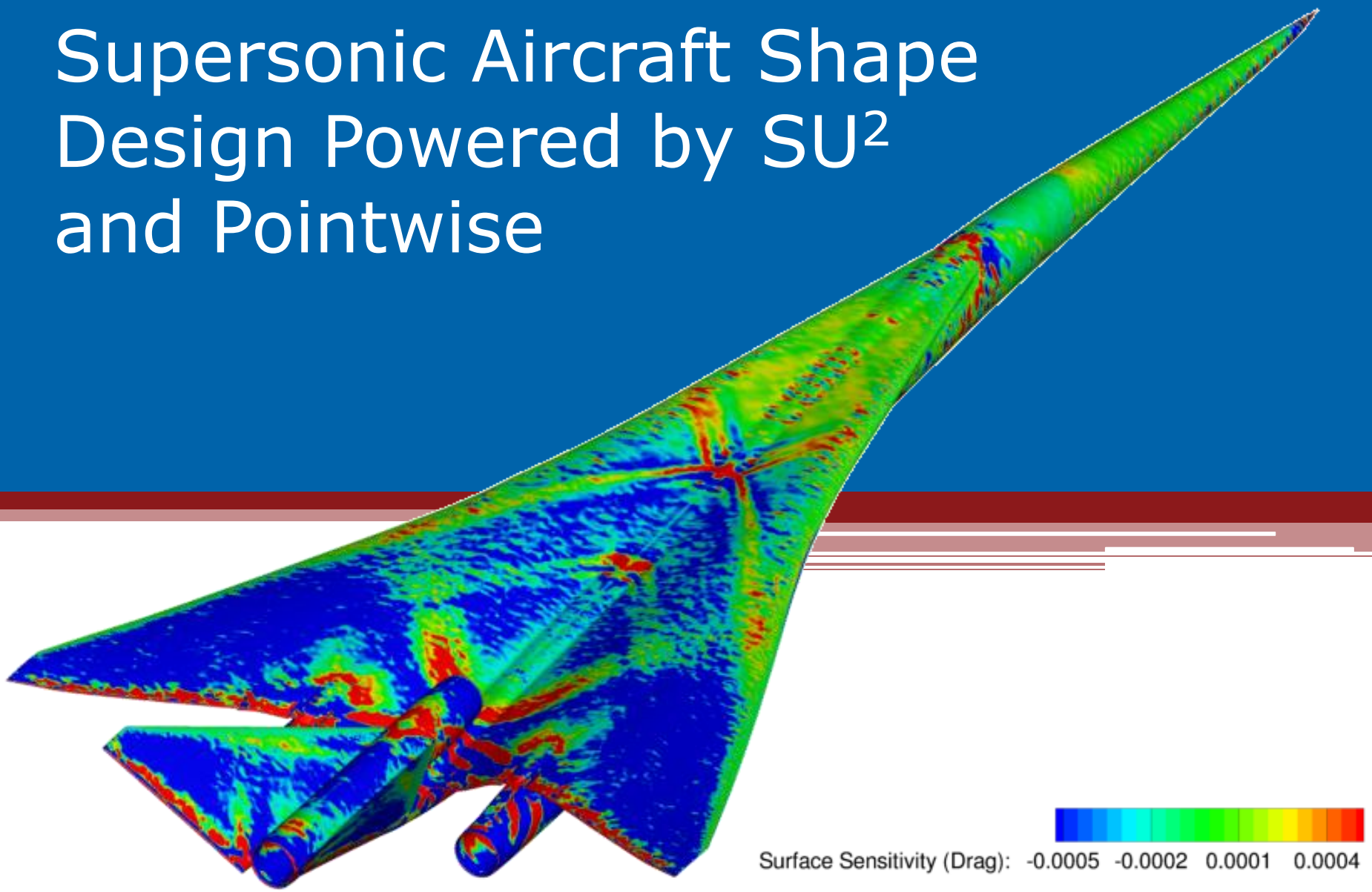
Supersonic Aircraft Shape Design Powered by SU² and Pointwise



SU²

POINTWISE®

Presenters

Travis Carrigan

Travis Carrigan joined Pointwise as a senior engineer after completing his M.S. in aerospace engineering at The University of Texas at Arlington in May 2011 where his graduate research involved aerodynamic design optimization. He interned at Pointwise beginning May 2008, producing demonstration and application videos and working in technical support, doing grid projects and quality assurance testing. During a prior internship at Vought Aircraft Industries, Mr. Carrigan worked as a quality engineer on the Boeing 787 Dreamliner Program. As a senior engineer at Pointwise, Mr. Carrigan works with clients to strengthen their CFD processes by developing interactive and automated meshing solutions.

Thomas Economon

Thomas D. Economon is currently a PhD candidate in the Aerospace Design Laboratory (ADL) within the Department of Aeronautics & Astronautics at Stanford University. His research focuses on the development of new design methodologies for aerospace systems, including high-fidelity, adjoint-based techniques for optimal shape design, as well as tools for design at the conceptual level. He has extensive experience with high performance computing and the development of CFD platforms, most notably as a member of the core development team for the open-source SU^2 software suite. He holds a BS in Aerospace Engineering from the University of Notre Dame (2008) and an MS in Aeronautics & Astronautics from Stanford University (2010).

SU^2

POINTWISE®

Presenters

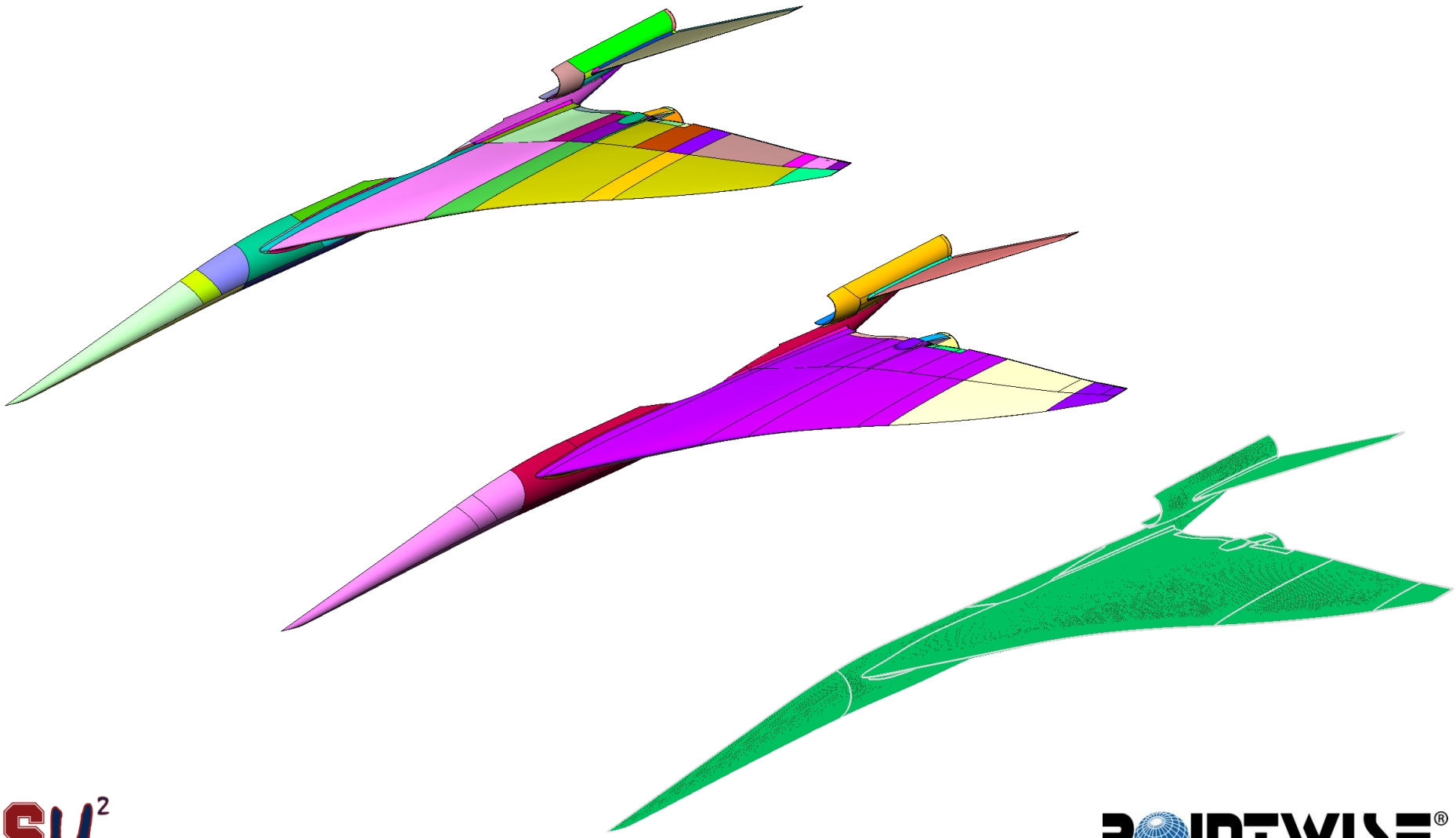
Francisco Palacios

Dr. Palacios is an Engineering Research Associate at the Department of Aeronautics and Astronautics at Stanford University. His main areas of expertise include optimal aerodynamic shape design, large-scale multi-physics CFD simulations, and numerical analysis. Prior to his arrival at Stanford University in 2011, Dr. Palacios was research lead for technological innovation at the Madrid Institute for Advanced Studies and coordinator of the Airbus program "Future Simulation Concepts". From 2011 to 2013, Dr. Palacios was responsible for the development of the main RANS solver for the Stanford PSAAP center, and for the research in adjoint-based methodologies to manage the uncertainties that are present in hypersonic combustion environments. Currently, Dr. Palacios research is focused on supersonic aircraft design and the development of novel shape design techniques applied to multi-physics problems.

Trent Lukaczyk

Trent Lukaczyk is a Ph.D. candidate in the Aerospace Design Lab at Stanford University. His core research interests are in aircraft design and optimization methods, and he is currently contributing to the design of NASA's next-generation Supersonic Passenger Jet. This work depends on experience of various disciplines that he gained from several internships and during his undergraduate education at Cornell University: developing meshing tools, simulating the aerodynamics of both aircraft and automobiles, testing those designs in the wind tunnel, and even designing combustion engines.

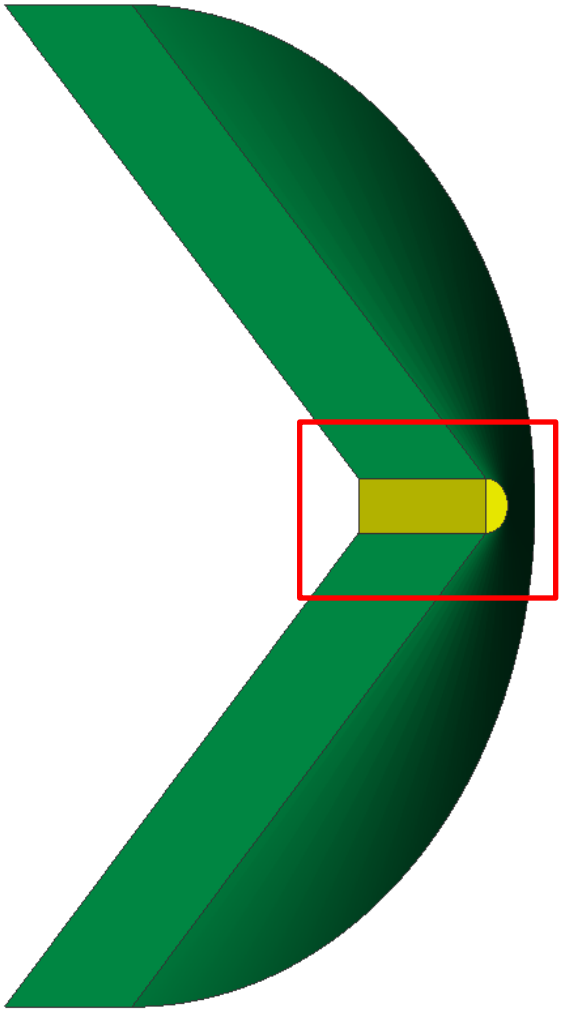
Solid Meshing

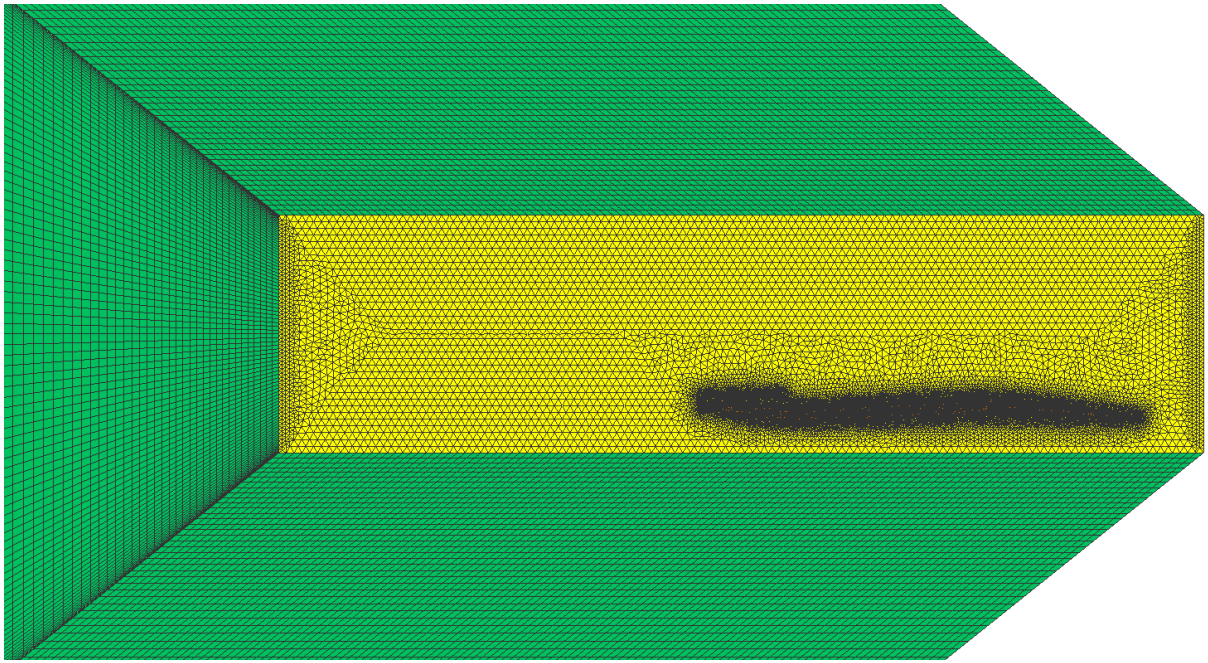


SU²

POINTWISE®

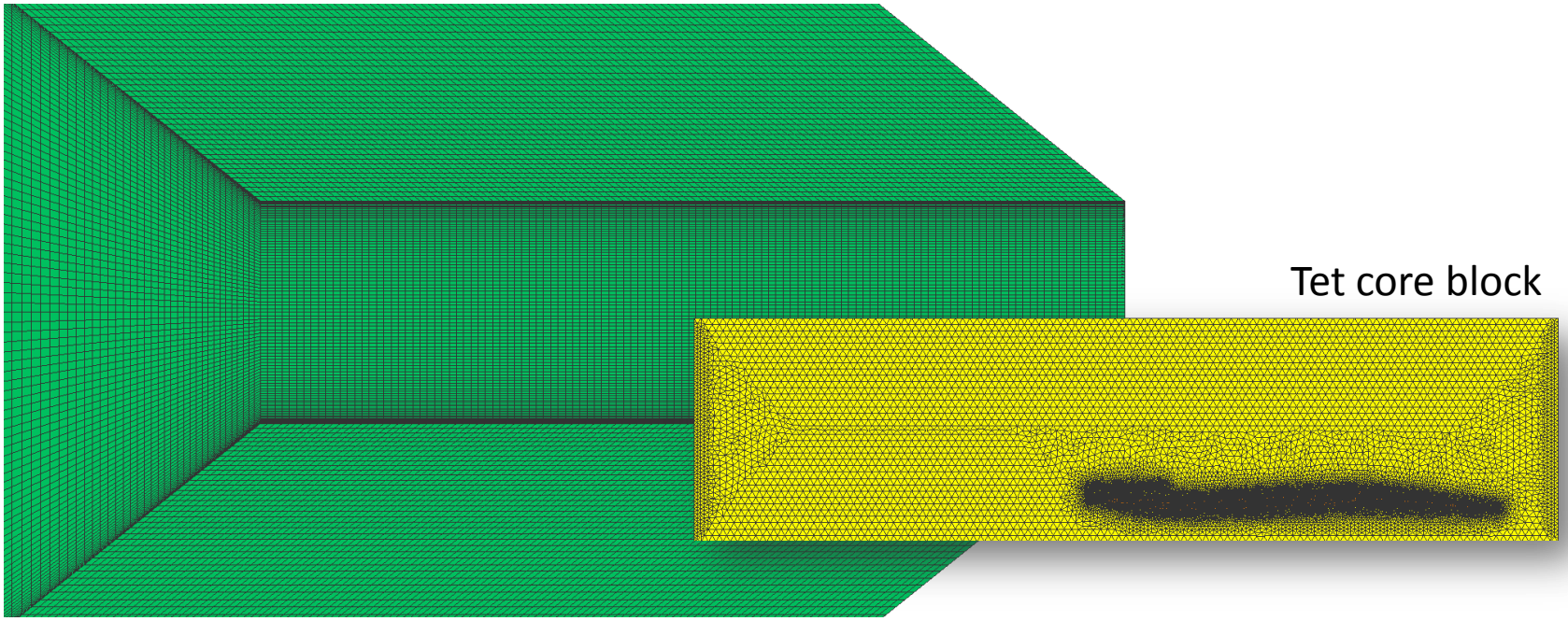
Volume Meshing



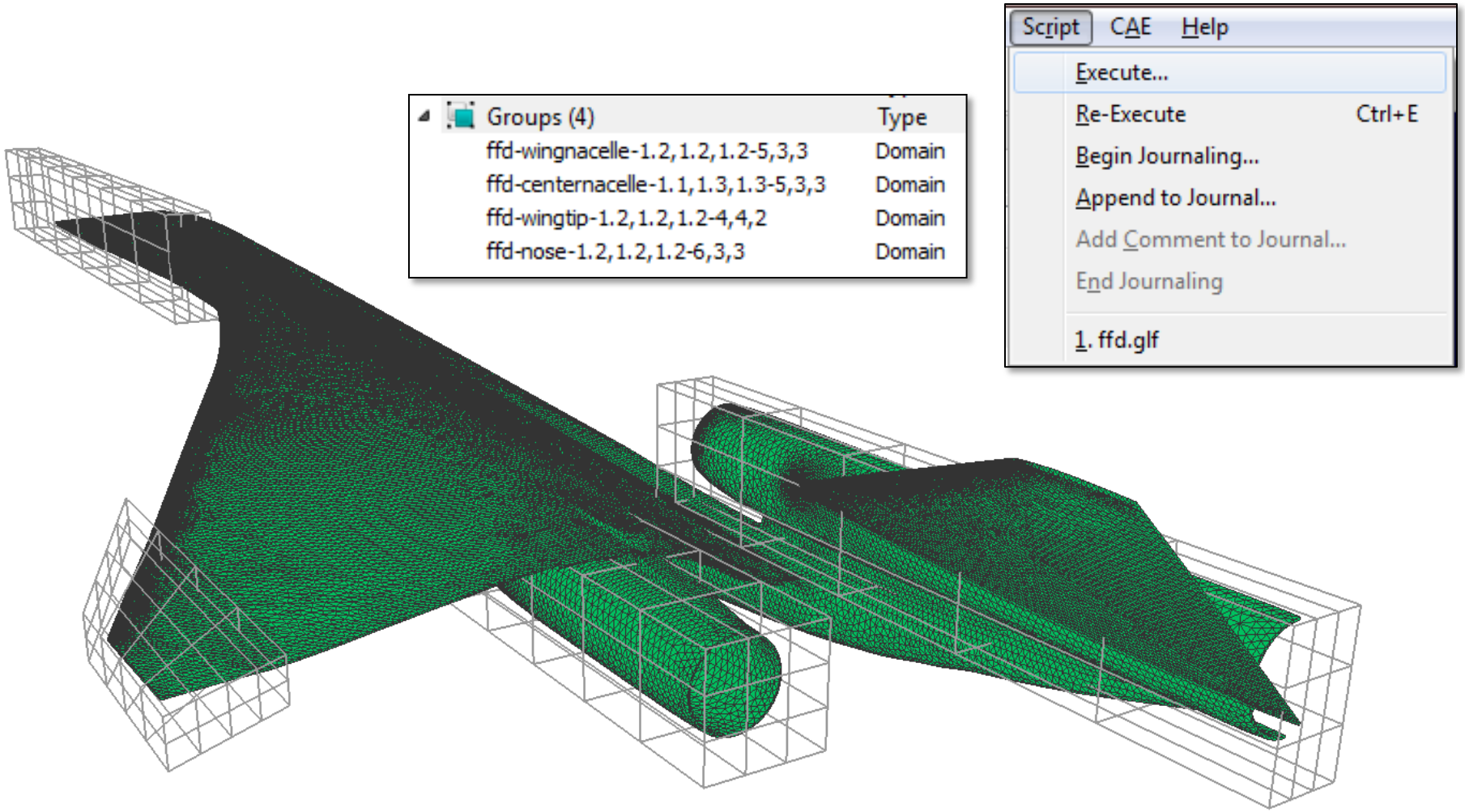


Volume Meshing

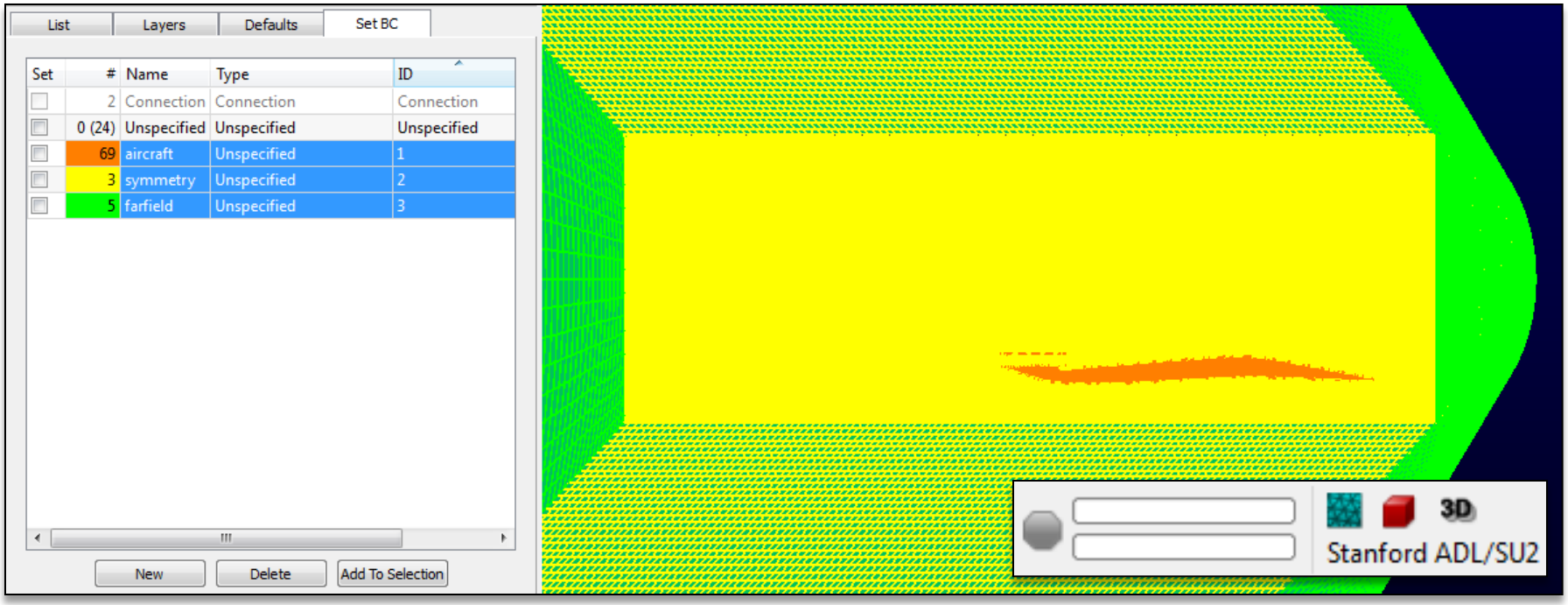
Mach-aligned hex elements



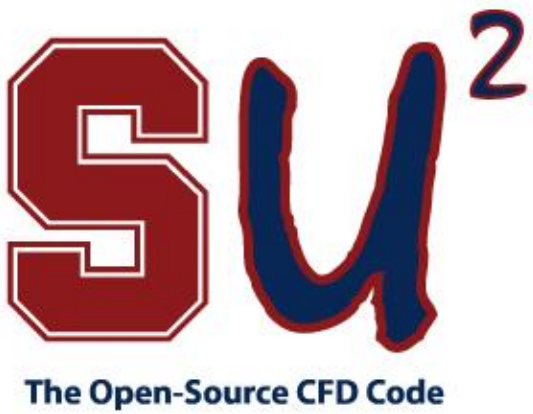
FFD Boxes


Groups (4)
Type
ffd-wingnacelle-1.2,1.2,1.2-5,3,3
Domain
ffd-centernacelle-1.1,1.3,1.3-5,3,3
Domain
ffd-wingtip-1.2,1.2,1.2-4,4,2
Domain
ffd-nose-1.2,1.2,1.2-6,3,3
Domain
Script
CAE
Help
Execute...
Re-Execute
Ctrl+E
Begin Journaling...
Append to Journal...
Add Comment to Journal...
End Journaling
1. ffd.glf

Native SU² Support



Shape Design in SU²



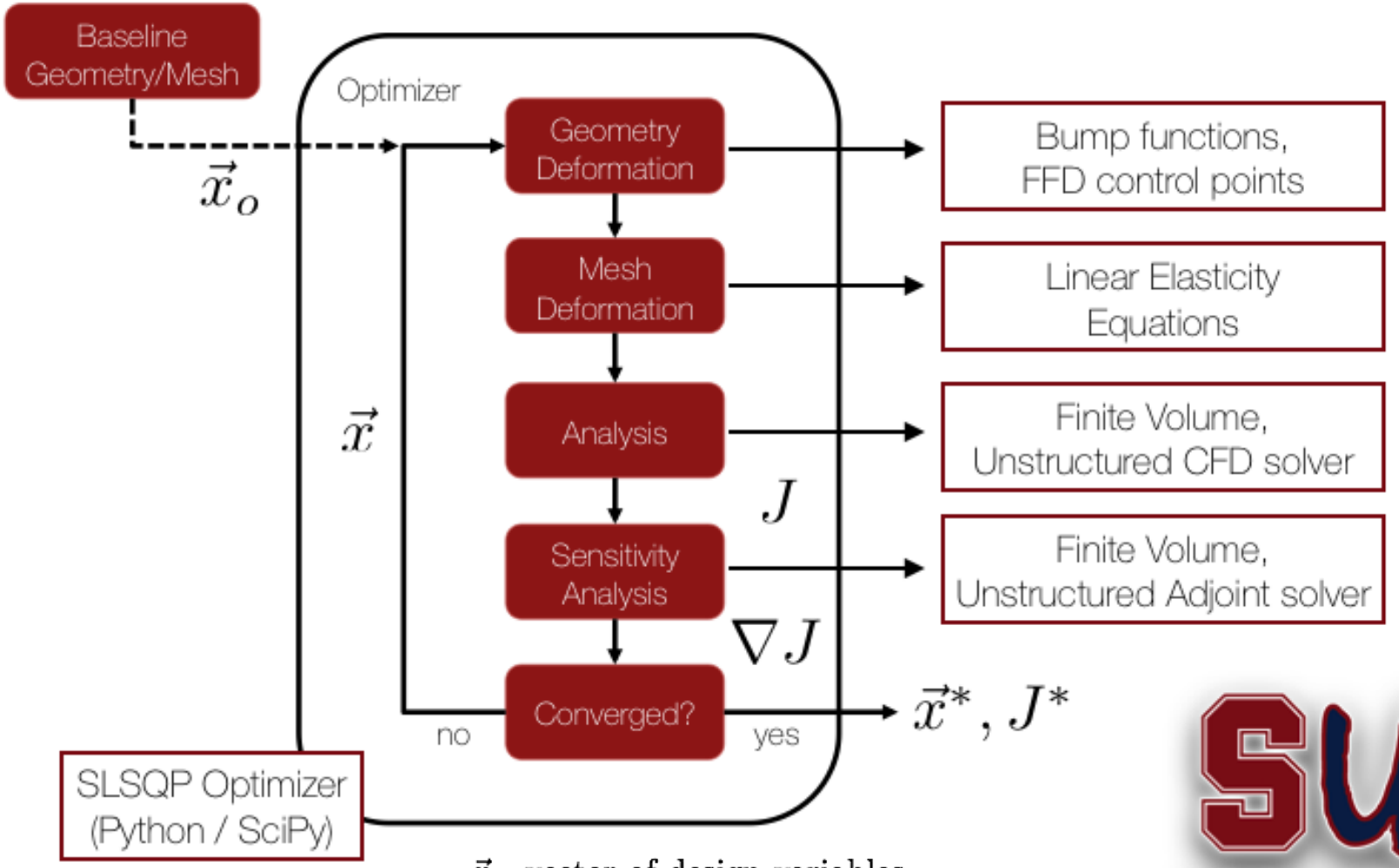
Today's webinar is based on SU² V3.1. See **http://su2.stanford.edu** for much more information!

- **Open-source suite** for CFD, shape design, + much more
- SU² is also capable of **multi-physics** calculations
- Industry-standard numerical methods on unstructured grids
- Object-oriented C++ modules and Python scripts
- Once a baseline geometry/mesh is available, SU² can be used for analyzing performance, evaluating sensitivities, and **automatic shape design**

SU²

POINTWISE®

Optimal Shape Design Loop



$\vec{x}$ vector of design variables

J objective/constraint function(s)

SU2 The New Open-Source CFD Code

POINTWISE®

Shape Design in SU²

- There are a lot of moving parts in the design loop…
- However, SU² provides everything you need for design
- Configure each module within a single **config file** (.cfg)
- With everything in place, call **shape_optimization.py**
- Python automates the shape design loop for you

```
$ shape_optimization.py –f LM-1021.cfg –p 96
```

‘-f’ specifies config filename

‘-p’ specifies number of procs for parallel jobs

Shape Design in SU²

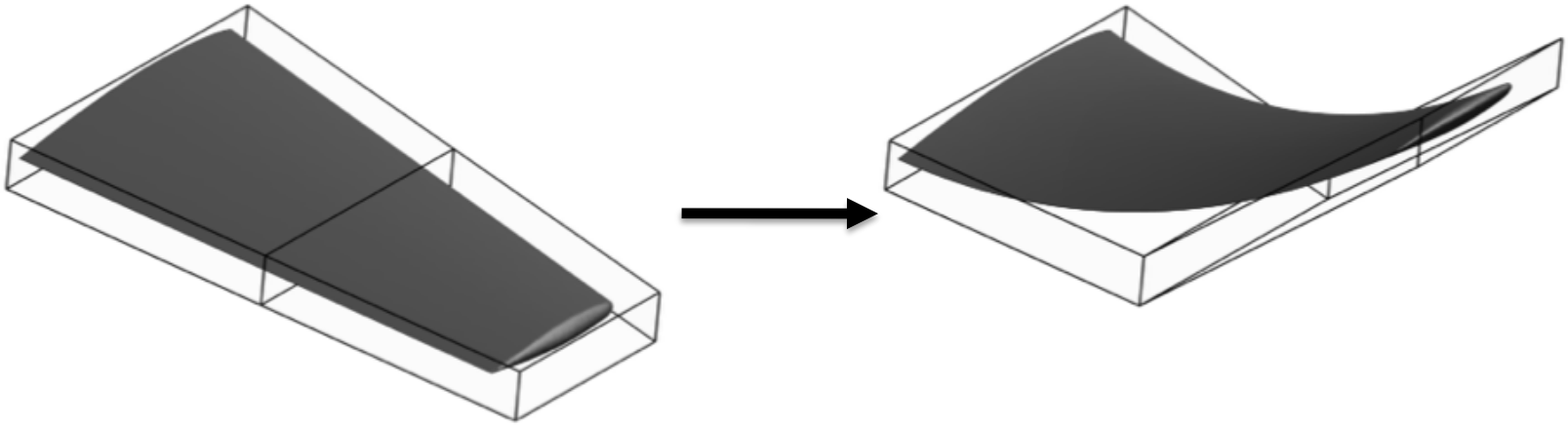
- Python orchestrates the automatic shape design loop
- Can easily adapt a config file from an existing case
- For the complex supersonic configuration, we will pose a realistic shape optimization problem:
  - Minimize drag (this objective drives the optimizer)
  - Decrease the pitching moment by 25%
  - Maintain 95% of the original lift
  - Maintain 90% of the original maximum thicknesses at 5 wing sections
  - It can be difficult to reduce drag given a good starting configuration based on supersonic theory
  - Our problem will trim the aircraft by changing the wing shape

So, what do you need to know in order to set up your own shape design problems?

SU²

POINTWISE®

Geometry Parameterization



- Parameterize by a **Free-Form Deformation** (FFD) approach
- Technique has origins in the computer graphics industry
- Encapsulate geometry in a bounding box and create a mapping between the FFD control points and the mesh surface nodes
- FFD control points become the design variables (DVs) with the surface inheriting a smooth deformation
- FFD variables are now available in both 2D & 3D with SU^2 V3.1

SU^2

POINTWISE®

Geometry Parameterization

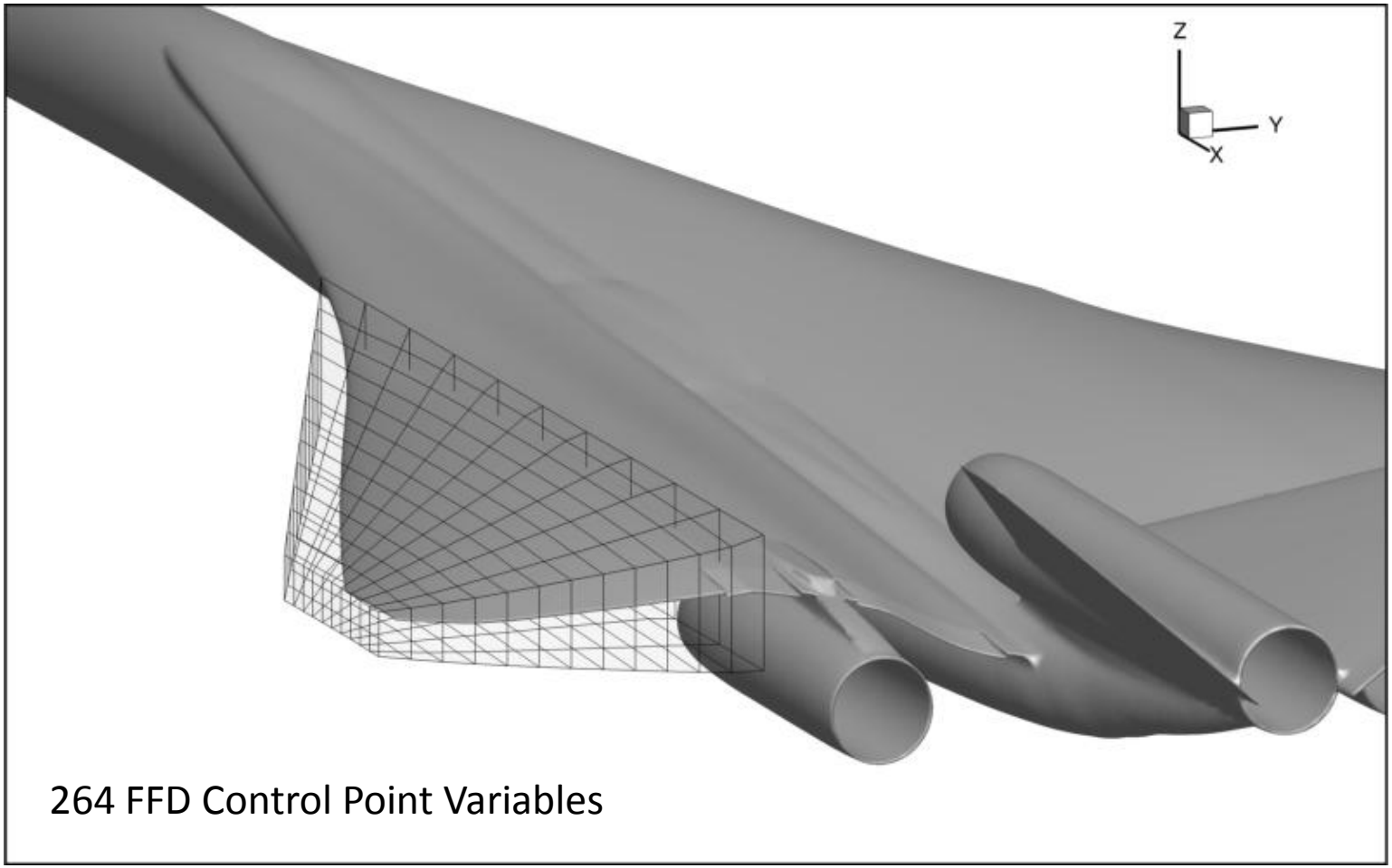
- Set up FFD boxes at bottom of the SU² native mesh file
- The new script for Pointwise can generate this information
- Initialize the mapping for the FFD box by running SU2_MDC once with option **DV_KIND= FFD_SETTING**
- After completion, a new copy of the mesh with all FFD information is written to disk
- Use this new mesh for shape optimization

FFD box specification for our LM 1021 problem

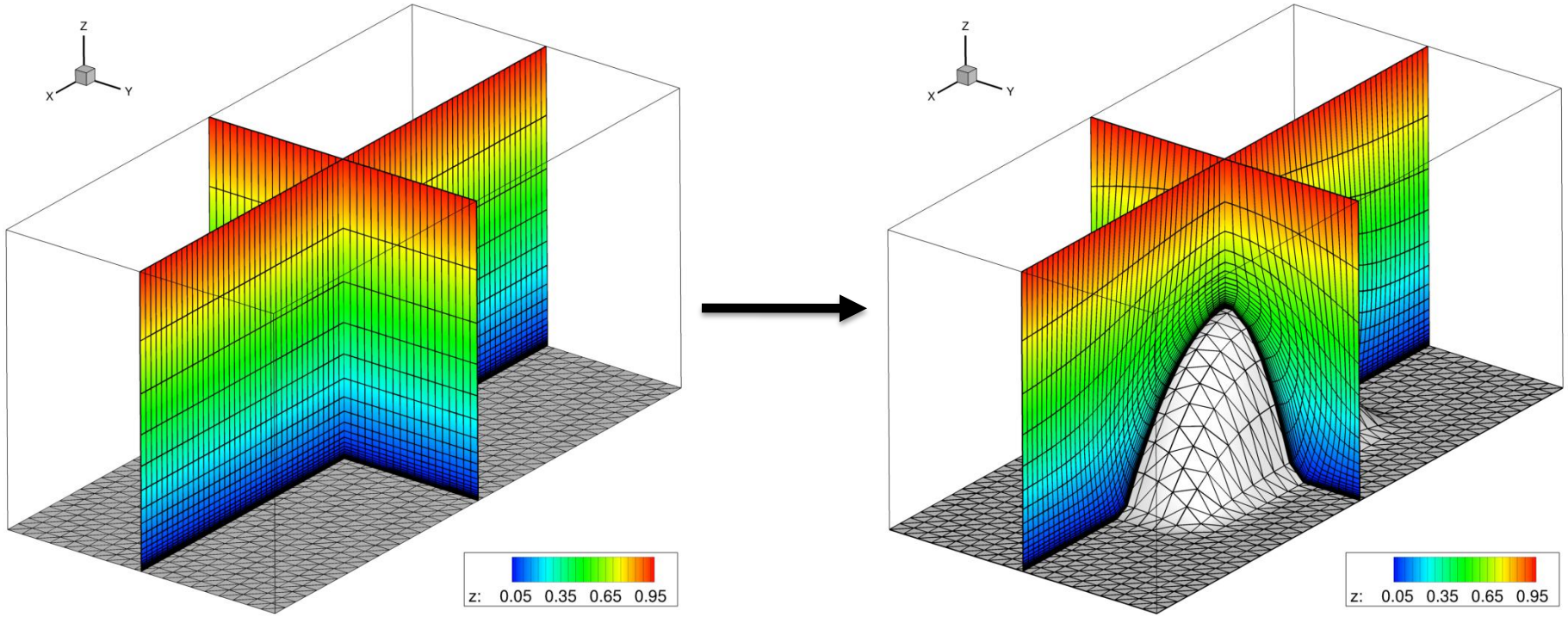
```
FFD_NBOX= 1
FFD_NLEVEL= 1
FFD_TAG= 0
FFD_LEVEL= 0
FFD_DEGREE_I= 5
FFD_DEGREE_J= 6
FFD_DEGREE_K= 1
FFD_PARENTS= 0
FFD_CHILDREN= 0
FFD_CORNER_POINTS= 8
0.502544    -0.044427    0.0488
0.352456    -0.044427    0.0488
0.503809    -0.107234    0.06271
0.53585     -0.107234    0.06271
0.502544    -0.044427    0.06534
0.352456    -0.044427    0.06534
0.503809    -0.107234    0.06828
0.53585     -0.107234    0.06828
FFD_CONTROL_POINTS= 0
FFD_SURFACE_POINTS= 0
```

Geometry Parameterization

FFD box for redesigning the Lockheed Martin 1021 configuration



Mesh Deformation



- Need robust mesh deformation technique…
- Treat the mesh as an elastic solid with non-uniform stiffness
- Solve the **linear elasticity equations** on the mesh for the nodal displacements using the movement of the boundaries as input

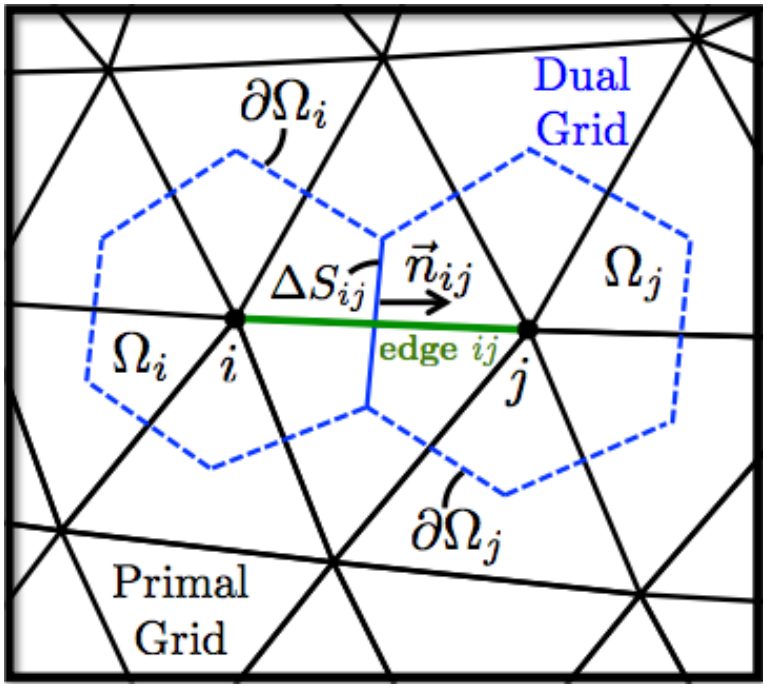
Mesh Deformation

```
% ------------------------ GRID DEFORMATION PARAMETERS -----------------------%
%
% Number of smoothing iterations for FEA mesh deformation
DEFORM_LINEAR_ITER= 500
%
% 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
%
% Factor to multiply smallest cell volume for deform tolerance (0.001 default)
DEFORM_TOL_FACTOR = 0.001
%
% Type of element stiffness imposed for FEA mesh deformation (INVERSE_VOLUME,
%                                      WALL_DISTANCE, CONSTANT_STIFFNESS)
DEFORM_STIFFNESS_TYPE= INVERSE_VOLUME
%
% Visualize the deformation (NO, YES)
VISUALIZE_DEFORMATION= NO
%
% Divide elemetns into triangles and tetrahedra
DIVIDE_ELEMENTS= YES
```

Controls type of cell stiffness. Smaller cells or those near walls will be more rigid under deformation (preserves mesh quality).

Mesh deformation options in SU^2 V3.1.

Flow & Adjoint Problem Setup



- Unstructured meshes with median-dual control volumes (vertex-based)
- Finite Volume Method with second-order schemes
- For the supersonic problem, we'll focus on **JST** spatial integration (centered) with **implicit time integration**
- Similar schemes for integrating the flow and adjoint equations (adjoint scheme is non-conservative)

SU²

POINTWISE®

Optimization Problem

- Lastly, define your optimization problem in the config file
- Choose an objective, constraints (including geometric), and specify your design variables
- Full specifications can be found in **config_template.cfg**
- Scaling factor helps optimizer take appropriate first step

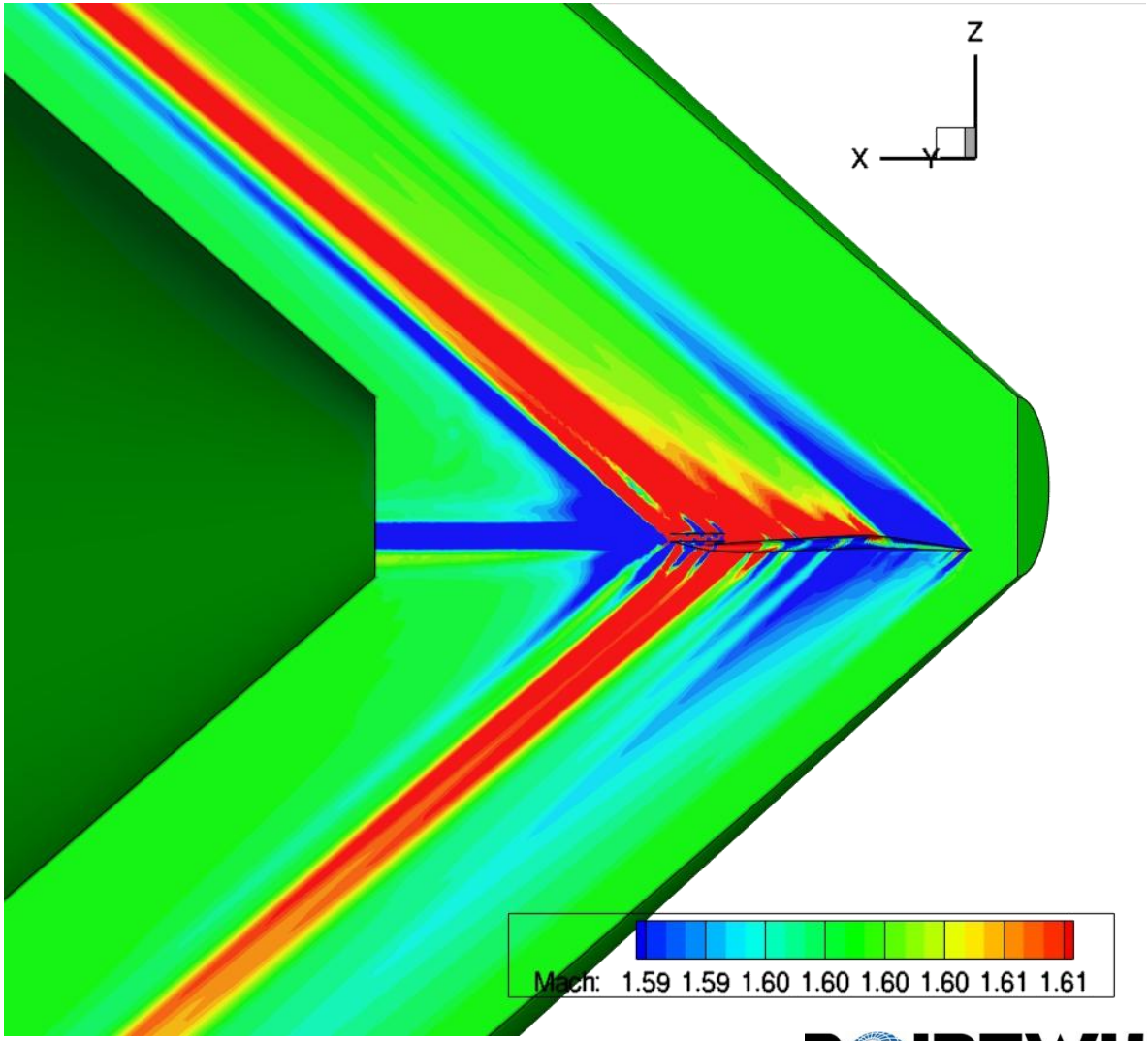
```
% Optimization objective function with scaling factor
% ex= Objective * Scale
OPT_OBJECTIVE= DRAG * 0.01
%
% Optimization constraint functions with scaling factors, separated by semic
olons
% ex= (Objective = Value ) * Scale, use '>','<','='
OPT_CONSTRAINT= (MOMENT_Y < 0.0135919515) * 0.01; (LIFT > 0.13212308464) * 0
.01; (MAX_THICKNESS_SEC1 > 0.0035208432) * 0.01; (MAX_THICKNESS_SEC2 > 0.002
7518184) * 0.01; (MAX_THICKNESS_SEC3 > 0.0020452032) * 0.01; (MAX_THICKNESS_
SEC4 > 0.0013014216) * 0.01; (MAX_THICKNESS_SEC5 > 0.00052060608) * 0.01
%
% Optimization design variables, separated by semicolons
DEFINITION_DV= ( 7, 1.0 | aircraft | 0, 0, 1, 0, 0.0, 0.0, 1.0 ); ( 7, 1.0 |
 aircraft | 0, 1, 1, 0, 0.0, 0.0, 1.0 ); ...
```

Shape Design in SU²

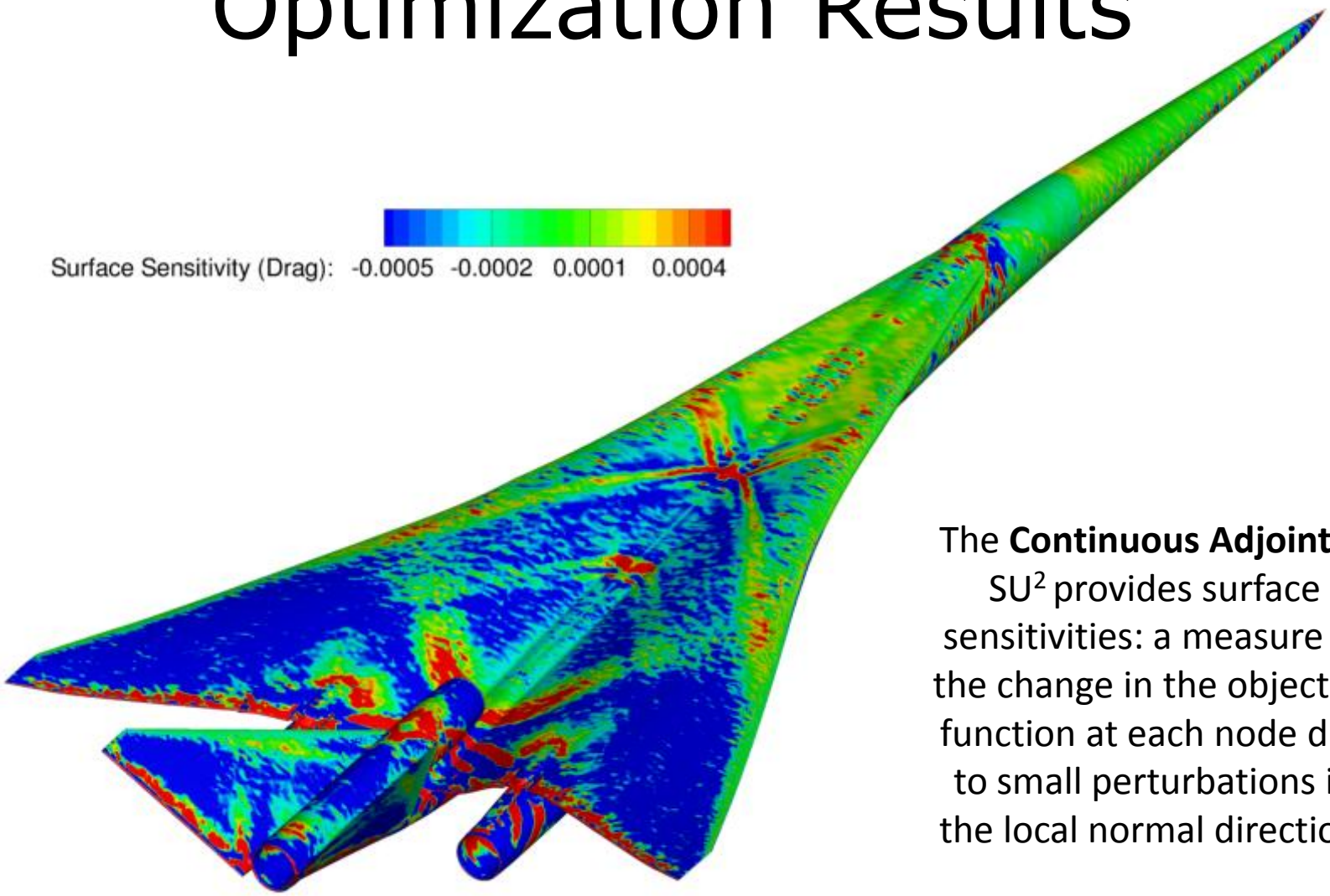
```
$ shape_optimization.py -f LM-1021.cfg -p 96
```

Optimization Results

Shock-aligned hex cells can help maintain accuracy and reduce cost for boom predictions when extracting off-body pressure signatures.

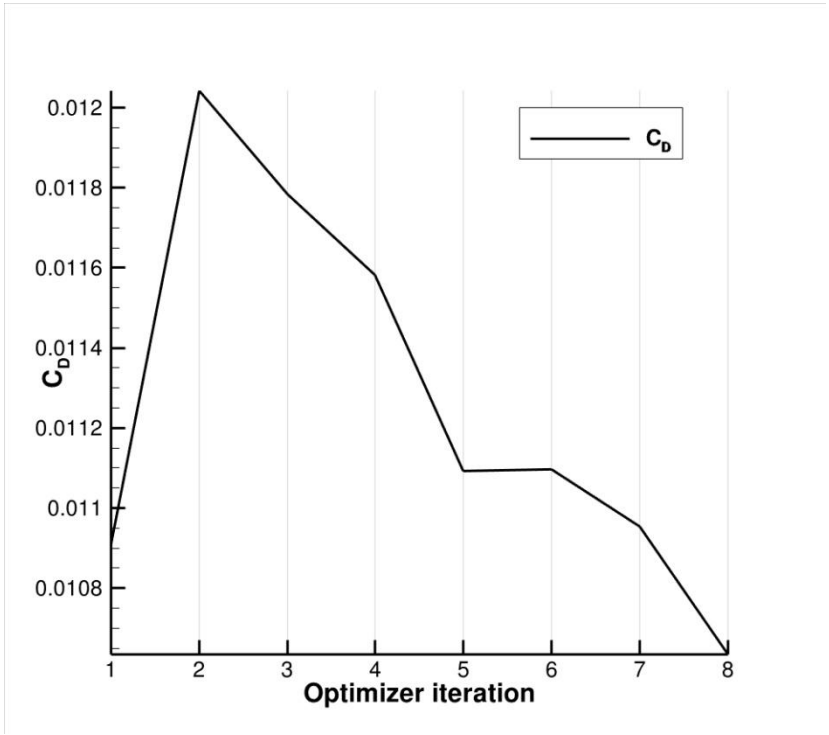


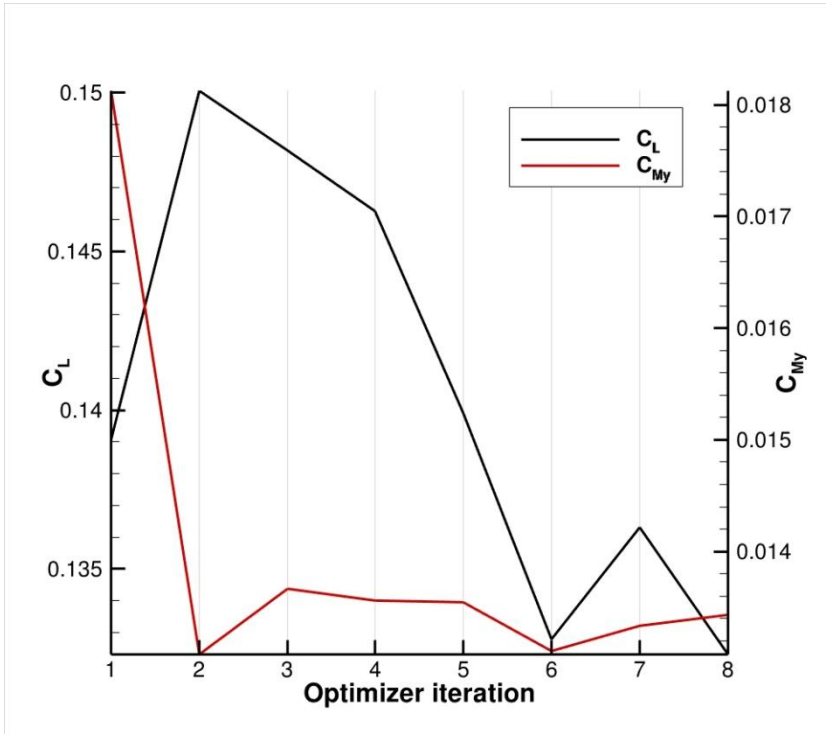
Optimization Results



The **Continuous Adjoint** in SU^2 provides surface sensitivities: a measure of the change in the objective function at each node due to small perturbations in the local normal direction.

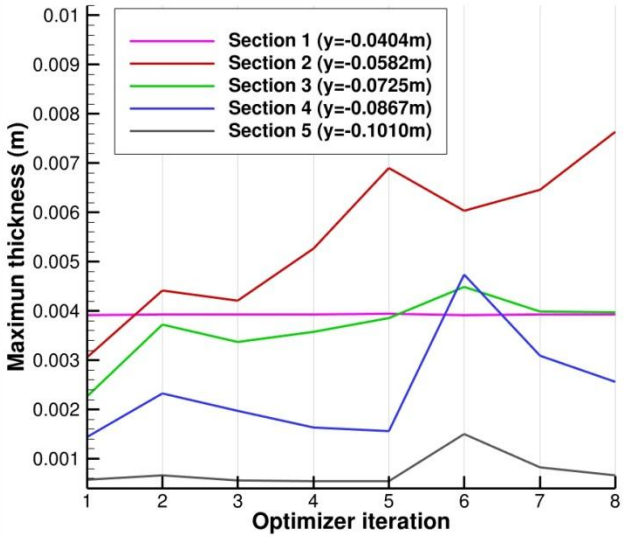
Objective & Flow Constraints



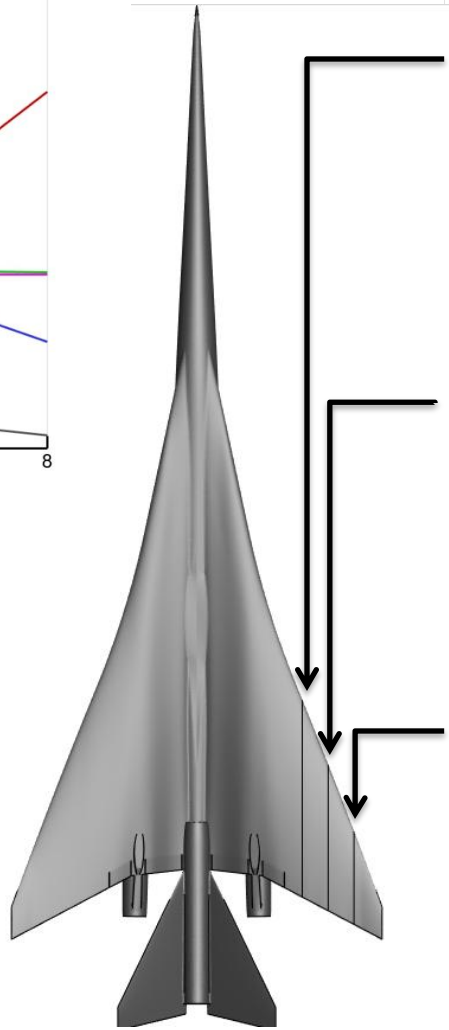


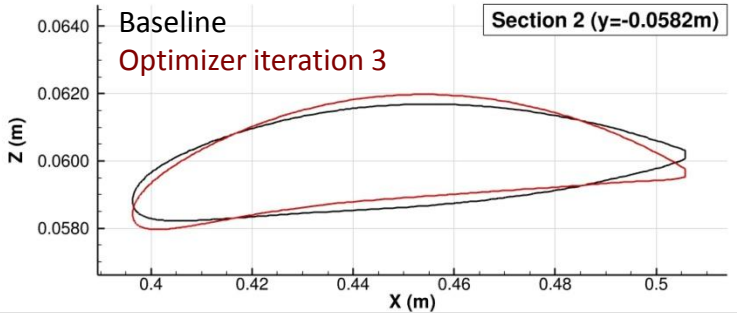
The final design (optimizer iteration 8) satisfies C_L and C_{My} constraints with a smaller C_D coefficient (3 drag counts less than the baseline value).

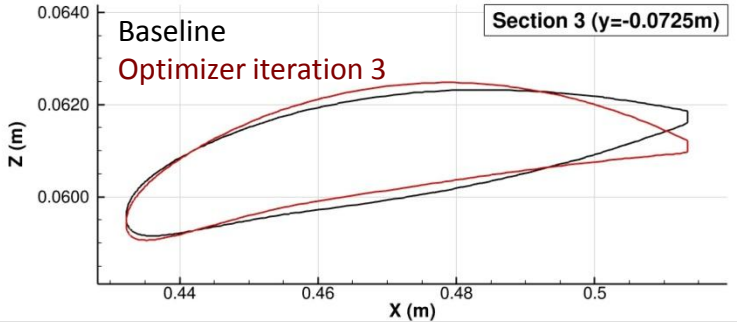
Geometric Constraints

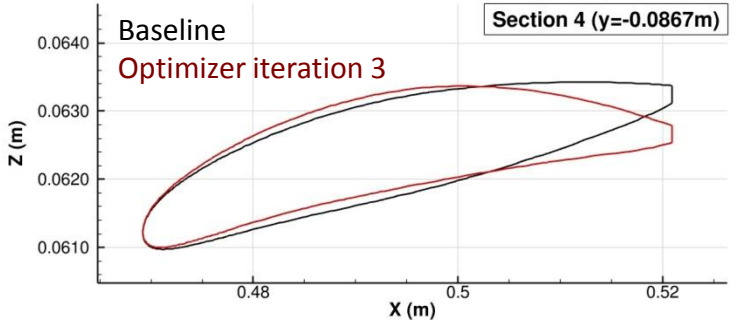


However, there is an inadmissible increase in the maximum thickness of section 2 at iteration 4 (a new geometrical constraint is required).

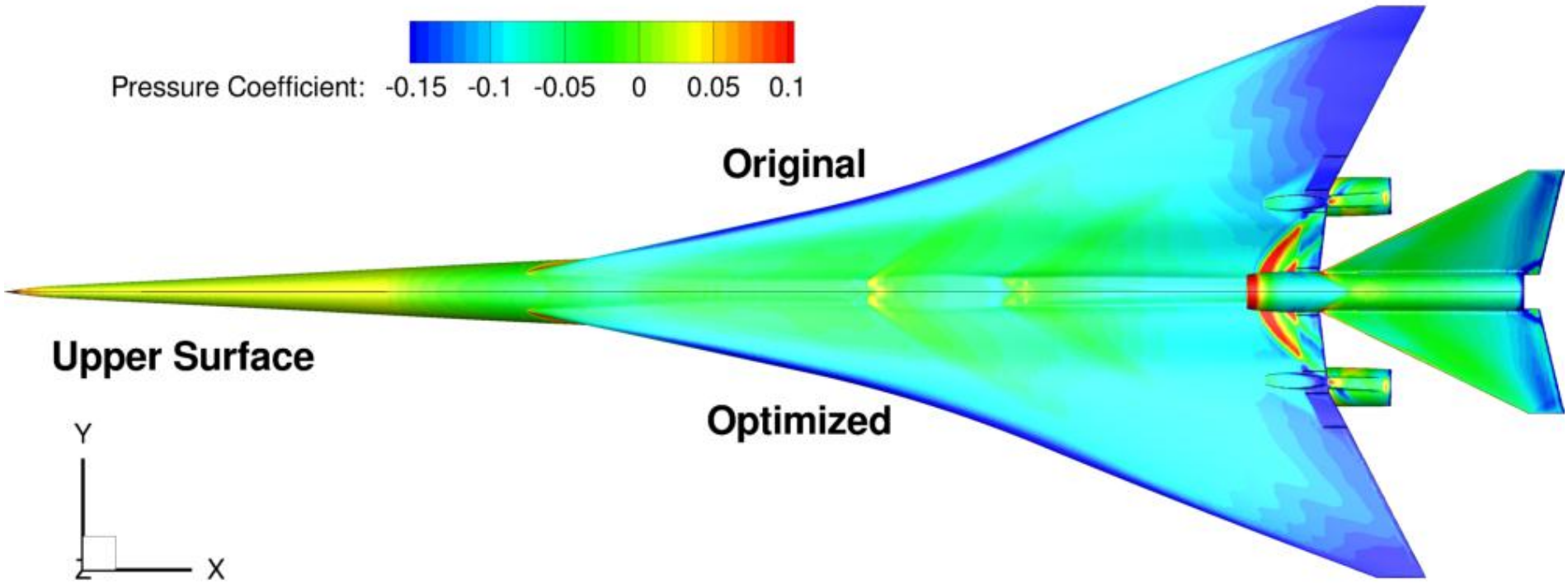




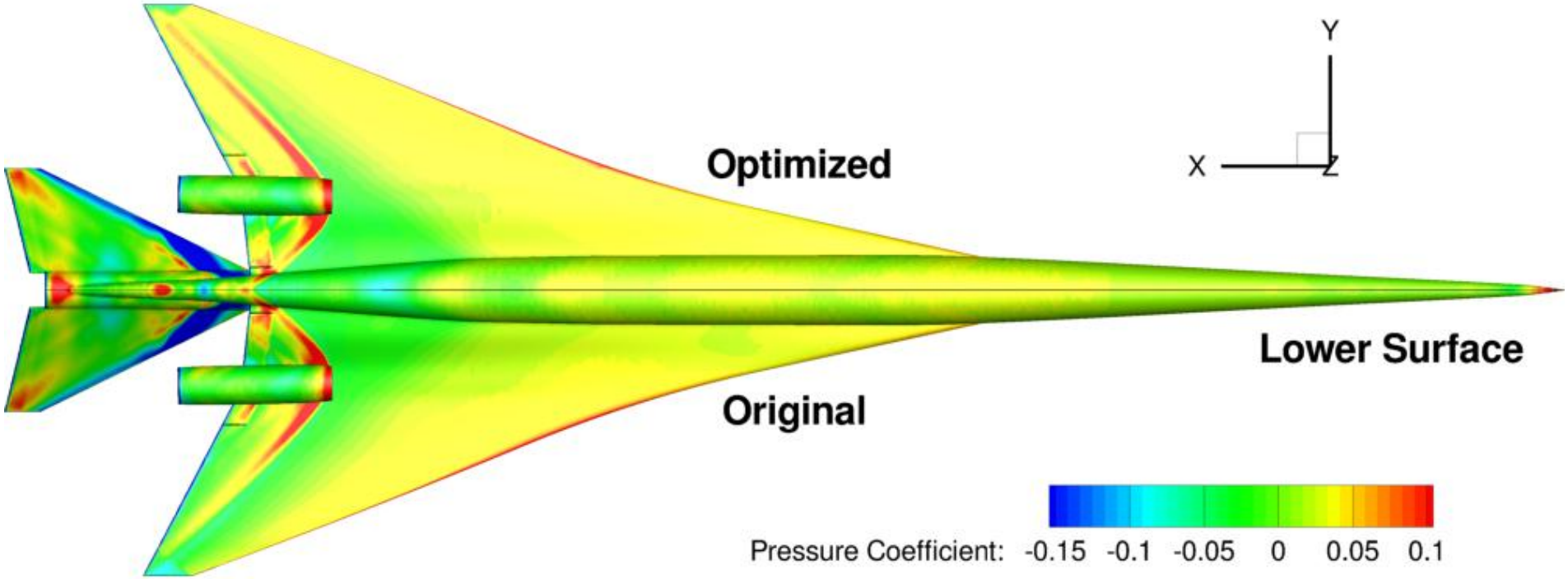




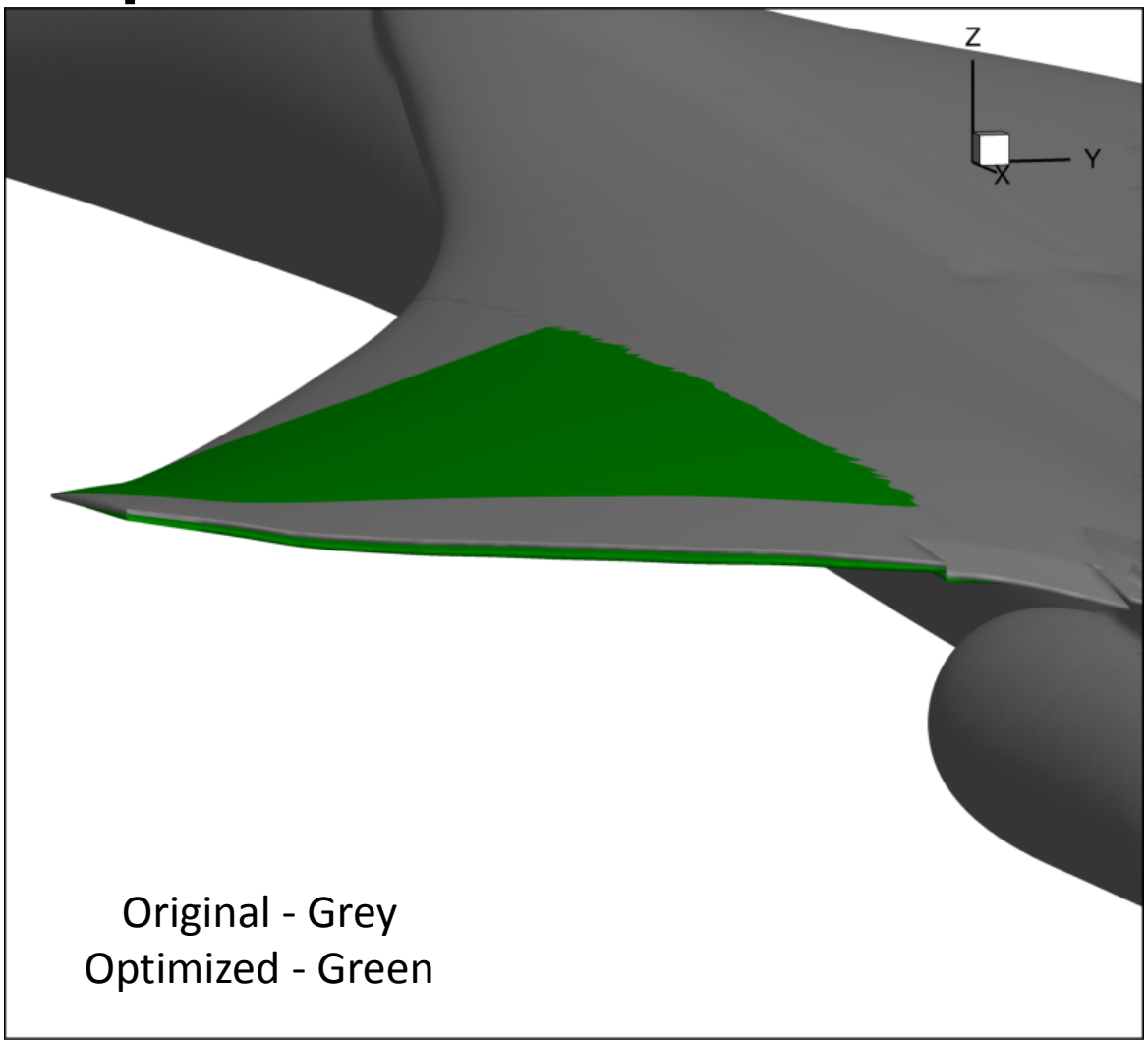
Optimization Results



Optimization Results



Optimization Results



We've Only Scratched the Surface…

- Promising results, but this is just the first design iteration
- Add more geometric constraints to the wing sections
- Consider angle of attack during optimization
- Adjust FFD design variables (easy w/ Pointwise script):
  - Change the shape or move the FFD box around on wing
  - Add more FFD control points
  - Hold some FFD control point variables fixed near inboard sections
  - Try the FFD camber or FFD thickness design variables
  - Add more FFD boxes to other locations (nacelles?)
  - Insert nested FFD boxes for finer control (local to leading edge?)
- Once a design is finalized, SU2_MDC can export .stl files of the geometry to reintegrate with CAD/manufacturing

SU²

POINTWISE®

Thank-you for attending our webinar

The grid files and scripts used in this webinar as well as the recorded video of the webinar will be available on our web sites tomorrow.

We would like to acknowledge the NASA N+2 Supersonic Program and the Lockheed Martin Corporation.

www.pointwise.com/webinar

http://su2.stanford.edu/training.html

If you have any questions about Pointwise, please email Travis Carrigan at webinar@pointwise.com.

If you have any questions about SU^2, please email Tom Economon at economon@stanford.edu.

SU^2

POINTWISE®