INRIA 18th month Technical report to HISAC 2.3.2

Youssef Mesri
Laurent Hascoet
Frédéric Alauzet
Adrien Loseille
Alain Dervieux

December 7, 2006

Summary

The role of INRIA in HISAC is to contribute to the developement of methods for the simulation and reduction of sonic boom. The contribution of INRIA to Task 2.3.2 concerns the application of Optimal Control methods to the optimisation of the nearfield flow around a supersonic aircraft. It is based on the application of an optimisation loop. We describe this optimisation loop, which based on an Euler model and an adjoint-based gradient descent algorithm. Adjoint and gradients are obtained by applying Automated Differentiation techniques. The main novelty is the combination with a new mesh adaptation technique introduced in Task 2.3.1. This new tool is demonstrated on the supersonic geometry HISAC_2_REF-IT3 provided by Dassault-Aviation.

1 Existing methods for sonic boom reduction

As most problems in Aerodynamics, the analysis of sonic boom could be performed by relying on the compressible Navier-Stokes. However, several obstacles should be solved before:
- of course this involves problems related to turbulence modeling: in fact in a first phase, the prediction of Euler model is considered as accurate enough,
- even with the Euler model, the integration from aircraft to the ground (without speaking about the shock reflected by stratosphere) is today out of reach with existing 3D Euler methods.
Instead, as appears from the Task 2.3.1 reports, today's simulation of sonic boom relies on two types of methods:
- simplified models from aircraft body to ground,
- composite models involving a 3D near field calculation (applying typically to 1-5 chords length around the aircraft), matching with a 2D simplified model performing the propagation from R/L=1 to ground.

As an example of the first approach, Farhat *et al.* [Farhat et al. 2002] [Farhat et al. 2002b] [Farhat et al. 2002c] [Farhat et al. 2004] iterate the optimisation loop on two state systems, the Euler near field for aerodynamical properties, and a Whitham model for sonic boom propagation to ground. An important difficulty on this way is the poor differentiability of certain propagation kernels with respect to shape or near field data. This is an obstacle to the use of gradient-based optimisation. This point can motivate the design of methods in which the propagation model would not be directly included inside the optimisation loop.
Intermediate to both approaches, Alonso *al.*, see [Nadarajah et al. 2004] [Choi et al. 2004] [Choi et al. 2004bis] [Choi et al. 2004ter] [Choi et al. 2005] [Nadarajah et al. 2005] propose to iterate on the near field shape and flow in order to reduce the deviation with respect to a target pressure on matching region. The target pressure is built from inversing a ground target signature.
Since at simulation level already, the accuracy of the ground signature prediction matching is a delicate problem related to a rough matching between both model, several teams in HISAC Task 2.3 try to improve the accuracy by combining a better 3D Euler output with improved Whitham type propagation models.

The role of INRIA in Task 2.3.1 is to improve the 3D Euler nearfield output. The approach chosen is the application of a novel adaptative mesh generator. For the present task (Task 2.3.2), the use of mesh adaptation makes the optimisation problem quite stiffer, and INRIA's contribution involves addressing these issues.

2 Methods of the proposed tool

The sonic boom reduction methods and tool that are developed by Partner INRIA in Task 2.3.2 is built from the simulation methods and tool developed in Task 2.3.1. It relies on an Euler numerical model and an optimisation loop defined from the Optimal Control Theory. We present now the main features of it.

2.1 Euler numerical model

2.1.1 Governing equations

The Euler equations, which express the conservation of mass, momentum, and energy for the flow of inviscid, compressible fluids, may be written in the following integral conservation law form

$$\frac{\partial}{\partial t}\int_V W dV + \oint_S \mathcal{F}\cdot\mathbf{n}dS = 0 \tag{1}$$

where W is the vector of conserved variables and $\mathcal{F}$ the flux of W across the bounding surface S with outward unit normal $\mathbf{n}$ of a any control volume V. The column vector W and flux vector $\mathcal{F}$ are given by

$$W = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}; \qquad \mathcal{F}(W) = \begin{pmatrix} \rho\mathbf{u} \\ \rho\mathbf{u}u + p\mathbf{i}_x \\ \rho\mathbf{u}v + p\mathbf{i}_y \\ \rho\mathbf{u}w + p\mathbf{i}_z \\ \rho\mathbf{u}H \end{pmatrix} \tag{2}$$

Here ρ, p, and E represent the fluid density, thermodynamic pressure, and total energy per unit mass. u, v, and w are the Cartesian components of the velocity vector $\mathbf{u}$ and H is the total enthalpy given by $H = E + \frac{p}{\rho}$. If the fluid is assumed to be a thermally perfect ideal gas, then the closure equation linking the pressure p and the conserved quantities ρ and E is provided by the equation of state

$$p = \rho(\gamma-1)[E - \frac{1}{2}(u^2+v^2+w^2)] \tag{3}$$

in which γ stands for the ratio of specific heats at constant pressure and volume.

2.1.2 Spatial discretization

The Euler system is solved by means of a vertex-centered Mixed-Element-Volume approximation on unstructured meshes, as in [Stoufflet et al. 1987]. The consistent part is a Galerkin formulation. In the corresponding discretization, the test function is taken into a V_h included in $(H^1(\Omega))^5$.

$$(\Psi_{Euler}(\gamma, W), \phi) \;=\; \int_\Omega \phi\nabla.\mathcal{F}_h(W)d\Omega - \int_{\partial\Omega}\phi\bar{\mathcal{F}}_h(W).nd\partial\Omega$$

where the integral involving $\mathcal{F}_h(W)$ holds for the upwind approximation of internal fluxes, which writes, for a particular vertex i in terms of flux through dual cells boundaries:

$$\int_\Omega \phi_i \nabla . \mathcal{F}_h(W) d\Omega \;=\; \Sigma\; \Phi^{ROE}(W_{ij}^{limited}, W_{ji}^{limited}, \eta_{ij})$$

where the sum is taken for edges ij around vertex i. The stabilising part relies on a Roe Riemann solver combined with a MUSCL reconstruction with TVD limiters. Symbol η_{ij} holds for the integral of normal vector to cell boundary between i and j.
The boundary condition are included inside the $\bar{\mathcal{F}}(W)$ and $\bar{\mathcal{F}}_h(W)$ boundary integrand. This produces a space accuracy of order two. Let us mention that for solution of the steady system, an implicit pseudo-time integration is applied, the approximate Jacobian of which will be re-used for solving the adjoint system.

2.2 Optimisation loop

The optimisation problem is stated as the research of the geometry γ_{opt} which will minimize an implicit objective functional $j(\gamma)$:

$$\gamma_{opt} = ArgMin\; j(\gamma)\;.$$

The implicit functional is expressed in terms of a state variable depending on the control variable γ:

$$j(\gamma) \;=\; J(\gamma, W(\gamma))$$

where the state $W(\gamma)$ is the solution of the state system, viz. Euler equations, written as follows:

$$W = W(\gamma) \;\Leftrightarrow\; \Psi_{Euler}(\gamma, W) \;=\; 0\;.$$

The objective function measures the deviation of the Euler solution from a pressure target on the bottom of the nearfield computational domain. This would be used for sonic boom reduction by applying a two-step process:
Step 1: derive a nearfield target pressure distribution which results after propagation down to the ground in an acceptable sonic boom. This can be done by an inverse iteration on a farfield propagation model.
Step 2: apply the present optimization loop for obtaining a shape giving a nearfield pressure close to the target pressure.

Reduced to the nearfield, this problem is much less difficult that an Euler attack down to the ground. But even in these simplified conditions, a good evaluation of the nearfield pressure remains difficult. To obtain an accurate answer, we shall combine the Euler solver with a mesh adaptation algorithm. This means that the shape optimization will be coupled with the mesh improvement algorithm. The mesh adaptation is developed in the context of Task 2.3.1. See T2.3.1 INRIA report.

2.2.1 The optimal control model

Our framework is the following general constrained minimization problem:

$$\text{Arg Min}\; J(\gamma, W), \quad \text{subject to} \quad \Psi(\gamma, W) = 0 \tag{4}$$

where the minimum is taken with respect to the composite variable $x = (\gamma, W)$. In other words, we want to find the $x_{opt} = (\gamma_{opt}, W_{opt})$ that minimizes the *objective functional* $J(x)$, where x_{opt} must in addition satisfy the *equality constraint* $\Psi(x) = 0$.

Let us assume that the Jacobian

$$A = \frac{\partial \Psi}{\partial W} \tag{5}$$

is always invertible. Then the minimum we are looking for is the solution of the following *Karush-Kuhn-Tucker* (KKT) system:

$$\left\{ \begin{array}{rcll} \Psi(\gamma, W) & = & 0 & \\ & & & \textit{(State)} \\ \dfrac{\partial J}{\partial W}(\gamma, W) - \left(\dfrac{\partial \Psi}{\partial W}(\gamma, W)\right)^* \cdot \Pi & = & 0 & \\ & & & \textit{(Adjoint state)} \\ \dfrac{\partial J}{\partial \gamma}(\gamma, W) - \left(\dfrac{\partial \Psi}{\partial \gamma}(\gamma, W)\right)^* \cdot \Pi & = & j'(\gamma) = 0 & \\ & & & \textit{(Optimality)} \end{array} \right. \tag{6}$$

This is the system that the Optimal Control loop must solve. Formally, this involves as usual an assembly step and a resolution step. The assembly step will take as input the current value of the variables that will eventually hold the result, which are:

- the control parameters γ,
- the state variables W,
- the *co-state* or *adjoint* variables Π.

The assembly step will compute each left-hand-side in system (6), and the resolution step will use them to update the variables until the residual is zero. Since the system is non-linear, this process will be iterative. Thus assembly and resolution will be called repeatedly.

We observe that parts of both steps are already available in the original simulation code. Specifically, the assembly of the Ψ residual, and the resolution for W, i.e. what concerns the non differentiated symbols.

For the assembly part, what is missing is the routine computing the terms that involve derivatives of Ψ and J. We derive their code from the assembly code of Ψ and the computation code of J, using Automatic Differentiation (**AD**). We remark that the two terms that involve derivatives of Ψ are indeed of the *transposed-Jacobian-times-vector* kind. The same holds for the terms that involve derivatives of J, only in the degenerate case of a single row Jacobian. Therefore, we use the so-called *reverse mode* of AD of Tapenade which is able to produce code that computes transposed-Jacobian-times-vector derivatives in remarkably few computations. See [Hascoet et al. 2003] for further details.

For the resolution part, we need a composite algorithm that will combine

1. the existing resolution of the state equations, yielding W,
2. with a resolution algorithm for the adjoint state equations, yielding Π,
3. and with a minimization algorithm for the optimality equations, yielding the optimal control parameters γ_{opt}.

We must develop the algorithms for Π and γ. In theory, the algorithm for Π and its usage in the assembly of $j'(\gamma)$ could be generated automatically, by reverse-mode AD of the existing algorithm for $j(\gamma)$. However, for efficiency reasons, we think it is better to write the resolution for Π by hand. Moreover, the resolution for Π can make use of crucial parts of the existing resolution for W, and is itself a key component to be re-used in many places, as we show for the second-order derivatives that are needed for robust optimization.

2.2.2 Resolution algorithms

Assume that, with the help of Automatic Differentiation applied to the assembly routines of the original simulation code, we have obtained the assembly routines for the different ingredients of the KKT system (6). Specifically, we now have routines that, given a γ and a W, compute efficiently

$$\frac{\partial J}{\partial W}(\gamma, W) \quad \text{and} \quad \frac{\partial J}{\partial \gamma}(\gamma, W) \ ,$$

and given an additional argument Π,

$$\left(\frac{\partial \Psi}{\partial W}(\gamma, W)\right)^* \cdot \Pi \quad \text{and} \quad \left(\frac{\partial \Psi}{\partial \gamma}(\gamma, W)\right)^* \cdot \Pi \ .$$

Computing the gradient of the objective functional with AD

Our goal is now to compute the gradient $j'(\gamma)$. We will apply a procedure that follows from system (6) line by line:

1. first solve the state equations, yielding W
2. then solve the adjoint state equations, yielding Π
3. finally assemble the residual of the optimality equations, yielding $j'(\gamma)$.

In this section we do not address the topmost optimization loop that reduces $j'(\gamma)$ to zero.

Resolution of the state equations (step 1) is of course already available in the initial simulation code. We assume, as it is generally the case, that this resolution uses a matrix-free iterative solver which repeatedly calls the assembly of the state residual Ψ. For example, it can be a pseudo-unsteady explicit time-stepping or a GMRES quasi-Newton iteration.

It is important to understand why we choose to go through step 2, i.e. explicitly solve for the adjoint state Π. Why don't we instead ask directly the AD tool to reverse-differentiate the routine that computes $j(\gamma)$? This would return the gradient $j'(\gamma)$. In fact, this has been done before with success, e.g. in [Mohammadi 1997]. But this straightforward approach has several severe drawbacks, that we shall put in two categories for discussion.

The first category of drawbacks is about efficiency. The differentiated code uses an enormous amount of memory, related to the reverse mode principle. Essentially, each of the non-converged iterates of the state W need be stored. In the present state of the art, even with data-flow analyses, radical manual post-processing of the differentiated code is necessary. Moreover, the systematic approach differentiates computations that are in fact irrelevant, such as evaluation of the time-step, and this hampers efficiency, requiring manual post-processing. The last drawback in this category is the fact that we cannot expect the

derivatives to converge at the same rate as W. In other words, it is questionable to perform the same number of iterations to converge the derivatives during the backward sweep, than to converge W during the forward sweep. Unfortunately, this is exactly what straightforward AD does.

One can think of an elegant way to overcome these drawbacks, which we might call "fixpoint-conscious-AD". We could modify the reverse-AD model for fixpoint iterations so that none of the iterates W_k of W is stored, except the final W_N, which is converged up to a tolerance ε. The backward sweep of the differentiated program would repeatedly use the values from W_N, even when reversing the computations of another time step $k \neq N$. This clearly solves the memory question. Moreover, this allows the backward sweep to perform a different number of iterations, and in particular to use a specific stopping criterion for the backward iterative loop, involving convergence of the derivatives themselves. Fixpoint-conscious AD could even be automated inside AD tools, freeing us from the tedious and error-prone post-processing task.

The second category of drawbacks comes from the iteration algorithm itself. If explicit pseudo-time stepping is used, the state iteration is a linear fixed point, and the transposed iteration performed by the differentiated code will also be stable and converging. On the other hand, if the state iteration is far from linear, typically because of line-searches or orthonormalization, then there is no guarantee that the differentiated transposed iteration is stable nor convergent, let alone efficient. Finally, in the case of non-linear iterations, there is very little mathematical insight of the consequences of freezing the state to W_N.

Therefore we recommend in general not to differentiate the fixed point iteration itself. We recommend instead to re-use the iteration algorithm, possibly changing the pre-conditioner which has to be simply transposed. In [Courty et al. 2003], this strategy is applied, using a first-order simplified Jacobian as a pre-conditioner.

One-shot optimization

Modern finite-dimensional optimization methods relying on adjoints are issued from the Sequential Quadratic Programming (SQP) methodology. A popular prototype is the Byrd-Omojokun algorithm, see [Nocedal et al. 1999]. This algorithm in its basic form assumes that the resolution of the different linearizations of state systems (Newton iteration of state and solution of adjoint) are not expensive. However, this assumption is not valid in Optimal Shape Design. This fact has lead some authors to attack the problem using *one-shot* (or *progressive*, or *simultaneous*) algorithms [Taasan et al. 1992, Dadone et al. 2000], which are based in the following two principles:

- Use discipline-specific iterative, maybe nonlinear solvers (for example, pseudo unsteady solvers for Fluid Mechanics) for state and co-state.
- Iterate simultaneously the three equations of the KKT system.

The cost by iteration of these algorithms is much lower with a comparable convergence. Assuming they converge in a number of iterations independent from the discretization fineness, it follows that they are potentially able to reach optimal complexity, in the sense that the solution costs k times the resolution of the state equation, k being independent from the number of control parameters. The efficiency of this method is shown for example in [Dervieux et al. 2004]. But the question of independence from the discretization fineness

remains to be addressed.

Multi-level Optimization

Large scale problems coming from Partial Differential Equations generally result in a conditioning which is poor and getting poorer as the number of degrees of freedom grows. The reason for this can be found either through a direct analysis of discrete eigenvalues as the number of unknowns increases, or through an analysis of the continuous -functional- problem and the continuous version of the algorithm. Shape design problem possess a continuous formulation and the corresponding sensitivity has been analyzed by Hadamard about one century ago. It appears that the gradient is a non-bounded operator. Its usage leads to ill-conditioned iterations. This problem can be tackled by applying an **additive multilevel pre-conditioner** B, which is applied to the iterative procedure:

$$\gamma^{n+1} = \gamma^n - \rho B g_{L^2}, \tag{7}$$

At each iteration n the correction coming from the optimization process is updated. The correction consists of a step-length factor ρ multiplying the preconditioned gradient. The self-adjoint invertible operator B is chosen in order to recover the degree of regularity lost by the L^2 gradient g_{L^2}. We refer to [Courty et al. 2005b, Courty et al. 2005a, Dervieux et al. 2004] for theoretical aspects and applications.

3 Work plan

The successive steps of the above program are:
a.Building a shape optimization loop.
b.Building a mesh improvement phase.
c. combining both previous steps. Step a can be splitt as follows:
a1. *Choosing a shape parameter.* this is done starting from the mesh of an aircraft skin and allowing the deformation of it determined by the motion of each vertex along a vector field which is normal to the surface.
a2. *Parameterizing an Euler flow solver with a shape parameter.* this is done with a transpiration condition.
a3. *Computation of sensitivities for functional J and residual Ψ.*
a4. *Solution of adjoint state*
a5. *Assembly of the gradient of j.*
a6. *Combination of the above steps with a gradient optimization algorithm.*

4 Work done on march 21st, 2006

At this date steps a1 to a3 are complete:

We have chosen a few main options already used in [Vàzquez et al. 2004]. We define the variable aircraft *shape* from the initial Workshop geometry as follows: a motion of skin mesh nodes normally to the initial geometry is applied with some amplitude. The shape variation is taken into account by means of a transpiration condition in order to continue using the initial geometry mesh. In this early stage of development, the functional is simplified as the

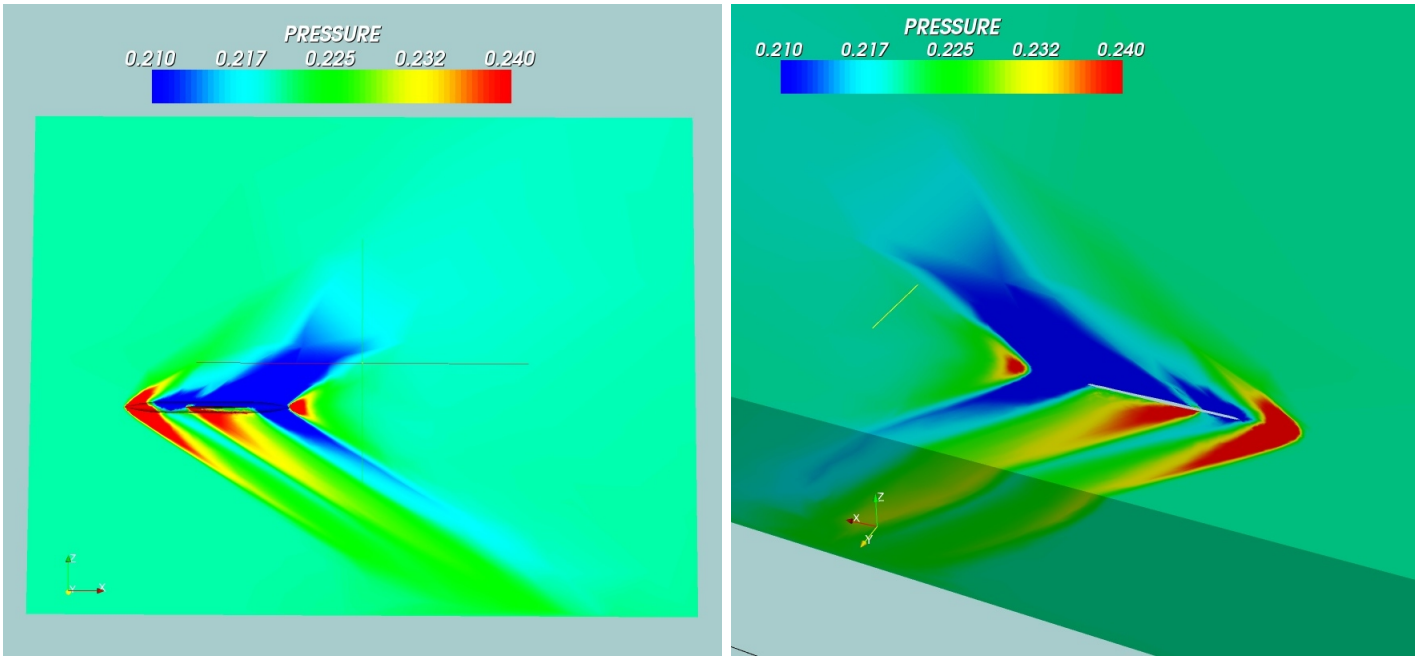

Figure 1: HISAC_2_REF-IT3. Pressure field on a vertical plan and on the horizontal one used for functional evaluation. Anisotropic mesh, 11K vertices

deviation between pressure and a target pressure measured between two planes under the aircraft.

We present some illustration of the recent developments on the proposed HISAC_2_REF-IT3 geometry. The initial mesh is adapted enough to provide a rather good pressure field in the vicinity of aircraft. see Fig. 1. The sensitivity of the functional to flow field, that is the right hand side of adjoint state is presented in Fig.2. We have validated the adjoint state and display on Fig.3 the values of its first component on an horizontal plane at aircraft level. The total gradient of the cost functional with respect to shape has been validated by comparison of a few component with divided differences. For this, flow convergence is pushed to 10^{-12} residual decay. Then gradient validation by comparison with divided differences is obtained with about 8 identical digits.

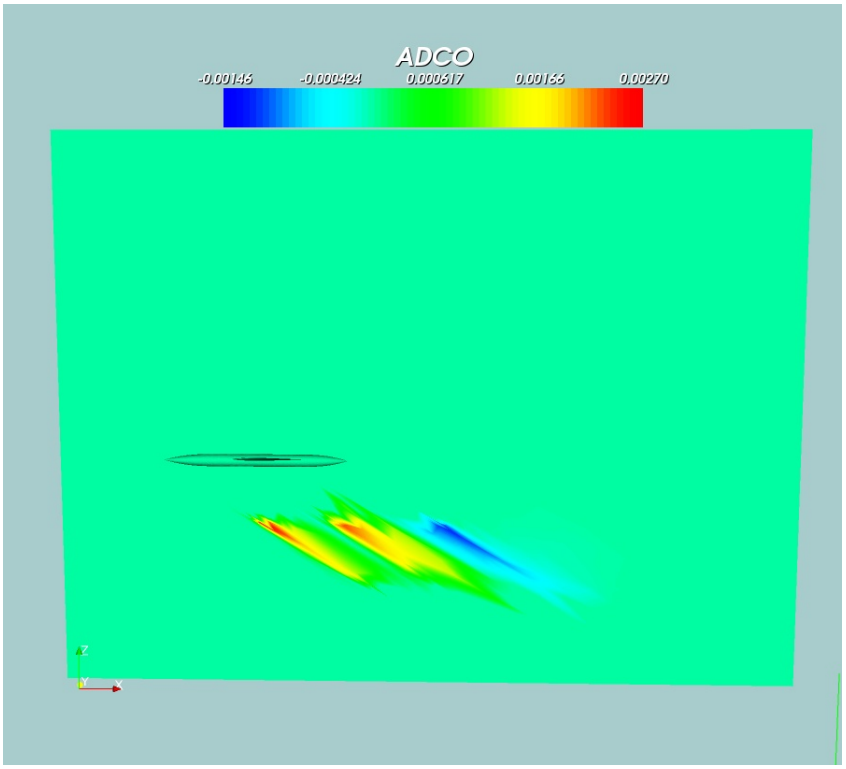

Figure 2: HISAC_2_REF-IT3: First component of functional derivative with respect to state, i.e. right hand side adjoint state. Values on symmetry plan.

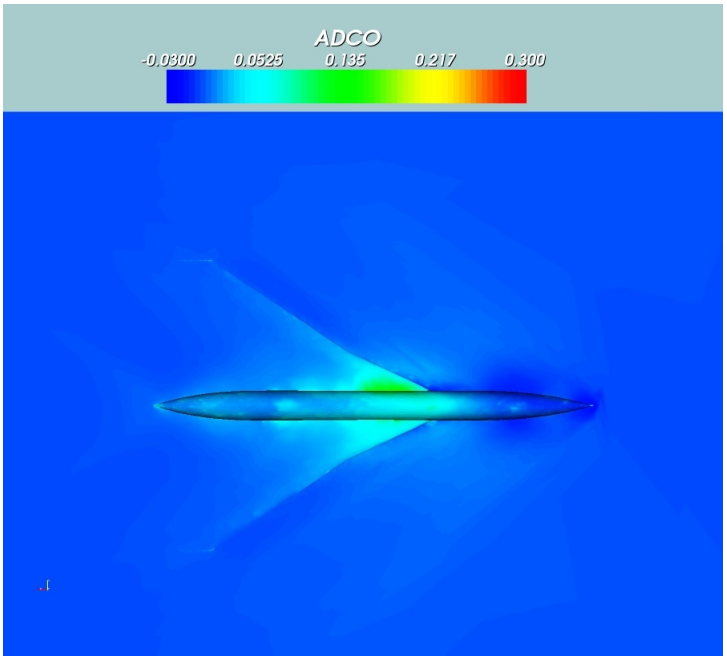

Figure 3: HISAC_2_REF-IT3: First component of adjoint state. Values at bottom of aircraft

5 Work done on june 14th, 2006

5.1 Validation of gradient with an adapted mesh

A new adapted mesh with strong stretching and $35,000$ nodes is considered. Both flow convergence and gradient accuracy are penalised of about one order of magnitude in the mesh adapted context:
- flow convergence is limited to 10^{-11} residual decay,
- gradient validation by comparison with divided differences looses one digit.
However, second-order divided differences produced the best matching, with a relative error between both around 10^{-7} (seven exact digits), see an example of this comparison in Fig.4. The cost functional gradient on the aircraft skin is displayed in Fig.5.

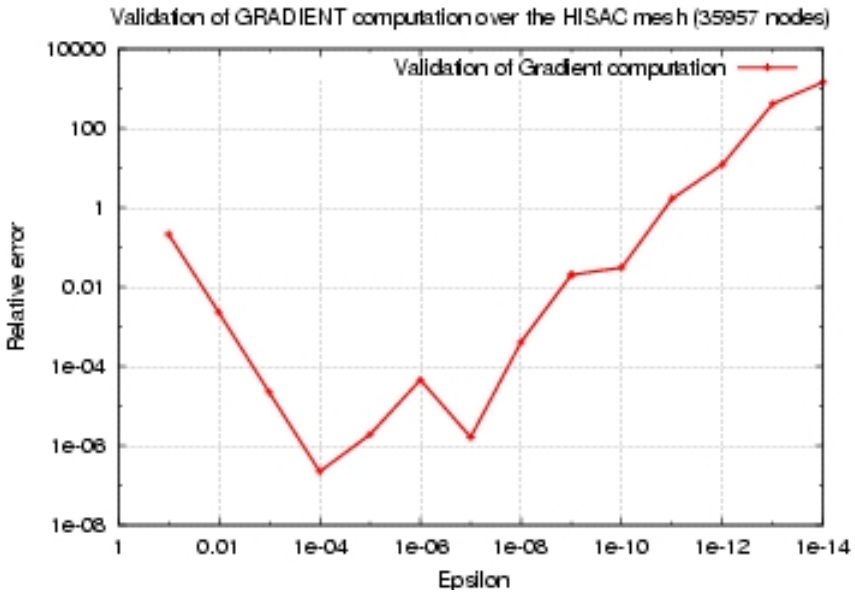

Figure 4: HISAC_2_REF-IT3, strongly adapted mesh (35 K vertices): validation of analytic gradient versus divided differences for a particular component

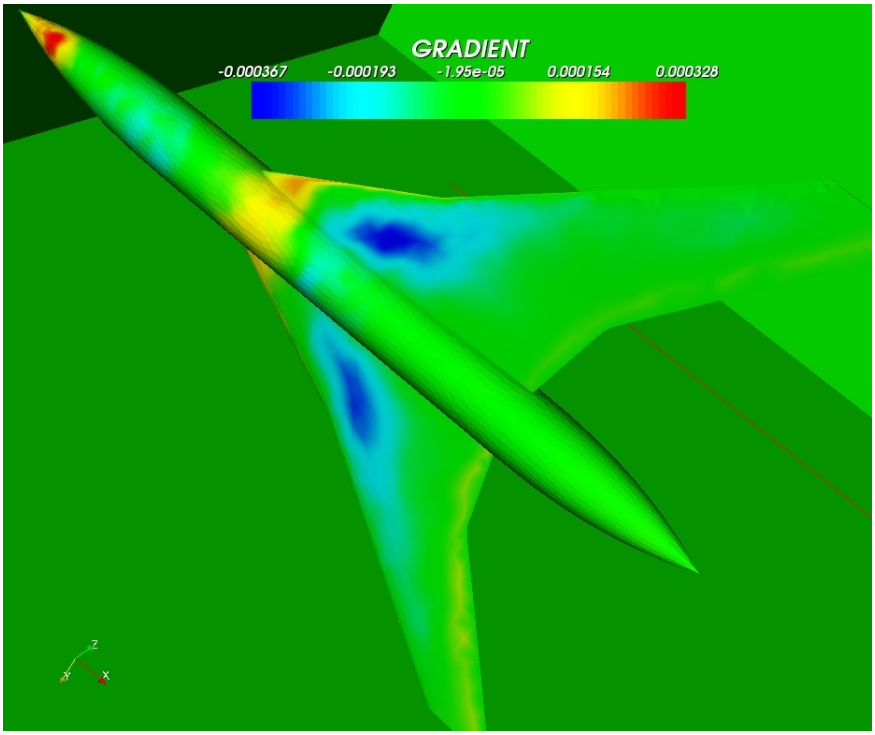

Figure 5: HISAC_2_REF-IT3, strongly adapted mesh: gradient of objective with respect to shape

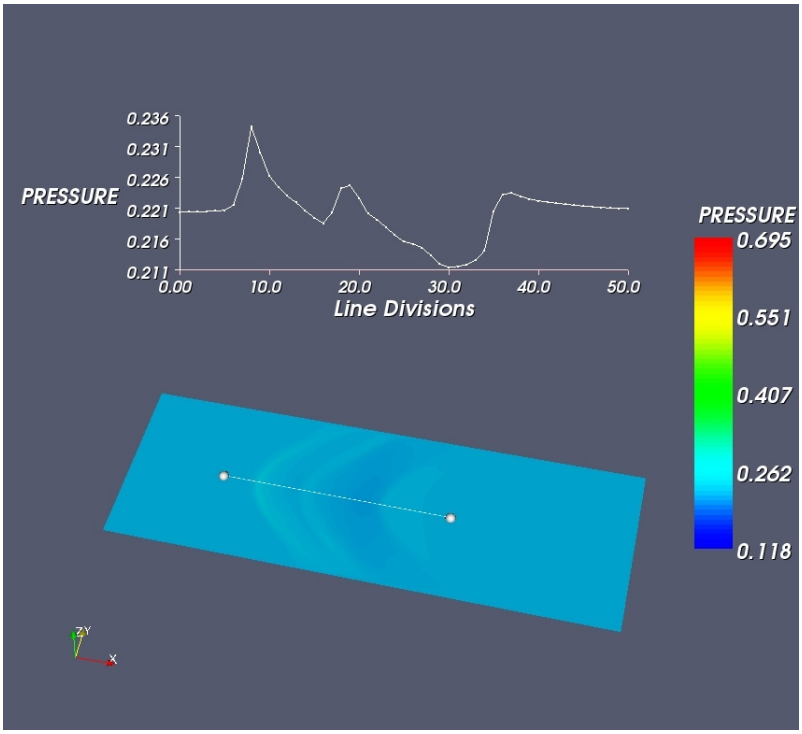

Figure 6: HISAC_2_REF-IT3, strongly adapted mesh: pression signature

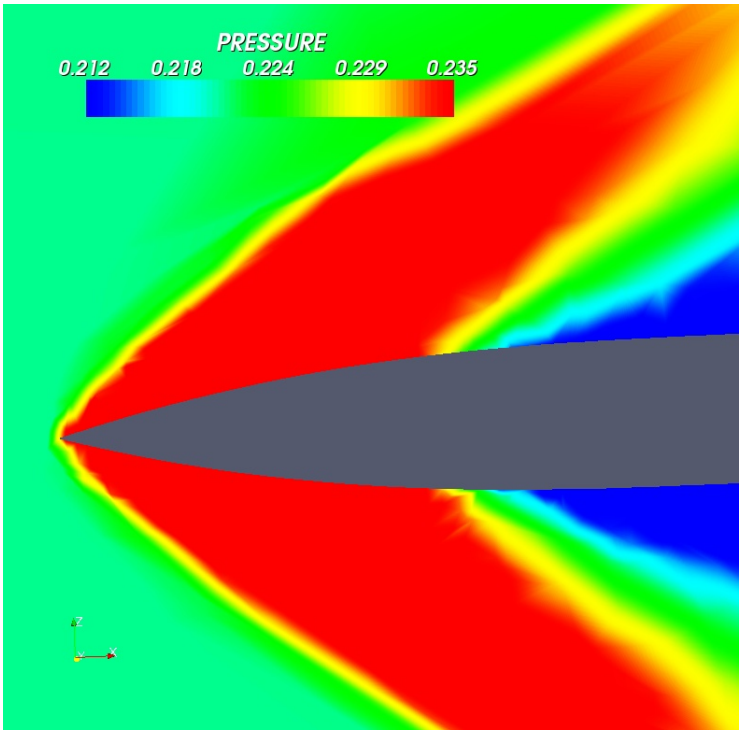

Figure 7: HISAC_2_REF-IT3, strongly adapted mesh: initial pressure at forebody

First optimisation:
The first optimisation experiment consists in trying to solve an inverse problem. The purpose is to build shapes giving a smaller initial pressure rise at one chord ($R/L = 1$) under the aircraft. The initial flow is depicted in Fig.6. Fig.7 presents a zoom at aircraft forebody. We choose as target pressure a pressure field equal to initial one, except that values larger than .225 are replaced by this value. After one iteration of optimization (steepest gradient), we observe that the desired effect is approached, that is that the initial pressure rise is smaller, but only of a few percents. Fig.8 presents the resulting pressure field around the aircraft. Fig.9 presents a zoom of it near forebody. We recall that the modified geometry is not shown since it is taken into account via boundary conditions. Fig.10 presents the resulting pressure signature at the $R/L = 1$ reference level.
This first calculation also demonstrates the strong coupling that is necessary between mesh adaptation and optimisation. Indeed, the initial shock is moved by the shape modification. But it is then less well captured by the mesh initially adapted for the initial flow.

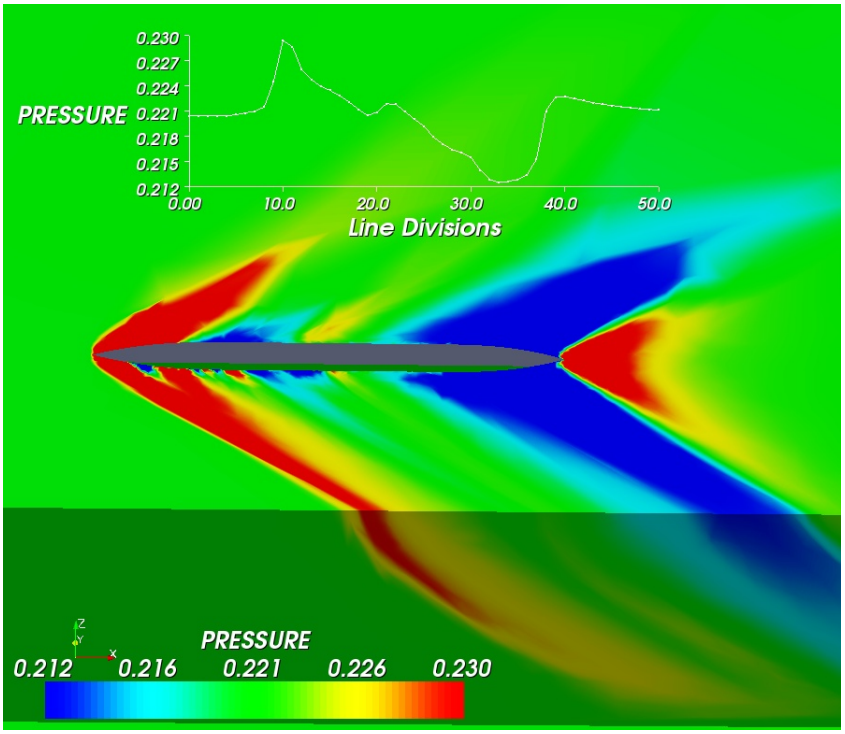

Figure 8: HISAC_2_REF-IT3, strongly adapted mesh: optimised pressure field

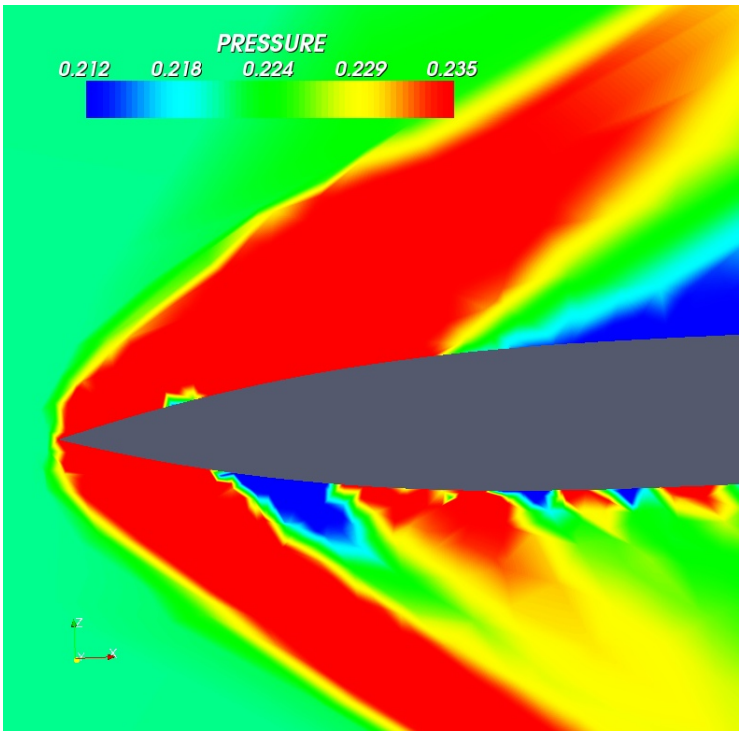

Figure 9: HISAC_2_REF-IT3, strongly adapted mesh: final pressure at forebody

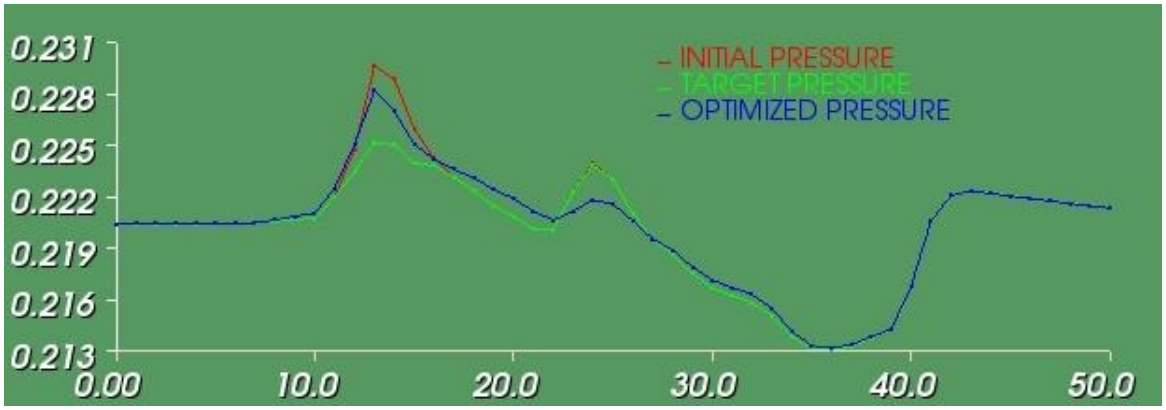

Figure 10: HISAC_2_REF-IT3, strongly adapted mesh: comparison of pression signature of initial flow (red), target flow (green), and optimised flow after 1 iteration (blue)

5.2 Preliminary study for propagation optimization

After a common reflexion with some partners, it appeared that it would be useful if INRIA also investigate the possibility to differentiate some propagation model.
A in-house propagation code `SonicBoom` based on the waveform parameter method of Thomas [Thomas, 1972] has been developed to propagate the near field perturbations to the ground. See the report for Task 2.3.1.
The application of TAPENADE differentiator ([Tapenade,2003]) to this model is currently studied.

6 Work done between june and november 2006

In the previous step, we checked that we could make an objective functional decrease for a fixed mesh. First with a standard mesh, second for a loosely adapted mesh. We have improved our solver in order to have an accurate gradient in the case of an adapted mesh.

The new step of the study addresses the main difficulty of it, viz. the coupling of shape optimisation and mesh adaptation.

6.1 Description of the toolbox

The basic block of our toolbox is the CFD solver. This routine is called by three other tools, the shape optimiser, the mesh adapter and the deformation kernel. We now present the details of each tool.

6.1.1 CFDSOLVER

CFDSOLVER: Given a mesh τ, an initial array W_{in}, and a shape γ, compute a steady flow $W(\tau, \gamma)$, solution of

$$\Psi(\tau, \gamma, W(\tau, \gamma)) = 0 \ .$$

The shape γ is taken into account by transpiration on the wall condition from the geometry defined by the mesh.

This kernel is the same as described below.

6.1.2 SHAPEOPTIMISER

SHAPEOPTIMISER: Given a mesh τ, a steady flow $W(\tau, 0)$, compute the optimal shape γ_{opt} and the corresponding steady flow $W(\tau, \gamma_{opt})$. This is performed by representing shape perturbation by means of a trfanspiration condition, that is without deforming the mesh.

This kernel is also the same as described below.

6.1.3 MESHADAPTER

MESHADAPTER: Given a mesh τ_{in},a steady flow $W(\tau_{in}, 0)$, compute a better adapted mesh τ_{out}, and a steady solution $W(\tau_{out}, 0)$.

This kernel is developed in INRIA's WP3-Task2.3.1 contribution. It contains a loop between the following steps:
a- calling CFDSOLVER for getting a converged solution on current mesh,
b- computing an adaptation criterion by the continuous metric method,
c- regenerating an adapted mesh with a controlled Voronoi principle,
d- transfering the previous CFD flow to the new mesh by interpolating it,
e- branching to a, if this process is not converged.

This mesh adaptation algorithm is run with a fixed mesh size, i.e. a fixed total number of nodes. The convergence arises when the change between two successive meshes is small enough.

6.1.4 DEFORMATION

DEFORMATION: Given a mesh τ_{in}, a transpired shape displacement γ, and a steady flow $W(\tau_{in}, \gamma)$, compute a mesh τ_{out} by deforming τ_{in} according to the transpired displacement in such a way that the mesh skin of τ_{out} follows the shape, and a steady flow $W(\tau_{out}, 0)$.

This kernel saves the mesh topology but find new coordinates for vertices by moving the boundary nodes parallel to initial mesh normals for a length equal to γ and solving a spring system for interior vertices for recovering a vertex distribution similar to the initial one. We refer to [Farhat et al. 1998] and [Vàzquez et al. 2004] for a detailed description of the deformation algorithm. Then the CFDSOLVER is called in order to equip the new mesh with a converged solution of the flow.

6.2 A global loop applied for adaptative optimisation

A straightforward adaptation/optimisation loop reads as follows:

0. Initial conditions: a mesh τ
1. Apply CFDSOLVER and compute a flow
2. Apply MESHADAPTER to obtain an adapted mesh and the (iteratively converged) flow on it.
3. Apply SHAPEOPTIMISER to obtain a transpired optimal shape and the (iteratively converged) flow on it.
4. Apply DEFORMATION to obtain a mesh following the optimal shape and the (iteratively converged) flow on it.
5. Go to 2.

The purpose of the loop is to get an optimal shape on an adapted mesh. In the present algorithm, flow convergence and adaptation convergence is forced at every time CFDSOLVER and MESHADAPTER are applied. DEFORMATION is also an iteratively converged process each time it is applied. Then stopping the above loop will rely on a test of the residual of the stationary condition $j' = 0$. Note that we are not any more minimizing a unique discrete functional since the discretisation is iteratively changed, but we try to converge towards the satisfaction of the KKT system for a family of discretisations.
This set another question, concerning the adaptation strategy: either we try to apply finer and finer meshes in adaptation, trying to get mesh-convergence to continuous limit, or our loop is restricted to minimise with the best adapted mesh of a given number of vertices. We prefer the second option: the global computing effort is mastered since the number of nodes is maintained moer or less fixed. Then the above loop can be applied three times, with three different number of nodes in order to evaluate mesh convergence.

6.3 Application to problem under study

Preliminary optimisation computations have been applied to the HISAC test cases. The common features of these calculations are:
- angle of attack is 3 degrees,
- six gradient iterations are applied starting from the reference HISAC_2_REF-IT3 geometry,
- mesh sizes are chosen around 30,000 vertices.

Figure 11 to 14 deal with the Mach=1.4 case:
Figure 11 depicts a pressure values over a vertical cut plane ($y = 0$), and over a horizontal cut plane under the aircraft ($z = -R/L$) in which we have constructed the target pressure. The vertical cut plane shows the propagation of the choc under the aircraft.
Figure 12, 13 and 14 depict contours associated to the pressure at several cut planes under the aircraft. In figure 14, the vertical plane has been taken over a wing section to represent the pressure propagation under it.

Figure 15 to 18 deal with the Mach=1.6 case:
The same figures shown in the previous test case (Mach= 1.4) have been re-produced here for a Mach= 1.6.

Figure 19 to 22 deal with the Mach=1.8 case:
As in the previous test cases (Mach=1.4 and Mach=1.6), figure 19 depicts the captured pressure at plane ($z = -R/L$), figure 20 and 21 depict contours associated to the pressure at several cut planes under the aircraft.

Figure 22, shows in blue the baseline shape and in black the optimized shape obtained at the sixth iteration of gradient.

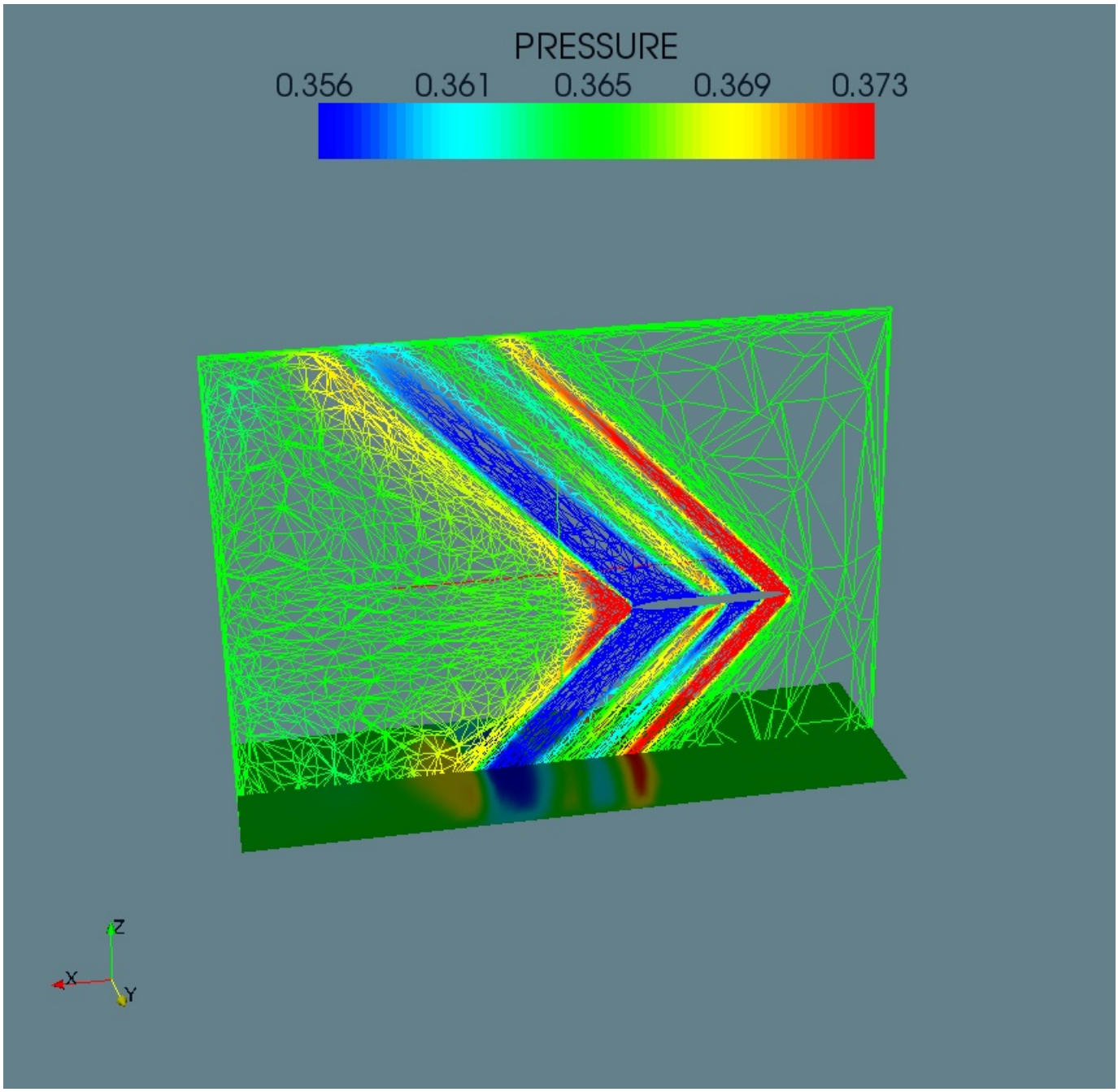

Figure 11: HISAC_2_REF-IT3,M=1.4,5

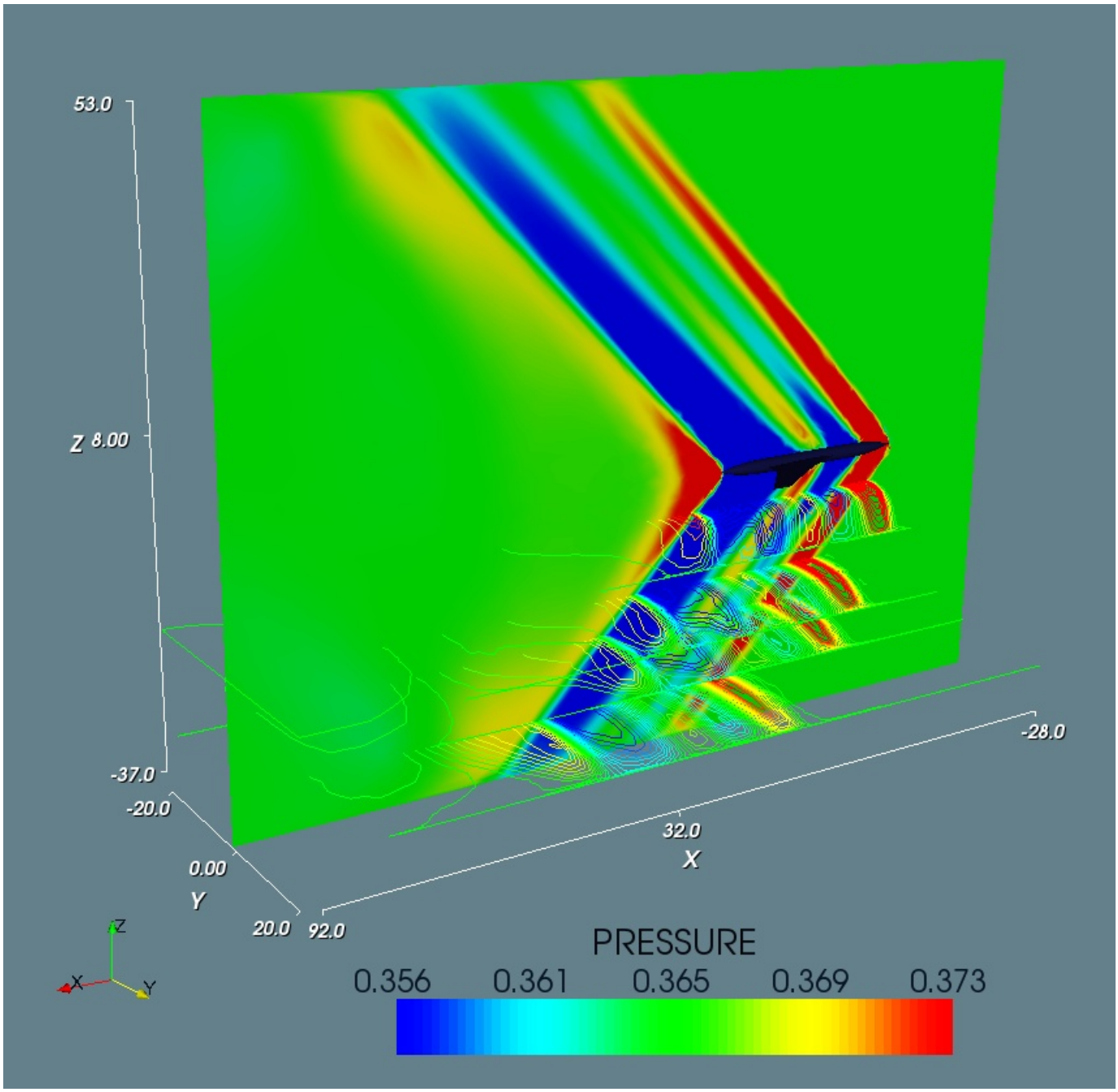

Figure 12: HISAC_2_REF-IT3,M=1.4,6

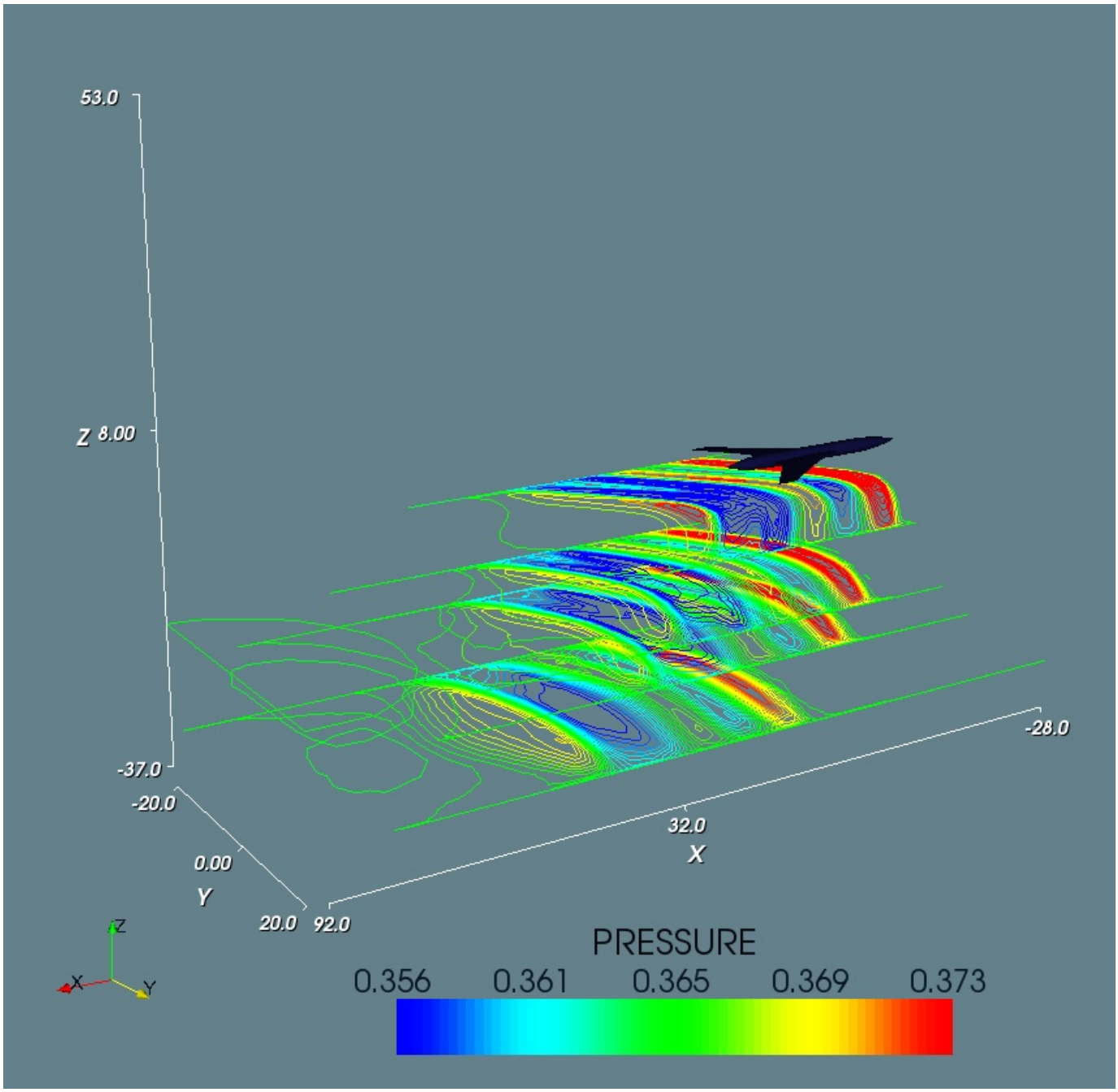

Figure 13: HISAC_2_REF-IT3,M=1.4,7

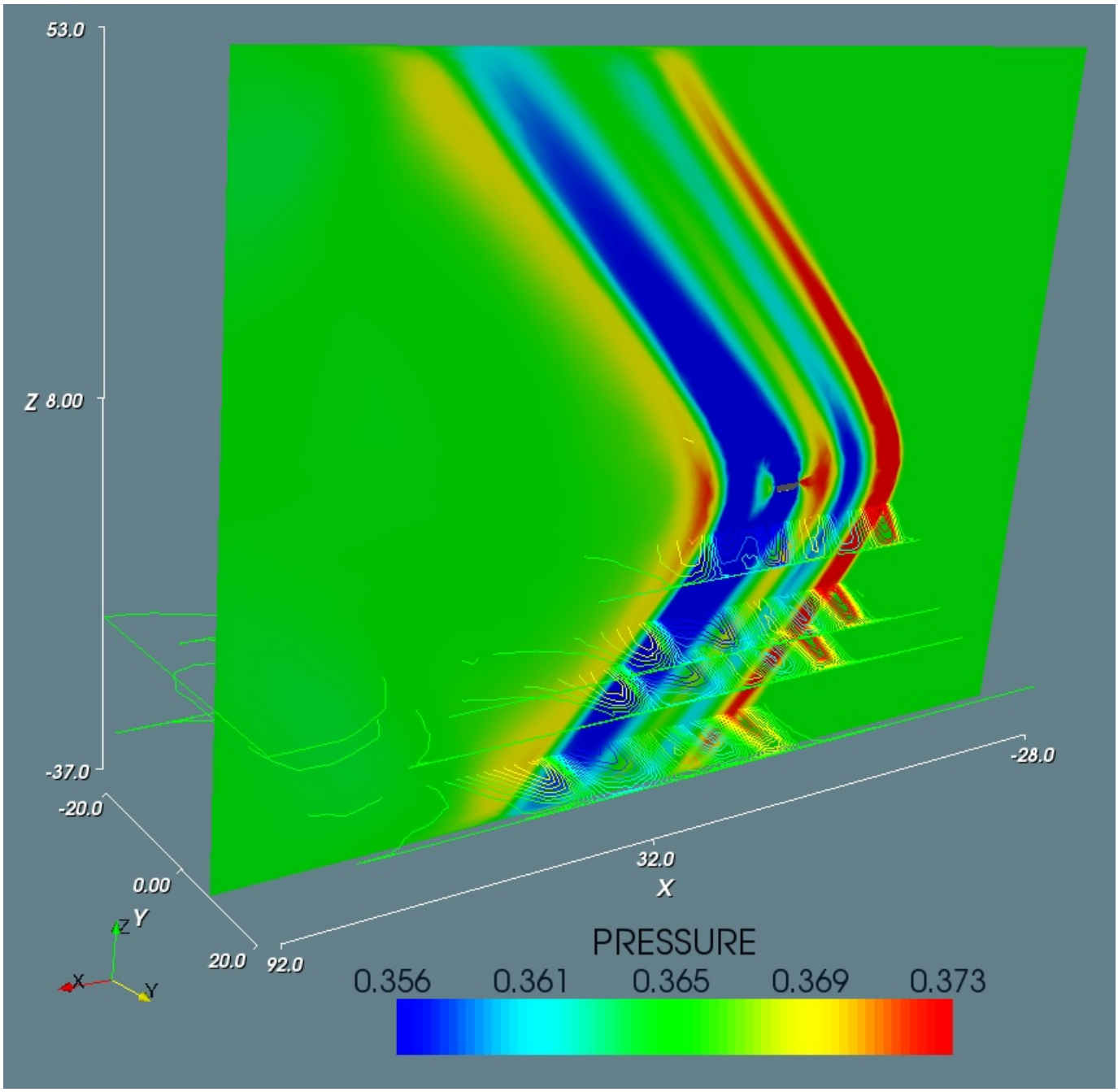

Figure 14: HISAC_2_REF-IT3,M=1.4,8

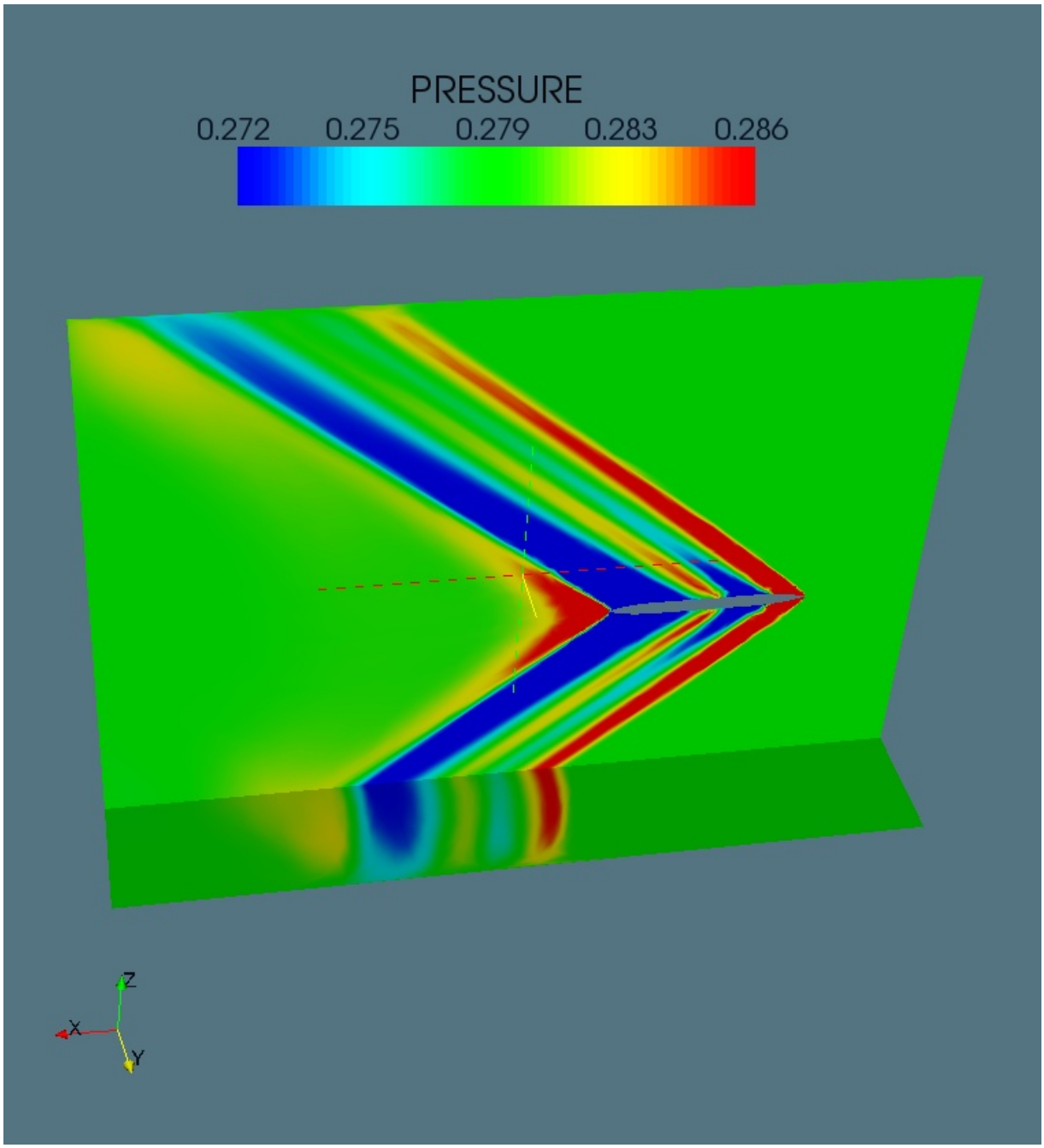

Figure 15: HISAC_2_REF-IT3,M=1.6,1

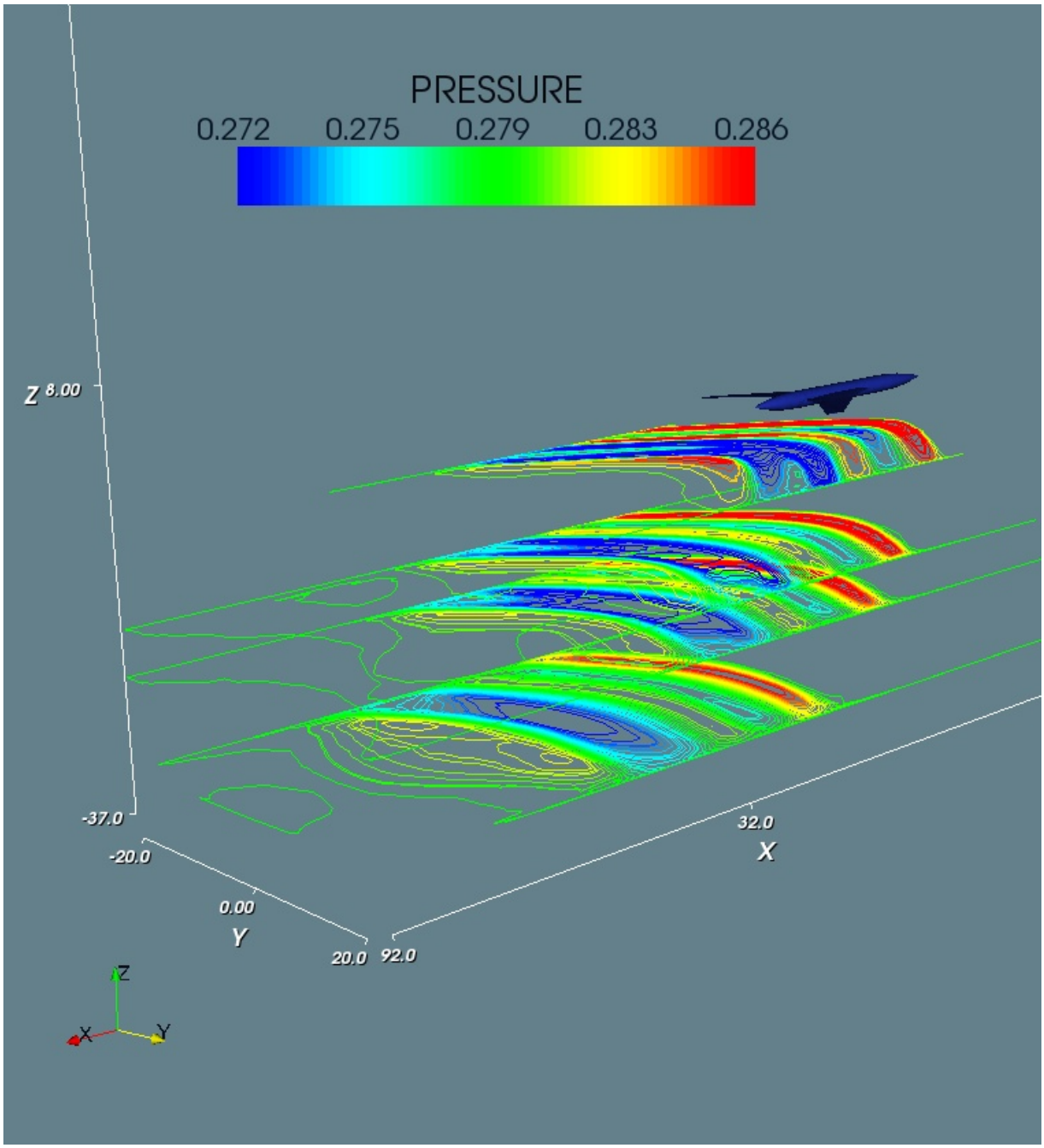

Figure 16: HISAC_2_REF-IT3,M=1.6,2

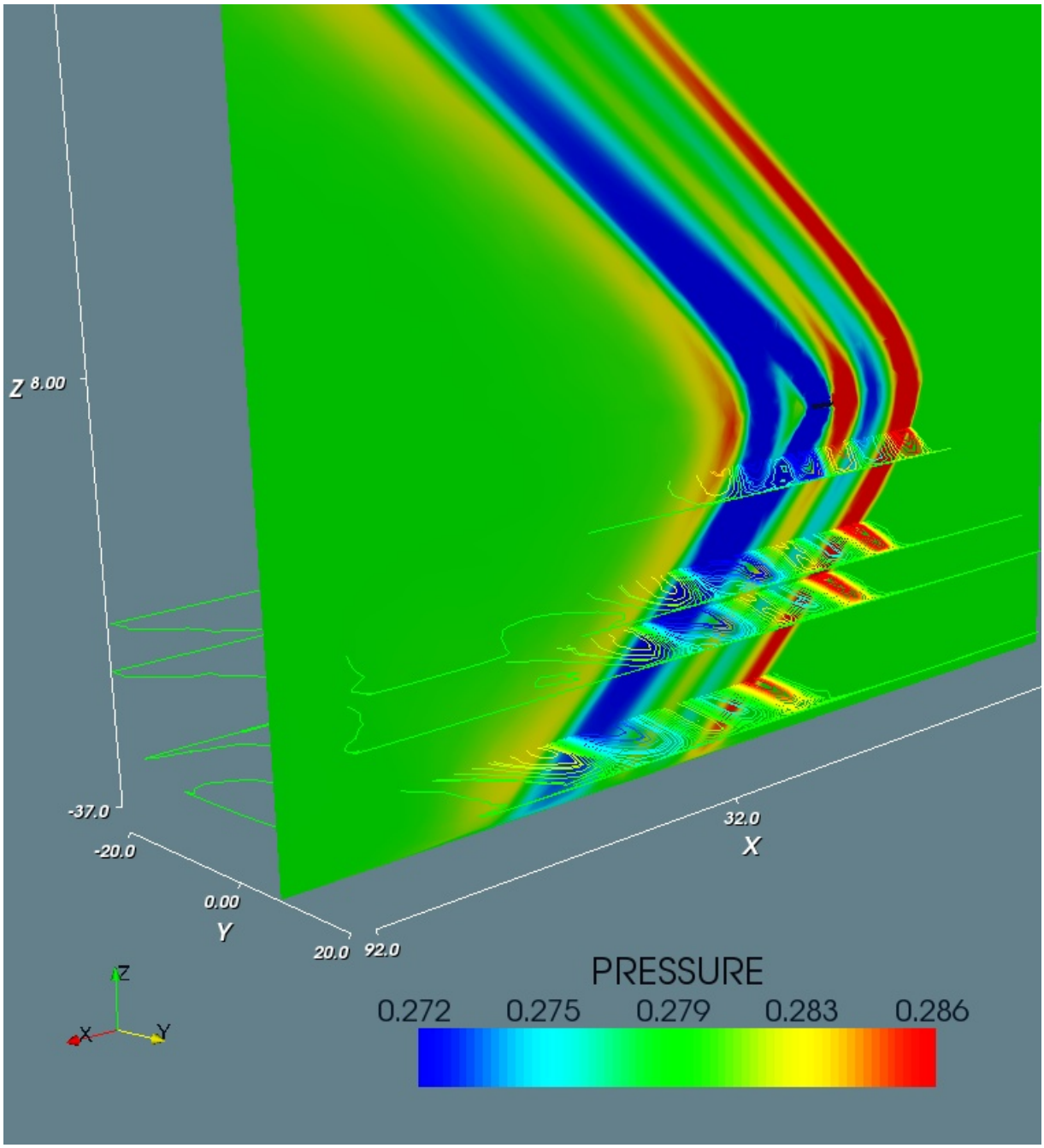

Figure 17: HISAC_2_REF-IT3,M=1.6,3

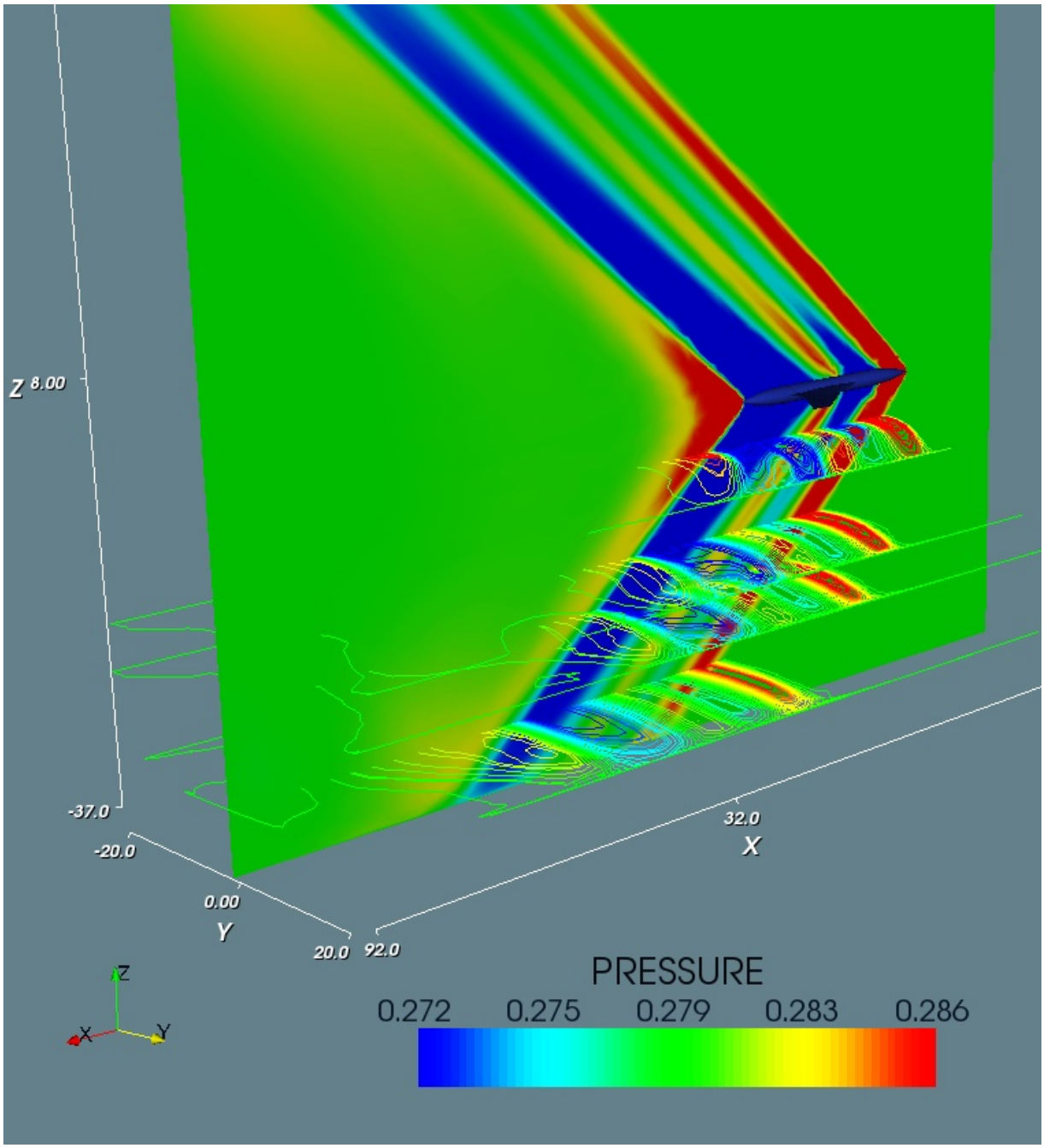

Figure 18: HISAC_2_REF-IT3,M=1.6,4

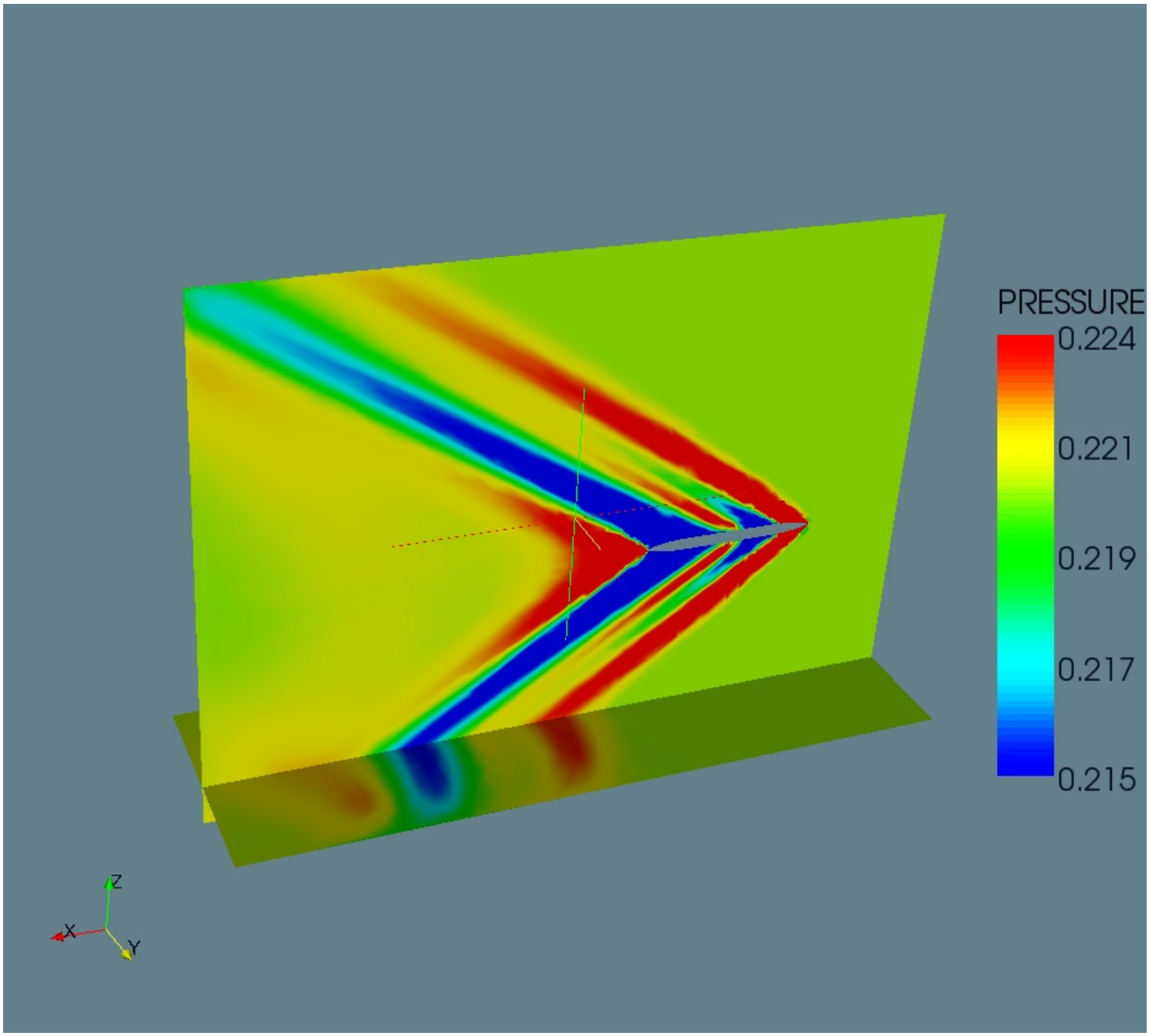

Figure 19: HISAC_2_REF-IT3,M=1.8,11

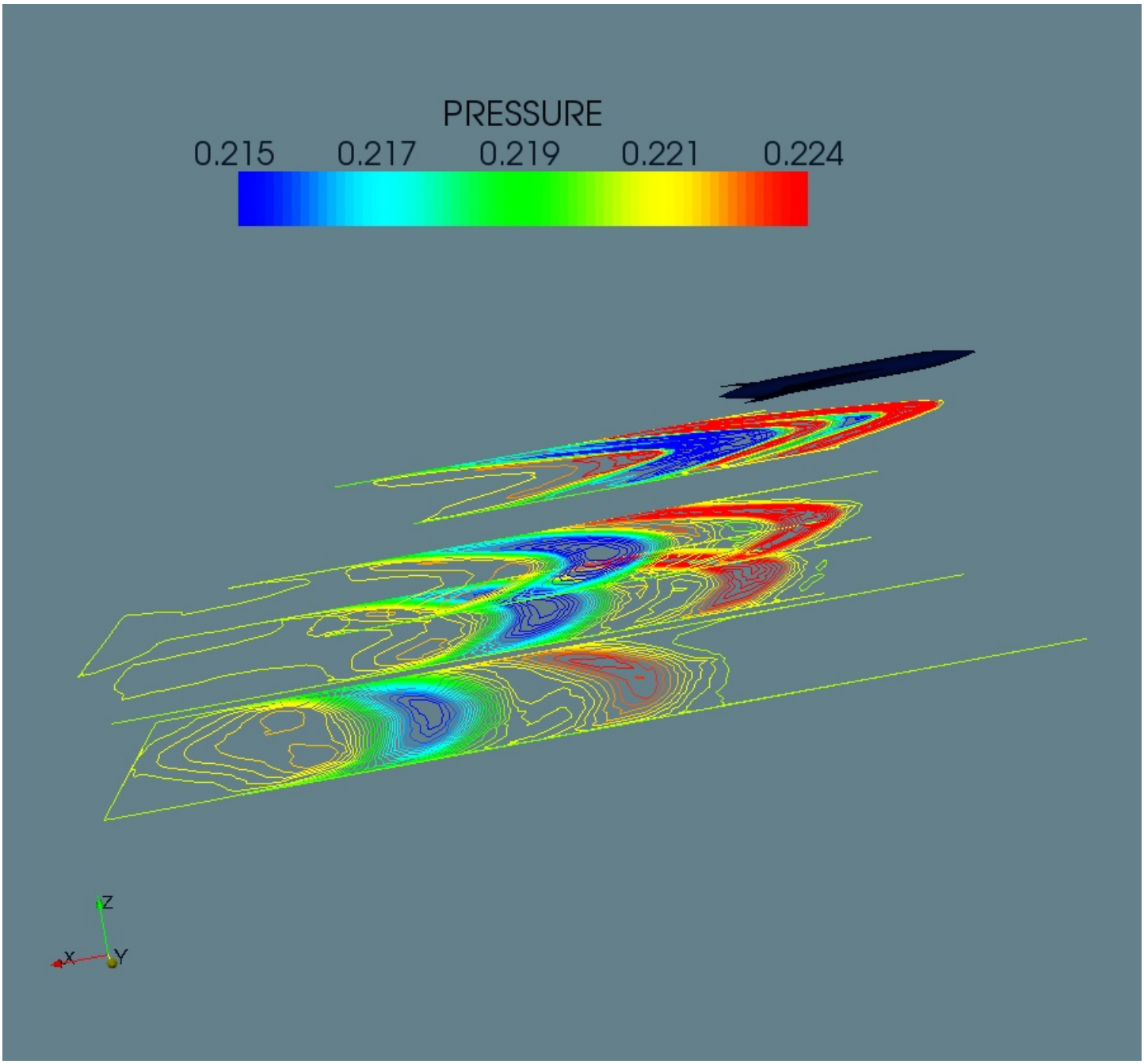

Figure 20: HISAC_2_REF-IT3,M=1.8,9

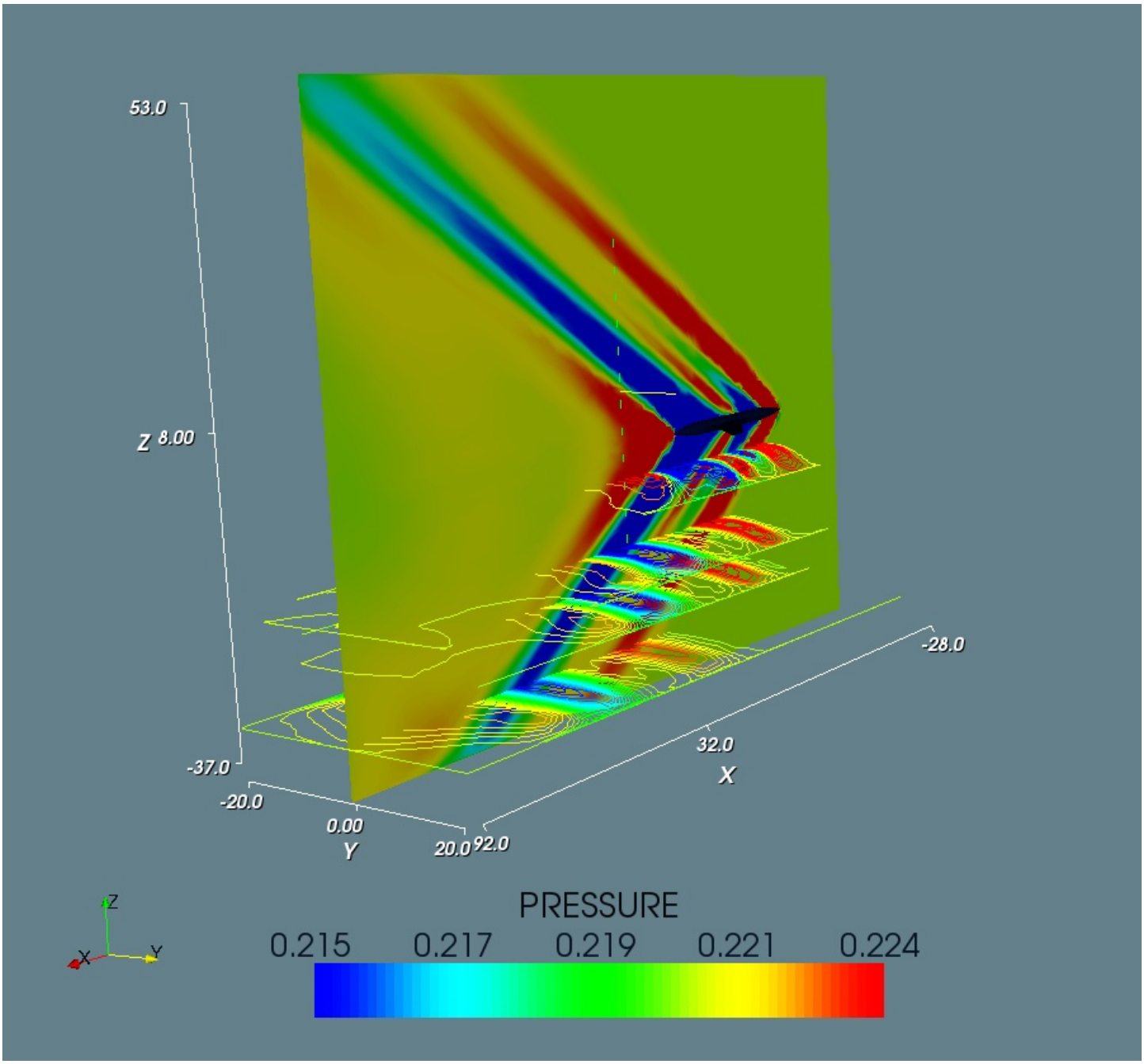

Figure 21: HISAC_2_REF-IT3,M=1.8,10

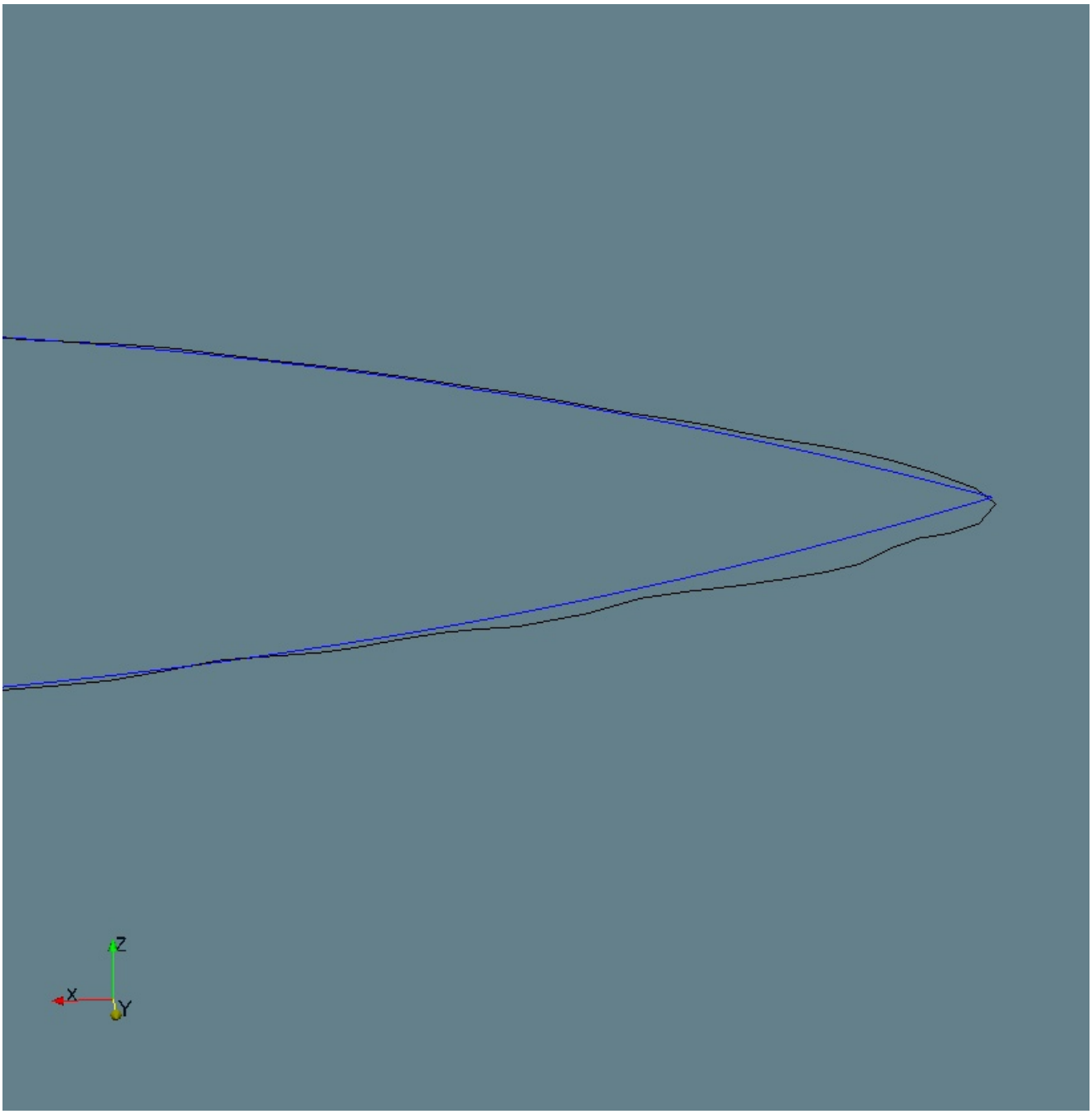

Figure 22: HISAC_2_REF-IT3,M=1.8,12

7 Concluding remarks

A first version of the proposed platform has been implemented. This first version involves the following functionalities:
- CAD-free reprepsentation of shape,
- Euler flow accounting for shape variations by transpiration conditions,
- gradient loop, using adjoint-based shape sensitivity developed with an AD tool.

The optimisation loop has been adapted and validated for the combination with stretched adapted meshes.
Preliminary experiments give indication that mesh adaptation is compulsory for a good flow evaluation.
However mesh adaptation makes the optimisation problem a little stiffer.
Further experimentations of this first version are being done.

We are now developing the mesh optimisation loop, consisting of an implicit approximation error model, playing the role of a state equation, an adjoint based sensitivity of error to mesh, and an optimisation loop analog to the present shape optimisation one.

References

[Farhat et al. 2002] C. Farhat, K. Maute, B. Argrow, and M. Nikbay , "A shape optimization methodology for reducing the sonic boom initial pressure rise", *AIAA paper* 2002-0145, *AIAA Journal of Aircraft, (in press)*

[Farhat et al. 2002b] C. Farhat, B. Argrow, M. Nikbay and K. Maute, "A Shape Optimization Methodology with F-function load balancing for Mitigating the Sonic Boom," *AIAA Paper 2002-5551, 9th AIAA/ISSMO Symposium on Multidisciplinary and Optimization*, Atlanta, Georgia, September 4-6 (2002)

[Farhat et al. 2002c] C. Farhat, K. Maute, B. Argrow and M. Nikbay, "A Shape Optimization Methodology for Reducing the Sonic Boom Initial Pressure Rise," *AIAA Paper 2002-0145, 40th Aerospace Sciences Meeting and Exhibit,* Reno, Nevada, January 14-17 (2002)

[Farhat et al. 2004] C. Farhat, B. Argrow, M. Nikbay and K. Maute, "Shape Optimization with F-Function Balancing for Reducing the Sonic Boom Initial Shock Pressure Rise", *The International Journal of Aeroacoustics*, Vol. 3, pp. 361-377 (2004)

[Nadarajah et al. 2004] S.K. Nadarajah, A, Jameson, J. Alonso, "An adjoint method for the calculation of remote sensitivities in supersonic flows", *AIAA paper* 2002-0261

[Nadarajah et al. 2005] S.K. Nadarajah, A, Jameson, J. Alonso, "Adjoint-based sonic boom reduction for wing-body configurations in supersonic flows", *Canadian aAeronautics and Space Journal*, Vol. 51, 4, 187-199 (2005)

[Choi et al. 2004] S. Choi, J.J. Alonso, E. Van der Weide, "Numerical and mesh resolution requirements for accurate sonic boom prediction of complete aircraft configurations", *AIAA paper* 2004-1060

[Choi et al. 2004bis] S. Choi, J.J. Alonso, S. Kim, I. Kroo, M. Wintzer "Multi-fidelity design optimization of low-boom supersonic business jets", *AIAA paper* 2004-4371

[Choi et al. 2004ter] S. Choi, J.J. Alonso, H.S. Chung, "Design of a low-boom supersonic business jet using evolutionary algorithms and an adaptive unstructured mesh method", *AIAA paper* 2004-1758

[Choi et al. 2005] S. Choi, J.J. Alonso, S. Kim, I. Kroo, M. Wintzer, " Two-level multi-fidelity design optimization studies for supersonic jets", *AIAA paper* 2005-0531

[Alauzet, 2003] F. Alauzet, "Adaptation de maillage anisotrope en trois dimensions. Application aux simulations instationnaires en Mécanique des Fluides", *Thèse de Doctorat de l'Université Montpellier II*, 2003.

[Farhat et al. 1998] C. Farhat, C. Degand, B. Koobus, M. Lesoinne, "Torsional springs for two-dimensional dynamic unstructured meshes" *Comput. Meths. Appl. Mech. Engrg.*,**163**,231-45 (1998)

[Frey and Alauzet, 2005] P.J. Frey and F. Alauzet, "Anisotropic mesh adaptation for CFD computations" *Comput. Methods Appl. Mech. Engrg.*, **194**, 5068-5082 (2005).

[Thomas, 1972] Ch. Thomas, "Extrapolation of sonic boom pressure signatures by the waveform parameter method".

[Stoufflet et al. 1987] B. Stoufflet, J. Periaux, F. Fezoui, A. Dervieux, "3-D Hypersonic Euler Numerical Simulation around Space Vehicles using Adapted Finite Elements", *25th AIAA Aerospace Meeting*, Reno (1987), AIAA Paper 86-0560

[Vàzquez et al. 2004] M. Vàzquez and B. Koobus and A. Dervieux, "Multilevel optimisation of a supersonic aircraft", *Finite Element in Analysis and Design*, **40**, 2101-2124 (2004)

[Vàzquez et al. 2004] M. Vàzquez and A. Dervieux and B. Koobus, "A methodology for the shape optimization of flexible wings" *Engineering Computations*, **23**:4, 344-367 (2006)

[Tapenade,2003] L. Hascoet and V. Pascual "Tapenade user Manual", INRIA Technical Report RT-0300 URL:http://www-sop.inria.fr/rapports/sophia/RT-0300.html

[Courty et al. 2003] F. Courty, A. Dervieux, B. Koobus, and L. Hascoet, Reverse automatic differentiation for optimum design: from adjoint state assembly to gradient computation. *Optimization Methods and Software*, 18(5):615–627, 2003.

[Courty et al. 2005a] F. Courty and A. Dervieux, Multilevel functional Preconditioning for shape optimisation. submitted to *Int. Journal CFD*, 2005.

[Courty et al. 2005b] F. Courty and A. Dervieux, *A SQP-like one-shot algorithm for optimal shape design*. Springer, 2005. to appear.

[Dadone et al. 2000] A. Dadone and B. Grossman, Progressive optimization of inverse fluid dynamic design problems. *Computer and Fluids*, 29:1–32, 2000.

[Dervieux et al. 2004] A. Dervieux, F. Courty, T. Roy, M. Vázquez, and B. Koobus, Optimization loops for shape and error control. In *PROMUVAL Short Course on Multidisciplinary Modelling, Simulation and Validation in Aeronautics, Barcelona, june 28-29, 2004*. CIMNE, 2004. extended version INRIA Research Report 5413.

[Hascoet et al. 2003] L. Hascoet, M. Vázquez, and A. Dervieux, Automatic differentiation for optimum design, applied to sonic boom reduction. In V.Kumar et al., editor, *Proceedings of the International Conference on Computational Science and its Applications, ICCSA'03, Montreal, Canada*, pages 85–94. LNCS 2668, Springer, 2003.

[Mohammadi 1997] B. Mohammadi, Practical application to fluid flows of automatic differentiation for design problems. *Von Karman Lecture Series*, 1997.

[Nocedal et al. 1999] J. Nocedal and S.-J. Wright, *Numerical Optimization*. Springer, Series in Operations Research, 1999.

[Taasan et al. 1992] S. Ta'asan, G. Kuruvila, and M.D. Salas, Aerodynamic design and optimization in one shot. In *30th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, AIAA Paper 91-0025*, 1992.