1 Introduction

Generating a good mesh is a hard work, especially if the mesh is structured and the geometry is complex. Furthermore, having a first guess of the flow behaviour is important to be able to refine the mesh in the desired region and to coarsen it in regions where high precision is not needed. The process thus requires a lot of time, both human time and computational one, and is often recursive. The concept of mesh adaptation aims at addressing this problem by allowing an unstructured mesh to adapt automatically to the flow behaviour, needing only a rough initial mesh and no apriori knowledge of the flow. The difficulty here relies on the adaptation tool, as realizing an unstructured mesh is since the 90’s quite an easy task, thanks to Automated Unstructured Tetrahedra Mesh Generation Methods, for instance the "Advancing Front Method" [1], the "Delaunay Method" [2, 3] or the "Octree Like Method" [4].

2 Mathematical tools for adaptation method

In order to catch the flow physical properties, the mesh has to be dense where there are strong variations. A naïve way to improve a coarse mesh in high variation area would be to uniformly divide the size of the cell by two in each direction. However, this leads to really higher computational time, as we also have to divide the time step by two in order to keep the same CFL constant, thus multiplying by \(2^4\) the amount of CPU or time needed.

A smarter approach is to take into account the fact that fluid mechanics problems often have geometric particularities and quantities do not vary with the same magnitude in every direction. We can thus create anisotropic meshes which will improve the accuracy of our solution without increasing the computational time too much.

2.1 The concept of metrics

To perform anisotropic mesh adaptation, a convenient tool is metrics on Riemannian metric space. The idea comes from the fact that in order to perform a good adaptation, one must be able to specify at each point of the domain privileged sizes and orientations for the mesh elements. It appeared first in 1991, in a paper written by George P., Hecht F., Vallet M. [5]. A detail explanation of metric-based notions and mathematical aspects can be found in [6, 7, 8]. We will just give here a short overview.

A metric is a real matrix \(M\), symmetric positive definite. It can be defined on a point \(P\) or for an element \(K\) of a mesh \(T_h\) meshing the domain \(\Omega\). In our application, \(M\) will be defined on every vertex \(P\) of the mesh \(T_h\), situated at the position \(x\) in space, we will thus write \(M(x)\).

\(M\) being symmetric positive definite, the matrix defines a scalar product on the domain \(\Omega\), and thus a length between every \((A, B) \in \Omega^2\):

\[
d_M(A, B) = \sqrt{(x_A - x_B)^T M(x_A - x_B)}
\]  \hspace{1cm} (1)

Equation (1), written for a \(M\) constant can be generalized for \(M(x)\), using an interpolation if \(M(x)\) is discrete (see [9]):

\[
\|\tilde{A}^T\tilde{B}\|_M = \int_0^1 \sqrt{(\tilde{A}^T\tilde{B})^T M(A + t\tilde{A}^T\tilde{B}) \tilde{A}^T\tilde{B}} \cdot dt
\]  \hspace{1cm} (2)

Furthermore, as \(M\) is symmetric positive definite, the matrix is diagonalizable. \(M\) has 2 (respectively 3) eigenvalues \((\lambda_i)_{i \in [1,2]}\) (respectively \((\lambda_i)_{i \in [1,3]}\)) and privileged direction in 2D (respectively 3D) and thus include information about anisotropy, as pictured in figure 1\(^1\). The unit ball is the

---

\*Safran Tech, E-mail: marc-antoine.bouvattier@polytechnique.edu

\*Safran Tech, E-mail: xavier.garnaud@safrangroup.com

\(1\) Thanks to Frederic Alauzet
set of points $M$ such that $\|PM\|_M = \sqrt{(PM)^*MPM} = 1$

![Figure 1: Unit ball of the metric](image)

If we now use an algorithm creating a mesh $T_h^{i+1}$, in which every edge $e_i$ of every element $K_k$ has a length equal to 1, with respect to the metric field $\mathcal{M}(x)$, we create an anisotropic mesh respecting the orientation and size prescribed by $\mathcal{M}(x)$, as pictured on figure 2, with axis lengths in the real space equal to $l_i = \frac{1}{\sqrt{\lambda_i}}$

![Figure 2: Triangle with edge length equal to 1](image)

2.2 Link between metric, hessian and numerical error

The concept of metric can thus prove itself very useful in order to anisotropically adapt a mesh. But how shall we define this metric for a RANS computation?

The idea of adaptation is to equally distribute the numerical projection error over the mesh and also to minimize this global error. The first goal aims at improving the consistency of the solution over the mesh while the second targets a better accuracy of the result.

One can show that the infinite norm of the numerical projection error $\text{err}$ on a element $K$ when calculating $u$ can be written, in every dimension [9]:

$$\|\text{err}\|_{\infty,K} \leq c \cdot \max_{v \in K} \max_{e \in E_K} \|\langle e, H_u(x) e \rangle\|$$  (3)

Where $c$ depends on the dimension of the space and $E_k$ is the edge set of the element $K$. $H_u(x)$ is the hessian of $u$ calculated in $x$.

In physics, one can assume that every variable we will use has the nice property to have a symmetrical hessian, meaning that its spatial derivatives commute. Thus the hessian is diagonalizable and can be written (in 2D) as

$$H_u = \mathcal{R} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \mathcal{R}^{-1}$$. The matrix denoted $|H_u|$ and equal to $\mathcal{R} \begin{pmatrix} |\lambda_1| & 0 \\ 0 & |\lambda_2| \end{pmatrix} \mathcal{R}^{-1}$ is a symmetric positively definite matrix.

This matrix defines a metric and is an upper bound of (3), as $|\lambda_1 e_1^2 + \lambda_2 e_2^2| \leq |\lambda_1 e_1^2| + |\lambda_2 e_2^2|$ (the demonstration also holds in 3D). We can finally rewrite the equation (3) as:

$$\|\text{err}\|_{\infty,K} \leq c \cdot \max_{e \in E_K} \|\tilde{e}\|_{H_u(K)}$$  (4)

This equation (4) shows that the projection error is linked to the length of every mesh edges through the hessian of our solution. Thus by setting a maximal value for the error $\|\text{err}\|_{\infty,K}$ and knowing the metric $|H_u|(K)$ from the solution $u$, we can adapt the length of every mesh edges $\{e \in E_K | K \in T_h\}$ with respect to that metric, such that the equation (4) is verified.

2.3 Final metric and mesh adaptation

To control the number of elements the new mesh will have, we have to modify the metric, realizing that (in 2D)

$$\text{Vol}(K) = h_1 \ast h_2 = \sqrt{\frac{1}{\lambda_1 + \lambda_2}} = \frac{1}{\sqrt{\det(|H_u(K)|)}}$$, which implies that $\sqrt{\det(|H_u(K)|)}$ is roughly the number of cells per unit mesh and thus $\left(\int_{\Omega} \sqrt{\det(|H_u(y)|)} dy\right) \approx N_{\text{cells}}$

We can then roughly have the number of desired faces $(N)$ in 2D, by using the following metric (5):

$$\mathcal{M}(x) = N \times \left(\int_{\Omega} \sqrt{\det(|H_u(y)|)} dy\right)^{-1} \times |H_u(x)|$$  (5)

An even better metric (equation 6 can be found in the literature[10], which allows the control of the number of elements in the future mesh, the cells’ size extrema and the accuracy of the caught variation, in different physical space dimensions $\text{dim}$ and in different $L^p$ norms.

$$\mathcal{M}_{L^p}(x) = \left(\int_{\Omega} \det(|H_u(y)|) \frac{d^p}{\pi^{\frac{p}{2} \times \text{dim}}} dy\right)^{-\frac{2}{\text{dim}}} \times N \frac{2}{\pi^{\frac{p}{2}}} \times \text{det}(|H_u(x)|) \frac{\pi^{\frac{p}{2} \times \text{dim}}}{\text{det}(|H_u(x)|)}$$  (6)

The adaptation is here done using the mmg library in an anisotropic way[11]. We provide the metric field we want our mesh to be adapted with, and mmg modifies the existing mesh such that every edge has a length equal to 1 in this metric field. The number of required faces $N$, inversely proportional to the edges length of every element (as the volume is constant), controls the precision of the solution we will have on the adapted mesh, (eq 4).

2Thanks to Frederic Alauzet

3https://www.mmgtools.org
3 Metric verification

3.1 Adaptation with an analytic solution

We studied first the adaptation of an analytic solution \( f \) given by (7) defined on a quite simple mesh (as pictured in figure 3). The origin of the coordinates are in the center of the hole, \( R \) is the hole radius, \((r, \theta)\) are the polar coordinates equivalent to \((x, y)\), \( A \) is the amplitude of the shock and \( \delta_0 \) is its characteristic length. (If nothing is stated, \( A = 1 \) and \( \delta_0 = 0.5R \))

\[
f(x, y) = \tanh \left( \frac{x - R}{R/2 \cdot (2 + \cos(\theta))} \right) + 1 + A \cdot \tanh \left( \frac{x - R/4}{\delta_0} \right)
\]

(7)

Using the metric given by (6) with \( N \) equal to the initial number of faces, in \( L^2 \) and \( L^\infty \) norm, we have convergence to the following meshes, after 10 iterations. The corresponding meshes are show in figures 4 and 5. One can observe that the mesh adapted with the \( L^\infty \) metric seems quite isotropic. This is due to the fact that \( \delta_0 \) is rather big and thus the infinite norm does not capture it.

3.2 Convergence with an analytic solution

The best way to analyze the convergence of the adaptation process (solution calculation, metric calculation, adaptation, solution calculation on the new mesh and so on) would be to verify that from a certain iteration the mesh does not move anymore, i.e. that every vertex has the same position and has the same links to other vertices. However this is a rather strong criteria which is not fulfilled in our meshes. Points can move a little aroud their previous positions and we also sometimes have periodic edges swap. To analyze the convergence, we thus analyze the number of faces we get after each adaptation. As one can see in figure 6, the convergence is obtained after 10 adaptations. We also note that the mesh seems to be converged in the stronger definition of convergence.

3.3 Number of mesh faces

Keeping the same mesh and the same analytical function (7), we studied the relation between the number of desired faces \((N \) roughly) and the number of faces we get after the mesh has converged. On figure 8 are two series of plots, one for the \( L^2 \) norm and one for the \( L^\infty \) one, with the stretching ratio \( h_{\text{grad}} \) varying from 1.1 to 100. As one can see, the ratio is not constant and not equal to 1.
ever this ratio tends to one as we increase the number of desired faces. Two parameters accounts for that: during the adaptation, we impose a maximal length ratio between two adjacent edges, named $h_{\text{grad}}$ and equal to 1.1 in most of our cases. As we increase its value, the number of faces ratio comes closer to 1. We also impose a set of maximal and minimal size for the edges $(h_{\text{min}}, h_{\text{max}})$, which modifies the one prescribed by the metric and thus the final number of cells.

### 3.4 Influence of the initial number of faces

One last important thing to verify is the influence of the initial number of faces on the different parameters of the adaptation (convergence speed, final mesh face number etc ...). Using three different meshes and the metric 6 in $L^2$ norm with the same $N$ for each norm, we show that the adaptation properties are independent of the first mesh quality.

#### Figure 8: Number of faces versus iteration for the two norms and three different initial meshes

### 4 Adaptation cycle

When it comes to genuine turbo-machinery RANS calculation, two things must be added: a boundary-layer mesh, made of quads in 2D and prims in 3D (B.L.) to capture the flow properties near the walls, and periodic boundaries (P.B.), as we often study a single blade and not the entire row of blades. This leads to two problems: the state of the art adaptation tools can’t handle the remeshing of prisms and can’t respect the periodicity.

#### 4.1 Managing the boundary layer

We thus developed a tool which allow the removal of existing boundary layer and another one which creates a B.L.

To create the B.L., we push the points on the physical boundary to the required distance, and we add a B.L. imposing the first prism height, the aspect ratio of the last prism and the maximal stretch ratio between two prisms. See figure 9

The tool removing the B.L. suppresses the existing prisms and expand the triangles/tetrahedrons previously linked to them to the physical boundary, by projecting the previous contact points on the physical boundary. See figure 9
4.2 Periodic adaptation

To realize the adaptation, we have to freeze the elements on the periodic boundaries, in order to easily keep the periodicity of the mesh. Thus the elements there can not be refined or coarsened. To address this problem, we developed a tool which shift the elements directly in contact with the boundary to the position where they should be by periodicity on the other periodic boundary (see figure 10). Then they became inside elements and can be adapted.

4.3 Adaptation workflow

We now have all the tools to realize the adaptation process pictured in figure 11. For an example of adaptation workflow on a genuine mesh, see figure 19 in the appendix. In our case, the initial mesh was done with ANSA \(^4\) and the solver used was elsA-H \(^5\), an hybrid solver developed at ONERA. We chose to use the Mach number to derive the metric, as it is a physical property which varies in B.L., in shocks and in wakes, thus allowing to capture most of the interesting features of the flow.

5 LS89

The first case is a 2D one, known as the LS89, which is designed and tested at the Von Karman Institute\(^12\). We used the condition associated to the test number MUR235\(^12\). The initial mesh and the case conditions are shown on figure 12, and has 6000 cells. We used the following numerical parameters (table 1).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turbulence model</td>
<td>Spalart</td>
</tr>
<tr>
<td>Riemann solver</td>
<td>roe</td>
</tr>
<tr>
<td>CFL</td>
<td>50</td>
</tr>
<tr>
<td>Limiter</td>
<td>Venkatakrishnan</td>
</tr>
<tr>
<td>(h_{grad})</td>
<td>10</td>
</tr>
<tr>
<td>(N_{faces})</td>
<td>6000</td>
</tr>
<tr>
<td>Norm</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 1: Parameters for the flow simulation and the adaptation

\(^4\)https://www.beta-cae.com/ansa.htm
\(^5\)http://elsa.onera.fr/
5.1 Adapted Mesh

The figure 13 shows the converged mesh. One can observe that the wake is well captured, and that the wakes of the other blades (obtained by periodicity) are also well defined. We have a nice B.L. and the shock which occurs mid span of the suction side is also taken into account.

![Figure 13: Adapted mesh](image)

5.2 Periodicity checking

Figure 20 (in appendix) shows three adapted meshes, shifted of the vein height, as it would be in the cascade of blades. One can observe that the periodicity is well respected and that there is a real continuity in the wakes.

Furthermore, a zoom on the periodic boundary (figure 21 in appendix) shows that there are no more signs of the periodic boundary as it was before the adaptation.

![Figure 14: Mach on the initial (left) and adapted (right) meshes](image)

![Figure 15: Slice in which we study the Mach number](image)

5.3 Solution comparison

Solution comparison between the Mach number calculated on the initial mesh and the one calculated on the final converged mesh is done on figure 14. As one can see, there are no major changes in the general solution behavior, but the wake seems to be captured better, especially regarding its width.

This is emphasized if we study the Mach number in the slice defined in figure 15, for every mesh corresponding to the different adaptation steps. We have both an improvement in the value of the Mach number and in the width of the wake, especially as we move further from the blade. We can also see that the convergence is rather quick.

![Figure 16: Mach number along the slice for different meshes](image)

6 RO37

The 3D case we studied is the NASA RO37 test case, designed and tested originally by Reid and Moore[13] at NASA Lewis. It is a low aspect ratio inlet stage for an eight stage core compressor with a 20:1 pressure ratio[14].

The CDF simulation was not done for this case, only the adaptation part was performed, on a coarse mesh solved with an LES method.

The results of the adaptation are shown on figure 17. One can observe both the coarsening of the cells at the inlet and outlet of the profile, and the refinement of the cells close
to the profile, which also captures the different shocks.

Figure 17: RO37 adapted once

When it comes to 3D adaptation, the cost in both computational time and storage of data increase a lot. We thus realized a tool allowing an adaptation by partition, as shown in figures in appendix. Figure 22 shows the initial mesh, figure 23 the first partition and 24 the consequent adaptation. We then realize 2 more partitions and adaptations to make sure we adapt the partition boundaries in a good way (figures 25, 26, 27, 28). A weighting function was also defined such that the new partition are different from the previous one.

7 Improvements

7.1 2D case

When analyzing the isentropic Mach number over the blade, we realized that the value obtained with our meshes (continuous lines on figure 18, for different total number of vertices) was varying around the desired value, obtained through DNS calculation (blue dots). This might be explained by the high variation in size for contiguous prims in the BL, as shown in figure 21. A way to control this stretching ratio in the B.L. will have to be defined.

The future improvements of the numerical solver elsA will allow the extraction of the $y^+$ value, which would be a relevant parameter to generate the B.L.

7.2 3D case

As said before, the workflow adaptation brick is working, but the convergence of the set of meshes still have to be verified.

It would also be really relevant to develop a tool which allows an adaptation respecting the case’s CAD regarding the periodic boundaries. Indeed the irregular periodic boundaries generated by the different adaptations might cause the CFD solver to crash.

8 Conclusion

The tools and methods explained and verified in this article are an application of the mesh adaptation process’ state of the art. They still have to be improved and tested on more cases, but the first examples give some really interesting results. At the moment, most companies use structured meshes, which require a really long time to be generated, an apriori knowledge of the flow and which can be over-refined. The adaptation process would allow a huge time saving in the mesh generation, and would give the assurance that the flow is captured in the best possible way, for a given number of cells, which is directly linked to the computational time.

References


