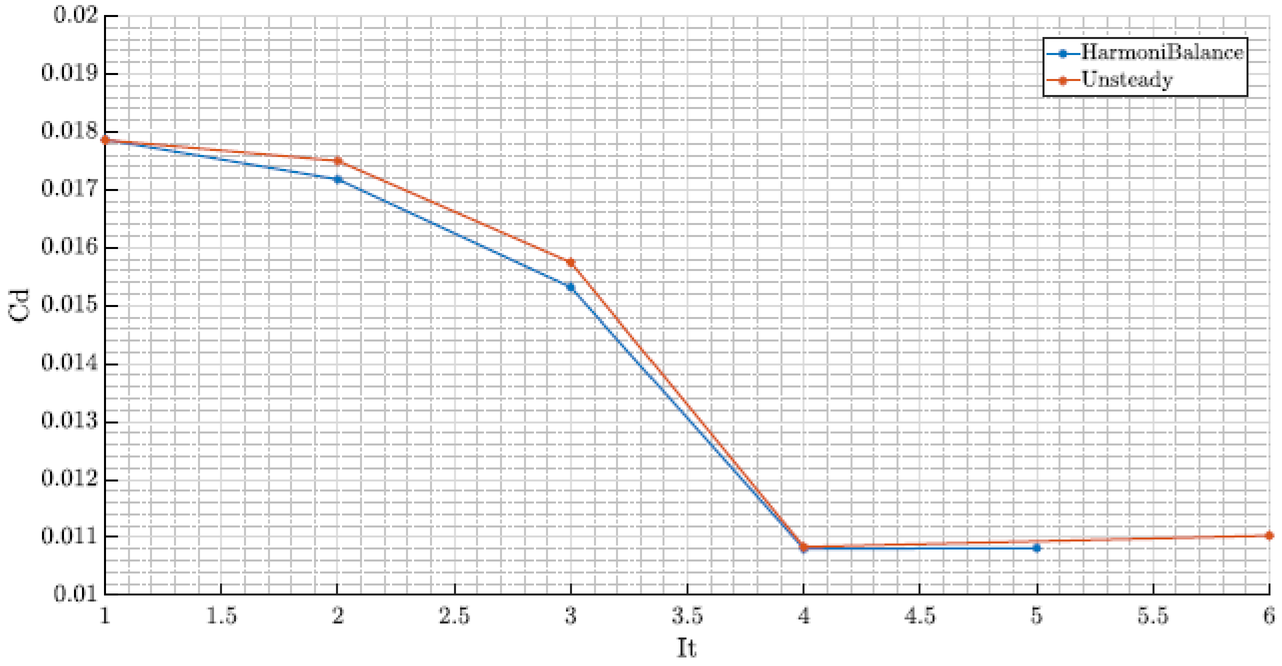


Fig.3 Optimization history

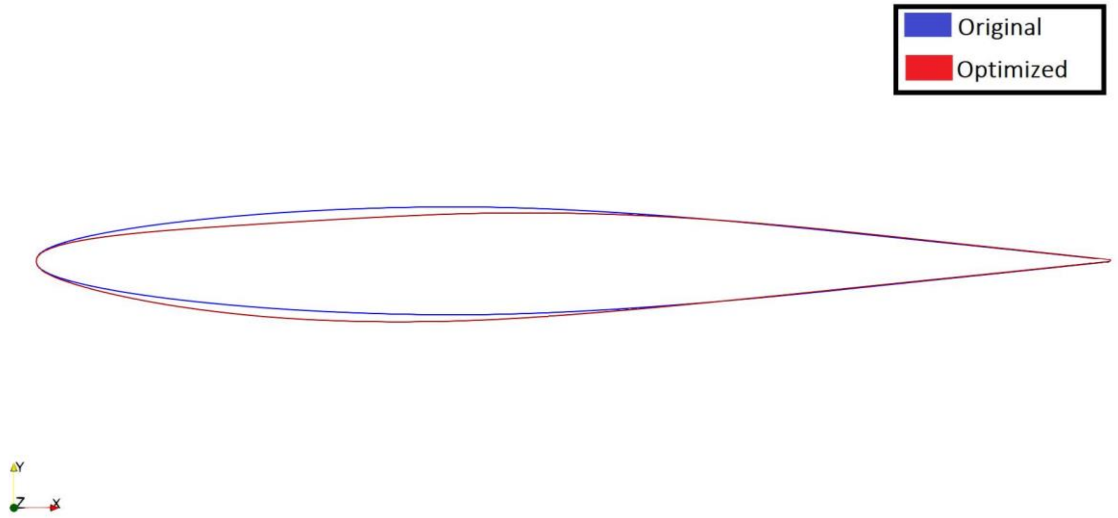


Fig.4 Optimized shape

AIRFOIL: Results

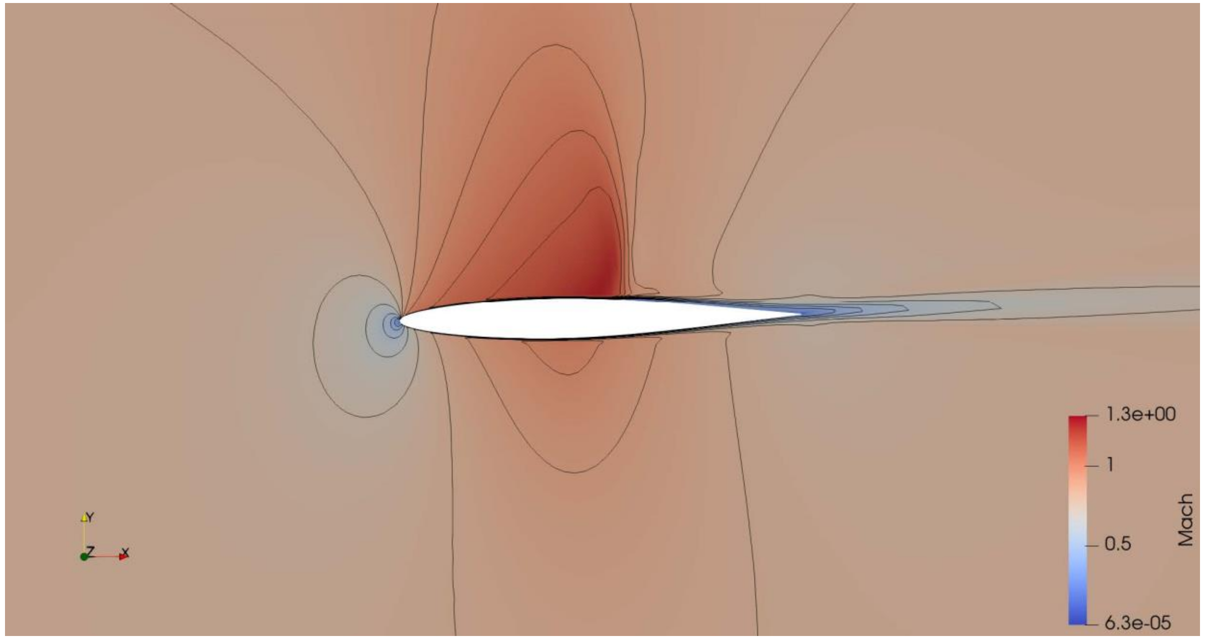


Fig.8 Original Mach field

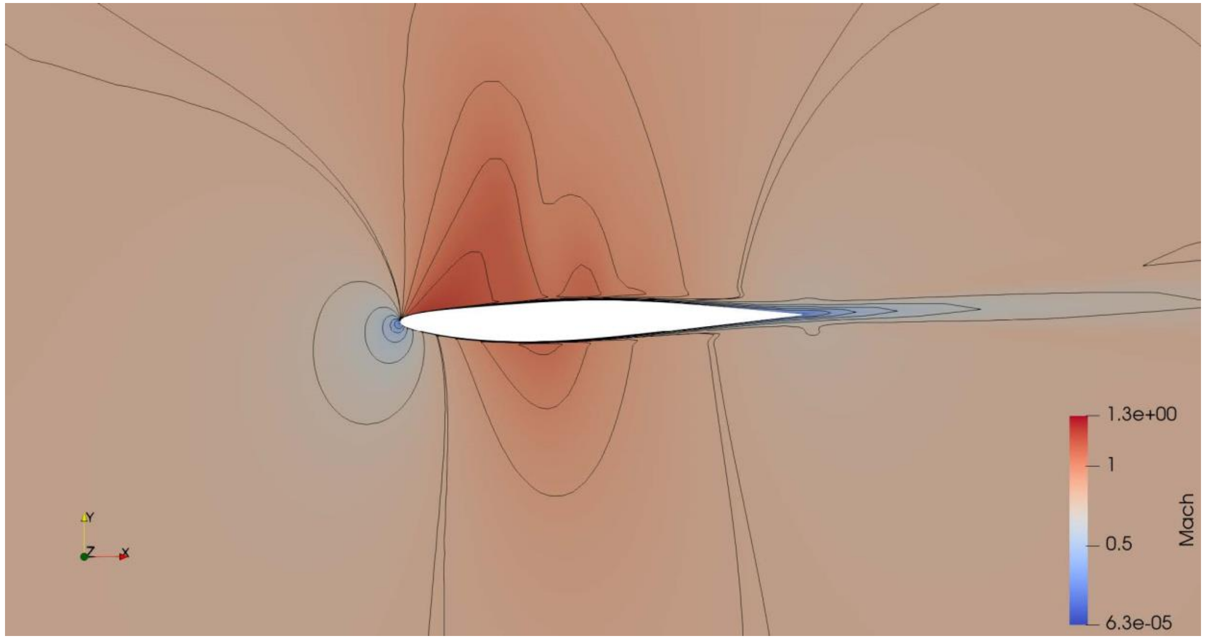
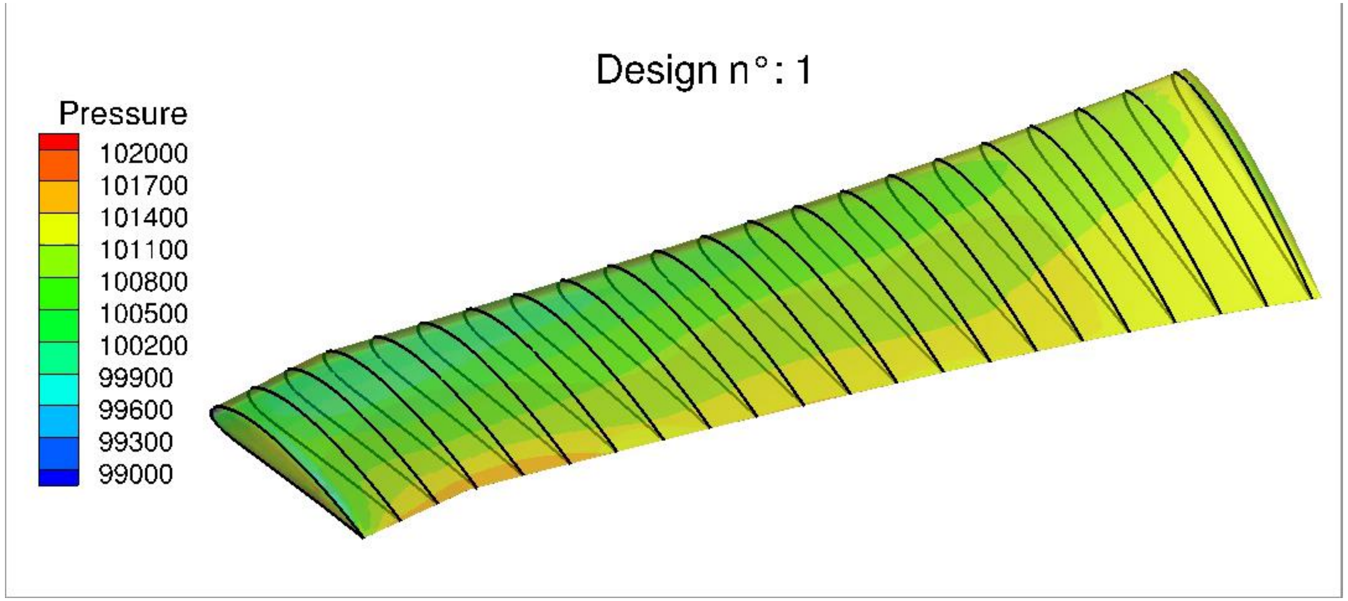
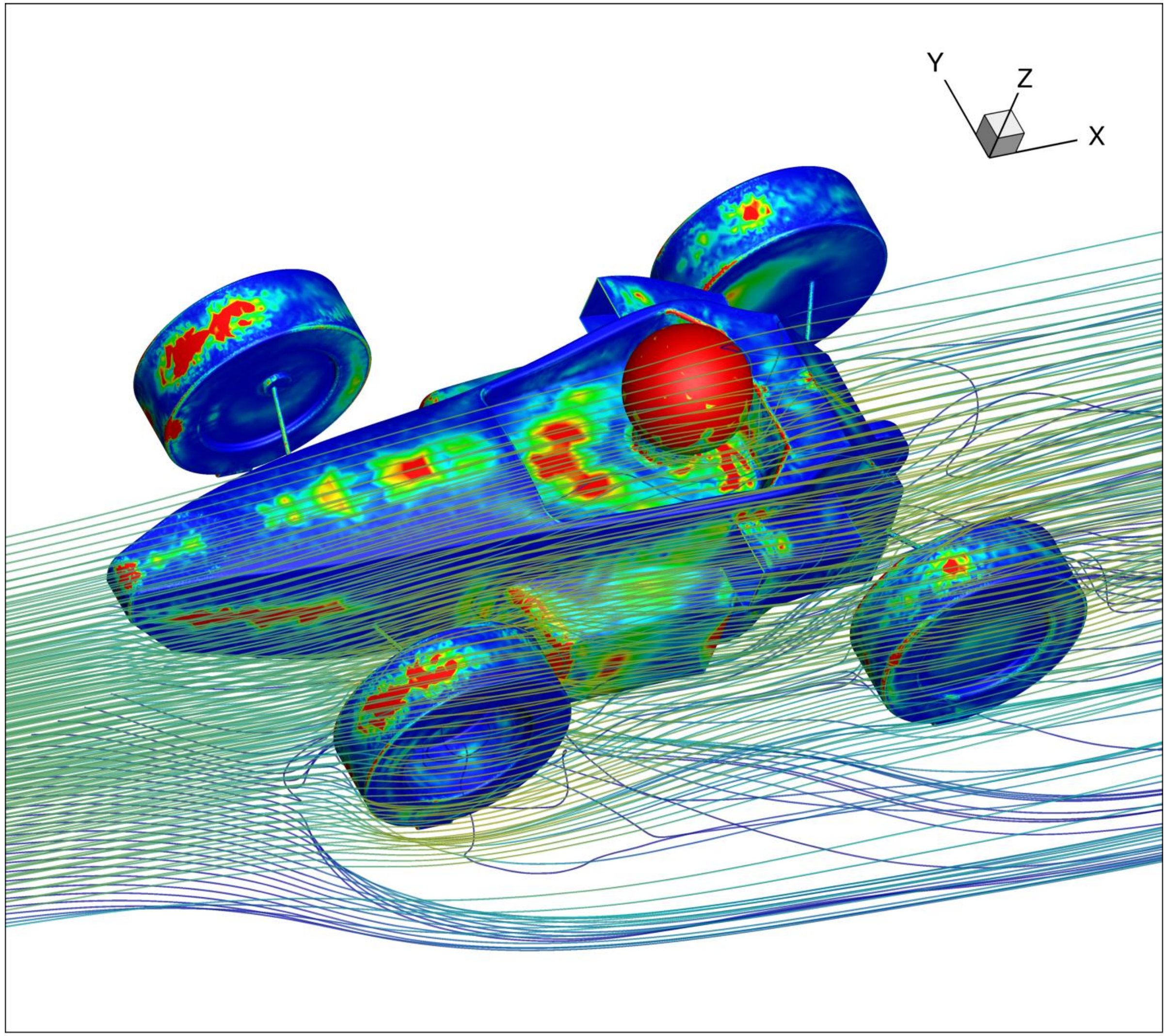


Fig.9 Optimized Mach field

NOISE MINIMIZATION



SURFACE SENSITIVITY



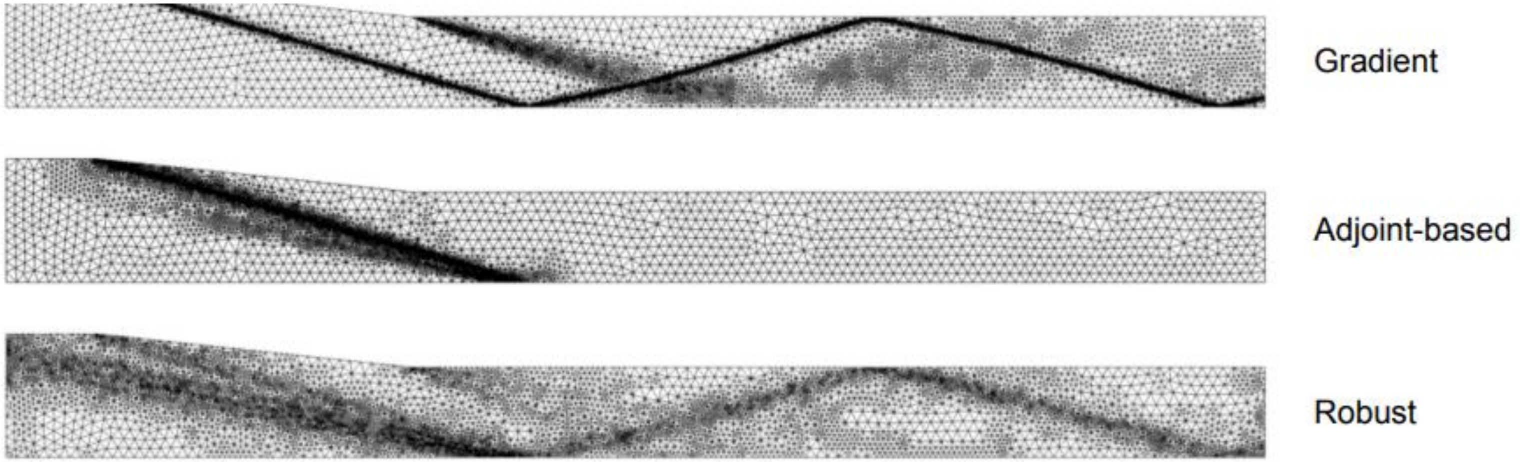
**WHAT MEANS
A POSITIVE VALUE?**

Surface sens= dJ/dn

CONCLUSIONS

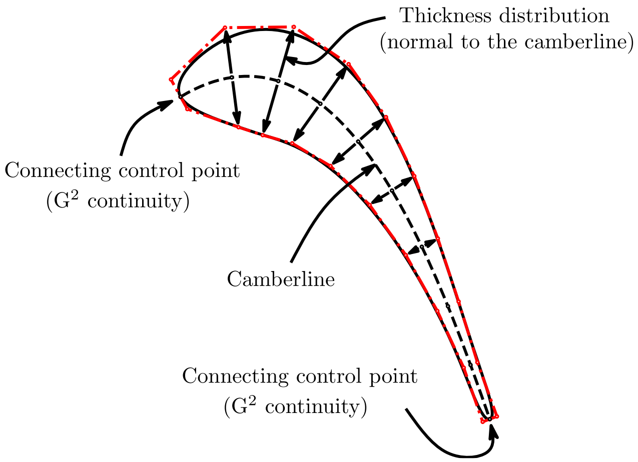
- GRADIENT BASED
- LOCAL MINIMUM
- RESULTS DEPENDS ON THE STARTING POINT
- COST DOES NOT SCALE WITH NUMBER OF DV
- COMPUTATIONALLY EXPENSIVE
- HIGH QUALITY RESULTS FOR SEVERAL INDUSTRIAL APPLICATIONS
- SURFACE SENSITIVITY IS USEFULL

- GOAL ORIENTED MESH ADAPTATION



- GEOMETRIC PARAMETRIZATION

ParaBlade
A python based blade parametrization tool



THANKS FOR THE ATTENTION

NACA 0012: FFD CONFIG

```
% -------------------- FREE-FORM DEFORMATION PARAMETERS -----------------------%
%
% Tolerance of the Free-Form Deformation point inversion
FFD_TOLERANCE= 1E-10
%
% Maximum number of iterations in the Free-Form Deformation point inversion
FFD_ITERATIONS= 100
%
% FFD box definition: 3D case (FFD_BoxTag, X1, Y1, Z1, X2, Y2, Z2, X3, Y3, Z3, X4, Y4, Z4,
%                              X5, Y5, Z5, X6, Y6, Z6, X7, Y7, Z7, X8, Y8, Z8)
%                     2D case (FFD_BoxTag, X1, Y1, 0.0, X2, Y2, 0.0, X3, Y3, 0.0, X4, Y4, 0.0,
%                              0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)
FFD_DEFINITION= (airfoil_box, -0.01, -0.2, 0.0, 1.01, -0.2, 0.0, 1.01, 0.2, 0.0, -0.01, 0.2, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)
%
% FFD box degree: 3D case (x_degree, y_degree, z_degree)
%                 2D case (x_degree, y_degree, 0)
FFD_DEGREE= (10, 1, 0)
%
% There is a symmetry plane (j=0) for all the FFD boxes (YES, NO)
FFD_SYMMETRY_PLANE= NO
%
% Surface grid continuity at the intersection with the faces of the FFD boxes.
% To keep a particular level of surface continuity, SU2 automatically freezes the right
% number of control point planes (NO_DERIVATIVE, 1ST_DERIVATIVE, 2ND_DERIVATIVE, USER_INPUT)
FFD_CONTINUITY= 1ST_DERIVATIVE
% ----------------------- DESIGN VARIABLE PARAMETERS -------------------------%
%
% Kind of deformation (FFD_SETTING, HICKS_HENNE, HICKS_HENNE_NORMAL, PARABOLIC,
%                      HICKS_HENNE_SHOCK, NACA_4DIGITS, DISPLACEMENT, ROTATION,
%                      FFD_CONTROL_POINT, FFD_DIHEDRAL_ANGLE, FFD_TWIST_ANGLE,
%                      FFD_ROTATION)
DV_KIND= FFD_SETTING
%
% Marker of the surface in which we are going apply the shape deformation
DV_MARKER= (AIRFOIL)
%
% Parameters of the shape deformation
%     - HICKS_HENNE_FAMILY ( Lower(0)/Upper(1) side, x_Loc )
%     - NACA_4DIGITS ( 1st digit, 2nd digit, 3rd and 4th digit )
%     - PARABOLIC ( 1st digit, 2nd and 3rd digit )
%     - DISPLACEMENT ( x_Disp, y_Disp, z_Disp )
%     - ROTATION ( x_Orig, y_Orig, z_Orig, x_End, y_End, z_End )
DV_PARAM= ( airfoil_box , 1, 0, 0, 1.0, 1.0, 0.0)
%
% Mesh input file
MESH_FILENAME= mesh_original.su2
%
% Mesh input file format (SU2, CGNS, NETCDF_ASCII)
MESH_FORMAT= SU2
%
% Mesh output file
MESH_OUT_FILENAME= mesh_ffd.su2
```

SU2_DEF NacaOpt.cfg

After:

DV_KIND: FFD_CONTROL_POINT

NACA 0012: MESH DEF AND GEO

```
% ------------------------ GEOMETRY EVALUATION PARAMETERS ----------------------%
%
% Marker(s) of the surface where geometrical based function will be evaluated
GEO_MARKER= ( AIRFOIL )
%
% Description of the geometry to be analyzed (AIRFOIL, WING, FUSELAGE)
GEO_DESCRIPTION= AIRFOIL
%
% Coordinate of the stations to be analyzed
GEO_LOCATION_STATIONS= (0.1 , 0.2 , 0.3 , 0.4 , 0.5)
%
% Geometrical bounds (Y coordinate) for the wing geometry analysis or
% fuselage evaluation (X coordinate)
GEO_BOUNDS= (0.08, 1.128)
%
% Plot loads and Cp distributions on each airfoil section
GEO_PLOT_STATIONS= NO
%
% Number of section cuts to make when calculating wing geometry
GEO_NUMBER_STATIONS= 25
%
% Geometrical evaluation mode (FUNCTION, GRADIENT)
GEO_MODE= GRADIENT
%
% ------------------------ GRID DEFORMATION PARAMETERS ------------------------%
%
%
% Linear solver or smoother for implicit formulations (FGMRES, RESTARTED_FGMRES, BCGSTAB)
DEFORM_LINEAR_SOLVER= FGMRES
%
% Number of smoothing iterations for FEA mesh deformation
DEFORM_LINEAR_SOLVER_ITER= 1000
%
% Number of nonlinear deformation iterations (surface deformation increments)
DEFORM_NONLINEAR_ITER= 1
%
% Print the residuals during mesh deformation to the console (YES, NO)
DEFORM_CONSOLE_OUTPUT= YES
%
% Minimum residual criteria for the linear solver convergence of grid deformation
DEFORM_LINEAR_SOLVER_ERROR= 1E-10
%
% Type of element stiffness imposed for FEA mesh deformation (INVERSE_VOLUME,
%                                           WALL_DISTANCE, CONSTANT_STIFFNESS)
DEFORM_STIFFNESS_TYPE= INVERSE_VOLUME
%
```

NACA 0012: J AND CONSTRAINTS

```
% Optimization objective function with scaling factor, separated by semicolons.
% To include quadratic penalty function: use OPT_CONSTRAINT option syntax within the OPT_OBJECTIVE list.
% ex= Objective * Scale
OPT_OBJECTIVE= DRAG
%
% Optimization constraint functions with pushing factors (affects its value, not the gradient
% in the python scripts), separated by semicolons
% ex= (Objective = Value ) * Scale, use '>','<','='
OPT_CONSTRAINT= (LIFT > 0.2849) * 0.01; (MOMENT_Z=0.0)* 0.01; (AIRFOIL_AREA> 0.04)
%
% Factor to reduce the norm of the gradient (affects the objective function and gradient in the python scripts)
% In general, a norm of the gradient ~1E-6 is desired.
OPT_GRADIENT_FACTOR= 1E-4
%
% Factor to relax or accelerate the optimizer convergence (affects the line search in SU2_DEF)
% In general, surface deformations of 0.01'' or 0.0001m are desirable
OPT_RELAX_FACTOR= 1E2
%
% Maximum number of optimizer iterations
OPT_ITERATIONS=100
%
% Requested accuracy
OPT_ACCURACY= 1E-10
%
%
% Optimization design variables, separated by semicolons
DEFINITION_DV= ( 11, 1.0 | AIRFOIL | airfoil_box, 1, 0, 0, 1.0, 1.0, 0.0 );( 11, 1.0 | AIRFOIL | airfoil_box, 1, 1, 0, 1.0, 1.0, 0.0
foil_box, 9, 0, 0, 1.0, 1.0, 0.0 );( 11, 1.0 | AIRFOIL | airfoil_box, 9, 1, 0, 1.0, 1.0, 0.0 )
```

NACA 0012: OUTPUTS

```
----------------- FFD technique (parametric -> cartesian) ---------------
Checking FFD box dimension.
Checking FFD box intersections with the solid surfaces.
SU2 is fixing the planes to maintain a continuous surface.
Update cartesian coord        | FFD box:| BLADE_BOX. Max Diff: 0.00210519.
Writing a Paraview file of the FFD boxes.
Writing a Tecplot file of the FFD boxes.

------------------- Volumetric grid deformation (ZONE 0) ----------------
Performing the deformation of the volumetric grid.
```

```
Sequential Least SQuares Programming (SLSQP) parameters:
Number of design variables: 18 ( 18 )
Objective function scaling factor: [1.0]
Maximum number of iterations: 100
Requested accuracy: 1e-15
Initial guess for the independent variable(s): [0.0, 0.0, 0.0, 0.0,
Lower and upper bound for each independent variable: [(-10000000.0,

  NIT     FC          OBJFUN            GNORM
    1      1     3.096491E-06     4.788515E-06
    2      2     3.050610E-06     4.631389E-06
    3      3     2.814069E-06     3.922372E-06
    4      4     2.175708E-06     2.857431E-06
    5      5     1.999863E-06     3.095495E-06
```