THE OPTIMIZATION PROBLEM

$$I(\boldsymbol{\beta}, \mathbf{u}(\boldsymbol{\beta})) \rightarrow min \quad ,$$

I: Objective Function
$\boldsymbol{\beta}$: Design Parameters
$\mathbf{u}$: Field Unknowns (e.g. Flowfield)

Subject To:

- PDE Constraints:

$$\mathbf{R}(\mathbf{u}) = 0$$

- Geometric Constraints:

$$\mathbf{g}(\boldsymbol{\beta}) \geq 0$$

- Physical Constraints:

$$\mathbf{h}(\mathbf{u}) \geq 0$$

EXAMPLES (1)

Functional I:

- Inviscid Drag (CFD):
  $I = \int_\Gamma p n_x d\Gamma$
- Prescribed Pressure (CFD):
  $I = \int_\Gamma (p - p_0)^2 d\Gamma$
- Weight (CSD):
  $I = \int_\Omega \rho d\Omega$
- Uniformity of Magnetic Field (CEM):
  $I = \int_\Omega (\mathbf{B} - \mathbf{B}_0)^2 d\Omega$

PDE Constraints $R(\mathbf{u})$:

- CFD: Euler/Navier-Stokes Equations
- CSD: Elasticity/Plasticity Equations
- CEM: Maxwell Equations
- ...

EXAMPLES (2)

Geometric Constraints $g(\boldsymbol{\beta})$:

- CFD: Wing Area Cross-Section (Stress, Fuel)
  $A > A_0$
- CFD: Trailing Edge Thickness (Cooling)
  - $w > w_0$
- CSD: Width (Manufacturability)
  $w > w_0$

Physical Constraints $h(\mathbf{u})$:

- CFD: Constrained Negative Pressure Gradient
  $\mathbf{s} \cdot \nabla p > p g_0$
- CFD: Constrained Pressure (Cavitation)
  $p > p_0$
- CFD: Constrained Shear Stress (Haemolysis)
  $|\tau| < \tau_0$
- CSD: Constrained Stress (Failure)
  $|\sigma| < \sigma_0$

MULTIPOINT OPTIMIZATION

Typical: Optimal for Single Point $\Rightarrow$ Bad for Other Points

Classic Example: Korn Airfoil

Approach: Multipoint Optimization

$$I = \sum_i \gamma_i I(Ma_i, \alpha_i)$$

Remarks:

- Additive Cost Function
- $\Rightarrow$ Gradient Evaluations Also Additive
- Apt for Low-Grain Parallel Processing
- Current Research Topic:
  - Robust Design;
  - Number of Design Points Required;
  - Optimal Choice of γ_i;

DESIGN VARIABLE PARAMETRIZATION

Parametrization: For Each Design Variable:

- Variable: $\beta^i_{min} \leq \beta^i \leq \beta^i_{max}$
- Define: $0 \leq \alpha^i \leq 1$

$\Rightarrow$ Instantiation:

$$\beta^i = (1 - \alpha^i)\beta^i_{min} + \alpha^i \beta^i_{max}$$

$\Rightarrow$

$$I(\boldsymbol{\beta}) = I(\boldsymbol{\beta}(\boldsymbol{\alpha}))$$

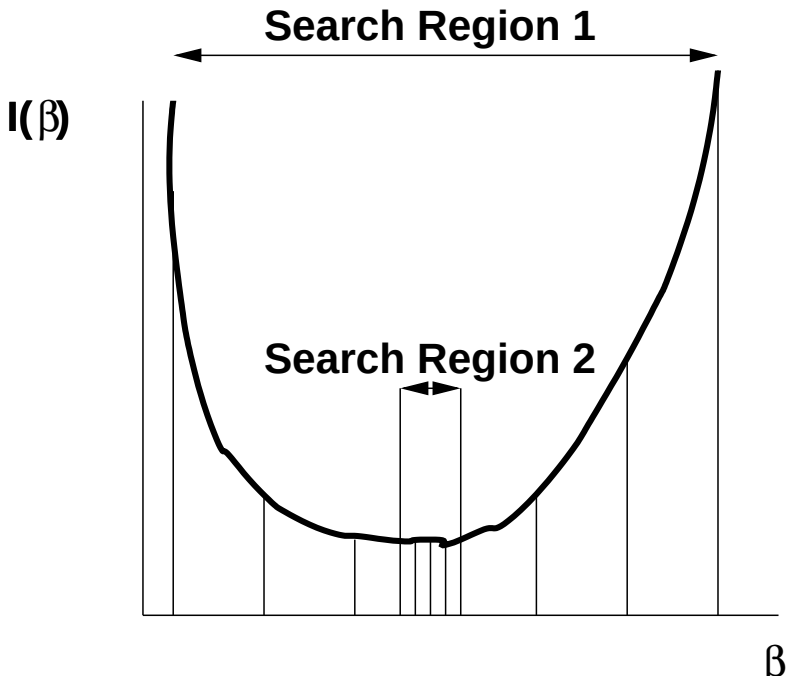
Work Only With α^i:

- Non-Dimensional
- Bounded $[0, 1]$
- Abstract

OPTIMIZATION TECHNIQUES

- Exhaustive Search
- Genetic
- Gradient-Based
    - Sensitivity Equations
    - Finite Difference
    - Response Surface
    - Adjoint

RECURSIVE EXHAUSTIVE SEARCH (1)



RECURSIVE EXHAUSTIVE SEARCH (2)

Key Ideas:

- Cover Parameter Space Uniformly
- Recursively Refine Regions of Interest

Pros:

- Completely General
- Suited for Rough Functionals
- Global Min Reachable

Cons:

- $O(n_d^N)$ Evaluations of $I \Rightarrow$ Extremely Slow
- CPU = f(n_d^N), But Accuracy = f(n_d)

RECURSIVE EXHAUSTIVE SEARCH (3)

Basic Steps:

- Define: n_d, $h = 1/n_d$
- `while`: $h > h_min$:
- $\forall$ Combinations of α_i: Evaluate $I(\boldsymbol{\beta}(\boldsymbol{\alpha}))$
- Retain Combination(s) $\boldsymbol{\alpha}_{opt}$ With Lowest I;
- Define New Search Range: $[\boldsymbol{\alpha}_{opt} - h/2, \boldsymbol{\alpha}_{opt} + h/2]$
- Define New Interval Size: $h := h/n$

  `end while`

GENETIC ALGORITHMS (1)

Key Ideas:

- Fitness Measure I
- Chromosome Coding
- Selection
- Crossover and Mutation

Pros:

- Completely General
- Suited for Rough Functionals
- Global Min Reachable

Cons:

- $O(N^2)$ Evaluations of $I \Rightarrow$ Very Slow
- Speed = f(Crossover/Mutation/Selection)

GENETIC ALGORITHMS (2)

Input Parameters Required:

- Fitness Measure (Objective Function)
- Chromosomes: $\alpha^i, i = 1, N$
- Population: $M > O(2N)$
- Percentage of Best Individuals Kept:
  $c_k = O(10\%)$
- Cutoff for Non-Reproduction of Worst Individuals:
  $c_c = O(75\%)$
- Mutation Frequency: $c_m = O(0.25/N)$

GENETIC ALGORITHMS (3)

Basic Steps per Generation:

- Evaluate Fitness Function $I(\boldsymbol{\beta}(\boldsymbol{\alpha}))$
- Sort in Ascending (Descending) Order
- Retain c_k Best Individuals for Next Generation
- `while`: Population Incomplete
    - Select (Randomly) a Pair i, j From c_c List
    - Obtain Pairing Factor (Randomly): $-\xi < \gamma < 1 + \xi$, $\xi = O(0.2)$
    - Chromosome of New Individual: $\boldsymbol{\alpha} = (1 - \gamma)\boldsymbol{\alpha}^i + \gamma\boldsymbol{\alpha}^j$
- `end while`

GENETIC ALGORITHMS (4)

Other Chromosome Combinations:

- Chromosome Splicing

$$\alpha_k = \alpha_k^i \quad , \quad 1 \leq k \leq l \quad ,$$
$$\alpha_k = \alpha_k^j \quad , \quad l \leq k \leq n \quad ,$$

- Arithmetic Pairing:

$$\boldsymbol{\alpha} = (1-\gamma)\boldsymbol{\alpha}^i + \gamma\boldsymbol{\alpha}^j$$

- Random Pairing:

$$\alpha_k = (1-\gamma_k)\alpha_k^i + \gamma_k\alpha_k^j$$

GENETIC ALGORITHMS (5)

Observation: Special Case of Single Optimum:

- Top Candidate Does Not Change Over Many Generations
  - Exception: Mutation (Slow)
- Top Candidates (e.g. Top 25% of Population) Become Uniform ⇒ Genetic Pool Collapses

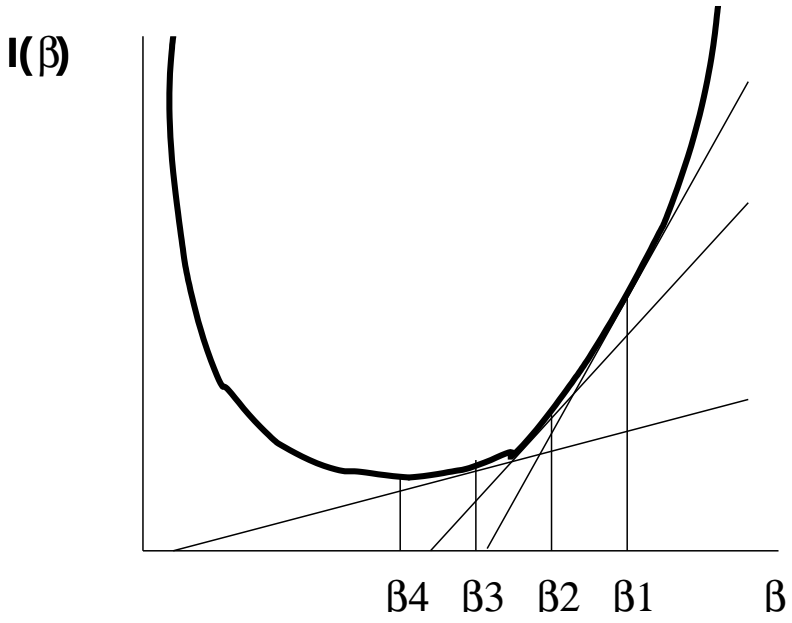
⇒ Detect and Correct Algorithm

- `while`:
  - `if`: For any i, j:
    $d_{ij} = |\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j| < \epsilon$
    Enlarge d_{ij} to Multiple of ϵ
  - `endif`
- `endwhile`

If Top Candidate Unchanged: Reduce ϵ

⇒ Automatic Stopping Criterion

GRADIENT-BASED ALGORITHMS (1)



GRADIENT-BASED ALGORITHMS (2)

Key Ideas:

$$I + \Delta I \approx I + I^T_{,\boldsymbol{\beta}} \Delta \boldsymbol{\beta}$$

$\Rightarrow$ Choose:

$$\Delta \boldsymbol{\beta} = -\lambda I_{,\boldsymbol{\beta}}$$

$\Rightarrow$

$$I + \Delta I = I - \lambda I^T_{,\boldsymbol{\beta}} I_{,\boldsymbol{\beta}} \leq I$$

Ways Of Computing Gradients:

- Sensitivity Equations
- Finite Differences
  - Explicit Evaluation of I
  - Automatic Differentiation (ADIFOR)
- Respose Surface
- Adjoint

GRADIENT-BASED ALGORITHMS (3)

Define: $\mathbf{G} = I_{,\boldsymbol{\beta}}$

Smoothed Gradients:

$$\tilde{\mathbf{G}} - \nabla\mu\nabla\tilde{\mathbf{G}} = \mathbf{G}$$

Choose:

$$\Delta\boldsymbol{\beta} = -\lambda\tilde{\mathbf{G}}$$

$\Rightarrow$

$$\Delta I = -\mathbf{G}\lambda\tilde{\mathbf{G}} == -\lambda\left[\tilde{\mathbf{G}} - \nabla\mu\nabla\tilde{\mathbf{G}}\right]\tilde{\mathbf{G}}$$

$\Rightarrow$

$$\Delta I = -\lambda\left[\tilde{\mathbf{G}}^2 + \mu\left(\nabla\tilde{\mathbf{G}}\right)^2\right]$$

$\Rightarrow$ Reduction in I $\forall\mu$ (!)

SENSITIVITY EQUATIONS

Key Ideas:

- Perform **Nonlinear** Stationary Solution $\Rightarrow \mathbf{u}$
- Differentiate w.r.t. Inflow Angles, Mach-nr., etc.
  $\Rightarrow \mathbf{s} = \mathbf{u}_{,\boldsymbol{\beta}}$
- Differentiate Fluxes and Boundary conditions
- Solve **Linear** Stationary Solution $\Rightarrow \mathbf{s}$

Pros:

- Fast Per Degree of Freedom (Linear)

Cons:

- Not General (Chain Rules, B.C.)
- May Stop at Local Min

FINITE DIFFERENCES

Key Ideas:

- $I(\boldsymbol{\beta}); I(\boldsymbol{\beta} + \Delta\boldsymbol{\beta}) \Rightarrow$ Gradient

Pros:

- General

Cons:

- Need $I(\boldsymbol{\beta} - \Delta\boldsymbol{\beta}), I(\boldsymbol{\beta} + \Delta\boldsymbol{\beta})$ for 2nd Order
- $O(2N)$ Evaluations of $I \Rightarrow$ Slow
- Stepsize (?)
- Not Suited for Rough Functionals
- May Stop at Local Min

COMPLEX VARIABLES

Key Ideas:

- $I(\boldsymbol{\beta} + ih) = I(\boldsymbol{\beta}) + ihI_{,\boldsymbol{\beta}} + HOT$
  $I_{,\boldsymbol{\beta}} \approx Im\left[I(\boldsymbol{\beta} + ih)\right]/h$

Pros:

- Faster than 2 Real Evaluations

Cons:

- Needs Re-Write of Codes (Complex Variables)
- $O(N)$ Evaluations of $I \Rightarrow$ Slow
- Stepsize (?)
- Not Suited for Rough Functionals
- May Stop at Local Min

AUTOMATIC DIFFERENTIATION (1)

Key Ideas:

- Perform Differentiation in Code
  (e.g. ADIFOR for FORTRAN Codes)
  `u = v * w` $\Rightarrow$ `du = dv* w + v * dw`

Pros:

- General

Cons:

- Not Suited for Rough Functionals
- Expensive in Memory
- Slow
- May Stop at Local Min

AUTOMATIC DIFFERENTIATION (2)

Complete Chain of Events:

- CAD-Data S From Design Variables $\boldsymbol{\beta}$
- surface/Volume Mesh Ω From CAD-data S
- Flow Solution $CFD(\Omega)$
- Cost Function I From $CFD(\Omega)$

Symbolically:

$$\boldsymbol{\beta} \rightarrow S \rightarrow \Omega \rightarrow CFD \rightarrow I$$

Practical Problems:

- Separate Codes
- General Codes

RESPONSE SURFACE

Key Ideas:

- Obtain $I(\boldsymbol{\beta})$ In Region of Interest
- Approximate $I(\boldsymbol{\beta})$ by Fitting a Surface
- Obtain Gradient
- Successively Reduce Region of Interest

Pros:

- General
- Better Suited for Rough Functionals

Cons:

- $O(N)$ Evaluations of $I \Rightarrow$ Slow
- May Stop at Local Min

GRADIENTS WITH APPROXIMATE COST FUNCTIONS

Key Ideas:

- Need Differences, Not Absolute Values
- Can Obtain from Lower Fidelity $I(\boldsymbol{\beta})$ (True ?)

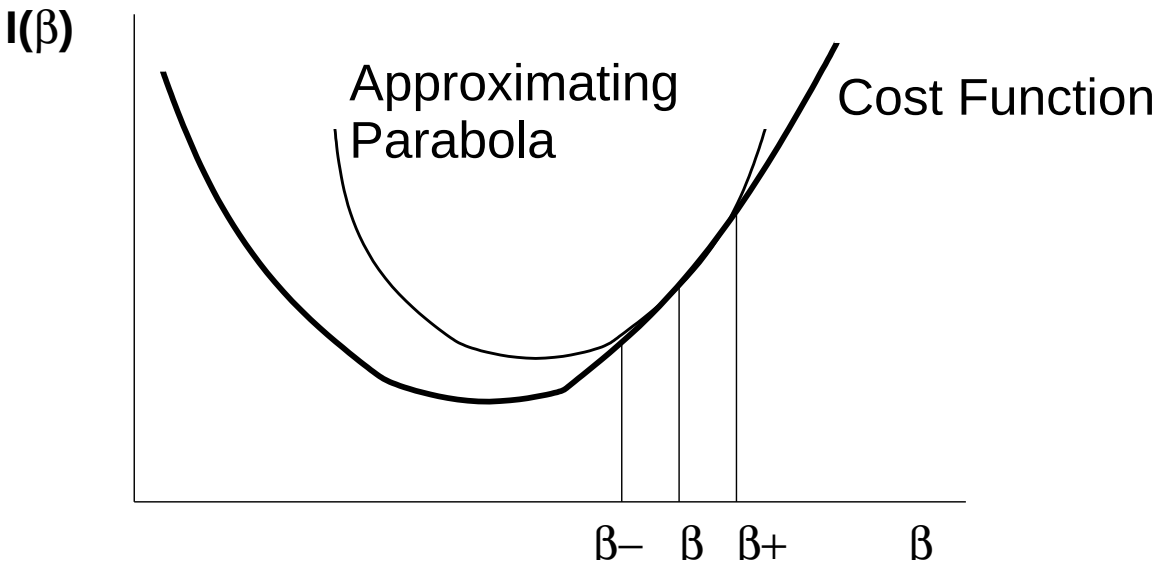
Approach:

- Evaluate Descent With High Fidelity $I(\boldsymbol{\beta})$
- Evaluate FD, CV, RS With Low Order $I(\boldsymbol{\beta})$

Alternatives for Low Order:

- Lower Physical Fidelity
  Potential, Euler, RANS, LES, ...
- Coarser Mesh
  (Same Physical Fidelity)

STEEPEST DESCENT (1)



Parabolic Extrapolation for Step Estimation

STEEPEST DESCENT (2)

Given:

- Current Objective Function: $I(\boldsymbol{\beta}(\boldsymbol{\alpha}_0))$
- Gradient: $I_{,\boldsymbol{\alpha}_0}$
- 1st Guess: Small $\lambda \Rightarrow \Delta\boldsymbol{\alpha} = -\lambda I_{,\boldsymbol{\alpha}_0}$

Attempt Quadratic Extrapolation:

$I_1 = I_0 + a\Delta\mathbf{u} + b\Delta\mathbf{u}^2 \quad , \quad I_{-1} = I_0 - a\Delta\mathbf{u} + b\Delta\mathbf{u}^2$

$a = (I_1 - I_{-1})/2\Delta\mathbf{u} \quad , \quad b = (I_1 - 2I_0 + I_{-1})/2\Delta\mathbf{u}^2$

$I_{,u} = 0 \quad \Rightarrow \quad \Delta\mathbf{u}_{min} = -a/2b \quad \Rightarrow$

$$\Delta\mathbf{u}_{min} = -\frac{I_1 - I_{-1}}{2(I_1 - 2I_0 + I_{-1})}\Delta\mathbf{u}$$

Use λ as $\Delta\mathbf{u}$

However: We Must Have: $\Delta\boldsymbol{\alpha} \cdot I_{,\boldsymbol{\alpha}_0} < 0$

STEEPEST DESCENT (3)

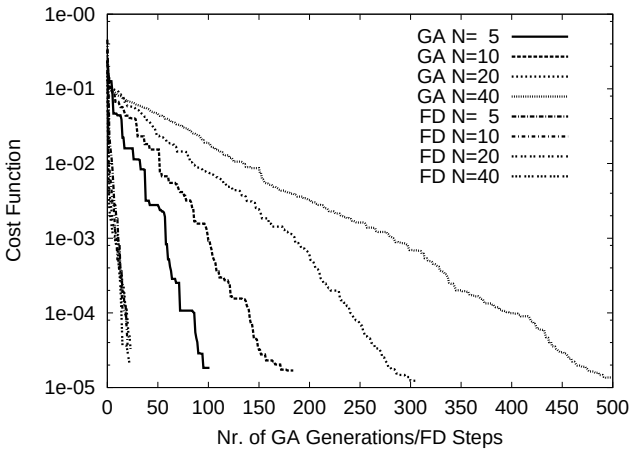
Basic Descent Step

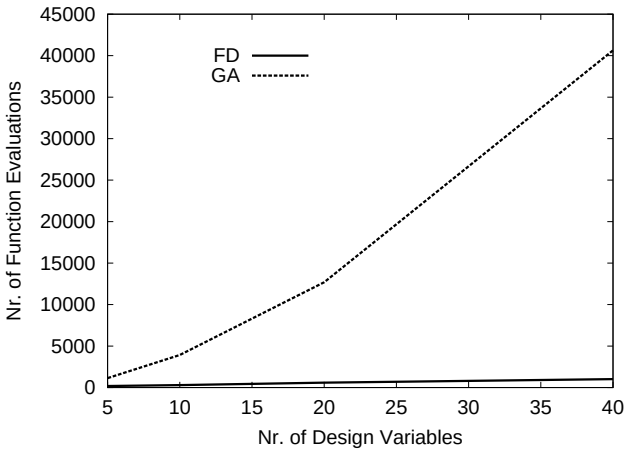
- Guess Small λ
- Evaluate I_1, I_{-1}
- Obtain λ_{min}
- $\boldsymbol{\alpha} = \boldsymbol{\alpha} - \lambda_{min} I_{,\boldsymbol{\alpha}_0}$
- Evaluate $I_2 = I(\boldsymbol{\beta}(\boldsymbol{\alpha}))$
- `If`: $I_2 > I_0 \Rightarrow$ Reduce λ (`go back`)
- Set: $\lambda = \lambda_{min}$
- `do:  icont=1,mcont` ! Continuation Steps
    - $\boldsymbol{\alpha}_3 = \boldsymbol{\alpha} - \lambda I_{,\boldsymbol{\alpha}_0}$
    - Evaluate $I_3 = I(\boldsymbol{\beta}(\boldsymbol{\alpha}_3))$
    - `If`: $I_3 > I_2$: `exit loop`
    - Replace $I_2 = I_3$, $\boldsymbol{\alpha} = \boldsymbol{\alpha}_3$
- `enddo`

SMALL EXAMPLE

Approximate: $sin(\omega x)$, $0 \leq x \leq 1$ Via Point Collocation (Uniform 1-D)

$$I = \frac{1}{N}\sqrt{\sum_{N}(a_i - sin(\omega x_i)^2}$$





Steps Required:

- Gradient Based: $O(1) \Rightarrow$ CPU $O(N)$
- Genetic: $O(N) \Rightarrow$ CPU $O(N^2)$

ADJOINT (1)

Key Ideas:

Use PDE Constraint to Obtain all Gradients at Once

$$\delta I = I_{,\boldsymbol{\beta}} \delta \boldsymbol{\beta} + I_{,\mathbf{u}} \delta \mathbf{u} \quad ,$$

$$\delta \mathbf{R} = \mathbf{R}_{,\boldsymbol{\beta}} \delta \boldsymbol{\beta} + \mathbf{R}_{,\mathbf{u}} \delta \mathbf{u} = 0$$

Introduce Lagrange Multiplier $\boldsymbol{\Psi}$:

$$\delta I = I_{,\boldsymbol{\beta}} \delta \boldsymbol{\beta} + I_{,\mathbf{u}} \delta \mathbf{u} - \boldsymbol{\Psi}^T \left[\mathbf{R}_{,\boldsymbol{\beta}} \delta \boldsymbol{\beta} + \mathbf{R}_{,\mathbf{u}} \delta \mathbf{u} \right]$$

ADJOINT (2)

Rearrange:

$$\delta I = \left[I_{,\boldsymbol{\beta}} - \boldsymbol{\Psi}^T \mathbf{R}_{,\boldsymbol{\beta}} \right] \delta \boldsymbol{\beta} + \left[I_{,\mathbf{u}} - \boldsymbol{\Psi}^T \mathbf{R}_{,\mathbf{u}} \right] \delta \mathbf{u}$$

If We Can Solve:

$$\boldsymbol{\Psi}^T \mathbf{R}_{,\mathbf{u}} = I_{,\mathbf{u}} \quad ,$$

$\Rightarrow$

$$\delta I = \left[I_{,\boldsymbol{\beta}} - \boldsymbol{\Psi}^T \mathbf{R}_{,\boldsymbol{\beta}} \right] \delta \boldsymbol{\beta} = \left[G^I \right]^T \delta \boldsymbol{\beta}$$

ADJOINT (3)

$\Rightarrow$

- Need Only Derivatives With Respect to $\boldsymbol{\beta}$
- 'Analytic' Evaluation of Gradient
- Cost of Gradient Evaluation Independent of Nr. of Design Variables

Pros:

- $O(1)$ Evaluations (!) of $I \Rightarrow$ Fast

Cons:

- Not General (Boundary Conditions)
- Not Suited for Rough Functionals
- May Stop at Local Min

ADJOINT: First Derivative Residuals

$$\mathbf{R} = \mathbf{F}^i_{,i} = 0$$

$\mathbf{F}(\mathbf{u})$: Flux Function
$\mathbf{A}^i = \mathbf{F}^i_{,\mathbf{u}}$: Flux Jacobian

$$\mathbf{\Psi}^T \mathbf{R}_{,\mathbf{u}} = \mathbf{\Psi}^T \left(\mathbf{F}^i_{,i}\right)_{,\mathbf{u}} = \mathbf{\Psi}^T \left(\mathbf{F}^i_{,\mathbf{u}}\right)_{,i} = \mathbf{\Psi}^T \left(\mathbf{A}^i\right)_{,i} = I_{,\mathbf{u}}$$

Integration by Parts:

$$-\left[\mathbf{A}^i\right]^T \mathbf{\Psi}_{,i} = I^{\Omega}_{,\mathbf{u}}$$

$$\int_\Gamma \left[\mathbf{A}^i n_i\right]^T \mathbf{\Psi} d\Gamma = I^{\Gamma}_{,\mathbf{u}}$$

Remarks:

a) **Linear** System of **Advection Equations**
b) Eigenvalues Same as Original PDE
c) Negative Sign in Front of $\mathbf{A}^i \Rightarrow$ 'Advection Direction' for Adjoint **Opposite** to Original PDE
d) Boundary Conditions Require Appropriate Change

ADJOINT: Second Derivative Residuals

$$\mathbf{R} = -\nabla\mu\nabla\mathbf{w} = 0 \quad ,$$

μ: Diffusion or Viscosity
$\mathbf{w}(\mathbf{u})$: (Non-Conserved) Variables
$\mathbf{B} = \mathbf{w}_{,\mathbf{u}}$: Jacobian of $\mathbf{w}$

$$\mathbf{\Psi}^T\mathbf{R}_{,\mathbf{u}} = -\mathbf{\Psi}^T\left(\nabla\mu\nabla\mathbf{w}\right)_{,\mathbf{u}} = -\mathbf{\Psi}^T\nabla\mu\nabla\mathbf{w}_{,\mathbf{u}}$$

$$= -\mathbf{\Psi}^T\nabla\mu\nabla\mathbf{B} = I_{,\mathbf{u}}$$

Repeated Integration by Parts:

$$-\mathbf{B}^T\nabla\mu\nabla\mathbf{\Psi} = I^{\Omega}_{,\mathbf{u}} \quad ,$$

$$-\int_\Gamma \mathbf{B}^T_{,n}\mu\mathbf{\Psi}d\Gamma + \int_\Gamma \mathbf{B}^T\mu\mathbf{\Psi}_{,n}d\Gamma = I^{\Gamma}_{,\mathbf{u}}$$

Remarks:

a) **Linear** System of **Diffusion Equations**
b) Sign of Operator **Unchanged**

STEPS REQUIRED PER DESIGN CYCLE

For Every Design Cycle:

1. Flow Solver
2. Adjoint Solver
3. Gradient Evaluator
4. Mesh Movement Module

FLOW SOLVER: COMPRESSIBLE FLOW (1)

Euler Equations:

$$\mathbf{u}_{,t} + \nabla \cdot \mathbf{F} = 0 \quad ,$$

$$\mathbf{u} = \left\{ \begin{array}{c} \rho \\ \rho v_i \\ \rho e \end{array} \right\} \ , \ \mathbf{F}^j = \left\{ \begin{array}{c} \rho v_j \\ \rho v_i v_j + p\delta_{ij} \\ v_j(\rho e + p) \end{array} \right\} \ .$$

ρ: Density
p: Pressure
e: Specific Total Energy
v_i: Velocity

Equation of State: Polytropic Gas

$$p = (\gamma - 1)\rho[e - \frac{1}{2} v_j v_j] \quad ,$$

γ: Ratio of Specific Heats

FLOW SOLVER: COMPRESSIBLE FLOW (2)

Define:

$$u = v_1 \quad , \quad v = v_2 \quad , \quad w = v_3 \quad , \quad q = u^2 + v^2 + w^2$$
$$c_1 = \gamma - 1 \quad , \quad c_2 = q(\gamma - 1)/2 \quad , \quad c_3 = 3 - \gamma$$
$$c_4 = \gamma e - c_1 q \quad , \quad c_5 = \gamma e - c_2$$

Jacobian Matrices:

$$\mathbf{A}^x = \left\{ \begin{array}{rrrrr} 0; & 1; & 0; & 0; & 0; \\ -u^2 + c_2; & c_3 u; & -c_1 v; & -c_1 w; & c_1; \\ -uv; & v; & u; & 0; & 0; \\ -uw; & w; & 0; & u; & 0; \\ -c_4 u; & c_5 - c_1 u^2; & -c_1 uv; & -c_1 uw; & \gamma u; \end{array} \right\}$$

$$\mathbf{A}^y = \left\{ \begin{array}{rrrrr} 0; & 0; & 1; & 0; & 0; \\ -uv; & v; & u; & 0; & 0; \\ -v^2 + c_2; & -c_1 u; & c_3 v; & -c_1 w; & c_1; \\ -uw; & 0; & w; & v; & 0; \\ -c_4 v; & -c_1 uv; & c_5 - c_1 v^2; & -c_1 vw; & \gamma v; \end{array} \right\}$$

$$\mathbf{A}^z = \left\{ \begin{array}{rrrrr} 0; & 0; & 0; & 1; & 0; \\ -uv; & w; & 0; & u; & 0; \\ -uw; & 0; & w; & v; & 0; \\ -w^2 + c_2; & -c_1 u; & -c_1 v; & c_3 w; & c_1; \\ -c_4 w; & -c_1 uw; & -c_1 uw; & c_5 - c_1 w^2; & \gamma w; \end{array} \right\}$$

FLOW SOLVER: COMPRESSIBLE FLOW (4)

Summary of Solver

- Discretization in Space: Linear FEM
- Discretization in Time: Implicit/LU-SGS
- Steady State Acceleration:
  Local Timestepping, GMRES
- Edge-Based Data Structures
- Approximate Riemann Solver: Roe
- MUSCL Limiting: vanAlbada

FLOW SOLVER: INCOMPRESSIBLE FLOW (1)

Euler/Navier-Stokes Equations:

$$\mathbf{u}_{,t} + \nabla \cdot \mathbf{F} = \nabla \mu \nabla \mathbf{w} \quad ,$$

$$\mathbf{u} = \left\{ \begin{array}{c} p/c^2 \\ v_i \end{array} \right\} \ , \ \mathbf{F}^j = \left\{ \begin{array}{c} v_j \\ v_i v_j + p\delta_{ij} \end{array} \right\} \ , \ \mathbf{w} = \left\{ \begin{array}{c} 0 \\ v_i \end{array} \right\}$$

p: Pressure
c: (Infinite) Speed of Sound
v_i: Velocity

Jacobian Matrices:

$$\mathbf{A}^x = \left\{ \begin{array}{l} 0;\ c^2;\ 0;\ 0; \\ 1;\ 2u;\ 0;\ 0; \\ 0;\ \ v;\ u;\ 0; \\ 0;\ \ w;\ 0;\ u; \end{array} \right\} \ , \ \mathbf{A}^y = \left\{ \begin{array}{l} 0;\ 0;\ c^2;\ 0; \\ 0;\ v;\ \ u;\ 0; \\ 1;\ 0;\ 2v;\ 0; \\ 0;\ 0;\ \ w;\ v; \end{array} \right\} \ ,$$

$$\mathbf{A}^z = \left\{ \begin{array}{l} 0;\ 0;\ 0;\ \ c^2; \\ 0;\ w;\ 0;\ \ u; \\ 0;\ 0;\ w;\ \ v; \\ 1;\ 0;\ 0;\ 2w; \end{array} \right\} \ ,$$

B-Matrix:

$$\mathbf{B} = \left\{ \begin{array}{l} 0;\ 0;\ 0;\ 0; \\ 0;\ 1;\ 0;\ 0; \\ 0;\ 0;\ 1;\ 0; \\ 0;\ 0;\ 0;\ 1; \end{array} \right\}$$

FLOW SOLVER: INCOMPRESSIBLE FLOW (3)

Summary of Solver

- Discretization in Space: Linear FEM
- Discretization in Time: Projection Scheme
- Steady State Acceleration:
  Local Timestepping
- Edge-Based Data Structures
- Approximate Riemann Solver: Roe
- MUSCL Limiting: vanAlbada
- Div-Limiting: 4th Order Pressure Damping

ADJOINT SOLVER

PDE to be Solved:

$$\mathbf{\Psi}_{,t} - \left[\mathbf{A}^i\right]^T \mathbf{\Psi}_{,i} = \mathbf{B}^T \nabla\mu\nabla\mathbf{\Psi} + I^{\Omega} \quad ,$$

$$\mathbf{\Psi} = \left\{ \begin{array}{c} \Psi_\rho \\ \Psi_{\rho v^i} \\ \Psi_{\rho e} \end{array} \right\} \quad , \quad \mathbf{A}^i = \mathbf{F}^i_{,\mathbf{u}} \quad , \quad \mathbf{B} = \mathbf{w}_{,\mathbf{u}}$$

Summary of Solver

- Discretization in Space: Linear FEM
- Discretization in Time: Explicit R-K
- Steady State Acceleration:
  Local Timestepping, Residual Smoothing
- Edge-Based Data Structures
- Artificial Viscosity:
  - Blend of 2nd/4th Order
  - Component-Wise

BOUNDARY CONDITONS FOR ADJOINT (1)

a) Inviscid Forces:

$$\mathbf{f} = \int_{\Gamma} p\mathbf{n} d\Gamma$$

Cost Function With Weights $\mathbf{c}_w = c_w^x, c_w^y, c_w^z$:

$$I = \mathbf{f} \cdot \mathbf{c}_w \quad ,$$

$\Rightarrow$

$$\mathbf{n} \cdot \mathbf{\Psi}_\mathbf{v} = \mathbf{c}_w \cdot \mathbf{n} \quad ,$$

Remarks:

- Normal Adjoint Velocity Prescribed
  Similar to No-Penetration for Flow Solver
- No Condition Required for Pressure

BOUNDARY CONDITONS FOR ADJOINT (2)

b) Prescribed Pressure:

$$I_{p_0} = \int_\Gamma (p - p_0)^2 d\Gamma \quad ,$$

$\Rightarrow$

$$\mathbf{n} \cdot \mathbf{\Psi_v} = 2(p - p_0)$$

Remarks:

- Normal Adjoint Velocity Prescribed
  Similar to No-Penetration for Flow Solver
- No Condition Required for Pressure

GRADIENT EVALUATION

Chain Rule:

$$\frac{\partial I}{\partial \beta_l} = \frac{\partial x_k^i}{\partial \beta_l} \frac{\partial I}{\partial x_k^i}$$

Remarks:

- 2nd Part: Gradient wrt Mesh Movement
  - Independent of Design Variables Chosen
  - Size: `ndimn*npoin`
  - $\Rightarrow$ Can Compute In Separate Run

GRADIENT WRT MESH MOVEMENT (1)

$$\frac{\partial I_L}{\partial x_k^i} = \left[\frac{\partial I}{\partial x_k^i} - \boldsymbol{\Psi}^T \frac{\partial \mathbf{R}}{\partial x_k^i} \right]$$

Finite Difference Evaluation

- Move Each Point in $(+/-)$ x, y, z Direction (1% of Characteristic Length)
- Evaluate Residual
- Evaluate Differences/Gradient
- Worst Case Scenario: New Residual for Each Point $\Rightarrow 2 \cdot 3 \cdot N_p$

GRADIENT WRT MESH MOVEMENT (2)

Key Ideas:

- Explicit RHS: Residual Only Influenced by Neighbours (+1)
- $\Rightarrow$ Colour Points ($O(25-30)$ Groups)

Evaluation of Gradient:

- `Do`: Loop Over Colours
- `Do`: Loop Over Dimensions
- `Do`: Loop Over +/− Increments
- Recalculate Geometrical Parameters
- Perform One (Explicit) Timestep/RHS
- `EndDo`
- Obtain Gradient
- `EndDo`
- `EndDo`

Number of RHS Evaluations: $O(150-180)$

GEOMETRIC CONSTRAINTS

Typical Constraints: Prescribed (Min/Max)

- Volume;
- Cross- Sectional Area;
- Sectional Thickness;
- Deviation from Original Shape;
- Surface Curvature;

Treatment Options:

- Inclusion in Cost Function;
- Projection of Gradient;
- Post-Processing of New Shape;

VOLUME CONSTRAINT (1)

$$I = I_0 + c_V \frac{|V - V_O|}{V_O} \quad ,$$

or

$$I = I_0 + c_V \frac{(V - V_O)^2}{V_O^2} \quad ,$$

Evaluation:

- Sum Up Element Volumes
  (+ Subtraction from Reference Volume);
- Sum Up Face Contributions
  (Divergence Theorem);

VOLUME CONSTRAINT (2)

Divergence Theorem:

$$\int_{\Omega} \nabla \cdot \mathbf{v} d\Omega = \int_{\Gamma} \mathbf{v} \cdot \mathbf{n} d\Gamma$$

Volume: $\int_{\Omega} d\Omega \Rightarrow$ Desire $\nabla \cdot \mathbf{v} = 1$
Typical: $\mathbf{v} = (x, y, z)/3$

$$V = \frac{1}{3} \int_{\Gamma} (x, y, z) \cdot \mathbf{n} d\Gamma$$

Separate Into x, y, z Components:

$$V = \frac{1}{3} \sum_{id=1,3} V_{id} = \frac{1}{3} \sum_{id=1,3} \int_{\Gamma} x_{id} n_{id} d\Gamma$$

VOLUME CONSTRAINT (3)

For Arbitrary (Surface) Gridpoint $\mathbf{x}_i$:

$$\frac{\Delta V}{\Delta \mathbf{x}_i^k} = \frac{1}{\Delta \mathbf{x}_i^k} \sum_{i \in jf} \left[\int_\Gamma x_{id} n_{id} d\Gamma \Big|_{\mathbf{x}_i^k + \Delta \mathbf{x}_i^k} - \int_\Gamma x_{id} n_{id} d\Gamma \Big|_{\mathbf{x}_i^k - \Delta \mathbf{x}_i^k} \right]$$

PROJECTION OF GRADIENTS

Typical Constraint: $V(\boldsymbol{\beta}) = V_0$

$\Rightarrow \delta V = \delta\boldsymbol{\beta}^T \cdot V_{,\boldsymbol{\beta}} = 0$

$\Rightarrow$ No Change of $\boldsymbol{\beta}$ in Direction $V_{,\boldsymbol{\beta}}$

Denote: $\mathbf{g}_i^c = V^i_{,\boldsymbol{\beta}} \quad , \quad i = 1, n$

Step 1: Orthogonalization of Gradients:

```
DO: For i = 1, n
      g_i^c' = g_i^c / |g_i^c|
      DO: For j = i + 1, n
            g_j^c = g_j^c - (g_j^c · g_i^c') g_i^c'
      ENDDO
ENDDO
```

Step 2: Projection of Cost Function Gradient:

$$I_{,\boldsymbol{\beta}} = I_{,\boldsymbol{\beta}} - (I_{,\boldsymbol{\beta}} \cdot \mathbf{g}_i^{c'})\mathbf{g}_i^{c'}$$

MULTIPOINT OPTIMIZATION

Typical: Optimal for Single Point $\Rightarrow$ Bad for Other Points

Classic Example: Korn Airfoil

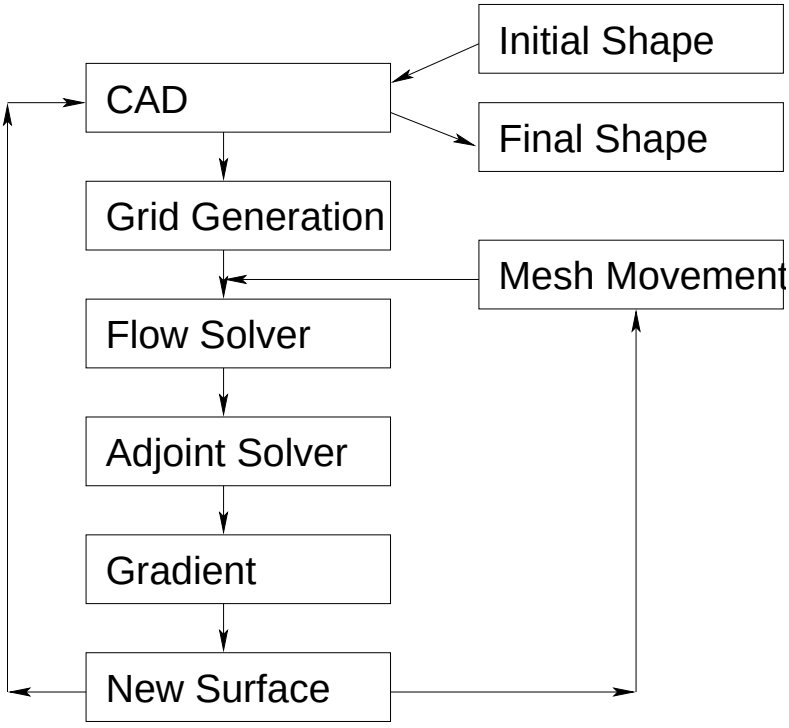
Approach: Multipoint Optimization

$$I = \sum_i \gamma_i I(Ma_i, \alpha_i)$$

Remarks:

- Additive Cost Function
- $\Rightarrow$ Gradient Evaluations Also Additive
- Apt for Low-Grain Parallel Processing
- Current Research Topic:
  - Robust Design;
  - Number of Design Points Required;
  - Optimal Choice of γ_i;

DESIGN CYCLE



REPRESENTATION OF DESIGN PARAMETERS

- Bezier/NURBS Control Points of CAD Model
- Family of Functions
  (e.g. Hicks-Henne Functions)
- Family of Bodies (Airfoils, Wings, ..)
- Gridpoints (Discrete Data)
- Basis Functions for Change

Options for All of These:

- Usual
- Hierarchical (One Shot)

PSEUDO-SHELL APPROACH (1)

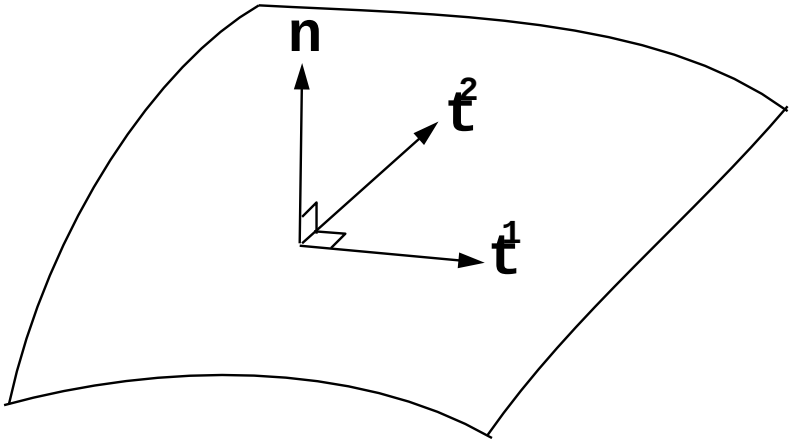


Figure 2 Pseudo-Shell Coordinates

PSEUDO-SHELL APPROACH (2)

$$\theta_l - \frac{\partial u_n}{\partial \mathbf{t}^l} = 0 \;\; ; \;\; l = 1,2 \;\; ; \theta_n = 0 \;\; ;$$

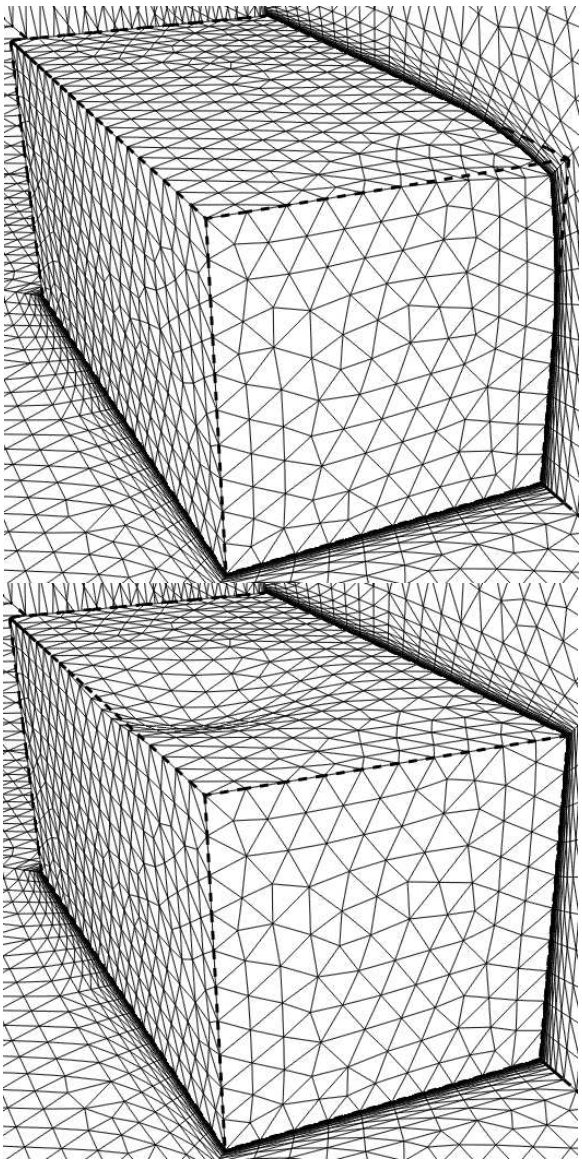
$$\frac{\partial}{\partial \mathbf{t}^l}\left(\theta_l - \frac{\partial u_n}{\partial \mathbf{t}^l}\right) = 0 \;\; ; \;\; l = 1,2 \;\; ;$$

$$\frac{\partial}{\partial \mathbf{t}^k}\left[\mu\left(\frac{\partial u_l}{\partial \mathbf{t}^k} + \frac{\partial u_k}{\partial \mathbf{t}^l}\right)\right] + \frac{\partial}{\partial \mathbf{t}^l}\left[\left(\frac{\mu\gamma}{1-2\gamma}\right)\frac{\partial u_k}{\partial \mathbf{t}^k}\right] = 0 \;\; ;$$

$$l, k = 1,2$$

- Solution by LU (Direct Method) on Γ_n
  $\Rightarrow$ Fast
- Assures **Smooth** Surface

PSEUDO-SHELL APPROACH (3)



MESH MOVEMENT (1)

Task:

Given: Surface Displacement $\mathbf{w}_\Gamma$

Obtain: Volume Displacement $\mathbf{w}_\Omega$

Options:

- Spring System
- Laplacian
- Elasticity + Near Incompressible Material

MESH MOVEMENT (2)

Elasticity:

$$\sigma^{ij}_{,j} = 0 \quad , \quad \sigma^{ij} = \mu(w^i_{,j} + w^j_{,i}) + \delta^{ij}\lambda w^k_{,k}$$

$\Rightarrow$

$$\mu w^i_{,jj} + (\mu + \lambda)(w^j_{,j})_{,i} = 0$$

Or:

$$\mu\nabla^2\mathbf{w} + (\mu + \lambda)\nabla\nabla\cdot\mathbf{w} = 0$$

Remarks:

- Laplacian Obtained for $\mu + \lambda = 0$
- Incompressibility Constraint: $\lambda >> \mu$

MESH MOVEMENT: LEVEL APPROACH

Define Levels: $l_i = 0, ..n$

- Surface: $l_i = 0$
- Any Point With $l_i > 0$: At Least One Connection to Point of l_{i-1}

Displacements $\mathbf{v}_i$:

$$\mathbf{v}_i = \frac{\sum_{j \in l_{i-1}} w_{ij} \mathbf{v}_j}{\sum_{j \in l_{i-1}} w_{ij}}$$

Edge-Weights w_{ij}: From Distance d_{ij}:

$$w_{ij} = d_{ij}^{-2}$$

Other Possibility: Local Weight Limiting

GRADIENT EVALUATION (1)

- Design Parameters: $\boldsymbol{\beta}$
- Surface Points: $\mathbf{y}$
- Mesh Points: $\mathbf{x}$

Chain Rule:

$$\frac{\partial I}{\partial \beta_l} = \frac{\partial y_q^p}{\partial \beta_l} \frac{\partial x_k^i}{\partial y_q^p} \frac{\partial I}{\partial x_k^i}$$

Obtain:

$\frac{\partial y_q^p}{\partial \beta_l}$: Surface Representation

$\frac{\partial x_k^i}{\partial y_q^p}$: Mesh Movement

$\frac{\partial I}{\partial x_k^i}$: Gradient Evaluation

GRADIENT EVALUATION (2)

Finite Differences $\Rightarrow \forall\ \beta_l$:

- Perturb $\boldsymbol{\beta}$ and Evaluate New Surface $\mathbf{y} \Rightarrow \frac{\partial y_q^p}{\partial \beta_l}$
- With New Surface $\mathbf{y}$: Get New Volume Point Placement $\Rightarrow \frac{\partial x_k^i}{\partial y_q^p}$
- Multiply by Pre-Stored $\frac{\partial I}{\partial x_k^i}$

Worst Case Scenario: $O(N_{\boldsymbol{\beta}} \cdot N_{\mathbf{y}} \cdot N_{\mathbf{x}})$

Reduce Cost by:

- Local Influence Factors
  (Weight Functions, Layers, ...)
- Layers for Mesh Movement (Damping)
  $\Rightarrow O(N_{\boldsymbol{\beta}} \cdot N_{l1} \cdot N_{l2})$
- Pre-Calculation of Influence Factors

APPROXIMATE GRADIENTS

Complete Gradient Composed of Two Parts:

- Gradient WRT Geometry
- Gradient WRT State

$$\frac{\partial I_L}{\partial x_k^i} = \left[\frac{\partial I}{\partial x_k^i} - \mathbf{\Psi}^T \frac{\partial \mathbf{R}}{\partial x_k^i} \right]$$

In Many Cases:

$$\frac{\partial I_L}{\partial x_k^i} \approx \frac{\partial I}{\partial x_k^i} \quad ,$$

Remarks:

- No Adjoint Required

BUMP EXAMPLE

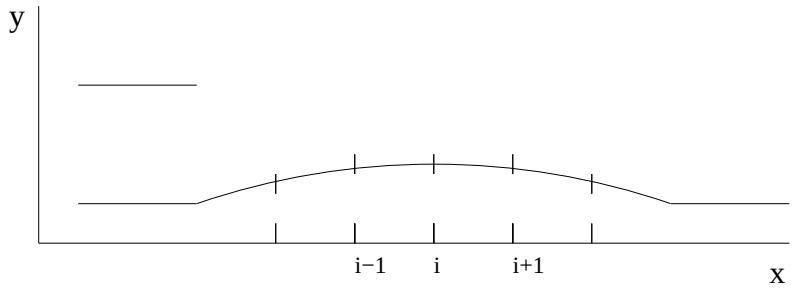


Figure 1a Bump Geometry

BUMP EXAMPLE: Inviscid Drag Force (1)

$$f_x^p = \int_\Gamma p n_x d\Gamma$$

Numerically:

$$f_x^p = \sum_{ij} \frac{p_i + p_j}{2}(y_j - y_i)$$

Sum Over Faces $\rightarrow$ Sum Over Boundary Points:

$$f_x^p = \frac{1}{2}\sum_i (p_{i-1} - p_{i+1}) y_i \quad ,$$

$\Rightarrow$

$$\left.\frac{\partial f_x^p}{\partial y}\right|_i = \frac{1}{2}(p_{i-1} - p_{i+1}) \quad ,$$

BUMP EXAMPLE: Inviscid Drag Force (2)

'Analytically':

$$\frac{\partial f_x^p}{\partial y} = -h_x \frac{\partial p}{\partial x}$$

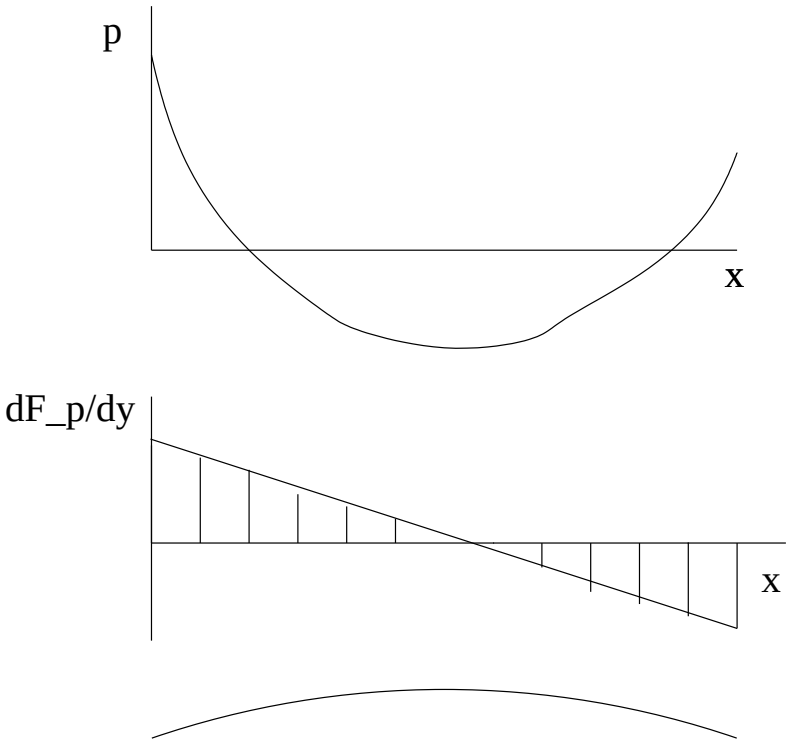


Figure 1b Gradient of Inviscid Drag Force

BUMP EXAMPLE: Viscous Drag Force (1)

$$f_x^v = \int_\Gamma \tau^x d\Gamma \quad ,$$

Numerically: $(l_{ij} = \sqrt{\Delta x^2 + \Delta y^2}\Big|_{ij})$

$$f_x^v = \sum_{ij} \tau_{ij}^x l_{ij}$$

Sum Over Faces $\rightarrow$ Sum Over Boundary Points:

$$I_p = \frac{1}{2} \sum_i \left[\tau_{i-1,i}^x l_{i-1,i} + \tau_{i,i+1}^x l_{i,i+1} \right]$$

$\Rightarrow$

$$\left. \frac{\partial I_p}{\partial y} \right|_i \approx \frac{\tau_{i-1,i}^x (y_i - y_{i-1})}{l_{i-1,i}} - \frac{\tau_{i,i+1}^x (y_{i+1} - y_i)}{l_{i,i+1}}$$

BUMP EXAMPLE: Viscous Drag Force (2)

Use: $\Delta y = l\Delta\phi, l = R\Delta\phi \Rightarrow$

$$\frac{\partial I_p}{\partial y} = \frac{\tau^x h_x}{R}$$

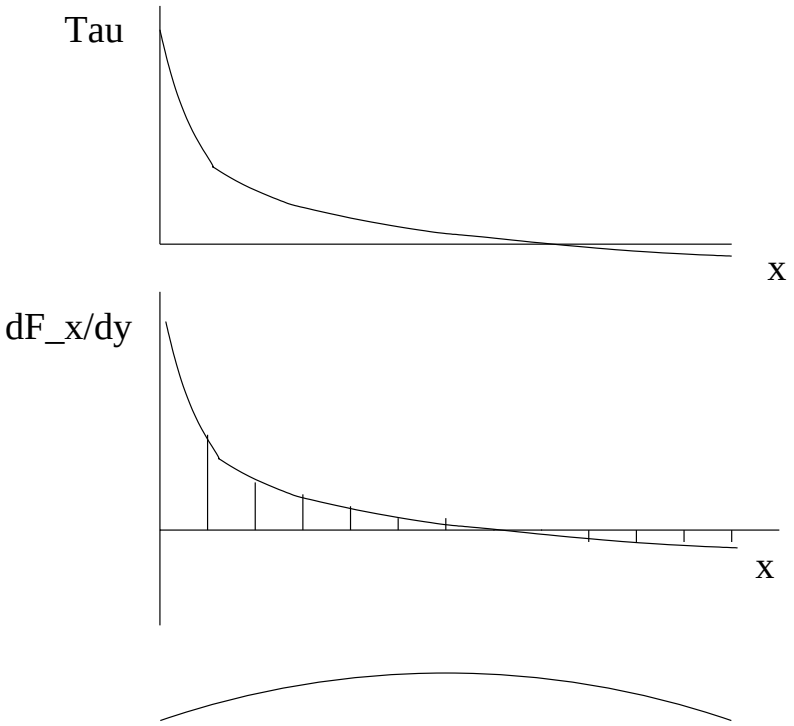


Figure 1c Gradient of Viscous Drag Force

BUMP EXAMPLE: Prescribed Pressure (1)

$$I_p = \int_\Gamma (p - p_0)^2 d\Gamma \quad ,$$

Numerically:

$$I_p = \sum_{ij} (p - p_0)_{ij}^2 l_{ij}$$

Sum Over Faces $\rightarrow$ Sum Over Boundary Points:

$$I_p = \frac{1}{2} \sum_i \left[(p - p_0)_{i-1,i}^2 l_{i-1,i} + (p - p_0)_{i,i+1}^2 l_{i,i+1} \right]$$

BUMP EXAMPLE: Prescribed Pressure (2)

$$\left.\frac{\partial I_p}{\partial y}\right|_i \approx \frac{(p-p_0)^2_{i-1,i}(y_i - y_{i-1})}{l_{i-1,i}}$$

$$-\frac{(p-p_0)^2_{i,i+1}(y_{i+1} - y_i)}{l_{i,i+1}}$$

Use: $\Delta y = l\Delta\phi, l = R\Delta\phi \Rightarrow$

$$\frac{\partial I_p}{\partial y} = \frac{(p-p_0)^2 h_x}{R}$$

BUMP EXAMPLE: Prescribed Pressure (3)

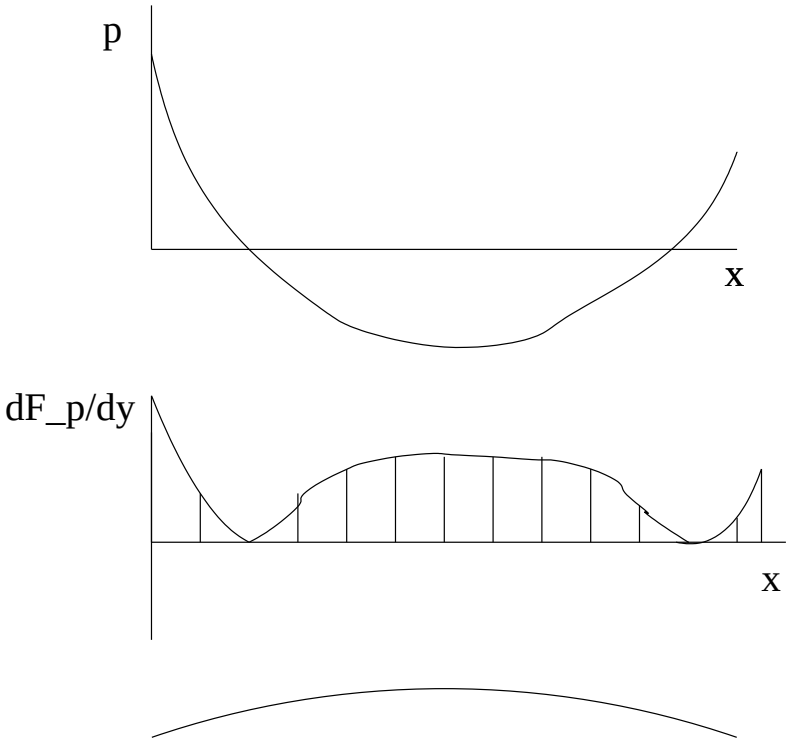


Figure 1d Gradient for Prescribed Pressure ($p_0 = 0$)

BUMP EXAMPLE: Volume Constraint (1)

$$V_y = \int_\Gamma y n_y d\Gamma$$

Numerically:

$$V_y = \sum_{ij} \frac{y_i + y_j}{2}(x_j - x_i)$$

Sum Over Faces $\rightarrow$ Sum Over Boundary Points:

$$V_y = \frac{1}{2}\sum_i y_i(\, h_x|_{i,i-1} + h_x|_{i+1,i})$$

$\Rightarrow$

$$\frac{\partial V_y}{\partial y} = h_x$$

BUMP EXAMPLE: Volume Constraint (1)

$$\frac{\partial V_y}{\partial y} = h_x$$

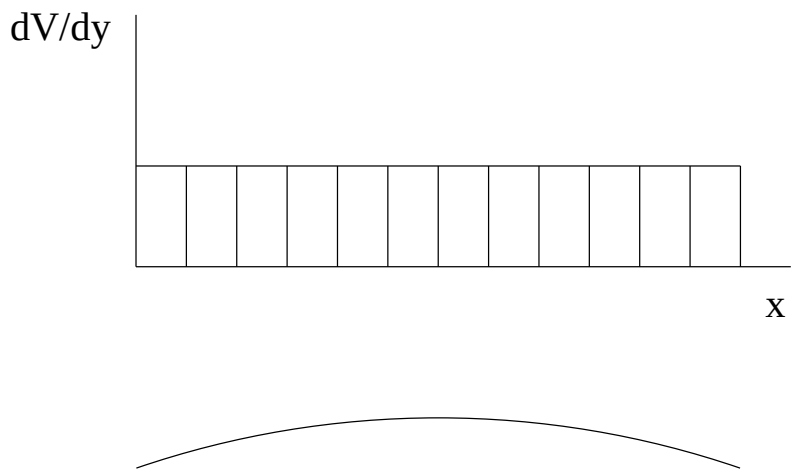


Figure 1e Gradient for Volume Constraint

SUBSONIC ENTRY NOZZLE (1)

- Objective: Prescribed Velocity (Entry)
- No Constraints on Shape
- Mesh: 4Kpts, 20Kels

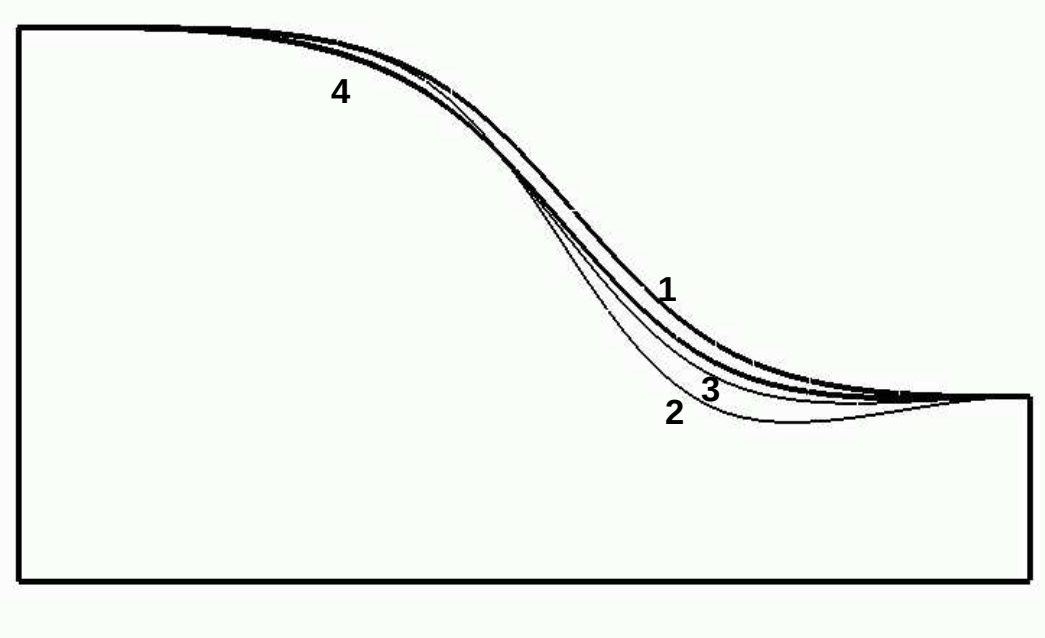


Figure 2a: Outlines for Subsonic Nozzle

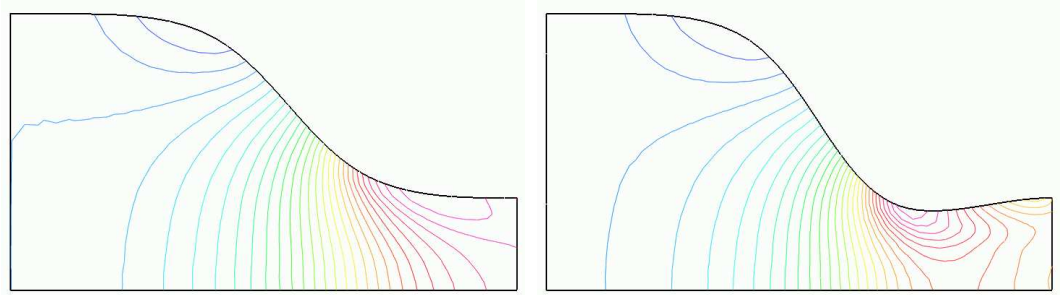


Figure 2b,c: Designs 0,1

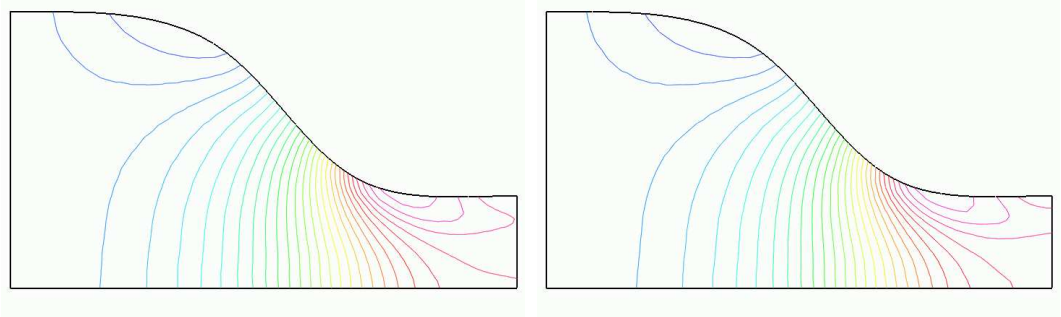


Figure 2d,e: Designs 2,4

SUBSONIC ENTRY NOZZLE (3)

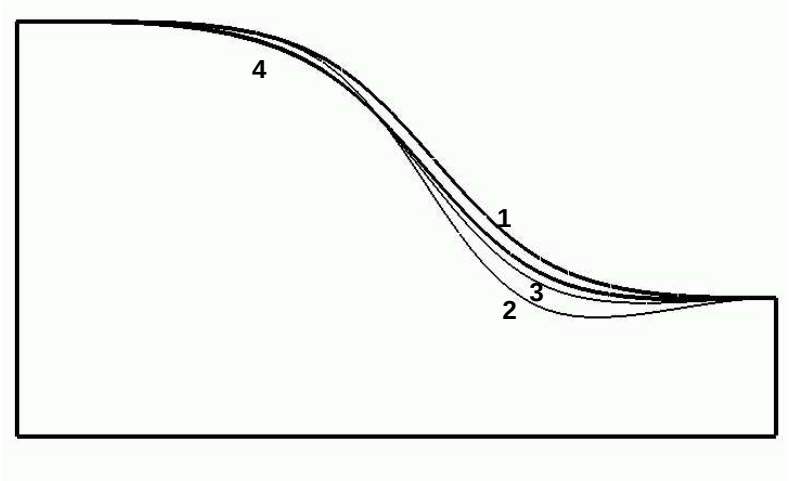


Figure 2f: Convergence History

EXTERNAL NOZZLE (1)

- Objective: Maximum Thrust
- Flow Conditions:
  - Inflow (external flowfield):
    $Ma = 3.00, \rho = 0.5, v = 1.944$
    $pr = 0.15, \alpha = 0.0^o$
  - Nozzle exit:
    $Ma = 1.01, \rho = 1.0, v = 1.000$
    $pr = 0.18, \alpha = -45.0^o$
- No Constraints on Shape
- Mesh: 51Kpts, 267Kels

EXTERNAL NOZZLE (2)

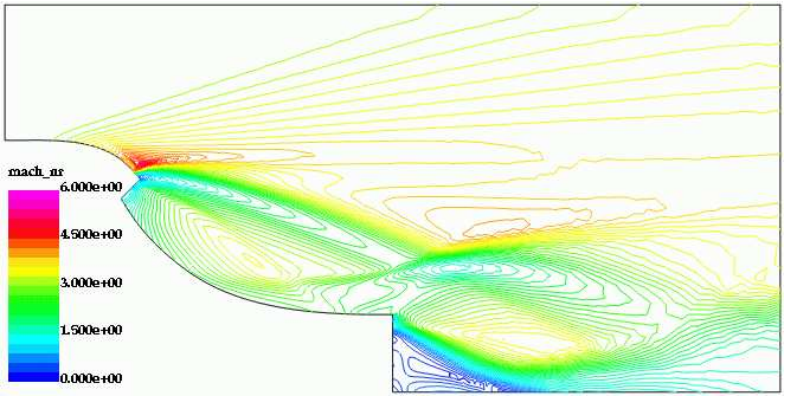


Figure 2a: Mach-Number for 1st Shape

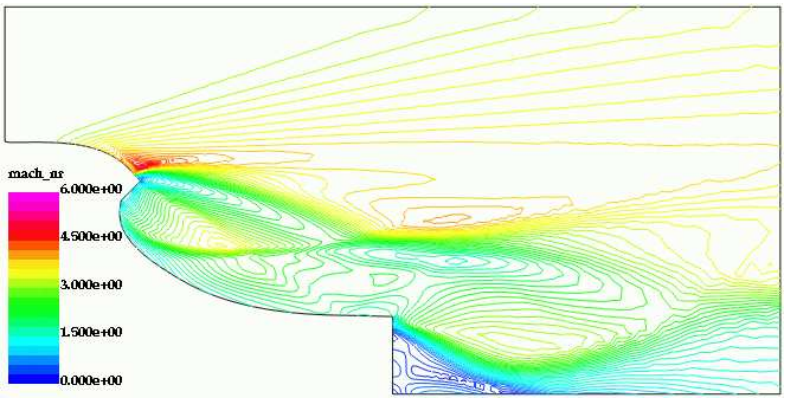


Figure 2b: Mach-Numbers for Last Shape

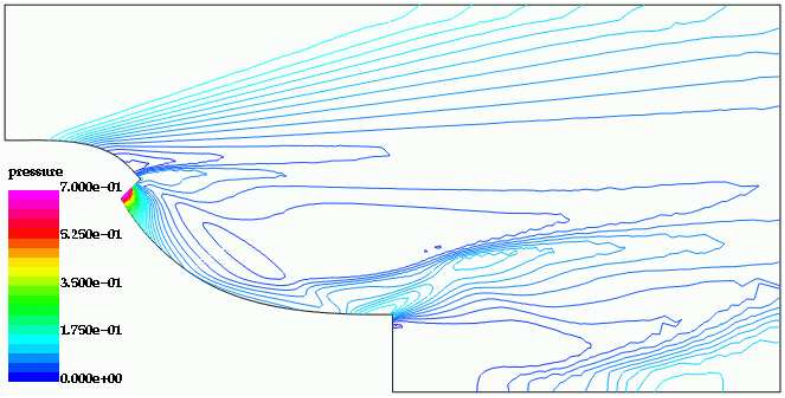


Figure 2c: Pressure for 1st Shape

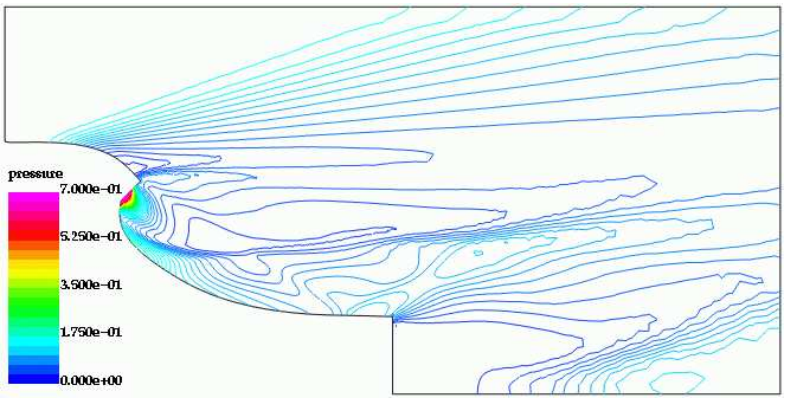


Figure 2d: Pressure for Last Shape

EXTERNAL NOZZLE (4)

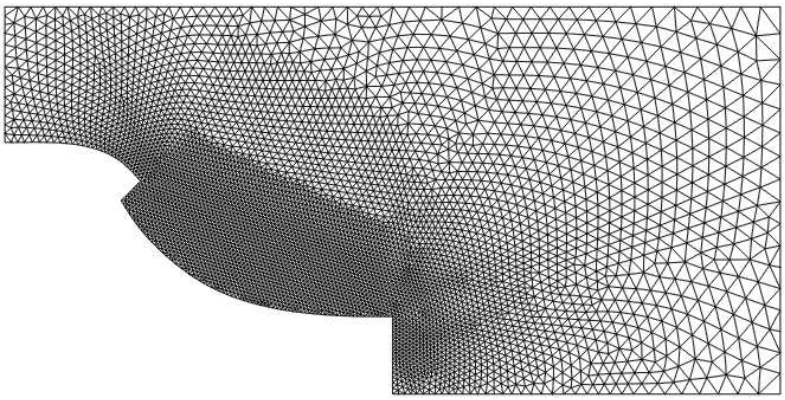


Figure 2e: Surface Mesh for 1st Shape

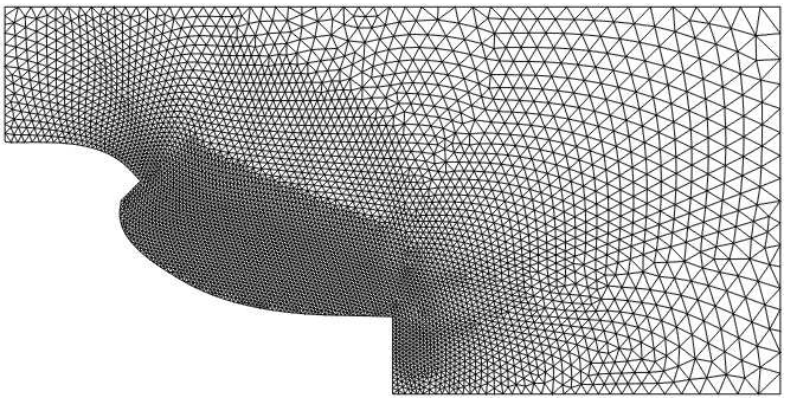


Figure 2f: Surface Mesh for Last Shape

EXTERNAL NOZZLE (5)

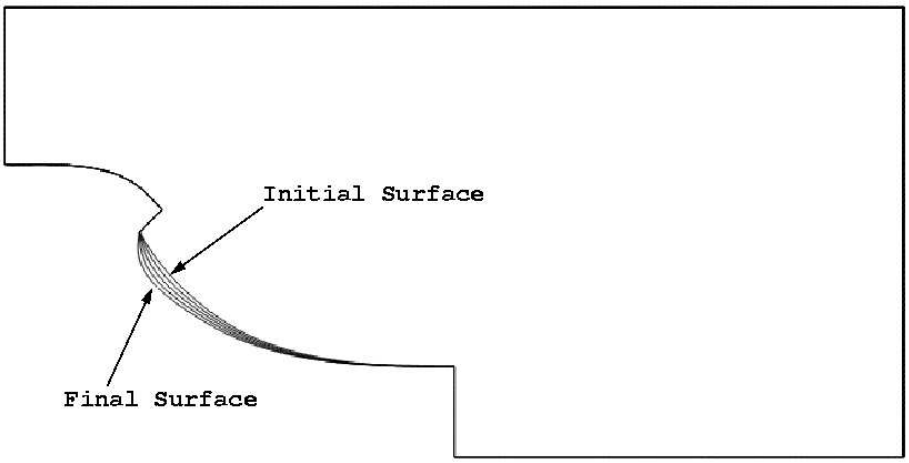


Figure 2g: Evolution of Shape

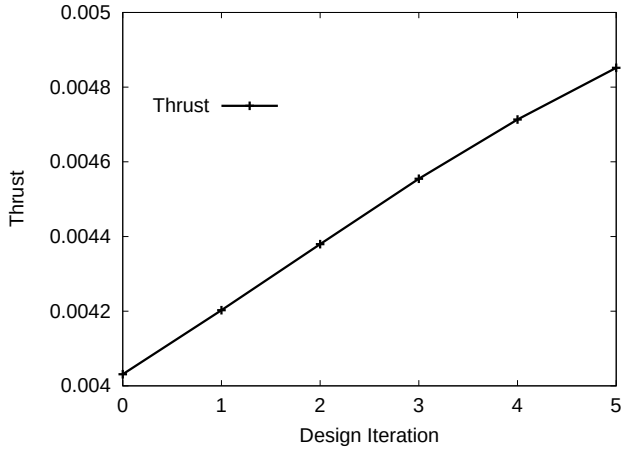


Figure 2h: Evolution of Thrust

HYPERSONIC FLYER (1)

- Objective: Maximum Thrust
- Flow Conditions [mks]:
  - Inflow (external flowfield):
    $Ma = 8.00, \rho = 0.004826, v = 2519.2$
    $pr = 341.9, \alpha = 2.0^o$
  - Nozzle exit:
    $Ma = 2.00, \rho = 0.048260, v = 630.0$
    $pr = 3419.0, \alpha = 0.0^o$
- No Constraints on Shape
- Mesh: 375Kpts, 2Mels

HYPERSONIC FLYER (2)

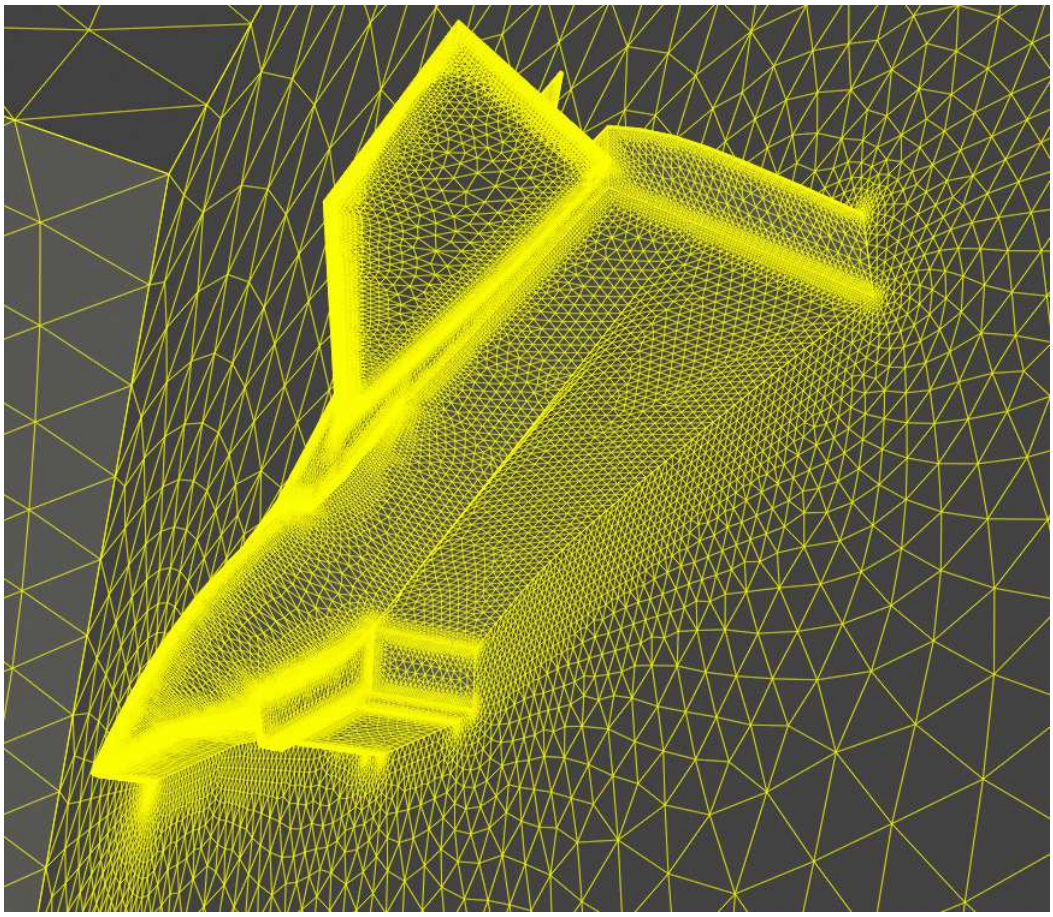


Figure 3a: Flyer: Surface Mesh

HYPERSONIC FLYER (3)

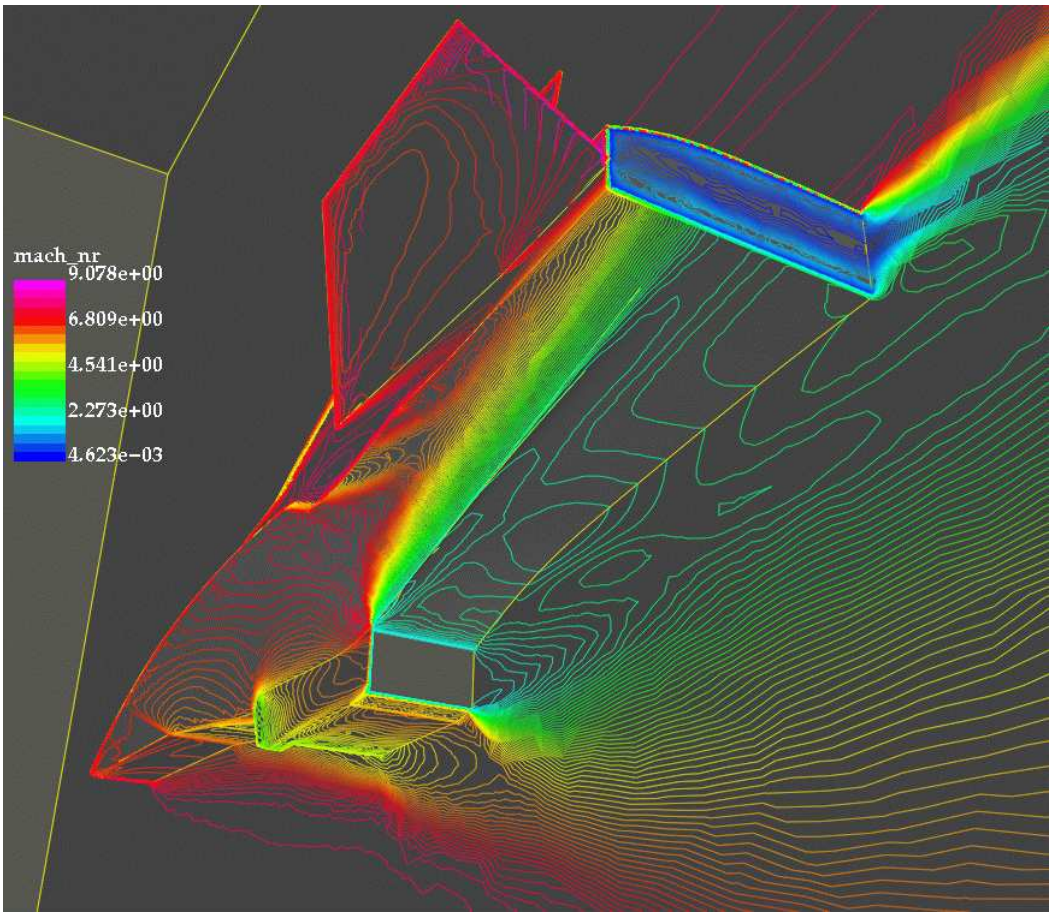


Figure 3b: Flyer: Surface Mach-Number

HYPERSONIC FLYER (4)

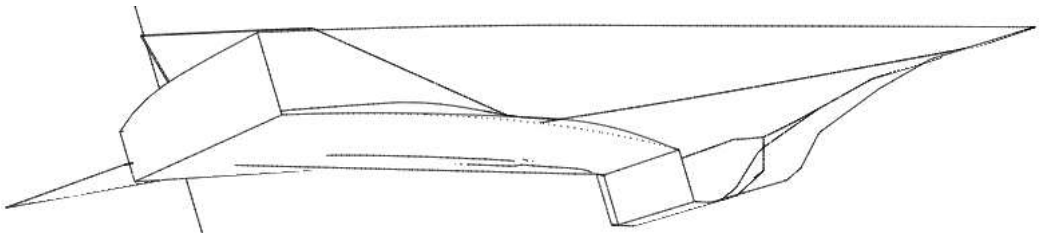
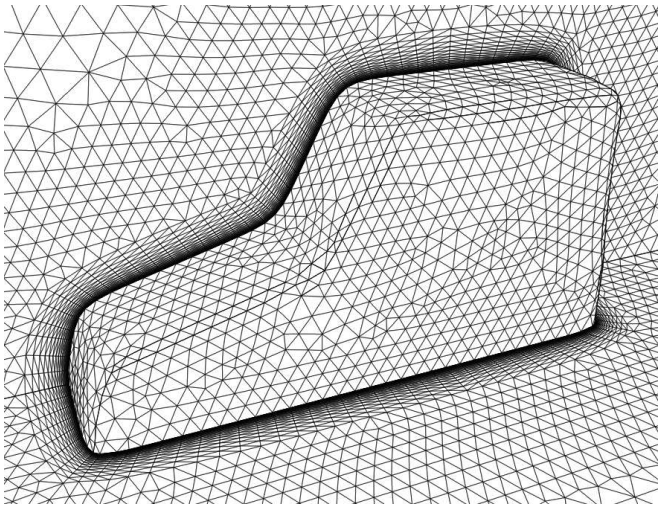


Figure 3c: Evolution of Shape

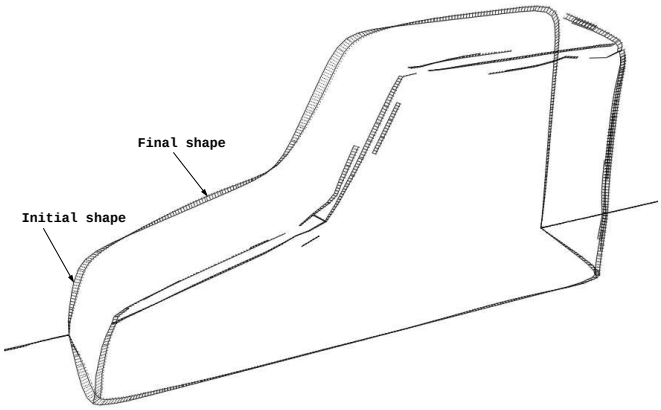
SUV-LIKE OBJECT (1)

- Objective: Minimum Drag
- Incompressible Flow
- Smagorinsky (Law of the Wall)
- Constraints: Minimum Shape Change
- Mesh: 375Kpts, 2Mels
- Nr. of Design Variables: 944

SUV-LIKE OBJECT (2)

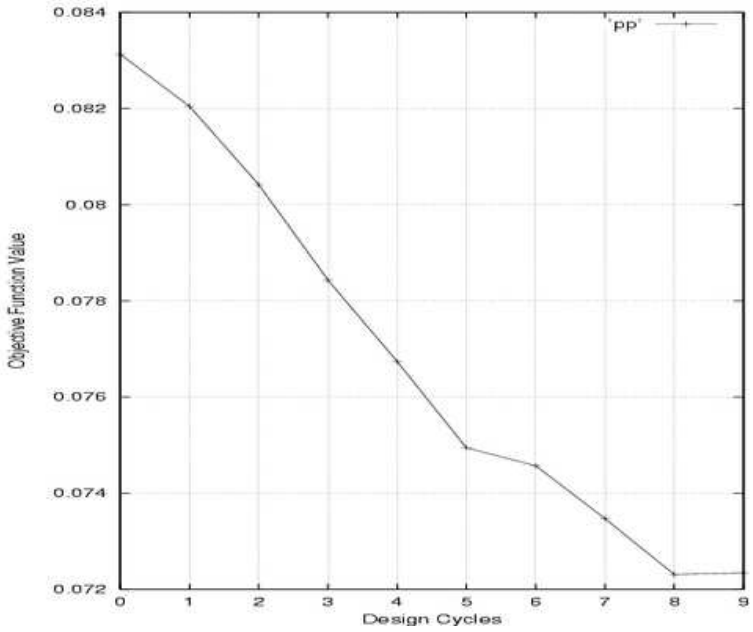


Mesh and Geometry

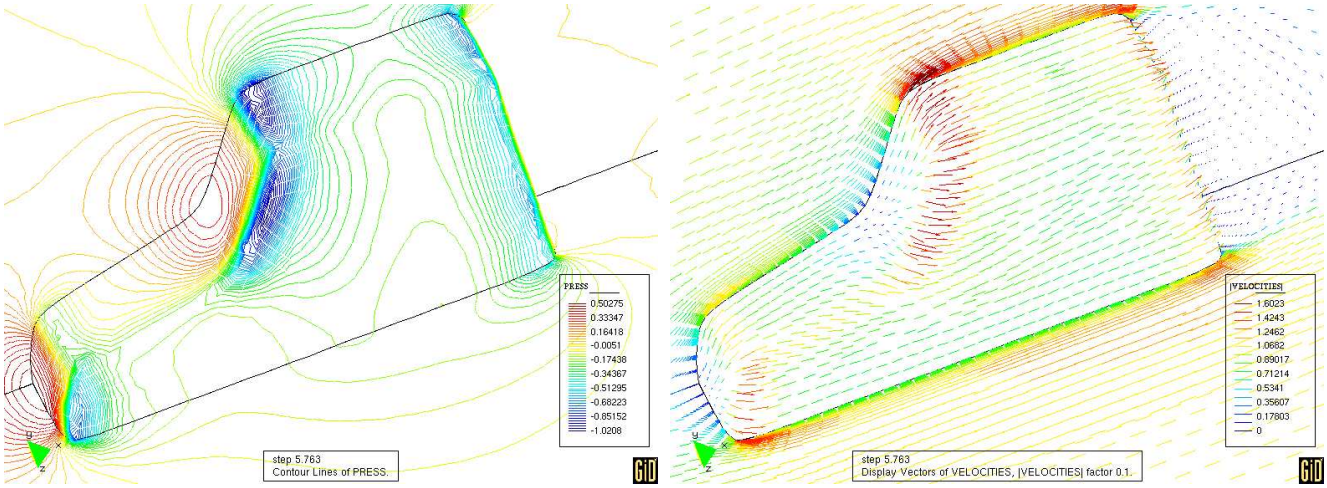


Shape Evolution

SUV-LIKE OBJECT (3)



SUV: Objective Function Evolution



Pressure and Velocity on Final Shape

OUTLOOK: HIERARCHICAL DESIGN

Key Ideas:

- Design: Information Creation Process
- Match Physics to Information Context
- Evolve Physics With Design

For CFD:

- Database
- Lifting Line/Surface
- Potential (Coarse/Fine)
- Euler (Coarse/Fine)
- RANS (Coarse/Fine)
- LES (Coarse/Fine)
- ...