Adjoint-based method for supersonic aircraft design using equivalent area distributions

Francisco Palacios*, Juan J. Alonso†, Michael Colonno‡, Jason Hicken§, Trent Lukaczyk¶
Stanford University, Stanford CA 94305, USA

This paper describes the efforts in the development of new adjoint-based techniques for the design of supersonic aircraft that must match a target equivalent area distribution at constant lift coefficients. This specific approach is meant to be part of a larger effort to create multidisciplinary analysis and optimization capabilities that can be used to design low-boom supersonic aircraft with adequate aerodynamic and structural performance. The paper focuses on the description of an inverse equivalent area distribution capability that includes the flow analysis, the formulation and implementation of an adjoint solver for the equivalent area distribution, and the shape design process of a trijet supersonic aircraft. This shape design effort is challenging due to three factors: the complexity of the geometric details involved that require a full unstructured flow and adjoint solver methodology, the *apriori* grid adaptation process needed for capturing the shock and expansion waves in the near-field (without the need for additional mesh adaptation), and the selected objective function for the calculation of sensitivities and the ultimate application in aircraft design. The viability of our approach is demonstrated with some preliminary examples. Ongoing work is targeting actual configurations that can result in an actual low-boom aircraft.

I. Introduction

As part of the NASA Supersonics Project N+2 effort, Stanford University has developed an adjoint-based capability for the inverse shape design of supersonic configurations that must match a target equivalent area distribution (chosen for its beneficial ground-boom properties). The long-term intent is to incorporate such a capability into a generic multi-disciplinary analysis and optimization (MDAO) framework that, in addition to aerodynamic performance and sonic-boom considerations, incorporates other important aspects of the design in order to reach a balanced, realizable, low-boom aircraft.

The work described in this article includes the derivation, implementation, and testing of an adjoint solver that is able to produce sensitivities (gradients) of cost functions derived from the aircraft equivalent area distribution with respect to arbitrary numbers of design variables that parameterize its shape. Owing to the properties of adjoint solvers, the entire vector of sensitivities (the gradient) can be obtained with a single flow solution followed by a single adjoint solution, thus leading to large computational efficiencies.

A schematic view of our approach to low-boom shape design is depicted in Fig. 1 below. This Figure emphasizes that the low-boom shape design module consists of various components that must be integrated themselves, prior to full integration with a true MDAO environment. In particular, geometries need to be created and modified for design purposes, meshes must be deformed during the optimization procedure, flow and adjoint solutions must be calculated (according to a new formulation described in this article) and

*Engineering Research Associate, Department of Aeronautics and Astronautics, AIAA Member.
†Associate Professor, Department of Aeronautics and Astronautics, AIAA Senior Member.
‡Engineering Research Associate, Department of Aeronautics and Astronautics, AIAA Member.
§Post-doctoral Researcher, Department of Aeronautics and Astronautics, AIAA Member.
¶Graduate student, Department of Aeronautics and Astronautics, AIAA Member.

gradients must be computed in order to supply them to the optimization procedure. In particular, this article focuses on:

- The description of our geometry treatment (using both a parametric CAD approach and a free-form deformation box approach),
- The mesh deformation algorithm (based on the work of Campbell) in order to resolve the near-field pressure distribution accurately,
- The formulation of the adjoint solver for low-boom design (based on cost functions related to the equivalent area distribution), and
- The results of various demonstration optimization runs that we have been pursuing and that leverage all other components.

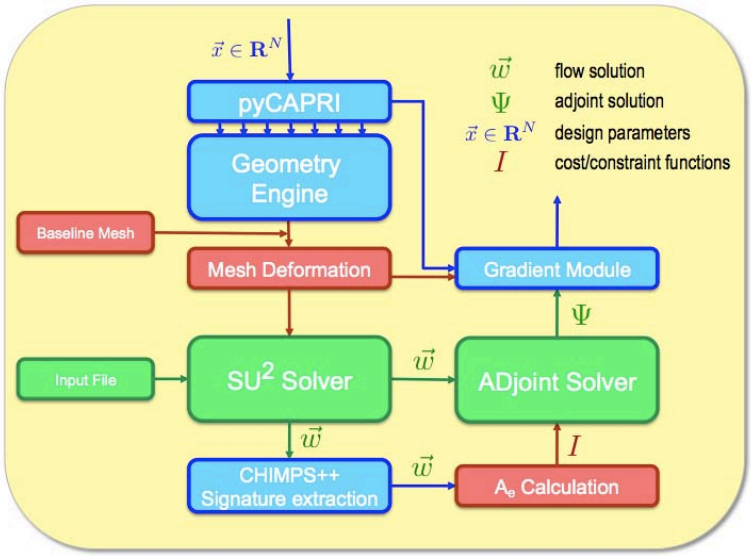


Figure 1. Overall Approach to Supersonic Low-Boom Shape Optimization

Various approaches to low-boom design have been followed in the past, starting from the pioneering work of Seebass, George, and Darden[1] to more recent work by other researchers.[2] A historical overview of such efforts can be found in.[3] An approach that is followed in this work is to rely on the construction (by other means available) of a target equivalent area distribution that is known to translate into acceptable ground-boom signatures. Once that target equivalent area distribution is available, it is possible to ask the question of which aircraft shape (volume and lift distribution) is needed to achieve that target area distribution. Efficient methodologies for such inverse design problems are the focus of this paper. Since CFD methods are likely to be used in the inverse design problem, an alternative approach would be to simply design the shape of the aircraft that leads to a near-field pressure distribution (that is known to translate into a ground boom signature with acceptable behavior). Both approaches have their merits and difficulties, but those are not elaborated further in this work. The reader is referred to[3] for additional references and more details of the pros and cons.

II. Parametric CAD Geometry Interface: pyCAPRI

Ideally, the designer would like optimization to be performed directly on variables that are convenient for design, here referred to as parametric variables. The sweep of a wing section or the width of the fuselage

at a given station are examples of parametric variables. These are distinguished from other design variables in that a change in their value results in a change of the parametric geometry throughout the model and not just in a local change to a given node or cell of a computational model. Hence, changing a parametric variable requires that the model geometry be regenerated (to use CAD terminology) or recomputed from the current parametric inputs via interaction with the relevant software.

In order to compute the sensitivity of the performance metrics of interest (derived from the computed equivalent area distributions) using CFD analysis, the following general procedure must be reliably performed for a given vector of design variables, $\vec{x}$:

1. Geometry generated for $\vec{x}$, $G(\vec{x})$
2. Surface mesh created and analysis conducted for $G(\vec{x})$
3. Geometry perturbed about $G(\vec{x})$ for each x_i of interest
4. Recreate or deform the surface mesh to obtain δG_i
5. Compute the sensitivities in the coordinates of the surface mesh corresponding to δG_i, $\partial\vec{r}/\partial x_i$.

With the sensitivities of the surface mesh to the geometry known, the sensitivities of the performance with respect to the parametric geometry can be found by combining these results in the appropriate way with the sensitivities of the performance with respect to the motion of the surface mesh. Assuming that the sensitivity of a performance metric, J (in our case, related to an aircraft equivalent area distribution), is known with respect to the position of a surface node, r_j, the parametric sensitivity of the performance metric, J, with respect to the parametric design variable i can be found via a dot product (chain rule),

$$\frac{\partial J}{\partial x_i} = \frac{\partial \vec{r}_j}{\partial x_i}\frac{\partial J}{\partial \vec{r}_j} = \frac{\partial x_j}{\partial x_i}\left(\frac{\partial J}{\partial x_i}\right) + \frac{\partial y_j}{\partial x_i}\left(\frac{\partial J}{\partial y_j}\right) + \frac{\partial z_j}{\partial x_i}\left(\frac{\partial J}{\partial z_j}\right) \tag{1}$$

Note that x is used in this context both as a Cartesian spatial coordinate (x, y, z) and as the vector of design variables $(\vec{x})$. In a later section of this article, the analytical tools to compute the sensitivities of the performance metric to the position of a surface node are discussed (which we will call surface sensitivities); here we discuss the tools developed to compute $\partial\vec{r}/\partial x_i$ for an arbitrary parametric solid or surface model.

Note also that, in this description, we compute the sensitivity of the location of a surface point in the model to changes in the parametric shape variables using a process of finite differencing. In general, and in order to minimize the number of CAD regenerations, it would be desirable to obtain these sensitivities analytically. While existing CAD packages do not provide such analytic sensitivities, efforts by Haimes are attempting to automate this process and a package that may be incorporated into this methodology will be available in the near future.

CAPRI was chosen as the basis for development of a geometry interface. CAPRI is a CAD-neutral software interface developed by Haimes[4] with a wide range of functionality for querying, modifying, and regenerating CAD models. In order to make CAPRI platform independent and compatible with the current and future framework, a Python interface was required and was developed as part of this work. In addition, multithreaded (locally parallel) execution of query commands involving the entire surface mesh was required to accelerate the performance of surface mesh regeneration. pyCAPRI is an implementation of CAPRI in Python which utilizes the NumPy high-performance Python array functionality. Internally, a combination of C++ and Fortran 90 code interfacing to the CAPRI API is wrapped in Python and exposed to the user as a single module. Parallelization is currently achieved through OpenMP.

pyCAPRI has been tested both with simple geometries and a few of the surface meshes used in this study for full vehicle configurations. Fig. 2 shows three variations of an SST solid geometry, varying only two parameters for visualization purposes (one of several fuselage thicknesses and the sweep of the leading edge / fuselage intersection). pyCAPRI itself is a general tool for the analysis involving parametric CAD models that is being applied to this work to determine spatial perturbations described in the steps above. Specifically, the perturbation of the surface mesh is achieved through a mapping to local surface parametric

coordinates (u, v) which are preserved through a regeneration of a model for small changes $x = x + \delta x$. The algorithm is summarized below:

1. Create a surface mesh corresponding to $\vec{x}$
2. Map the spatial coordinates of the surface mesh to (u, v) surface coordinates in the local surface
3. Regenerate the geometry with the new vector of design variables, $x = x + \delta x$
4. Map the (unchanged) (u, v) coordinates to new spatial coordinates that can be subsequently used for further analysis

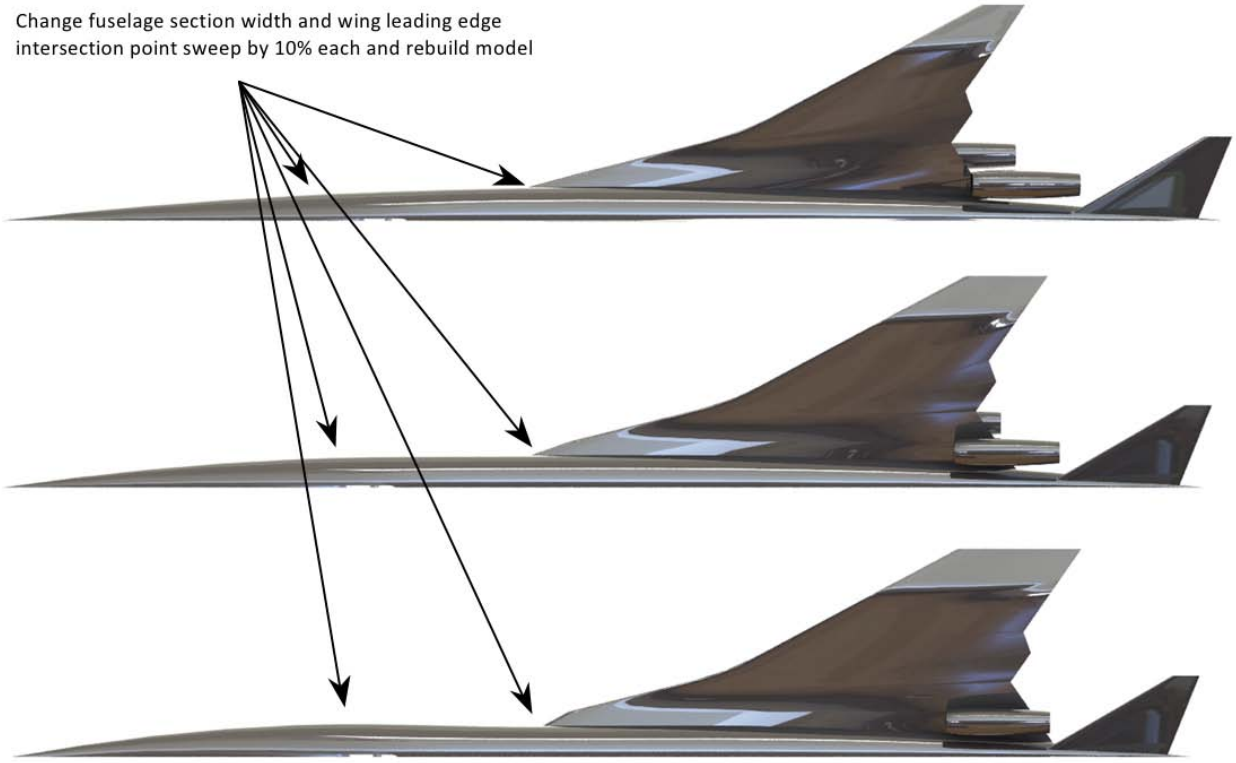


Figure 2. Parametrically varying surface geometry of SST concept vehicle via pyCAPRI API calls from Python. Note that much of the surface varies significantly though only two of approximately 150 parameters are varied.

III. A priori anisotropic grid adaptation for sonic-boom simulations

Sonic-boom propagation methods typically rely on the CFD solver to model the strongest nonlinearities in the flow. Consequently, the near-field pressure signature must be extracted a sufficiently large distance from the configuration such that the non-axisymmetric effects have largely died down: one or more body lengths below the aircraft is common. Capturing the pressure signature accurately at these distances is challenging, because strong shocks can be dissipated by the numerical scheme before they reach the extraction location.

Grid adaptation can be used to reduce numerical dissipation and thereby improve the pressure-signature accuracy. Using isotropic adaptation based on the solution gradient, Choi *et al.* found that the number of nodes required to accurately capture the near-field pressure is of order 10^7 for a node-based unstructured

solver. This motivates the use of anisotropic output-based adaptation; indeed, several studies have shown that the adjoint-weighted residual method coupled with metric-based anisotropic adaptation can reduce the number of solution degrees of freedom an order of magnitude relative to isotropic adaptation.[5,6]

While automated *a posteriori* grid adaptation is attractive for general flows, *a priori* adaptation can be more efficient if the locations of critical flow features are well known. A common example is refinement of the boundary layer in Reynolds-averaged Navier-Stokes simulations. Recently, Campbell *et al.*[7] proposed a simple, yet remarkably effective *a priori* mesh adaptation strategy for sonic-boom prediction. Numerical experiments suggest that their approach produces accurate pressure signatures with order 10^6 degrees of freedom, which is comparable to *a posteriori* output-based adaptation methods. Therefore, for this work we follow reference[7] and use the *a priori* grid adaptation approach described in the following sections.

A. Radial stretching and conical shearing

The *a priori* grid adaptation method of Campbell *et al.*[7] consists of two transformations applied to a baseline grid: (1) a radial stretching to reduce the number of nodes required to reach the near-field analysis location, and (2) a conical shearing to align the cells with the free-stream Mach angle. These transformations are described in more detail below.

Applying the stretching and shearing transformation too close to the aircraft may create volume cells that intersect with the geometry. Consequently, a cylindrical region with its axis parallel to the x-axis is placed around the configuration, and nodes inside this region are not modified. The sectional shape of the cylinder is composed of two ellipses with a common spanwise (major) axis length of $(1+\epsilon)b$, where b is the span and $\epsilon = 0.1$ is a safety margin. The upper and lower ellipses have semi-minor axis lengths of $(1+\epsilon)z_{\max}$ and $(1+\epsilon)z_{\max}/2$, respectively, where $z_{\max}$ denotes the upper bound of z on the geometry.

Stretching in the radial direction procedes as follows. Consider an arbitrary point in cylindrical coordinates $\mathbf{x} = (x, r, \beta)$. Let $\mathbf{x}_{\text{cyl}} = (x, r_{\text{cyl}}, \beta)$ denote the point where the line through $\mathbf{x}$ and $(x, 0, \beta)$ pierces the cylinder defining the fixed-node region. Stretching is applied if $r > r_{\text{cyl}}$, in which case the stretched coordinates are given by $\mathbf{x}_{\text{str}} = (x, r_{\text{str}}, \beta)$ where

$$r_{\text{str}} = r_{\text{cyl}} + \lambda(r - r_{\text{cyl}}),$$

and λ is the stretching factor. We have used $\lambda = 3$ in the present work.

Once a node is stretched radially, it is sheared to align the grid with the Mach cone. As with stretching, this transformation is applied only to nodes outside the fixed-node cylinder described above. First, the stretched point $\mathbf{x}_{\text{str}}$ and its corresponding $\mathbf{x}_{\text{cyl}}$ are expressed in a coordinate system that has its x-axis parallel with the free-stream velocity and its origin at the nose of the configuration. In this new coordinate system, the Mach cone that $\mathbf{x}_{\text{cyl}}$ lies on is determined, and $\mathbf{x}_{\text{str}}$ is projected onto this Mach cone along the line parallel to the free-stream.

B. Repairing invalid cells after adaptation

The stretching and shearing transformation may create inverted cells, i.e. cells with negative volumes, particularly along the boundary of the fixed-node region. To repair these invalid cells, we use the simultaneous untangling and smoothing procedure proposed by Escobar *et al.*.[8] For completeness, we describe this procedure briefly below.

Let v denote a vertex in the grid and $N(v)$ the local submesh of tetrahedral elements connected to v. We define an objective function for the node v using a sum of modified quality measures for the elements in $N(v)$:

$$K_v = \sum_{m=1}^{N(v)} \eta_m^*,$$

where η_m^* is a modified version of the inverse mean ratio:

$$\eta_m^* = \frac{|S_m|^2}{3\left[h(\sigma_m, \delta)\right]^{(2/3)}}, \qquad h(\sigma_m, \delta) = \frac{1}{2}\left(\sigma_m + \sqrt{\sigma_m^2 + 4\delta^2}\right).$$

Here, S_m denotes the nodally-invariant Jacobian matrix of element m. The scalars $|S_m|$ and σ_m are the Frobenius norm and determinant of S_m, respectively.

If the parameter δ is set to zero then $h = \sigma_m$, and we recover the usual inverse mean ratio quality measure, i.e. $\eta^* \to \eta$ as $\delta \to 0$. However, η is singular for collapsed elements whereas η^* remains defined for all elements provided $\delta > 0$. Hence, the role of the function h is to globalize the objective function K_v, so that we may apply suitable continuous optimization algorithms to minimize K_v.

For each cell that is inverted, we loop through its nodes v and attempt to minimize K_v using a safe-guarded Newton's method. Nodes on the surface of the geometry are not permitted to move, and nodes on the symmetry plane have their y-coordinate held fixed.

We begin the untangling process with $\delta = 0.1$ and decrease δ after each sweep through the inverted cells; we have found that this iterative process helps keep the objective function Hessian well conditioned during the first few iterations. All inverted cells are typically removed after three or fewer iterations.

C. Baseline and transformed grids

The baseline grid for this work was created using the StarCCM meshing package, which is based on Delaunay tetrahedralization. A strength of this software is its ability to precisely define volume, surface, and edge refinement regions. Volume regions of cells may be sized with sources generated from primitive shapes or imported shapes from a CAD file. Surfaces may be sized by region by splitting imported surface patches. Edges may be sized by defining feature curves on which cell edges are forced to snap.

The meshing process is characterized by the following three steps. First, a surface mesh of the aircraft is imported. Typically the surface is in Parasolid file format, because it provides StarCCM with patch information useful for splitting the surface into refinement regions. Next, the the aircraft surface is integrated with the farfield geometry. In the case of external symmetric flow problems like the present study, the surface is intersected with a symmetry plane. The result of this operation is shown in Fig. 3. Finally, the mesh is populated with information to control the size of cells in specified locations.

Several regions of the baseline mesh were targeted for refinement. In particular, a volume refinement source was added below the aircraft to capture the near-field pressure signature in the final transformed grid. In addition, the surface mesh was refined in regions of high curvature using feature curves. The refinement achieved around the leading and trailing edges is shown in Fig. 4.

The baseline grid used for this study is shown in Fig. 5(a). For a Mach number of 1.8 and angle of attack of 3 degrees, the stretching and shearing operations transform the baseline grid into the grid shown in Fig. 5(b). The anisotropic cells are clearly aligned with the Mach angle.

IV. Surface sensitivities using a continuous adjoint methodology

The objective of this section is to describe the way in which we quantify the influence of geometry modifications on either the pressure distribution on the trijet surface or at an arbitrary location within the domain of interest (typically the near-field). Conventional adjoint implementations are typically aimed at reducing a cost function computed from the pressure distribution on the surface that is being modified. In our current work, apart from the traditional functionals defined on the airplane surface (drag, lift, etc.), we will use two objective functions that depend on the near-field pressure distribution: the pressure coefficient on the near-field,[9] and the equivalent area distribution calculated from this pressure distribution.

The fluid domain Ω is bounded by a disconnected boundary $\delta\Omega$ which is divided into a "far field" component, Γ_∞, and a solid wall boundary, S. Ω has been further divided into two subdomains Ω_i and Ω_o separated by the near-field boundary Γ_{nf}. Note that Γ_{nf} will remain fixed throughout the optimization process, but the solid surface S will change as needed to meet the optimization criteria.

A typical optimization problem seeks the minimization of a certain cost function J with respect to changes in the shape of the boundary S. We will concentrate on functionals defined as integrals over the solid surface

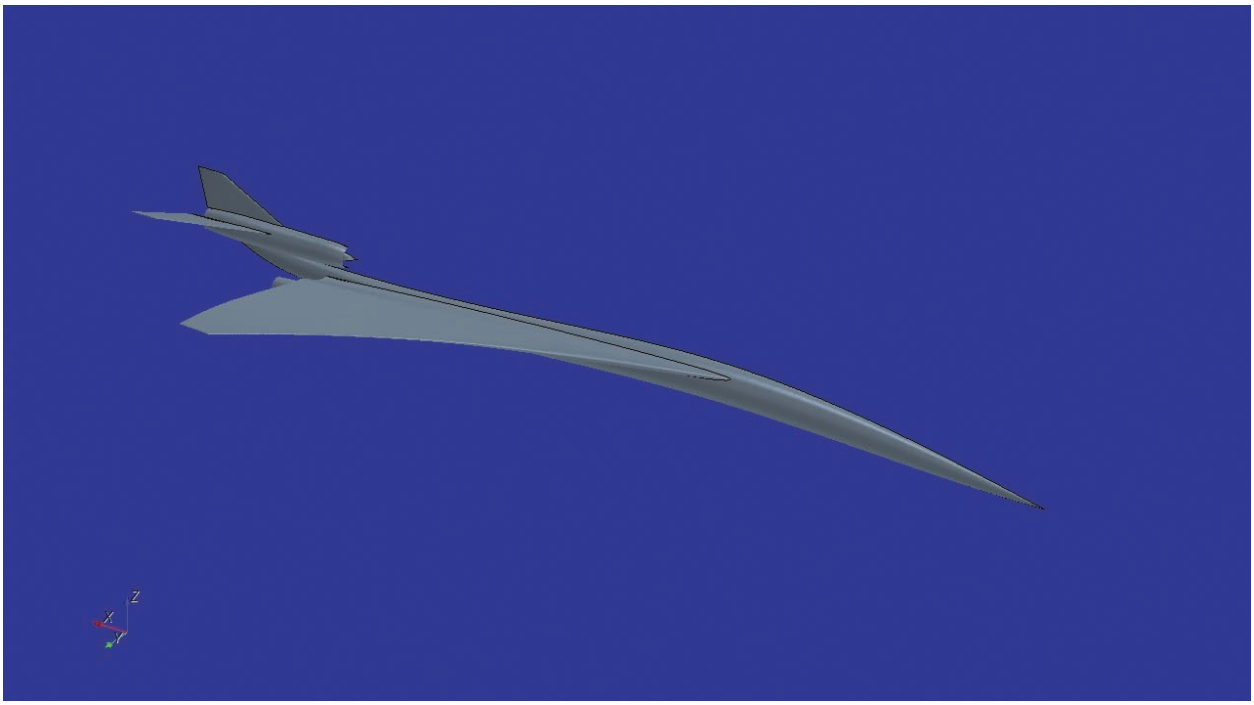


Figure 3. Half-body aircraft geometry.

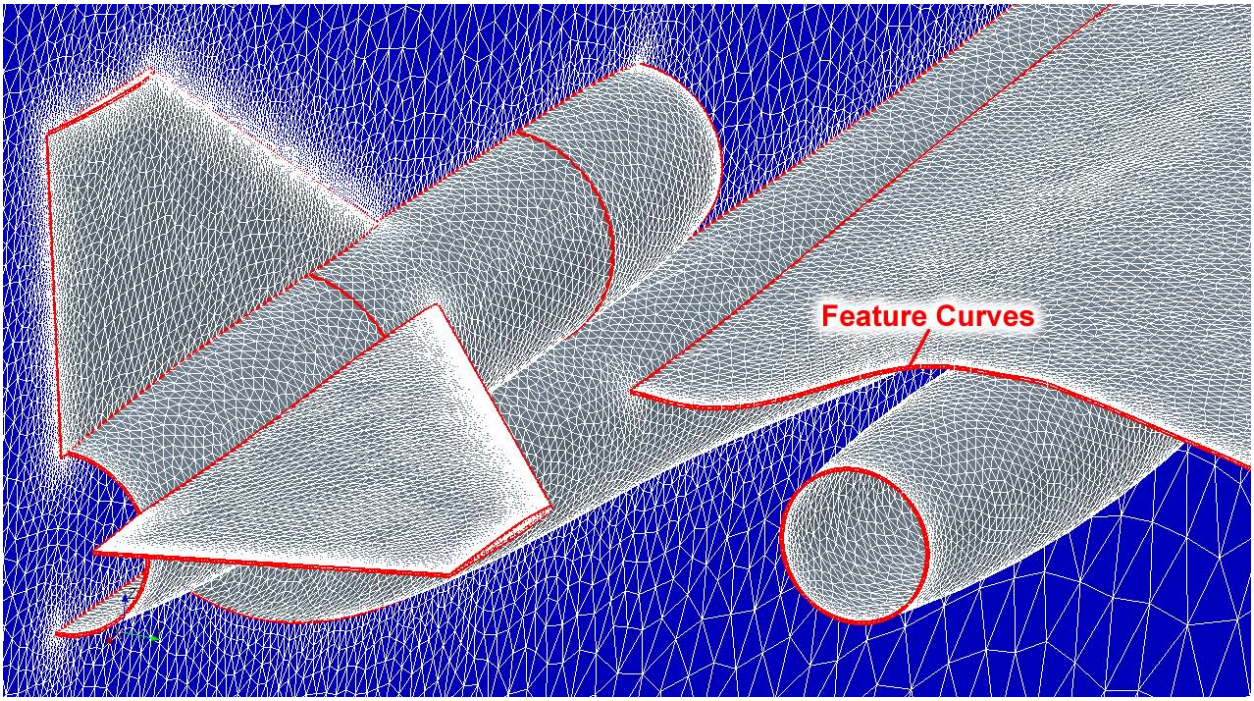
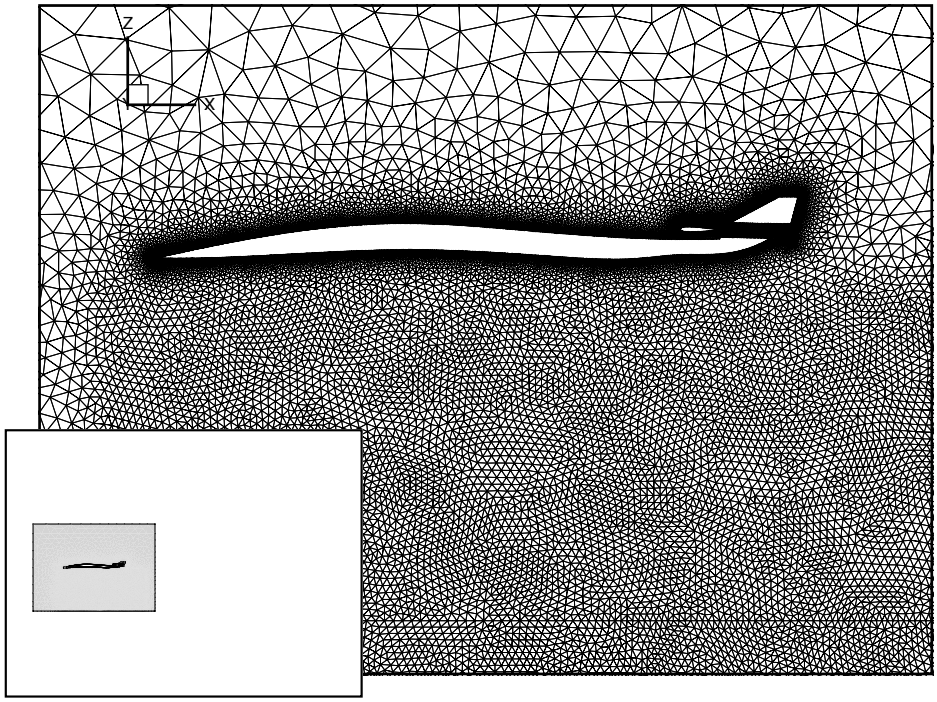
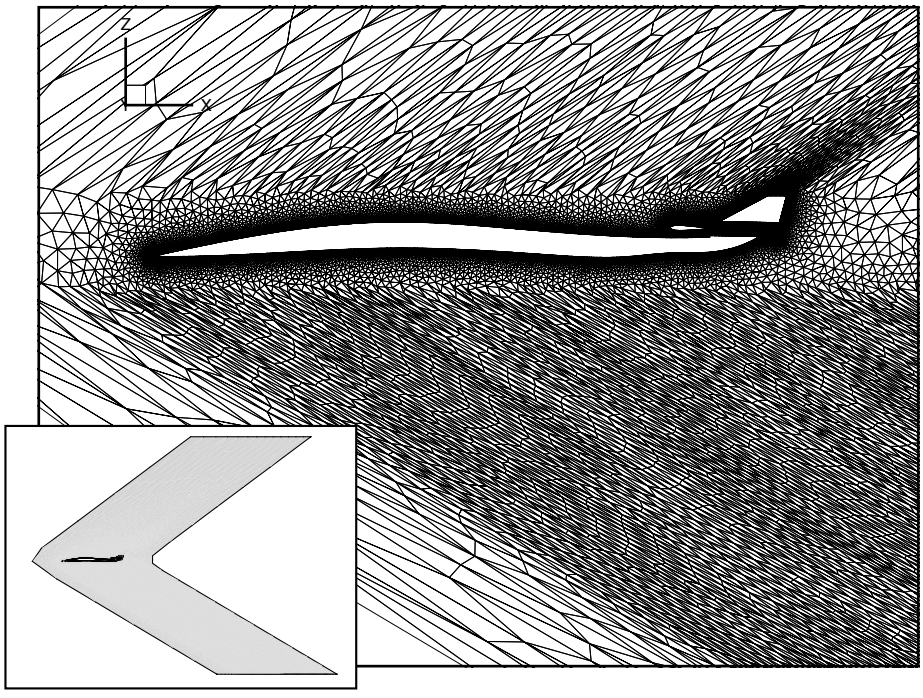


Figure 4. Detail showing surface and symmetry-plane mesh near empennage.



(a) initial grid



(b) transformed grid

Figure 5. Cells on the symmetry plane of the initial and transformed grids. The inset plots show the shape of the far-field boundary at the symmetry plane.

S, and integrals over the near-field boundary Γ_{nf},

$$J = \int_S \vec{d}\cdot(P\vec{n}_S)\,ds + \int_{\Gamma_{nf}} \frac{1}{2} f(x)(P-P_\infty)^2\,ds = \int_S j_S\,ds + \int_{\Gamma_{nf}} j_{nf}\,ds, \tag{2}$$

where P is the value of the pressure, $\vec{n}_S$ is a normal vector to the solid surface S, $f(x)$ is a function that only depends on the spatial coordinates, P_∞ is the pressure in the free stream, and $\vec{d}$ is an arbitrary constant vector that we will define later on. The goal is to compute the variation of the above functional caused by arbitrary (and multiple) deformations of S.

Upon an infinitesimal deformation δS of the control surface S along the normal direction $\vec{n}_S$, the cost function varies due to the changes in the solution induced by the deformation:[10]

$$\begin{aligned}
\delta J &= \int_{\delta S} j_S\,ds + \int_S \delta j_S\,ds + \int_{\Gamma_{nf}} \delta j_{nf}\,ds \\
&= \int_S (\partial_n j_S - 2H_m j_S)\delta S\,ds + \int_S \vec{d}\cdot(\delta P\vec{n}_S + P\delta\vec{n}_S)\,ds + \int_{\Gamma_{nf}} \frac{\partial j_{nf}}{\partial P}\delta P\,ds \\
&= \int_S \vec{d}\cdot(\partial_n(P\vec{n}_S) - 2H_m(P\vec{n}_S))\delta S\,ds + \int_S \vec{d}\cdot\delta P\vec{n}_S\,ds - \int_S \vec{d}\cdot P\nabla_S(\delta S)\,ds + \int_{\Gamma_{nf}} f(x)(P-P_\infty)\delta P\,ds \\
&= \int_S \vec{d}\cdot\delta P\vec{n}_S\,ds + \int_{\Gamma_{nf}} f(x)(P-P_\infty)\delta P\,ds + \int_S \vec{d}\cdot(\partial_n(P\vec{n}_S) + \nabla_S P - 2H_m(P\vec{n}_S))\,\delta S\,ds \\
&= \int_S \vec{d}\cdot\delta P\vec{n}_S\,ds + \int_{\Gamma_{nf}} f(x)(P-P_\infty)\delta P\,ds + \int_S (\vec{d}\cdot\vec{\nabla}P)\delta S\,ds,
\end{aligned} \tag{3}$$

where we have used $\delta\vec{n} = -\nabla_S(\delta S)$ that holds for small deformations,[11] and H_m is the mean curvature of S computed as $(\kappa_1+\kappa_2)/2$, where (κ_1,κ_2) are curvatures in two orthogonal directions on the surface. Here ∇_S represents the tangential gradient operator on S. Note that we have done an integration by parts and that we have used the gradient operator on local coordinates on S.

Once we are able to evaluate the variation of the objective function, we note that the variation is subject to the steady Euler flow equations, $\vec{\nabla}\cdot\vec{F} = 0$, where $\vec{F}$ are the convective fluxes. To tackle the variation of the flow variables δP in the objective function evaluation Eq. 3 we resort to the adjoint state $\Psi = (\psi_1, \vec{\varphi}, \psi_5)$, where $\vec{\varphi} = (\psi_2,\psi_3,\psi_4)$. Starting with the linearized form of the Euler's equations, taking the inner product with Ψ, and integrating over the domain one gets

$$0 = \int_\Omega \Psi(\vec{\nabla}\delta\vec{F})\,d\Omega = \int_\Omega \Psi\vec{\nabla}(\vec{A}\delta U)\,d\Omega = \int_{\Omega_i} \Psi\vec{\nabla}(\vec{A}\delta U)\,d\Omega + \int_{\Omega_o} \Psi\vec{\nabla}(\vec{A}\delta U)\,d\Omega, \tag{4}$$

where $\vec{A}$ is the Jacobian of $\vec{F}$ in conservative variables, and the entire domain has been split between the subdomains Ω_i and Ω_o. Integrating by parts

$$\begin{aligned}
0 &= -\int_{\Omega_i} (\vec{\nabla}\Psi)\vec{A}\delta U\,d\Omega_i + \int_S \Psi\left(\vec{n}_S\vec{A}\right)\delta U\,ds + \int_{\Gamma_{nf}} \Psi_i\left(\vec{n}_{nf}\vec{A}\right)\delta U\,ds \\
&- \int_{\Omega_o} (\vec{\nabla}\Psi)\vec{A}\delta U\,d\Omega_o - \int_{\Gamma_{nf}} \Psi_o\left(\vec{n}_{nf}\vec{A}\right)\delta U\,ds + \int_{\Gamma_\infty} \Psi_i\left(\vec{n}_\infty\vec{A}\right)\delta U\,ds.
\end{aligned} \tag{5}$$

Note that the integral over the far-field can be forced to vanish with the appropriate choice of boundary conditions, and the integral over the solid boundary S gives

$$\int_S \Psi\left(\vec{n}_S\vec{A}\right)\delta U\,ds = \int_S (\vec{n}\cdot\delta\vec{v})(\rho\psi_1 + \rho\vec{v}\cdot\vec{\varphi} + \rho H\psi_5)\,ds + \int_S (\vec{n}\cdot\vec{\varphi})\,\delta P\,ds, \tag{6}$$

where we have used the boundary condition $\vec{v}\cdot\vec{n}_S = 0$ to evaluate the jacobian $\vec{A}$. To eliminate the

dependence on $\vec{n}\cdot\delta\vec{v}$ we will use the linearized boundary condition on the surface S to obtain

$$\begin{aligned}\int_S \Psi\left(\vec{n}_S\vec{A}\right)\delta U\,ds &= -\int_S((\delta S\partial_n\vec{v})\cdot\vec{n}_s+\delta\vec{n}_s\cdot\vec{v})(\rho\psi_1+\rho\vec{v}\cdot\vec{\varphi}+\rho H\psi_5)\,ds+\int_S(\vec{n}\cdot\vec{\varphi})\,\delta P\,ds\\ &= -\int_S((\delta S\partial_n\vec{v})\cdot\vec{n}_s-\nabla_s(\delta S)\cdot\vec{v})\,\vartheta\,ds+\int_S(\vec{n}\cdot\vec{\varphi})\,\delta P\,ds\\ &= -\int_S((\partial_n\vec{v}\cdot\vec{n}_s)\,\vartheta+\nabla_s(\vec{v}\,\vartheta))\,\delta S\,ds+\int_S(\vec{n}\cdot\vec{\varphi})\,\delta P\,ds. \end{aligned}\tag{7}$$

To obtain this last expression we have used the linearized boundary conditions, the value of $\delta\vec{n}_S$ at the surface, and we have also done an integration by parts. On the other hand, the integrals supported on Γ_{nf} can be combined into the single integral

$$\int_{\Gamma_{nf}}\Delta\Psi\left(\vec{n}_{nf}\cdot\vec{A}\right)\delta U\,ds,\tag{8}$$

where $\Delta\Psi=\Psi_i-\Psi_o$ is the difference between the values of Ψ above and below the near-field boundary. Finally the domain integrals vanish provided the adjoint equation

$$\vec{\nabla}\Psi\cdot\vec{A}=\vec{A}^T\cdot\vec{\nabla}\Psi=0\tag{9}$$

is satisfied. And Eq. 4 can be written as

$$-\int_S((\partial_n\vec{v}\cdot\vec{n}_S)\,\vartheta+\nabla_S(\vec{v}\,\vartheta))\,\delta S\,ds+\int_S(\vec{n}_S\cdot\vec{\varphi})\delta P\,ds+\int_{\Gamma_{nf}}\Delta\Psi\left(\vec{n}_{nf}\vec{A}\right)\delta U\,ds=0,\tag{10}$$

where $\vartheta=(\rho\psi_1+\rho\vec{v}\cdot\vec{\varphi}+\rho H\psi_5)$. The last step is to subtract Eqs. 3 and 10 to obtain the complete variation of the functional as:

$$\begin{aligned}\delta J &= \int_S\vec{d}\cdot\delta P\vec{n}_S\,ds+\int_{\Gamma_{nf}}f(x)(P-P_\infty)\delta P\,ds+\int_S(\vec{d}\cdot\vec{\nabla}P)\delta S\,ds\\ &+ \int_S((\partial_n\vec{v}\cdot\vec{n}_s)\,\vartheta+\nabla_S(\vec{v}\,\vartheta))\,\delta S\,ds-\int_S(\vec{n}_S\cdot\vec{\varphi})\delta P\,ds-\int_{\Gamma_{nf}}\Delta\Psi\left(\vec{n}_{nf}\vec{A}\right)\delta U\,ds\\ &= \int_S(\vec{d}\cdot\vec{n}_S)\delta P\,ds-\int_S(\vec{n}_S\cdot\vec{\varphi})\delta P\,ds+\int_{\Gamma_{nf}}f(x)(P-P_\infty)\delta P\,ds-\int_{\Gamma_{nf}}\Delta\Psi\left(\vec{n}_{nf}\vec{A}\right)\delta U\,ds\\ &+ \int_S\left(\vec{d}\cdot\vec{\nabla}P+(\partial_n\vec{v}\cdot\vec{n}_S)\,\vartheta+\nabla_S(\vec{v}\,\vartheta)\right)\delta S\,ds,\end{aligned}\tag{11}$$

where we have rearranged the different terms to have a more clear view of those which are involved in the evaluation of the objective function gradient.

A. Surface and near-field pressure-based functionals

Eq. 11 is the key to evaluate the functional sensitivity with respect to deformations on solid surface S. In order to do that, we firstly define the vector $\vec{d}$ as

$$\vec{d}=\begin{cases}\left(\frac{1}{C_\infty}\right)(\cos\alpha\,\cos\beta,\sin\alpha\,\cos\beta,\sin\beta), & C_D \quad \text{Drag coefficient},\\ \left(\frac{1}{C_\infty}\right)(-\sin\alpha,\cos\alpha,0), & C_L \quad \text{Lift coefficient},\\ \left(\frac{1}{C_\infty}\right)(-\sin\beta\,\cos\alpha,-\sin\beta\,\sin\alpha,\cos\beta), & C_{SF} \quad \text{Side-force coefficient},\end{cases}\tag{12}$$

where α is the angle of attack, β is the angle of sideslip, $C_\infty=\frac{1}{2}v_\infty^2\rho_\infty A_z$, and v_∞ and ρ_∞ denote the infinity values of velocity and density. All the positive components of the normal surface vectors in the z-direction

are summed up in order to calculate the projection A_z, but a pre-specified reference area can also be used in a similar fashion.

The admissible adjoint boundary conditions that eliminates the dependence on the fluid flow variations are:

$$\begin{cases} \vec{n}_S \cdot \vec{\varphi} = \vec{d} \cdot \vec{n}_S \\ \vec{\nabla}\Psi \left(\vec{n}_{nf} \cdot \vec{A} \right) = f(x)(P - P_\infty), \end{cases} \tag{13}$$

and the variation of the objective function is

$$\delta J = \int_S \left(\vec{d} \cdot \vec{\nabla} P + (\partial_n \vec{v} \cdot \vec{n}_s)\, \vartheta + \nabla_s (\vec{v}\, \vartheta) \right) \delta S \, ds \tag{14}$$

where $\frac{\partial J}{\partial S} = \left(\vec{d} \cdot \vec{\nabla} P + (\partial_n \vec{v} \cdot \vec{n}_s)\, \vartheta + \nabla_s (\vec{v}\, \vartheta) \right)$ is what we have called the surface sensitivity, which provides a measure of the variation of the objective function with respect to infinitesimal variations of the surface shape in the direction of the local surface normal. This value is computed at each surface node of the numerical grid with negligible computational cost: in a typical supersonic aircraft mesh, this involves the computation of approximately tens of thousands of derivatives at little computational expense. In Fig. 6 and Fig. 7 the surface shape sensitivities for the drag and lift coefficients are shown respectively using the baseline configuration.

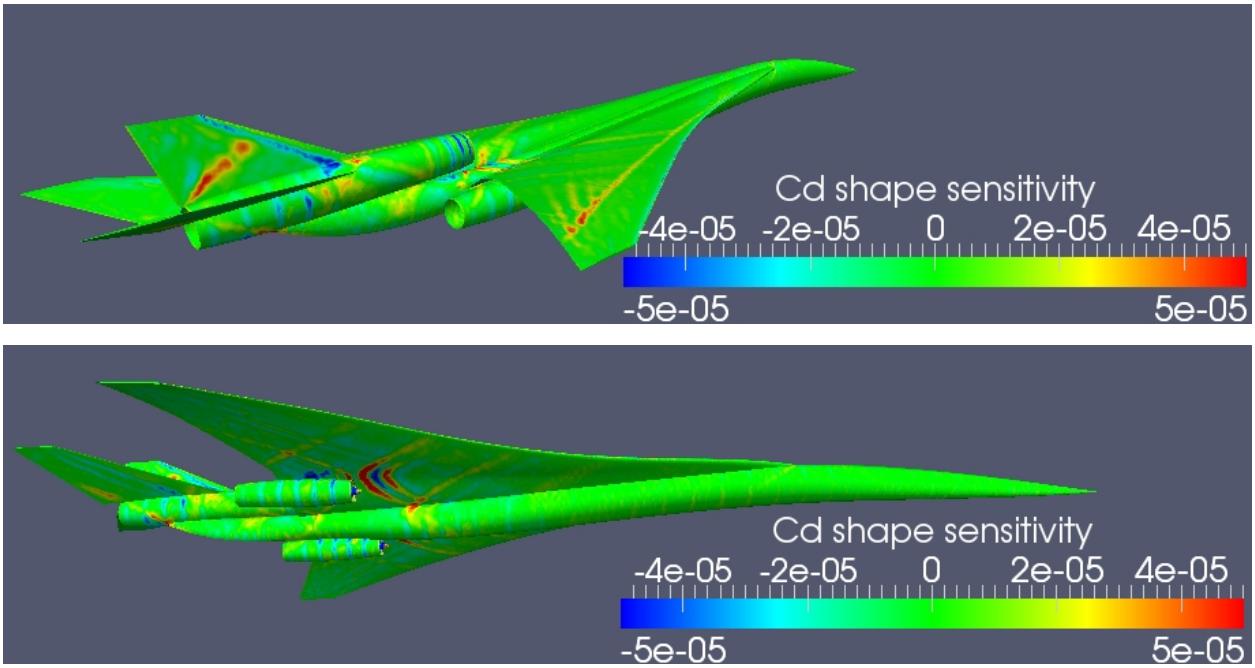


Figure 6. Surface shape sensitivity of the Drag coefficient (baseline configuration).

B. Equivalent Area-based functionals

As we have seen earlier in this article, in optimal shape design problems, continuous adjoint formulations restrict the form of the objective functions that can be computed in the near-field. Essentially, the variation of the cost function with respect to the pressure must have the following form so that the appropriate

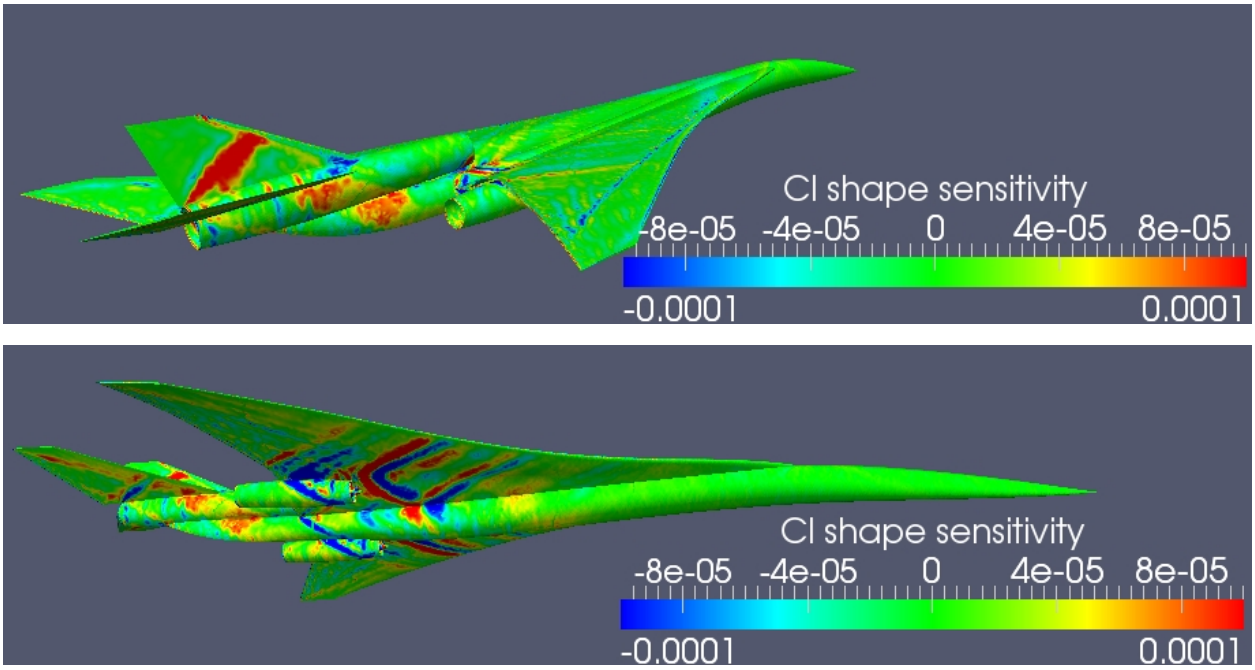


Figure 7. Surface shape sensitivity of the Lift coefficient (baseline configuration).

boundary terms are cancelled in the formulation:

$$\delta J = \int_{-L}^{L} f(x)(P - P_\infty)\delta P \, dx, \; -L < x < L, \tag{15}$$

where $f(x)$ is a function that can only depend on the spatial coordinates at the near-field plane. Note that, if $f(x) = 2/\rho_\infty A U_\infty^2$ the objective function is just the area integral of the pressure coefficient on the near-field, where the area within the limits of integration is given by A. Using classical supersonic aerodynamics, we know that the equivalent area distribution $A_e(x;\theta)$ at a particular azimuthal angle θ is given by

$$A_e(x;\theta) = \int_0^x C(P - P_\infty)(x - t)^{1/2} \, dt, \tag{16}$$

where $C = 4\frac{\sqrt{2\beta r}}{\gamma \rho_\infty M_\infty^2}$. In the following we assume that we are on an azimuthal plane $\theta = \theta_0$ and drop the dependence of the area distribution on the azimuthal angle θ. In order to pose an inverse design problem whereby we attempt to match a target equivalent area distribution and, at the same time, satisfy the requirement expressed by Eq. 15, the logical approach is to formulate the problem using a least-squares minimization formulation. For purposes of illustration, let's minimize the weighted sum of the square of the differences between the computed equivalent area and the target equivalent area at four different points (x_0, x_1, x_2, x_3) in the near field, with weights $(\omega_0, \omega_1, \omega_2, \omega_3)$, and target equivalent areas $(A_t(x_0), A_t(x_1), A_t(x_2), A_t(x_3))$. Obviously, the procedure can be easily extended to an arbitrary number of such points, thus making it completely general, even for cases where we might be interested in designing equivalent area distributions at multiple azimuthal planes, θ, as would be the case to avoid simply reducing the sonic boom under the flight track while the boom intensity increases off the flight track. This is precisely the way in which our implementation has been carried out.

For this problem formulation, the objective function can be written as:

$$J = \omega_1 \left[A_e(x_1) - A_t(x_1)\right]^2 + \omega_2 \left[A_e(x_2) - A_t(x_2)\right]^2 + \omega_3 \left[A_e(x_3) - A_t(x_3)\right]^2, \tag{17}$$

where we have assumed that $A_e(x_0) = A_t(x_0) = 0$. In order to check if the variation of this objective function satisfies the requirement in Eq. 15, the variation of Eq. 17 is given by:

$$\begin{aligned} \delta J &= 2\omega_1 \left[A_e(x_1) - A_t(x_1)\right] \delta A_e(x_1) + 2\omega_2 \left[A_e(x_2) - A_t(x_2)\right] \delta A_e(x_2) \\ &+ 2\omega_3 \left[A_e(x_3) - A_t(x_3)\right] \delta A_e(x_3). \end{aligned} \tag{18}$$

On the other hand, the variation of the equivalent area, $A_e(x)$, with respect to the pressure yields:

$$\delta A_e(x) = \int_0^x C(x-t)^{1/2} \delta P \, dt. \tag{19}$$

Using this last expression it is possible to separately evaluate the variation of the non-zero equivalent areas at the three chosen points as follows:

$$\begin{aligned} \delta A_e(x_1) &= \int_{x_0}^{x_1} C(x_1 - x)^{1/2} \delta P \, dx, \\ \delta A_e(x_2) &= \int_{x_0}^{x_1} C(x_2 - x)^{1/2} \delta P \, dx + \int_{x_1}^{x_2} C(x_2 - x)^{1/2} \delta P \, dx, \\ \delta A_e(x_3) &= \int_{x_0}^{x_1} C(x_3 - x)^{1/2} \delta P \, dx + \int_{x_1}^{x_2} C(x_3 - x)^{1/2} \delta P \, dx \\ &+ \int_{x_2}^{x_3} C(x_3 - x)^{1/2} \delta P \, dx, \end{aligned} \tag{20}$$

where, for the sake of simplicity, we have substituted the dummy variable of integration t by x. Using the variation of the equivalent area that we have computed in Eq. 18, it is possible to rewrite the variation of the objective function as:

$$\begin{aligned} \delta J &= 2\omega_1 \left[A_e(x_1) - A_t(x_1)\right] \int_{x_0}^{x_1} C(x_1 - x)^{1/2} \delta P \, dx \\ &+ 2\omega_2 \left[A_e(x_2) - A_t(x_2)\right] \left(\int_{x_0}^{x_1} C(x_2 - x)^{1/2} \delta P \, dx + \int_{x_1}^{x_2} C(x_2 - x)^{1/2} \delta P \, dx \right) \\ &+ 2\omega_3 \left[A_e(x_3) - A_t(x_3)\right] \left(\int_{x_0}^{x_1} C(x_3 - x)^{1/2} \delta P \, dx + \int_{x_1}^{x_2} C(x_3 - x)^{1/2} \delta P \, dx \right. \\ &+ \left. \int_{x_2}^{x_3} C(x_3 - x)^{1/2} \delta P \, dx \right), \end{aligned} \tag{21}$$

To simplify the notation, we will use:

$$\begin{cases} \Delta A_e(x_1) = 2\omega_1 \left[A_e(x_1) - A_t(x_1)\right] C \\ \Delta A_e(x_2) = 2\omega_2 \left[A_e(x_2) - A_t(x_2)\right] C \\ \Delta A_e(x_3) = 2\omega_3 \left[A_e(x_3) - A_t(x_3)\right] C. \end{cases}$$

Rearranging the integrals in Eq. 21, it is possible to write the variation of the objective function as

$$\begin{aligned} \delta J &= \int_{x_0}^{x_1} \left(\Delta A_e(x_1)(x_1 - x)^{1/2} + \Delta A_e(x_2)(x_2 - x)^{1/2} + \Delta A_e(x_3)(x_3 - x)^{1/2} \right) \delta P \, dx \\ &+ \int_{x_1}^{x_2} \left(\Delta A_e(x_2)(x_2 - x)^{1/2} + \Delta A_e(x_3)(x_3 - x)^{1/2} \right) \delta P \, dx \\ &+ \int_{x_2}^{x_3} \left(\Delta A_e(x_3)(x_3 - x)^{1/2} \right) \delta P \, dx. \end{aligned} \tag{22}$$

In conclusion, the inverse equivalent area shape design problem can be reinterpreted as an optimization of a functional of the near-field pressure distribution with a particular weighting function:

$$J = \int_{-L}^{L} f(x)(P - P_\infty)\, dx, \; -L < x < L, \tag{23}$$

where

$$f(x) = \begin{cases} 0 & , \text{ if } -L < x < x_0, \\ \Delta A_e(x_1)(x_1 - x)^{1/2} + \Delta A_e(x_2)(x_2 - x)^{1/2} + \Delta A_e(x_3)(x_3 - x)^{1/2} & , \text{ if } x_0 < x < x_1 \\ \Delta A_e(x_2)(x_2 - x)^{1/2} + \Delta A_e(x_3)(x_3 - x)^{1/2} & , \text{ if } x_1 < x < x_2 \\ \Delta A_e(x_3)(x_3 - x)^{1/2} & , \text{ if } x_2 < x < x_3, \\ 0 & , \text{ if } x_3 < x < L, \end{cases}$$

and $\Delta A_e(x_i) = 2\omega_i \left[A_e(x_i) - A_t(x_i)\right] C$ as given in Eq. 22, $C = 4\frac{\sqrt{2\beta r}}{\gamma \rho_\infty M_\infty^2}$, and where ω_1, ω_2, and ω_3 are arbitrary weights chosen by the designer. It is important to note that this formulation of the problem fits perfectly within a continuous adjoint framework. In addition, a similar procedure can be used to produce a formulation that is amenable to discrete adjoint methods (with little or no modifications). The application of this formula to an arbitrary number of points in the near-field is straightforward. Furthermore, this formula also gives us some clues about how to choose $f(x)$ in a three-dimensional environment, should we be interested in matching a number of different equivalent area distributions simultaneously at different azimuthal angles.

1. *Sample 2D Equivalent Area Distribution Redesign*

Using the L-2 norm of the difference between the computed equivalent area distribution and a pre-specified target area distribution as a functional on an optimal shape design process requires the imposition of a jump in the adjoint variables on the near-field plane. In order to do that, the original grid is divided by a horizontal line (2D) or plane (3D), and the points on the line/plane are duplicated: both planes on either side of the near field as well as the coordinates of the nodes that lie on them are exactly the same, except that the connectivity of the elements is altered to allow the imposition of the jump on the adjoint variables. Given an existing mesh, this process is done automatically with an in-home tool developed for this project. After the domain division, the jump condition is imposed using the fluxes that connect both domains.

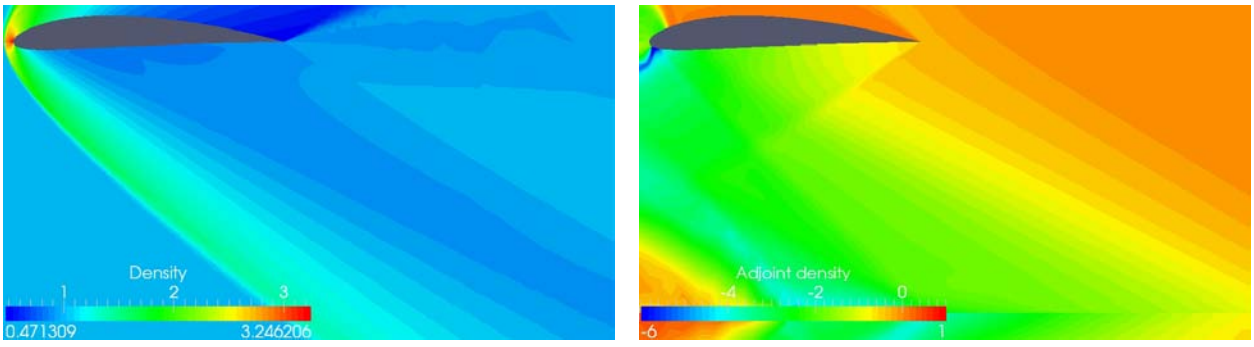


Figure 8. Flow (left), and adjoint (right) solutions for the NACA 4412 and the equivalent area distribution functional.

To test the complete methodology we have used a simple equivalent area inverse design problem: starting with a NACA 4412 airfoil, at Mach 2.0 and at an angle of attack of 0.0deg, the objective is to modify the shape of this airfoil in order to obtain the equivalent area distribution produced by a NACA 0012 airfoil

at the same flow conditions. The target equivalent area distribution (for the NACA 0012) is computed in a pre-processing step and guarantees that the optimal value of the cost function is indeed zero. In other words, appropriate shape modifications of the starting NACA 4412 can result in a NACA 0012 airfoil that exactly achieves the target area distribution specified.

This problem is chose as a validation step for our entire equivalent area adjoint formulation: if the gradients are computed properly and our implementation is correct, then the optimization procedure should achieve exactly both the shape of the NACA 0012 airfoil and its equivalent area distribution.

In Fig. 8 the flow solution (left) and the adjoint solution (right) for the NACA 4412 are plotted (using the density and adjoint-of-density variables). The plot for the adjoint variable reflects the regions of the flow that affect the computation of the cost functional: the L-2 norm of the difference between the computed and target equivalent area distributions. To reduce the computational cost an apriori-adapted grid is used (see Fig. 9, left).

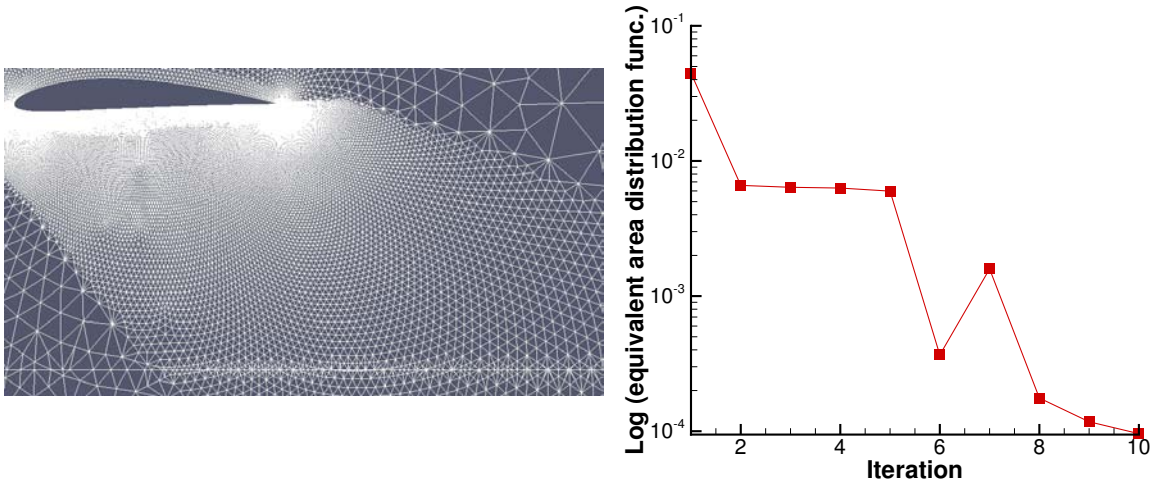


Figure 9. A priori adapted numerical grid (left), and optimization evolution using a logarithmic scale (right).

In this problem the near-field is located at one chord length below the airfoil and, as mentioned earlier, the objective is to recover the equivalent area of a NACA 0012 airfoil. As design variables we have used a total of 50 Hicks-Henne bump functions distributed along the entire airfoil (upper and lower surfaces), and a quasi-Newton method is used as the optimizer with gradients computed using the technique described earlier in this article.

In Fig. 10 the original NACA4412 and the designed airfoil with the area distribution of a NACA0012 are plotted. Finally, in Fig. 9 (right) the optimization history is shown, after 10 iterations we have almost completely recovered the NACA 0012 equivalent area distribution. It is important to note that, although the equivalent area distribution is the same as in the NACA0012, the designed airfoil shape differs from the NACA0012: this is a problem with non-unique solutions. Using gradient-based optimization techniques, the optimizer has found the closest local minimum to the starting point of the optimization, namely the NACA4412. If a NACA 0012 airfoil is used as a starting point, the optimizer exits immediately having found another local minimum in the design space.

The results of this simple optimization problem prove that the gradient information supplied by our adjoint solver (and boundary conditions) for the equivalent area distributions of the problem are correct and can be used for more complicated three-dimensional designs.

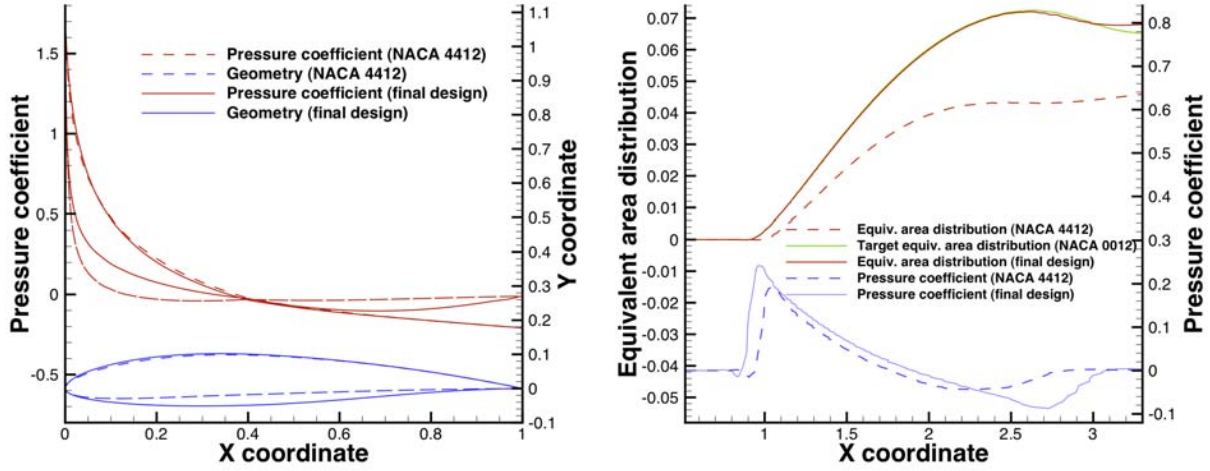


Figure 10. Optimization results, geometry and pressure distribution (left), equivalent area distribution, and near-field pressure (right).

V. Design variable definition and mesh deformation

For these initial results, a Free-Form Deformation (FFD)[12] strategy has been adopted, although the surface shape sensitivities can also be used in conjunction with the pyCAPRI approach described earlier in this article. In FFD an initial box encapsulating the object (wing, fuselage, etc.) that we want to redesign is parameterized as a Bézier solid. A set of control points are defined on the surface of the box, the number of which depends on the order of the chosen Bernstein polynomials. The solid box is parameterized by the following expression

$$X(u,v,w) = \sum_{i,j,k=0}^{l,m,n} P_{i,j,k} B_i^l(u) B_j^m(v) B_k^n(w), \tag{24}$$

where $u, v, w \in [0,1]$, and the B^i being the Bernstein polynomial of order i. The Cartesian coordinates of the points on the surface of the object are then transformed into parametric coordinates within the Bézier box. Control points of the box become design variables, as they control the shape of the solid, and thus the shape of the surface grid inside. The box enclosing the geometry is then deformed by modifying its control points, with all the points inside the box inheriting a smooth deformation. Once the deformation has been applied, the new Cartesian coordinates of the object of interest can be recovered by simply evaluating the mapping inherent in Eq. 24.

Once the boundary displacements have been computed using the FFD strategy, a classical spring method is used in order to deform the rest of vertices of the unstructured mesh. The method is based on the definition of a stiffness matrix, k_{ij}, that connects the two ends of a single bar (mesh edge). Equilibrium of forces is then imposed at each mesh node

$$\left(\sum_{j \in \mathcal{N}_i} k_{ij} \vec{e}_{ij} \vec{e}_{ij}^{\mathsf{T}} \right) \vec{u}_i = \sum_{j \in \mathcal{N}_i} k_{ij} \vec{e}_{ij} \vec{e}_{ij}^{\mathsf{T}} \vec{u}_j, \tag{25}$$

where the displacement $\vec{u}_i$ is unknown and is computed as a function of the known surface displacements $\vec{u}_j$, $\mathcal{N}_i$ is the set of neighboring points to node i, and $\vec{e}_{ij}$ the unit vector in the direction connecting both points. The system of equations is solved iteratively by a conjugate gradient algorithm with Jacobi preconditioning.

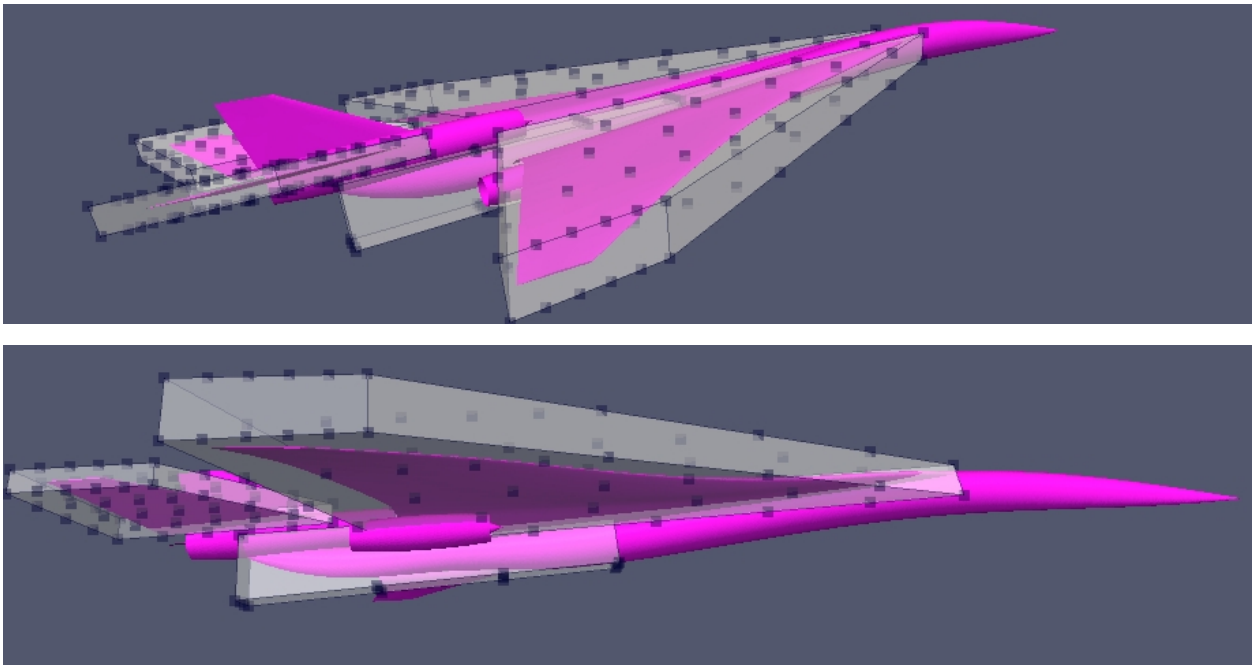


Figure 11. Free Form Deformation boxes used in the trijet optimization.

VI. Simulation and shape design framework

The simulations and optimal shape design cases presented in this work have been carried out using the SU^2 suite (Stanford University Unstructured). SU^2 is a computational tool for the optimal shape design of airplanes composed by a collection of software modules (written in C++), and scripts (written in Python). Next, a brief description of the main software modules is presented:

- SU2 CFD (Computation Fluid Dynamics Code). This is the software for performing the fluid dynamics simulation. This code is able to solve the direct, continuous adjoint, and linearized problems for the potential, Euler, Navier-Stokes and RANS equations. It uses a Finite Volume Method formulation, and an edge-based data structure. The code is fully parallel, explicit/implicit, and it uses an agglomeration multigrid approach to accelerate the convergence to the steady state.

- SU2 GPC (Gradient Projection Code). This is the software module for computing the partial derivative of an objective function with respect to variations of the shape of the aerodynamic surface. SU2 GPC uses the surface sensitivity, the flow solution, and the definition of the geometric variables to evaluate the derivative of a particular functional.

- SU2 MDC (Mesh Deformation Code). This is the software module for both performing the geometric deformation of an aerodynamic surface, as well as the volumetric grid deformation. Once the kind of deformation is defined, SU2 MDC performs the grid deformation using a torsional spring analogy. To perform the 3D surface deformations a Free Form Deformation method is used. Work is being completed to also interface SU2 MDC with the pyCAPRI method described earlier, as well as other choices for surface parameterization.

- SU2 MAC (Mesh Adaptation Code). This is the software module used for grid adaptation using different techniques based on the analysis of the fluid solution, adjoint solution and linearized problem.

- SU2 GDC (Geometric Design Code). This is the software module for computing the value of geometric functional and its gradient that depend exclusively on the geometry of the aircraft.

All these software modules interact with each other using a series of Python scripts that automate the entire optimization loop.

VII. Shape optimization using Trijet configuration

In this section we conclude the article by presenting examples of aerodynamic shape optimization of the trijet supersonic configuration using our adjoint-based techniques. The governing equations are the Euler equations, and the baseline configuration is in a free stream of Mach 1.8, and an angle of attack of $3.0deg$ that, using the reference area for the configuration, results in a $C_L = 0.14$. An unstructured grid, with all the geometric details (and a total of approximately 6 million cells) is used. Using our methodology , 15 iterations of a complete trijet optimization with more than 100 design variables can be completed in 24 hrs using only 10 cores. Much faster designs can be achieved using larger computational resources.

In this particular design problem we have used a maximum number of 108 design variables, which produce a variation of the shape in three different areas of the aircraft: the wing, horizontal tail, and the rear portion of the fuselage. The Free Form Deformation boxes for the wing and horizontal tail have $(5, 4, 1)$ degrees in the local (i, j, k) directions, and the rear fuselage box has $(3, 3, 1)$ degrees of freedom in their local coordinate directions. Fig. 12 shows the outline of the three Free Form Deformation boxes used in this work, together with the location of the actual control points that we use as design parameters (seen on the surfaces of the boxes).

For demonstration purposes we have performed three different optimization tests: an unconstrained drag minimization, a drag minimization with a constrained lift coefficient ($C_L = 0.14$), and inverse design problem to obtain a target equivalent area distribution. It is important to highlight that, after definition of the design problem, the process is completely automated.

The optimization results presented in this work make use of the *SciPy* library (`http://www.scipy.org`), a well-established open-source software for mathematics, science, and engineering. The SciPy library provides many user-friendly and efficient numerical routines for the solution of non-linear constrained optimization problems, such as conjugate gradient, Quasi-Newton or sequential least-squares programming algorithms. At each design iteration, the SciPy routines only require as inputs the values and gradients of the objective functions, computed by means of our continuous adjoint approach, as well as the set of chosen constraints.

A. Drag minimization problem

Starting with the baseline geometry, and using two Free Form Deformation boxes (main wing, and horizontal tail plane), we have performed an unconstrained drag minimization problem to check the quality of the gradients obtained with the continuous adjoint methodology. In Fig. 13 one can see the upper and lower surfaces of the trijet before and after the shape optimization.

In Fig. 14 (left) we show the rear part of the aircraft with especial emphasis in the drag reduction on the nacelles and the modifications of the horizontal tail plane. These modifications would have to be considered within the context of stability and control considerations that we have not included here. Finally in Fig. 14 (right) we show the evolution of the optimization. In this case the drag coefficient has been reduced by about 50% with respect to the original configuration, although the lift coefficient has also decreased by approximately 40%. Note that we have used the quasi-Newton method of Broyden, Fletcher, Goldfarb, and Shanno (BFGS) as optimization method.

B. Drag minimization problem with lift constraint

As in the previous case we have started with the baseline configuration (mesh with a total of 6 million cells). The objective is to reduce the drag coefficient while maintaining the original lift coefficient ($C_L = 0.14$). In this case we will use a Sequential Least Squares Programming optimization algorithm (SLSQP) with

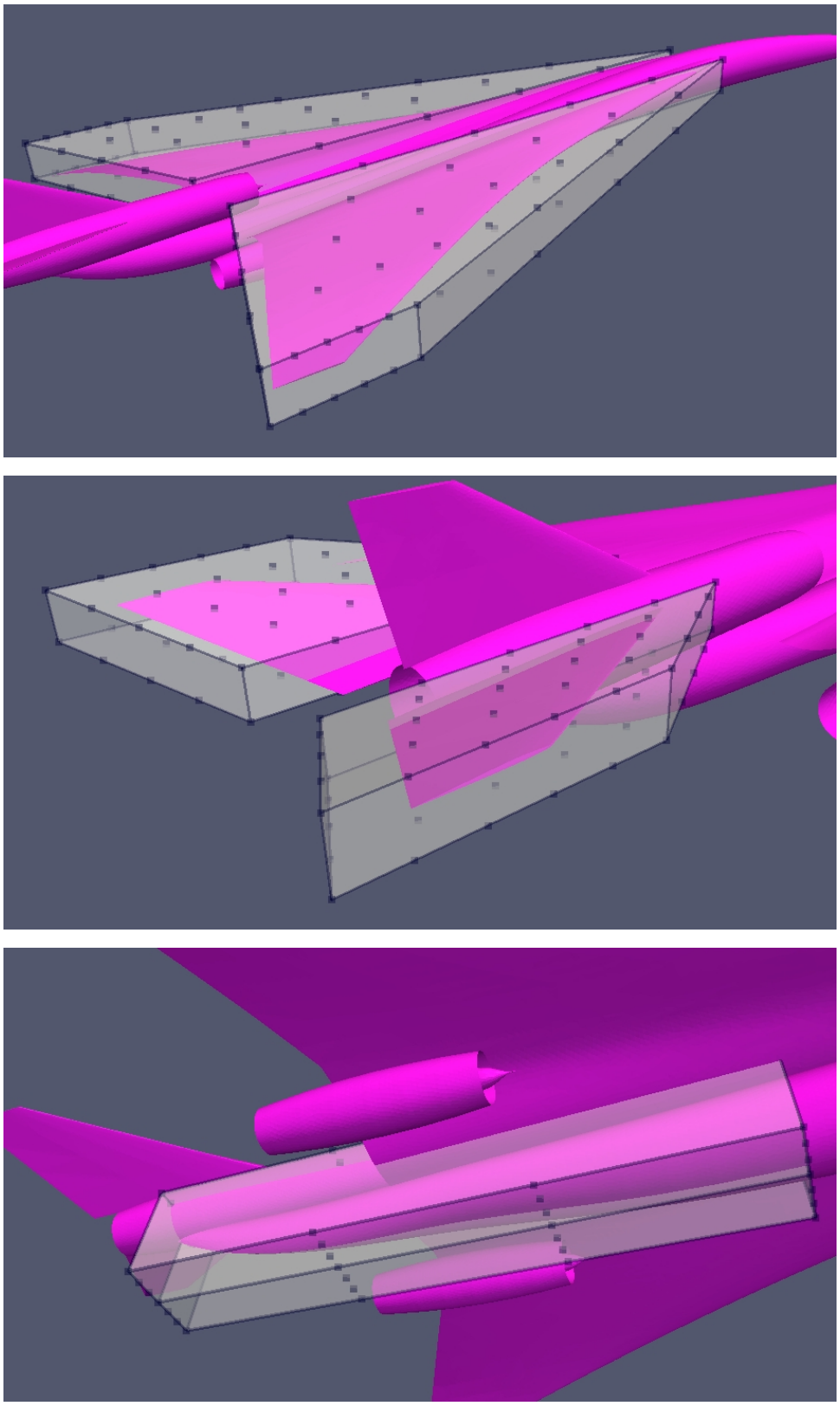


Figure 12. Free Form Deformation boxes used in the optimization process. Wing box (top), horizontal tail (middle), aft fuselage (bottom).

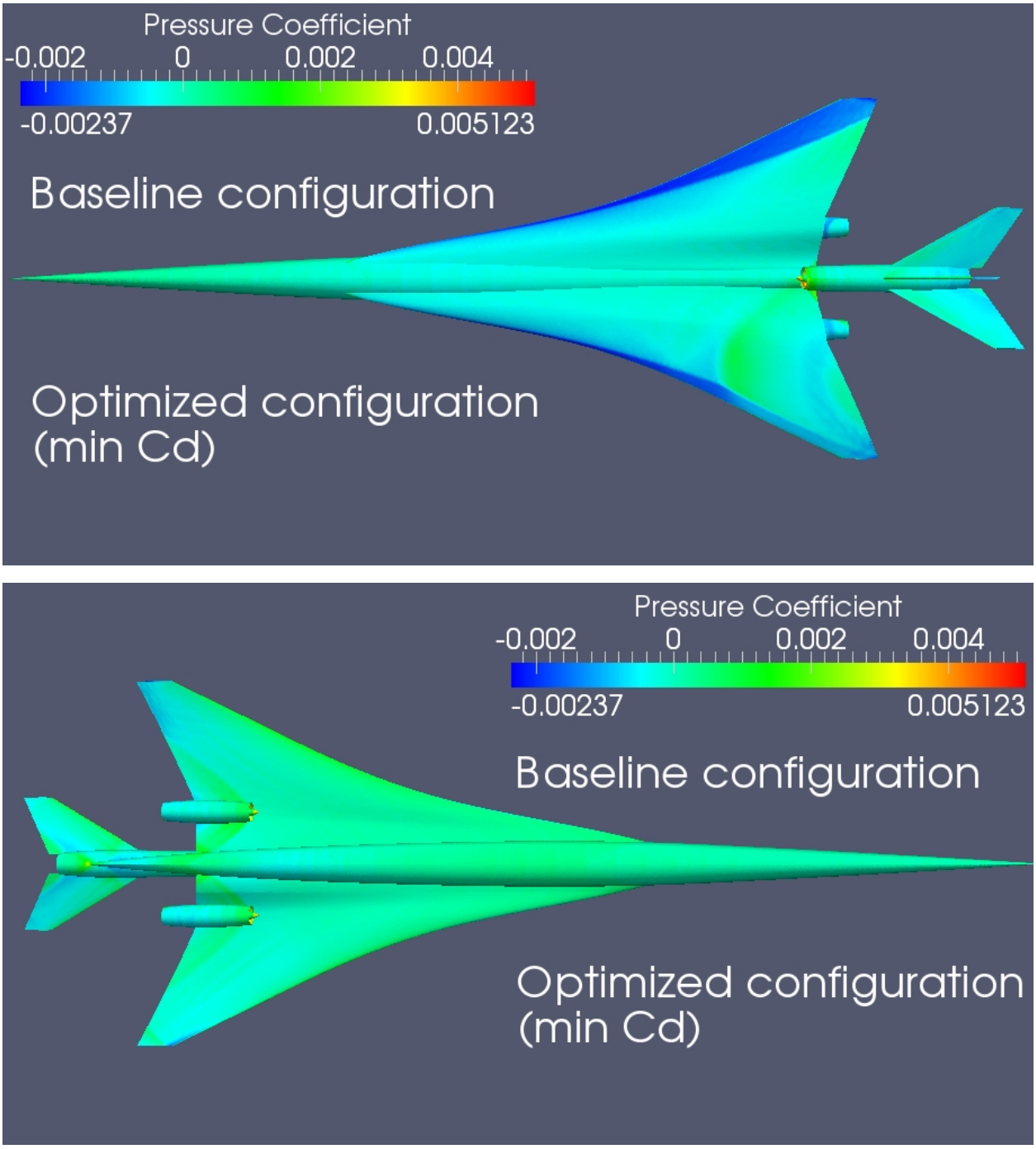


Figure 13. Shape design optimization (min Cd), upper and lower surfaces of the trijet.

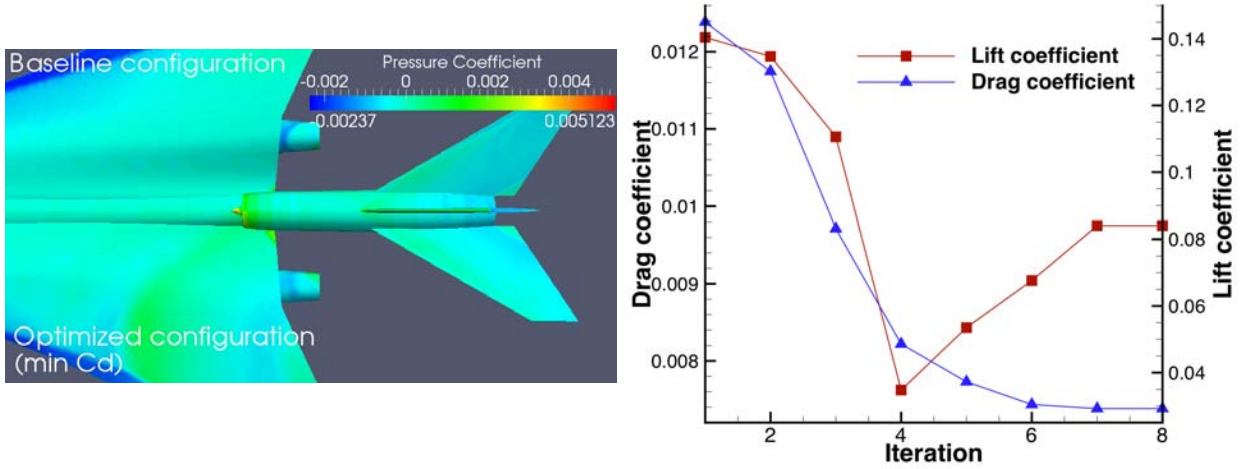


Figure 14. Detail of the shape design optimization (min Cd, left), and evolution of the optimization (right)

constraints as the optimizer. The design variable definition contains two Free Form Deformation boxes (main wing and horizontal tail plane). In Fig. 15 we show the upper and lower surfaces of the trijet (baseline and optimized configurations).

In Fig. 16 (left) we show the rear part of the aircraft with especial emphasis in the drag reduction on the nacelles and the modification of the horizontal tail plane. Finally in Fig. 16 (right) we show the evolution of the optimization, in this case the drag coefficient has been reduced by 20% with respect to the original configuration, while the C_L remains fixed at 0.14.

C. Equivalent Area inverse design problem

As an initial example of the entire methodology for equivalent-area-based adjoint design applied to complex three-dimensional problems, this section shows a sample calculation where the baseline trijet configuration is analyzed, the resulting equivalent area distribution is arbitrarily altered (including changes to the area of the base of the equivalent area distribution and, therefore, the lift of the configuration), and the configuration is redesigned using the FFD parameterization described in the next section in order to match as closely as possible the target equivalent area distribution for the entire aircraft.

This test case is viewed as a validation of the entire design procedure we have followed to compute the adjoint sensitivities of functions of the equivalent area distribution. It is, however, not intended to be a realistic design test case: a more typical use of this design procedure may focus on fixing shortcomings of the equivalent area distribution in a localized portion of the signature. In addition, realistic design processes may (or may not) require that the overall lift be fixed. In this example we were interested in finding out if, even with lift changes that may require angle of attack variations, the target equivalent area distributions could be achieved. To be completely clear, in order to show true demonstrations of the capability just developed additional efforts are being pursued in order to fix a target area distribution that exhibits the properties desired by the aircraft designer.

Fig. 17 shows the results of the initial analysis and surface shape sensitivity calculations of our sample design process. After an initial flow and adjoint solutions, these two solutions are combined to compute the surface shape sensitivity of the cost function. In other words, Fig. 17 shows both the upper surface (top) and the lower surface (bottom) sensitivities: by how much will our cost function (based on the difference between the actual equivalent area and the target equivalent area) change if each point on the surface of

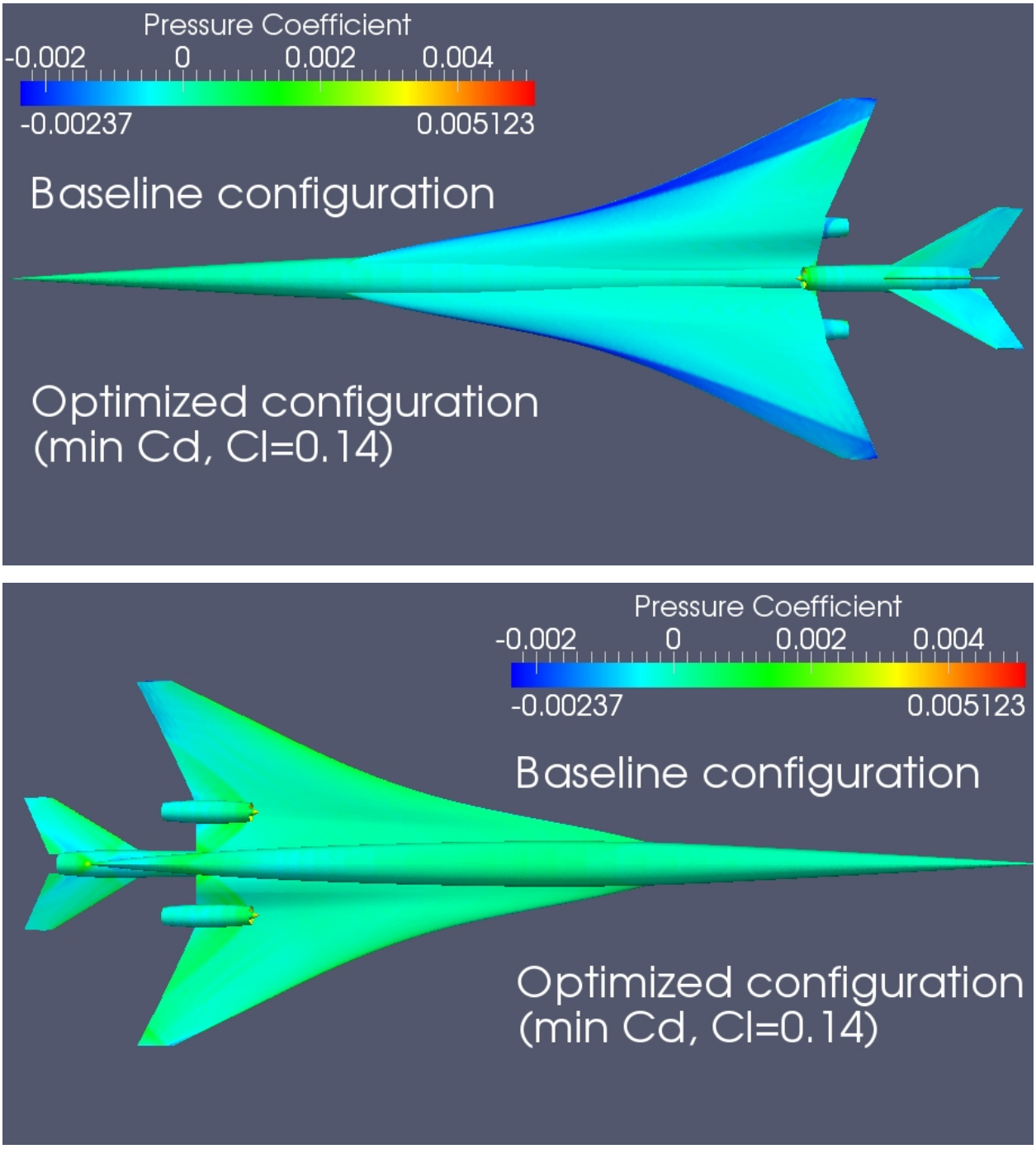


Figure 15. Shape design optimization (min Cd, $C_L = 0.14$), upper and lower surfaces of the trijet.

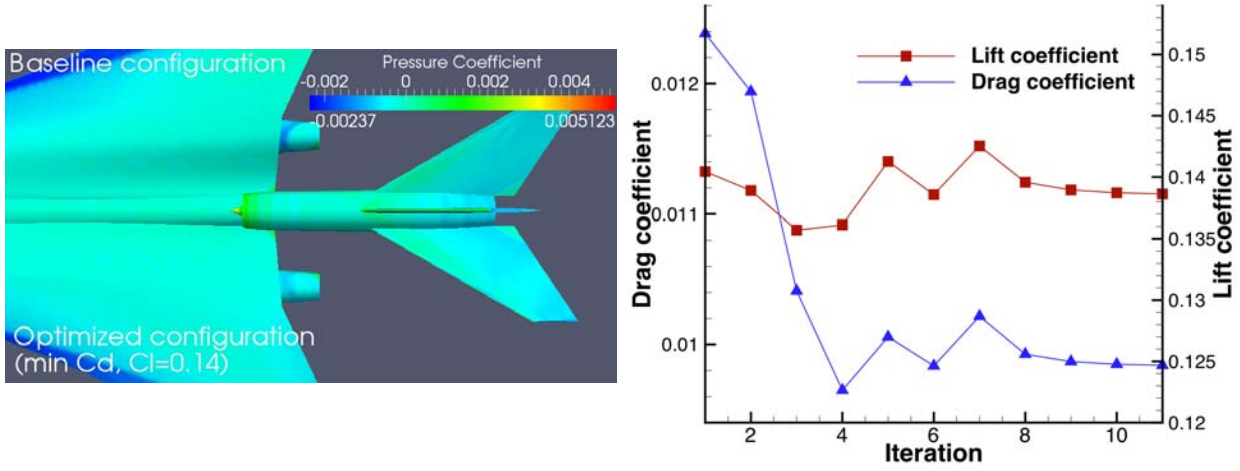


Figure 16. Detail of the shape design optimization (min Cd, $C_L = 0.14$, left), and evolution of the optimization (right)

the aircraft were to move along the local (positive) normal? The plots also show the C_p distribution on the upper and lower surface of the configuration for reference.

Note that these results (even without an actual optimization) are quite revealing: they point to the areas of the configuration where the impact on the cost function of the equivalent area distribution is largest and, therefore, the areas where the parameterization of the surface shape should be active. Owing to the supersonic flow nature of this problem (regions of dependence/regions of influence) it is clear that the largest contributions in order to reduce the difference between the target area distribution and the existing one are on the lower surface of the configuration ahead of the nacelles/inlets. In fact these areas of high influence can be seen to reflect in complicated ways off of the various surfaces of the configuration, making it nearly impossible to be able to figure such patterns out using designer's intuition alone. Note that, since the upper surface hardly influences the computation of the equivalent area distribution on the azimuthal plane right below the aircraft, the values of the shape sensitivities are rather small (though not identically zero).

Fig. 18 shows a large view and some details of the adjoint variable distribution (for the density variable) in the symmetry plane, on the surface of the aircraft, and on a cutting plane perpendicular to the incoming free stream. The adjoint variables are used to compute sensitivities of the equivalent area cost function to the deformation of the surface of the aircraft configuration.

Fig. 19 shows the Mach number distribution in the symmetry plane and the location of the near field plane.

Finally, Fig. 20 shows the initial, target, and final (after 5 iterations of the optimizer) equivalent area distributions and near-field C_p distributions under the flight track. Given the limited extent of the FFD boxes, the optimizer makes progress towards the target area distribution and manages to find a compromise (with lower C_L) of the best match it can find (within the provided parameterization) for the equivalent area distribution. After 5 iterations, the optimizer ceases to make progress (no better solution is available). As mentioned earlier, further study of this test case is warranted, using more realistic target area distributions that may be better suited for industrial design. The evolution of the value of the cost function is also show in Fig. 20 (right).

Fig. 22 (left and right) show the initial and final configuration that result from the optimization process. Both the lower and upper surface C_p distributions are presented in the Figure.

The differences between the initial and final geometries can be seen in Fig. 22 below. At this moment,

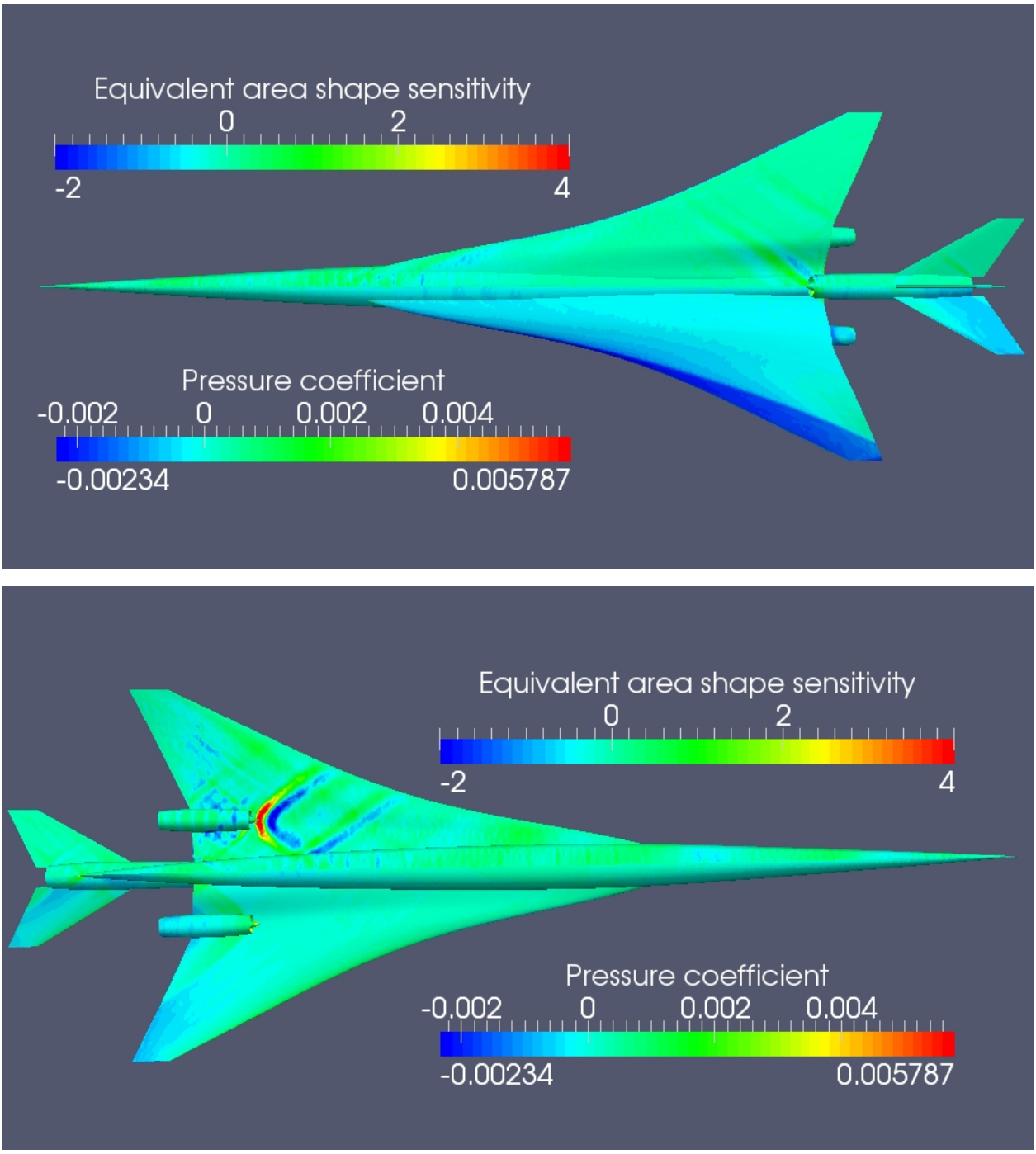


Figure 17. Upper and lower surface C_p and shape sensitivity distributions

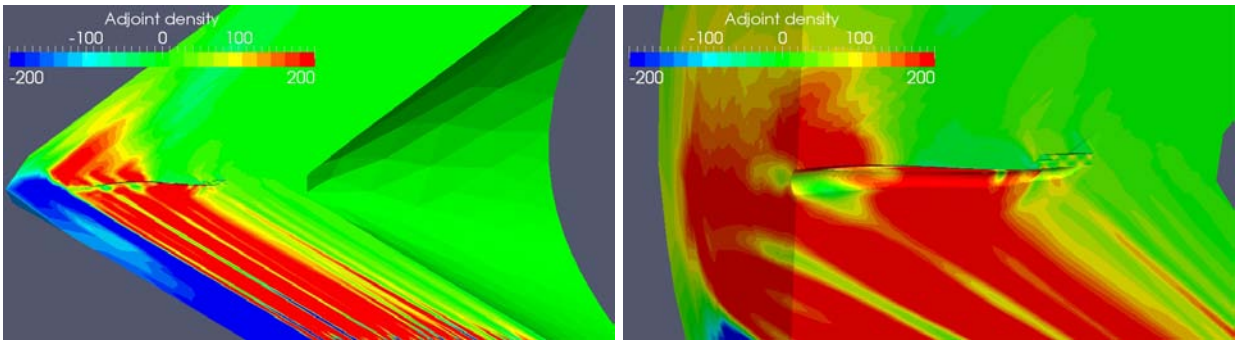


Figure 18. Adjoint of density variable: overall view and detail.

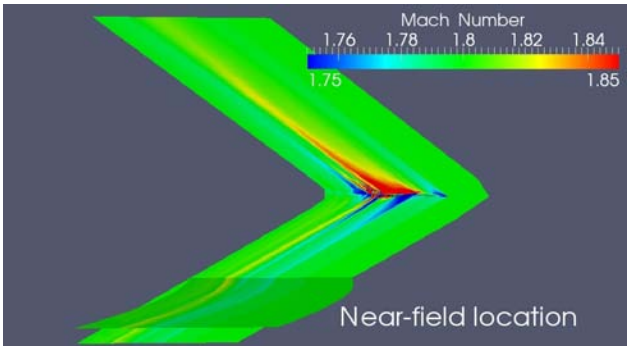


Figure 19. Near-field Mach number distribution (including near-field plane where adjoint variable jumps are imposed).

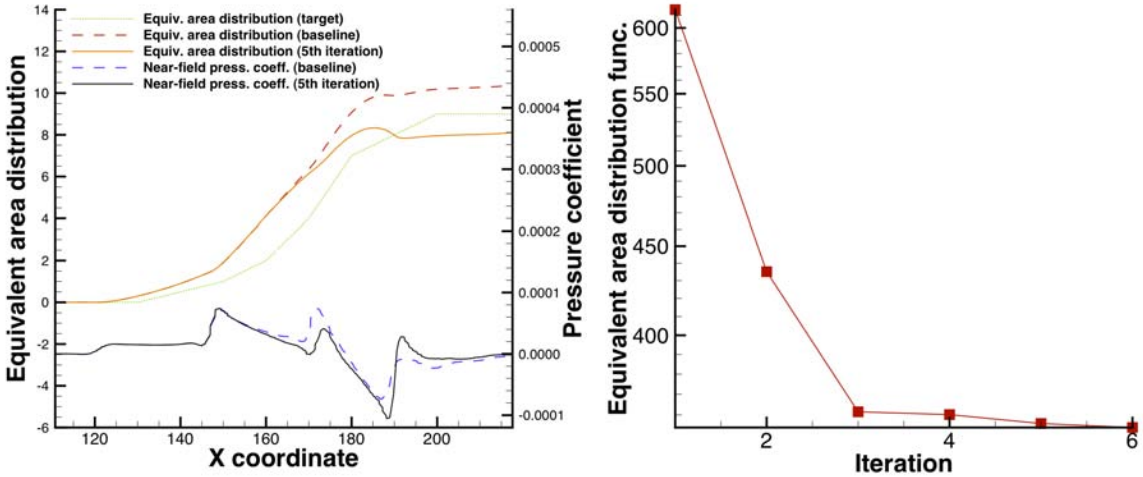


Figure 20. Equivalent area optimization results including equivalent areas and C_p distributions (left) and evolution of the cost function during optimization (right).

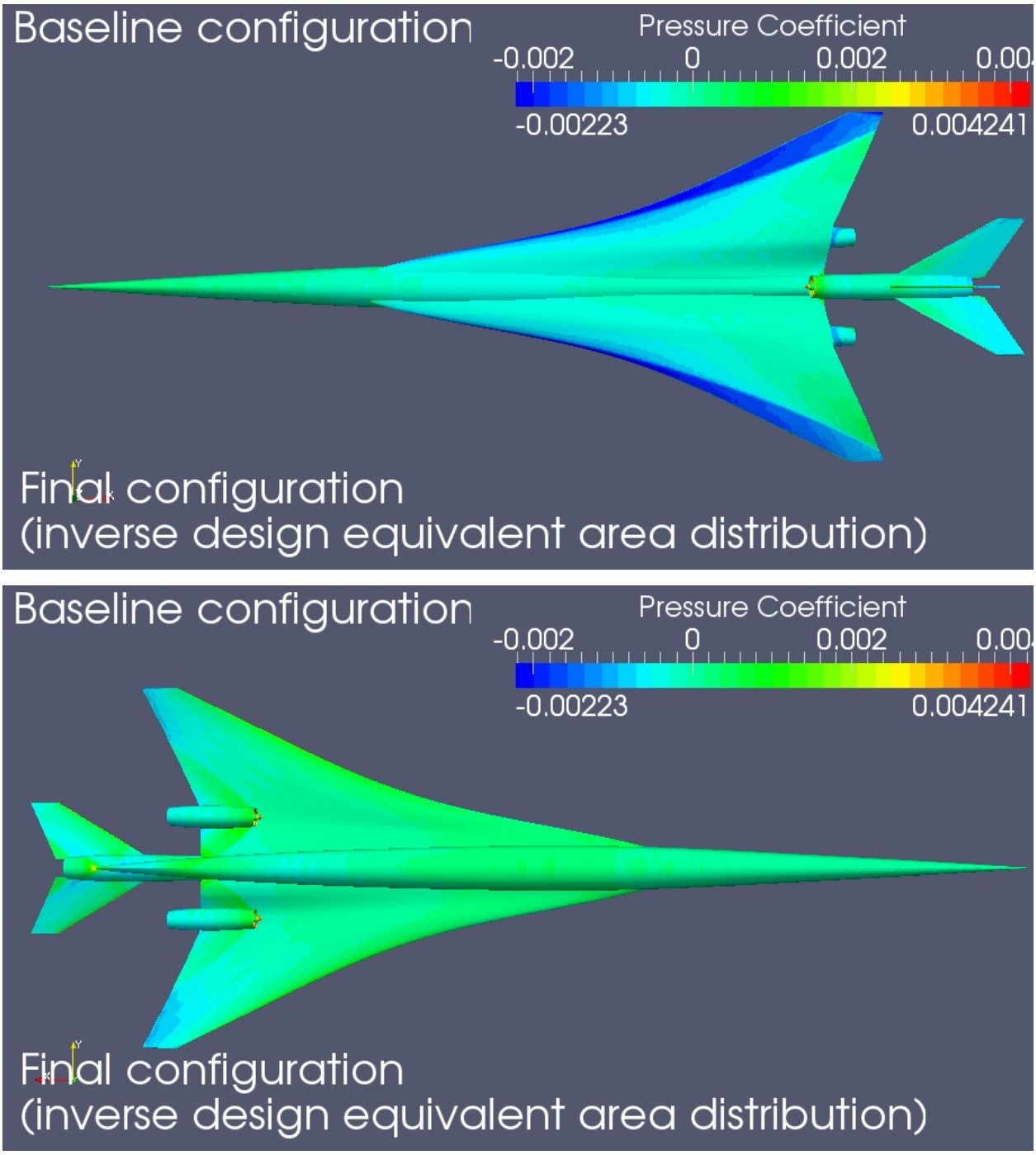


Figure 21. Upper (top) and lower (bottom) surface C_p distributions for the baseline configuration and the optimized configuration using the target equivalent area distributions.

these geometry differences are limited to the ones selected by the optimizer from within the range of geometric deformation possibilities offered by the free form deformation boxes we have selected for this study. Note that the optimizer has chosen to alter the twist distribution of the wing and to thin the rear section of the fuselage (which is obscured behind the initial grey surface.

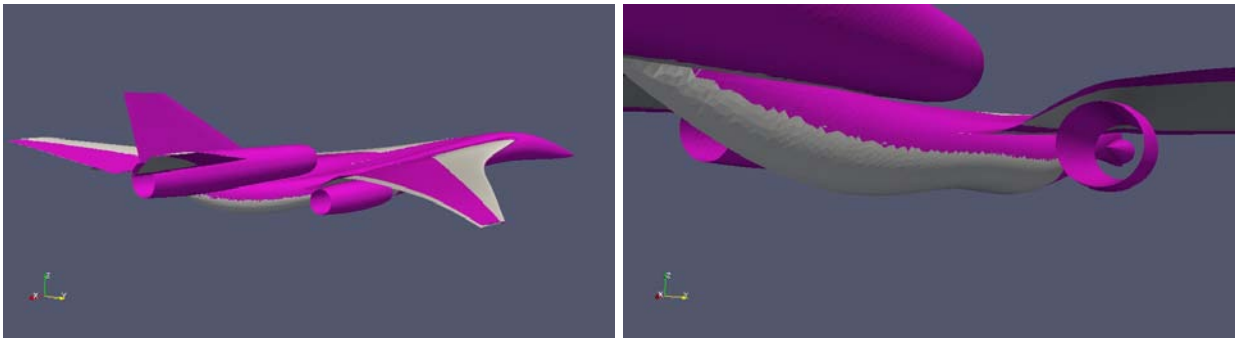


Figure 22. Geometric differences between baseline and optimized (according to equivalent area criterion) configurations. Such geometry perturbations are chosen by the optimizer from within the range of possibilities offered by the FFD boxes used in this study. Baseline geometry is shown in grey. Optimized geometry is shown in purple.

VIII. Conclusions and future work

In this work we have developed the methodology for the simulation and optimal shape design of supersonic aircraft. In order to close the design loop it has been necessary to treat the geometry modifications (CAD and a free-form deformation approaches), create apriori-adapted computational grids to resolve the near-field pressure distribution accurately, and formulate and implement the adjoint solver for low-boom design (based on cost functions related to the equivalent area distribution).

To the best of our knowledge this is the first application of adjoint methods to functionals based on the equivalent area distribution. Moreover, the use of Campbell's apriori-adapted grid methodology, combined with the design of the complete geometry and the computation of the so-called surface sensitivity is a contribution of this work.

Finally, we believe that we have demonstrated the viability of the technique in real 2D and 3D geometries, but work remains to be done to show that complex and realistic configurations can be treated by our methods. This work is currently ongoing.

IX. Acknowledgements

The authors would like to thank the inputs from and useful discussions with Nicole Norstrud, John Morgenstern, and Michael Bounanno of the Lockheed-Martin Corporation and the financial support of the NASA Supersonics Project of the NASA Fundamental Aeronautics Program.

References

[1]Darden, C., "Sonic Boom Theory: Its Status in Prediction and Minimization," *Journal of Aircraft*, Vol. 14, 1977, pp. 569–576.

[2]Aftosmis, M., Nemec, M., and Cliff, S., "Adjoint-based low-boom design with CART3D," *AIAA paper 2011-3500*, 29th AIAA Applied Aerodynamics Conference, Honolulu, Hawaii, June 2011.

[3]Alonso, J. J. and Colonno, M. R., "Multidisciplinary Optimization with Applications to Sonic-Boom Minimization," *Annual Review of Fluid Mechanics*, Vol. 44, 2012, pp. 505–526.

[4]Alonso, J. J., Martins, J. R. R. A., Reuther, J. J., Haimes, R., and Crawford, C., "High-Fidelity Aero-Structural Design Using a Parametric CAD-Based Model," *AIAA paper 2003-3429*, 16th AIAA Computational Fluid Dynamics Conference, Orlando, Florida, June 2003.

[5]Loseille, A., Dervieux, A., and Alauzet, F., "Fully anisotropic goal-oriented mesh adaptation for 3D steady Euler equations," *Journal of Computational Physics*, Vol. 229, No. 8, 2010, pp. 2866–2897.

[6]Park, M. A. and Darmofal, D. L., "Validation of an output-adaptive, tetrahedral cut-cell method for sonic boom prediction," *AIAA Journal*, Vol. 48, No. 9, Sept. 2010, pp. 1928–1945.

[7]Campbell, R. L., Carter, M. B., Deere, K. A., and Waithe, K. A., "Efficient unstructured grid adaptation methods for sonic boom prediction," *26th AIAA Applied Aerodynamics Conference*, No. AIAA–2008–7327, Honolulu, Hawaii, Aug. 2008.

[8]Escobar, J. M., Rodríguez, E., Montenegro, R., Montero, G., and González-Yuste, J. M., "Simultaneous untangling and smoothing of tetrahedral meshes," *Computer Methods in Applied Mechanics and Engineering*, Vol. 192, No. 25, 2003, pp. 2775–2787.

[9]Nadarajah, S., Jameson, A., and Alonso, J. J., "An Adjoint Method for the Calculation of Remote Sensitivities in Supersonic Flow," *40th AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA–2002–0261, Reno, Nevada, Jan. 2002.

[10]Bueno-Orovio, A., Castro, C., Palacios, F., and Zuazua, E., "Continuous Adjoint Approach for the Spalart-Allmaras Model in Aerodynamic Optimization," *49th AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA–2011–1299, Orlando, Florida, Jan. 2011.

[11]Sokolowski, J. Zolesio, J.-P., *Introduction to shape optimization*, Springer Verlag, New York, 1991.

[12]Samareh, J., "Aerodynamic shape optimization based on Free–Form deformation," *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, No. AIAA–2004–4630, Albany, New York, Aug. 2004.