Mesh Adaptation, Fluid Dynamics and Aeronautics

Inria Junior Seminar

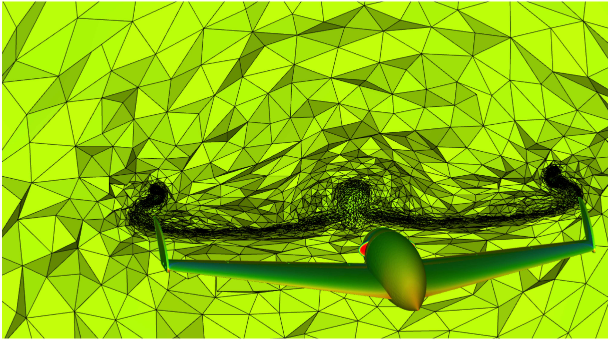
by

Victorien Menier (Gamma3 team)

November 18, 2014

My PhD

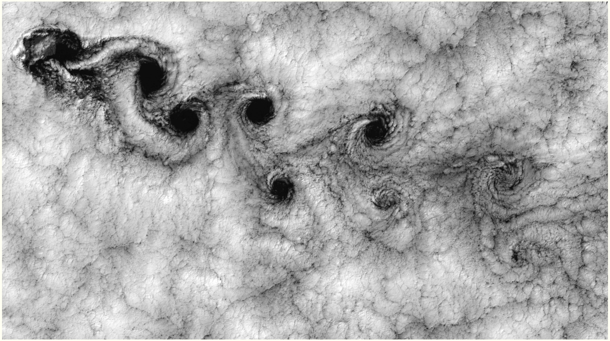
- Subject: **Adaptive Methods** for the Numerical Prediction of Viscous Phenomena and their Interactions. Application to Aeronautics.



Mesh adaptation: automatically taylored meshes.

My PhD

- Subject: Adaptive Methods for the Numerical Prediction of **Viscous Phenomena** and their Interactions. Application to Aeronautics.



Navier-Stokes equations, turbulence modeling...

My PhD

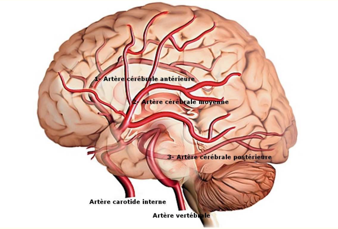
- Subject: Adaptive Methods for the Numerical Prediction of Viscous Phenomena and their Interactions. **Application to Aeronautics.**

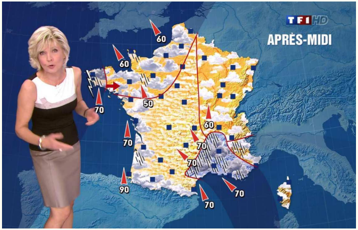
Planes, wings, space shuttles...

Contents

Scientific Context

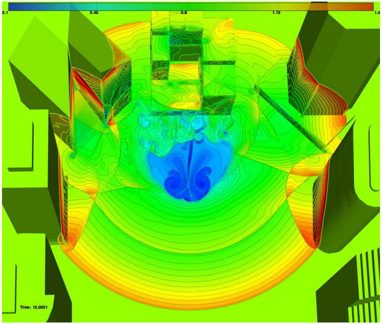
Numerical Simulations Everywhere

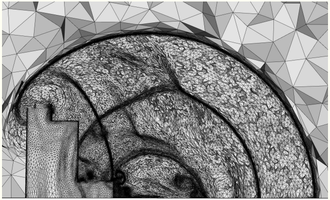


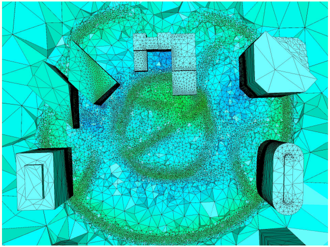


Numerical Simulations Everywhere

- **Blast in a city**

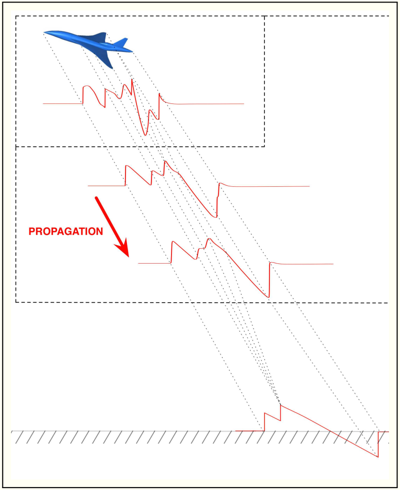


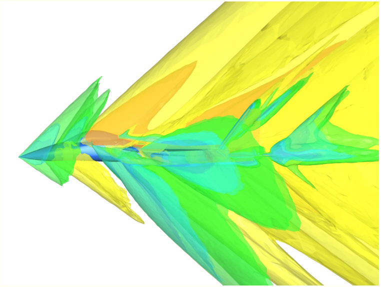




Numerical Simulations Everywhere

Sonic Boom Prediction

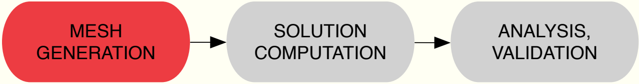




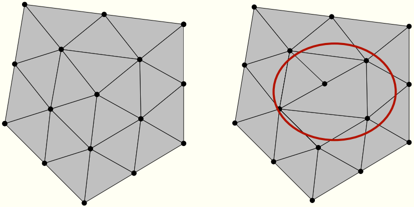
Numerical Simulation Pipeline

- Gamma3 team's main research axes: Mesh generation and Computational Fluid Dynamics.

What is a mesh?

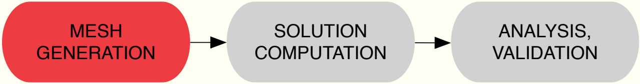


- Solving a Partial Differential Equation (P.D.E.) requires the **discretization of the continuous problem.**
  - The computational domain Ω is replaced by a union of elements such as quadrilaterals, triangles, tetrahedra, etc.

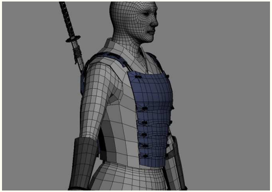


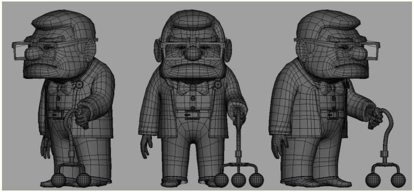
Conformal **Not conformal**

What a mesh is not



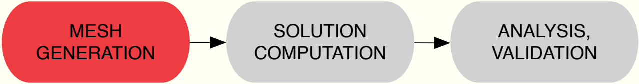
- **Computational meshes $\neq$ visu meshes**



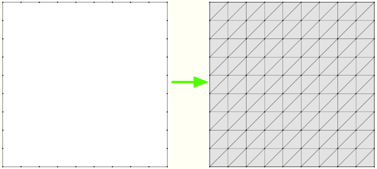


- Visu meshes are not made for numerical simulations : they are **not conformal** and with no consideration for **mesh quality**

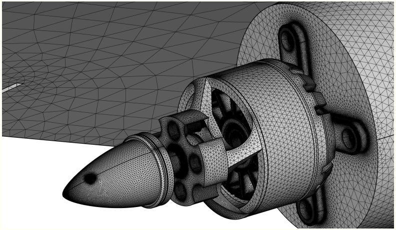
Mesh Generation is hard



- **Problem**: Given a surface mesh, ”fill” the volume with vertices and tetrahedra
  - 3D complex geometries are challenging

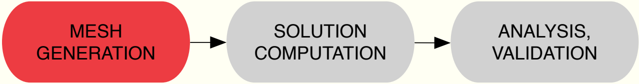


EASY!

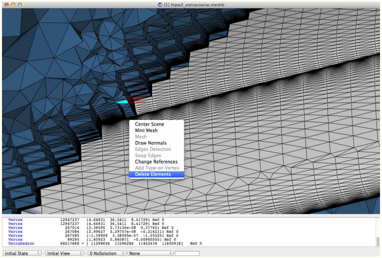


EASY?

Mesh generation is hard

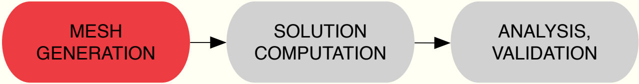


- Method 1 : **Semi automatic**
  - The algorithm **fails** generating a conformal mesh
  - The mesh obtained is **manually corrected** (add/delete mesh elements)



$\Rightarrow$ Months of ”clicking” by an engineer.

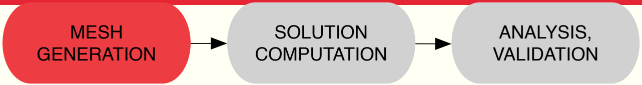
Mesh generation is hard



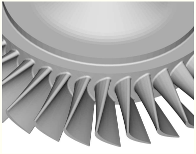
- Method 1 : **Semi automatic**
  - The algorithm **fails** generating a conformal mesh
  - The mesh obtained is **manually corrected** (add/delete mesh elements)

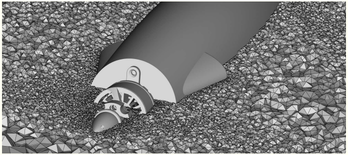
$\Rightarrow$ Months of ”clicking” by an engineer.

Fully automatic mesh generation



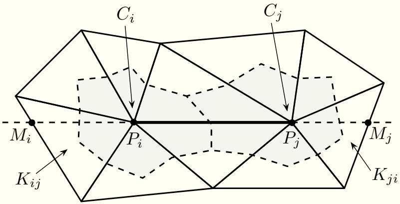
- Method 2 : **Fully automatic** (Gamma3 team)
  - An algorithm robust enough to generate a valid volume mesh from any surface mesh
    - 20 years of research (mathematics, informatics)
- Example at Inria : **GHS3D** is used by Dassault Systèmes (CATIA), Siemens, ANSYS, Autodesk, EDF, Safran, Alcan, etc.





Solution Computation

- Flow solver: Modeling equations solved using the finite element or finite volume method.
  - I contribute to our in-house flow solver `Wolf`
  - $\frac{\partial W}{\partial t} + \nabla \cdot \mathcal{F}(W) = \mathcal{S}(W)$



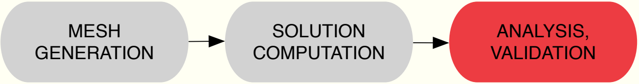
Finite volume cells constructed on unstructured meshes:

$$\Omega_h = \bigcup_{i=1}^{N_T} K_i = \bigcup_{i=1}^{N_S} C_i$$

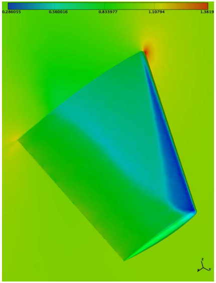
Solution Computation

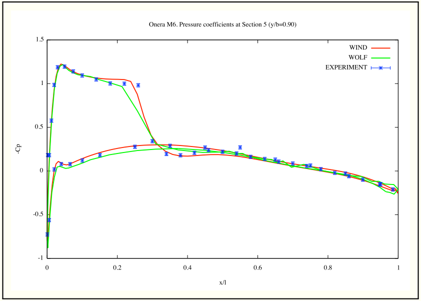
- The solution is iteratively converged.
  - Example of evolution of a solution during a computation:

Analysis, Validation



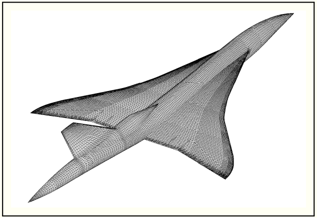
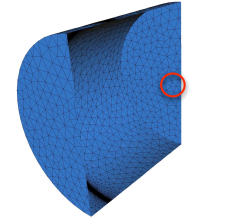
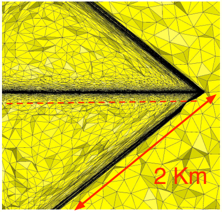
- The solutions obtained are compared with experimental data and other codes.





Mesh Adaptation, why?

- **Anisotropic** physical phenomena located in **small areas** of the computational domain

- Aircraft: 36m. Domain: 2km.
- Required mesh size around the aircraft: 1mm
- Adapted mesh → **0.1 Billion DoF**
- Uniform mesh with 1m edges → **200 Billion DoF**

Generation of Adapted Meshes

- Main idea : change mesh generator's **distance and volume computation**.

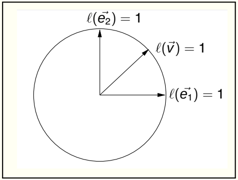
Fundamental concept: Unit mesh

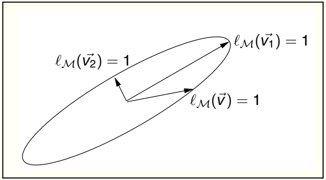
Adapting a mesh

$\Updownarrow$ Work in an adequate Riemannian metric space

Generating a uniform mesh w.r. to $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$

$\mathcal{H}$ unit mesh $\iff \forall \mathbf{e},\ \ell_{\mathcal{M}}(\mathbf{e}) \approx 1$ and $\forall K,\ |K|_{\mathcal{M}} \approx \begin{cases} \sqrt{3}/4 & \text{in 2D} \\ \sqrt{2}/12 & \text{in 3D} \end{cases}$

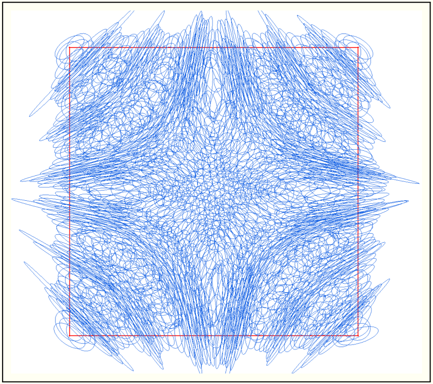




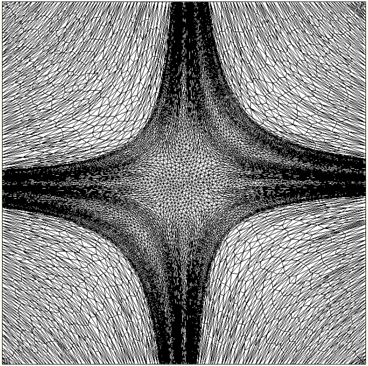
Generation of an adapted mesh

- Example of adapted mesh

Input : Metric Field



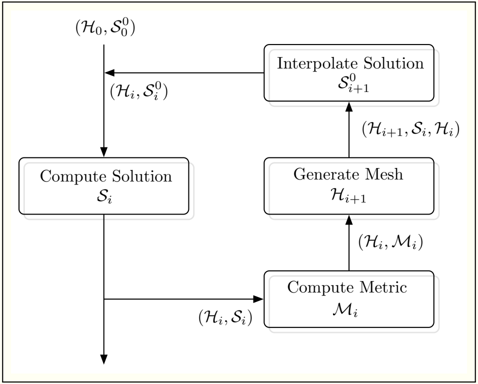
Output: Adapted mesh



$\mathcal{H}$ unit mesh $\iff \forall \mathbf{e},\ \ell_{\mathcal{M}}(\mathbf{e}) \approx 1$

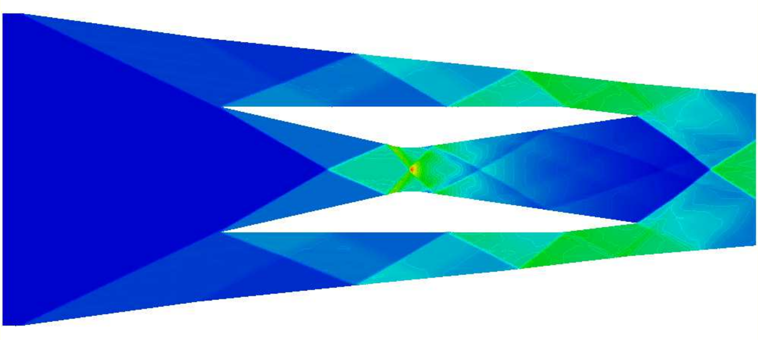
Mesh adaptation loop

- Mesh adaptation is a **non linear problem**
  - An iterative process is required to converge the couple mesh-solution



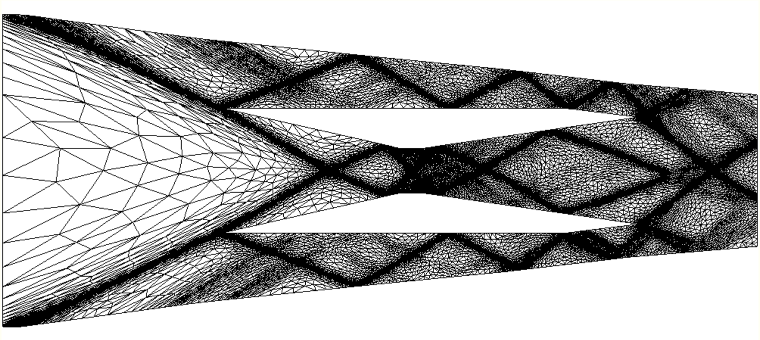
Example of an adapted mesh

- Mesh adaptation of anisotropic **supersonic shocks**
  - Mapping of the fluid’s density (strong discontinuities around shocks)



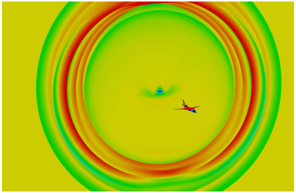
Example of an adapted mesh

- Mesh adaptation of anisotropic **supersonic shocks**
  - Corresponding anisotropic adapted mesh

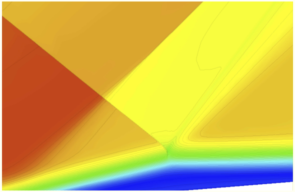


Turbulent Viscous Flow Simulations

Inviscid vs. Viscous Simulations



- **Inviscid simulations** do not take into account the viscosity of the fluid.
  - Modelized by the Euler equations



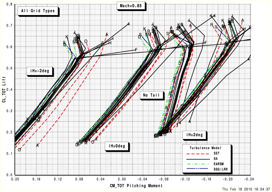
- **Viscous simulations** do. And we couple them with turbulence modeling.
  - Modelized by the Navier-Stokes equations

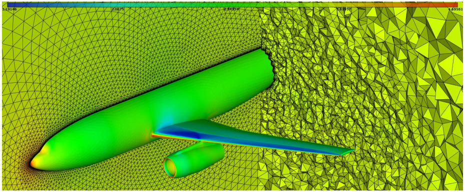
- My work consists in improving numerical methods and meshing strategies for turbulent viscous flow simulations.

Viscous Simulations, why?

- Two examples of targeted applications which require a viscous simulation and turbulence modeling:
  - **Example 1 :** Drag Prediction Workshop

- **Drag :** the aerodynamic force that opposes an aircraft's motion through the air
  - A fully turbulent simulation is mandatory

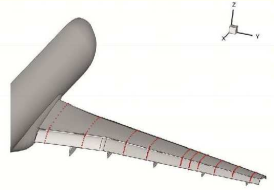


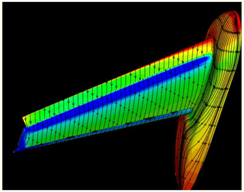


Viscous Simulations, why?

- Two examples of targeted applications which require a viscous simulation and turbulence modeling:
  - **Example 2 :** High Lift Prediction Workshop

- **Lift :** the force that directly opposes the weight of an airplane and holds it in the air
  - A fully turbulent simulation is mandatory

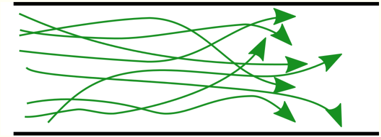




Turbulence modeling

"When I meet God, I am going to ask him two questions: Why relativity? And why turbulence? I really believe he will have an answer for the first."
- Heisenberg

Laminar Flow



Turbulent Flow

- Turbulence modeling using the **Spalart-Allmaras** one equation model:

$$\frac{\partial \rho \tilde{\nu}}{\partial t} + \underbrace{\mathbf{u} \cdot \nabla \rho \tilde{\nu}}_{\text{convection}} = \underbrace{c_{b1} \tilde{S} \rho \tilde{\nu}}_{\text{production}} - \underbrace{c_{w1} f_w \rho \left(\frac{\tilde{\nu}}{d} \right)^2}_{\text{destruction}} + \underbrace{\frac{\rho}{\sigma} \nabla \cdot ((\nu + \tilde{\nu}) \nabla \tilde{\nu})}_{\text{dissipation}} + \underbrace{\frac{c_{b2} \rho}{\sigma} \|\nabla \tilde{\nu}\|^2}_{\text{diffusion}}$$

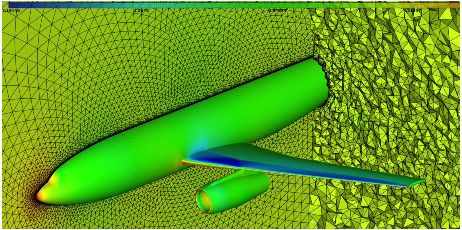
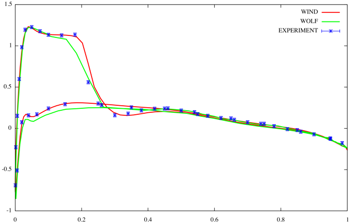
Solving Navier-Stokes

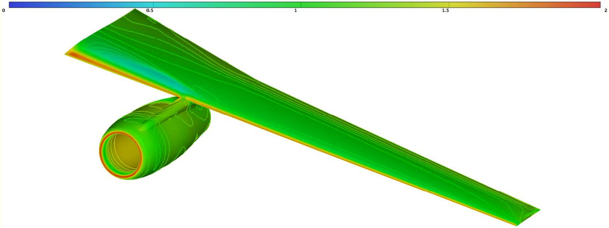
$$
\begin{cases}
\dfrac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) & = & 0\,, \\
\dfrac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) + \nabla p & = & \nabla \cdot (\mu \mathcal{T})\,, \\
\dfrac{\partial (\rho e)}{\partial t} + \nabla \cdot ((\rho e + p) \mathbf{u}) & = & \nabla \cdot (\mu \mathcal{T} \mathbf{u}) + \nabla \cdot (\lambda \nabla T)\,,
\end{cases}
$$

- A few features of our flow solver:
  - Spatial Discretization based on a vertex-centered finite element / finite volume formulation
  - Finite volume cells constructed on unstructured meshes: $\Omega_h = \bigcup_{i=1}^{N_T} K_i = \bigcup_{i=1}^{N_S} C_i$.
  - Flux computation : HLLC approximate Riemann solver
  - **Time integration: Matrix-free implicit LU-SGS.**
  - CFL law: Linear, geometric, or bounding of primitive variables

Example of simulation using Wolf

- Result: **Drag Prediction Workshop**

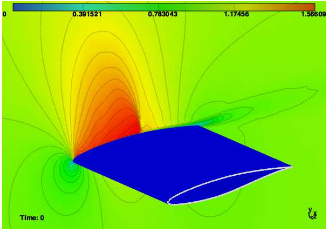



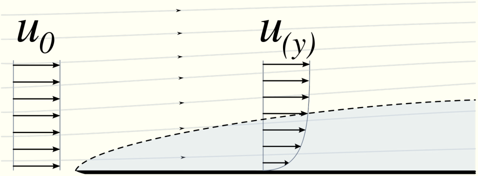


Meshing Strategies for Viscous Simulations

Specific meshes, why?

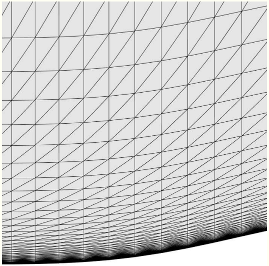
- In near-wall regions - **boundary layers** - , the dramatic variation of the velocity in the normal direction requires the use of quasi-structured meshes.

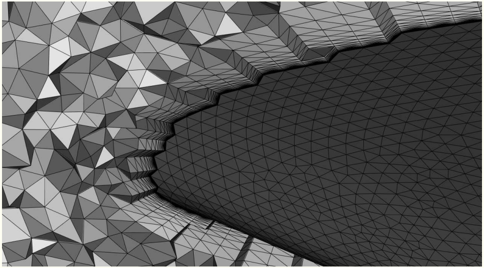




Boundary Layer Meshes

- Examples of quasi-structured **boundary layer meshes**

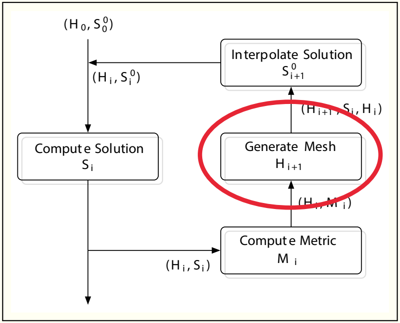




- Their generation is challenging for **complex geometries**

Coupling with mesh adaptation?

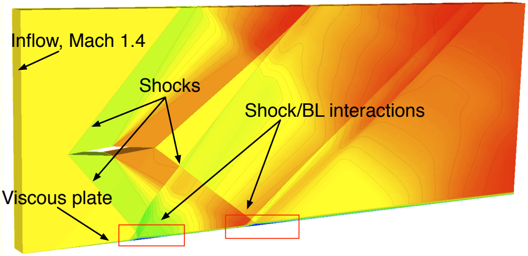
- Only little work exists on **coupling boundary layer meshes with adaptivity**



- We proposed 3 approaches:
  1. Fully unstructured
  2. Mixed approach
  3. Metric-aligned approach

Viscous mesh adaptation (1/3)

- 1st Approach : Fully-unstructured
  - Test case : Shock/Boundary Layer interaction

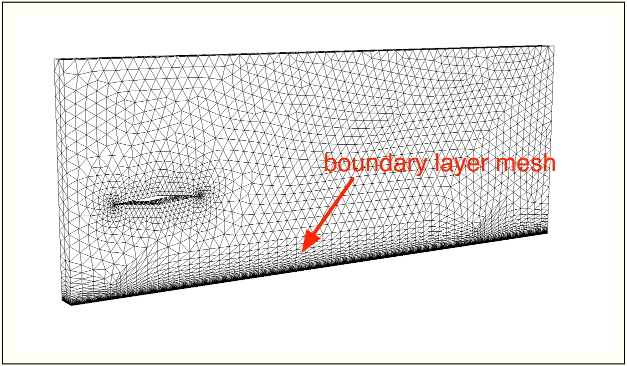


- A supersonic flow (Mach 1.4, *Re* $= 2.710^7$) is applied around a diamond

- Objective : Performing a mesh adaptation in order to accurately capture the shock/boundary layer interactions

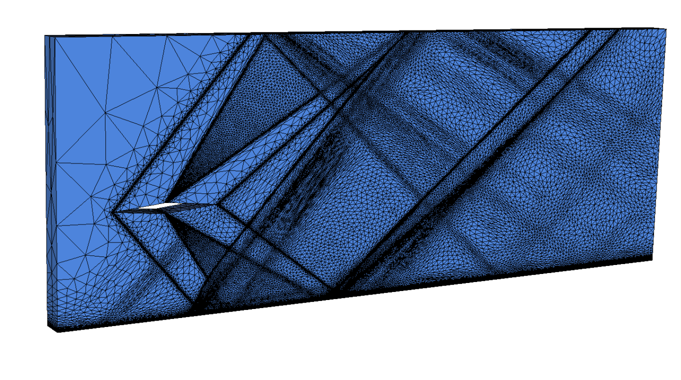
Viscous mesh adaptation (1/3)

- 1st approach : Fully-unstructured
  1. Initial mesh with boundary layer : compute its geometric metric $\mathcal{M}_{bl}$
  2. Intersect $\mathcal{M}_{bl}$ with the computational metric
  3. Use the usual unstructured mesh operators



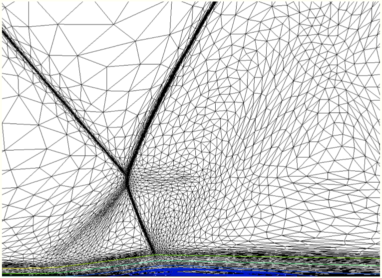
Viscous mesh adaptation (1/3)

- 1st approach : Fully-unstructured
  1. Initial mesh with boundary layer : compute its geometric metric $\mathcal{M}_{bl}$
  2. Intersect $\mathcal{M}_{bl}$ with the computational metric
  3. Use the usual unstructured mesh operators



Viscous mesh adaptation (1/3)

- 1st approach : Fully-unstructured
  1. Initial mesh with boundary layer : compute its geometric metric $\mathcal{M}_{bl}$
  2. Intersect $\mathcal{M}_{bl}$ with the computational metric
  3. Use the usual unstructured mesh operators

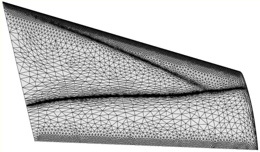


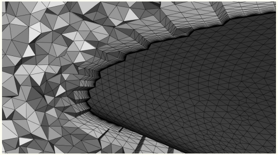
Close-up view of the interaction

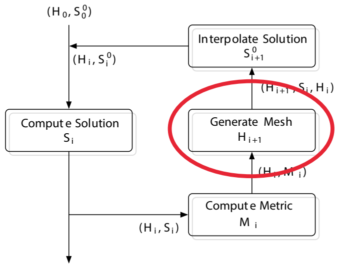
- **Successfully** captured the interactions
  - 280 000 Vertices and 1.3 M tets.
  - Total cpu 1h (on this laptop)
- **But:** Too many vertices inserted and lacks robustness

Mixed approach (2/3)

- The quasi-structured boundary layer mesh is regenerated at each step in the mesh adaptation loop

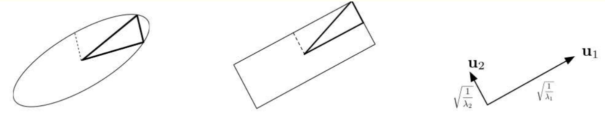




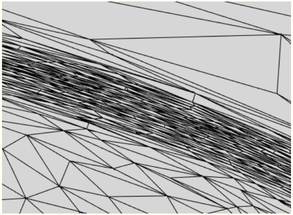


- **Acceptable** number of inserted vertices
- **But** lacks robustness
  - It is difficult to build a boundary layer mesh from an anisotropic surface

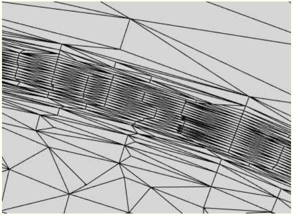
Metric-aligned approach (3/3)



1. Use **metric's eigenvectors** to better control the mesh elements' alignment
2. Modification of the **local remeshing algorithm**
   - $\rightarrow$ Favor the creation of quasi-structured elements



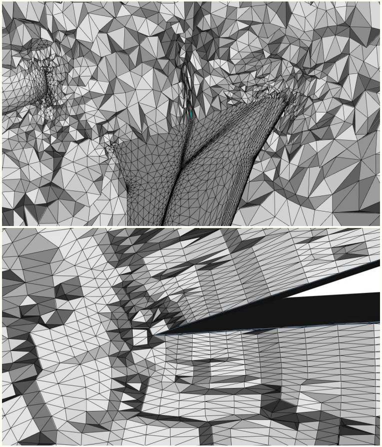
Classical operator



Metric-aligned operator

Metric-aligned approach (3/3)

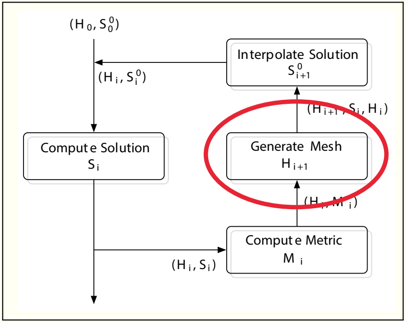
- **Example:** anisotropic metric-aligned mesh adaptation of a wing



- **Pros**
  - Only one mesh operator for the BL and the rest of the domain
  - Robustness
- **Ongoing**
  - Surface metric-aligned operator
  - Validation using experimental data

Parallel Mesh Adaptation

- Problem : Turbulent flow simulations require **heavy meshes**. Their generation is too **CPU consuming** in the mesh adaptation loop.

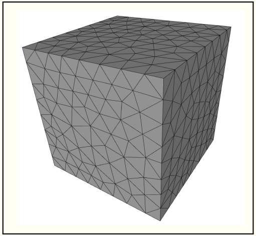


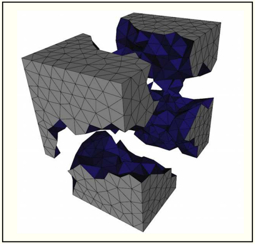
- Parallel Mesh Adaptation:
  - Small scale architectures ($\sim$ 100 nodes)
  - Mesh migration: MPI
  - Target : 150M vertices in one hour

- Didactic example : refinement of a cube.

Parallel Mesh Adaptation : Outline

- The initial mesh is split in N partitions. Each partition is given to a processing core (or node).

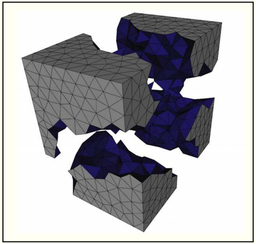


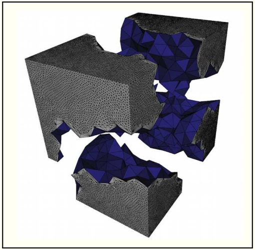


In blue: **interfaces** between partitions.

Parallel Mesh Adaptation : Outline

- Each processing core performs a mesh adaptation on its partition. Without modifying the interfaces (in blue).

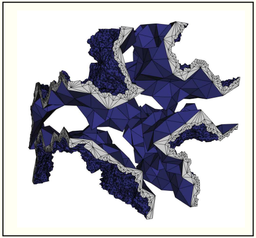


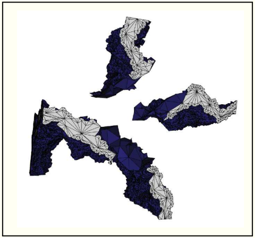


Problem : mesh elements close to an interface are not correctly adapted.

Parallel Mesh Adaptation : Outline

- Interface elements remain to be adapted.
  - To do so, they are migrated using MPI in order to be put inside the volume.

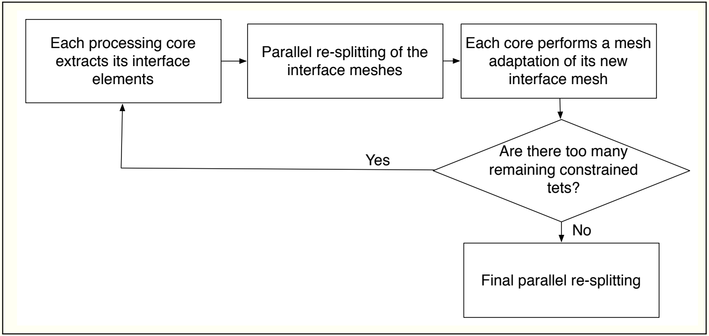




NB : once not in an interface anymore, the elements can be correctly adapted.

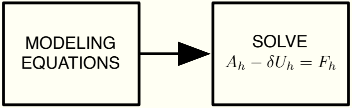
Parallel Mesh Adaptation : Outline

- Some elements still remain constrained
  - They are extracted, re-splitted in parallel once again
  - This process is iterated until the number of constrained elements is small enough

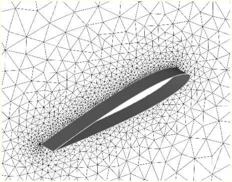


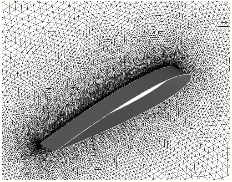
Adaptive Multigrid Methods (Ongoing)

Multigrid Methods

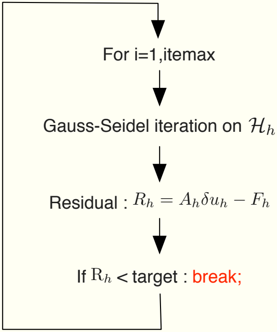


- Multigrid methods : use **coarser meshes to smooth** to obtain a **faster convergence** and an **increased robustness**
  - $A_h \delta u_h = F_h$ is solved on the finest mesh $\mathcal{H}_h$ using an iterative method ($\sim$ Gauss-Seidel)

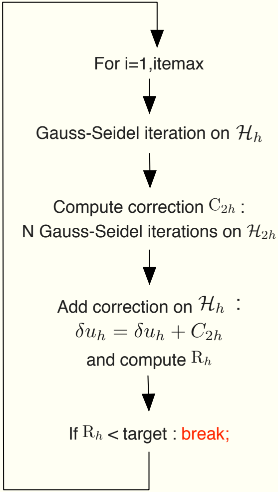




- **Solving** $A_h \delta u_h = F_h$



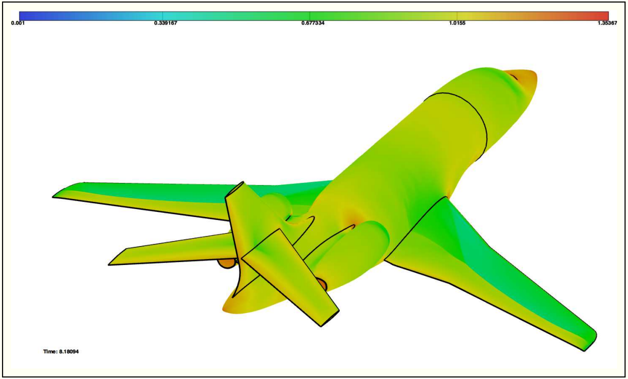
- **Monogrid method**



- **2-grid method**

Results (uniform)

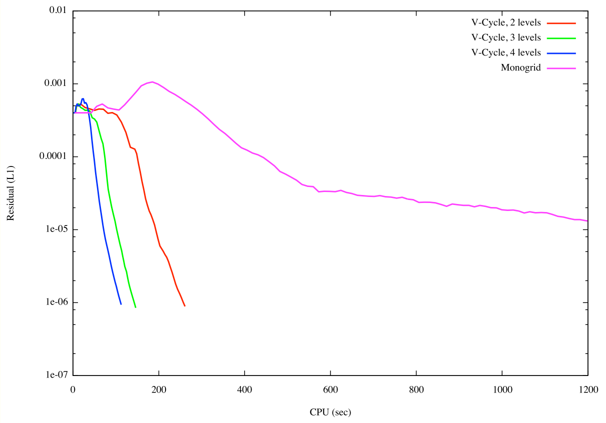
- **Transsonic flow around a falcon**



- A single-grid method is compared to multigrid methods

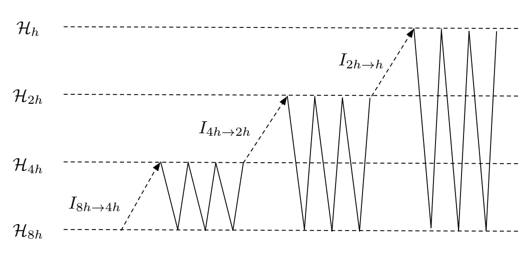
Results (uniform)

Transsonic flow around a falcon



- Significant acceleration of the convergence speed using multigrid methods and better convergence

Coupling with adaptivity (Ongoing)



- In the previous example, the coarser meshes are **uniformly generated**
- Coupling with adaptivity : **each mesh is adapted**
- Ideal relaxation : generate the adapted coarser mesh leading to the best correction

Conclusion

- Done :
  - Contribution to the implementation of a turbulent flow solver
  - Validation of the flow solver
  - Experimentations of new remeshing strategies for boundary layers
  - Parallel mesh adaptation
  - Uniform full multigrid
- Ongoing :
  - Adaptive multigrid methods
  - MPI parallelization of the flow solver

- **Thank you!**