Periodic adjoints and anisotropic mesh adaptation in rotating frame for high-fidelity RANS turbomachinery applications

F. Alauzet\textsuperscript{a,∗}, L. Frazza\textsuperscript{a}, D. Papadogiannis\textsuperscript{b}

\textsuperscript{a}INRIA Saclay Ile-de-France, Projet Gamma, 1, rue Honoré d’Estienne d’Orves, 91126 Palaiseau, France
\textsuperscript{b}Safran Tech, Rue des jeunes bois, Châteaufort, 78772 Magny-les-Hameaux, France

Abstract

The scope of this paper is to demonstrate the viability and efficiency of metric-based unstructured anisotropic mesh adaptation techniques to turbomachinery applications. The main difficulty in turbomachinery is the periodicity of the domain that must be taken into account in the mesh-adaptive solution process. The periodicity is strongly enforced in the flow solver using ghost entities to minimize the impact on the source code. For the mesh adaptation, the local remeshing is done in two steps. First, the inner domain is remeshed with frozen periodic frontiers, and, second, the periodic surfaces are remeshed after moving geometric entities from one side of the domain to the other. One of the main goals of this work is to demonstrate that mesh-independent certified numerical solutions can be obtained thanks to anisotropic mesh adaptation and that it is possible to run high-fidelity CFD on unstructured adapted meshes composed only of tetrahedra. This paper demonstrates how mesh adaptation, thanks to its automation, is able to generate meshes that are extremely difficult to envision and almost impossible to generate manually, leading to highly accurate numerical solutions. This study considers feature-based error estimate based on the standard multi-scale $L^p$ interpolation error estimate and goal-oriented error estimate using an adjoint state to control the error on turbomachinery output functionals. A description of the the flow solver and the adjoint solver is given in this work as they are very different from what is encountered in the turbomachinery community. We also present all the specific modifications that have been introduced in the adaptive process to deal with periodic simulations used for turbomachinery applications. The periodic mesh adaptation strategy is then tested and validated on the LS89 high pressure axial turbine vane and the NASA Rotor 37 test cases.

Keywords: Periodic anisotropic mesh adaptation, periodic flow solver, periodic adjoint, Reynolds average Navier-Stokes, turbomachinery, NASA Rotor 37, feature-based error estimate, goal-oriented error estimate.

1. Introduction

In modern Reynolds-Averaged Navier-Stokes (RANS) numerical simulations, the mesh generation and CAD discretization are known to be one of the main bottlenecks for many applications. This is particularly the case for the generation of suitable meshes for turbomachinery geometries which remains a difficult task. Turbomachinery flows are wall-bounded flows that are characterized by multiple privileged directions. Such directions can be found at the near-wall boundary layers across the blades, hub and casing, at the wakes as well as at the shocks in the cases with transonic operating points. The impact of the near-wall boundary layers and wakes in particular on the overall predictions, combined with the necessity for periodic boundary conditions in the azimuthal direction (to simulate only a single blade per row), renders the simulations of turbomachinery on structured meshes very effective as such meshes allow for high resolution on those areas. It is usually addressed by the mean of multi-block structured meshes composed only of hexahedra \cite{7,8,12,13,22,27,32,61}. Traditional processes rely on the experience and intuition of a skilled engineer to predict and adapt the mesh prescription to the flow. The generation of multi-block structured meshes requires a careful management but automatic processes have been set-up and work very well when a simple geometry like a blade is considered. However, a standard structured mesh will likely be of insufficient resolution at a shock or around the secondary flows developing across a blade passage, phenomena crucial for the performance and the stability of turbomachinery. Additionally, convecting correctly the wakes far from the blade would require a very large number of cells. Furthermore, the current trend is towards more realistic turbomachinery simulations that include complex technological effects (cavities, seals, squealers, fillets, cooling, etc.) as these effects, previously usually taken into account via correlations, are shown to impact considerably the flow field. Meshing such complex geometries using structured meshes is very difficult, while the flows in such effects are no longer characterized by the privileged directions observed across a blade, for example. Following such
meshing guidelines slows down the mesh generation process leading to a prohibitive cost in pre-processing time of the numerical simulation pipeline. This lack of automation is an impediment for many applications.

These constraints imply that introducing anisotropic mesh adaptation techniques with unstructured meshes can be an attractive solution to achieve improved predictions, as it can combine easy initial mesh generation with the ability to follow the flow’s preferred directions while being capable of refining in all relevant flow phenomena. This work proposes a metric-based anisotropic mesh adaptation strategy based on unstructured meshes composed only of tetrahedra which is able to automatically manage complex geometries and technological effects, the periodicity of the domain, and adapt the mesh in size and in direction to the flow.

Metric-based anisotropic mesh adaptation is a well known topic for external flows [4, 5, 26, 36, 51, 47, 71]. We have demonstrated that high-fidelity prediction for viscous flows can be obtained using fully unstructured meshes [2], which until now was considered impossible in the community. Starting from an initial coarse non-adapted mesh which is generated without any a priori knowledge on the solution, it consists in iteratively modifying the computational mesh in order to better capture all the flow features and to obtain an improved numerical solution for a given number of degrees of freedom. The size and the orientation of the elements of the mesh is given by an error estimate which evaluates the error in the computation of the solution due to the discretization. This process can be combined with a convergence study by incrementing gradually the size of the mesh, and thus giving the capability to check the independence of the numerical solution to the domain discretization.

However, there are few efforts on turbomachinery mesh adaptation reported in the literature. One of the first studies [20] performed Adaptive Mesh Refinement (AMR) on unstructured meshes by splitting the edges of the cells on areas where a variety of criteria (geometric factors or flow field gradients) suggested that refinement was necessary. The approach was tested on steady and unsteady problems. However, it was limited to isotropic AMR, which cannot follow the privileged directions of the flow, and the mesh quality after multiple cell divisions can be an issue. A more flexible approach on unstructured meshes, allowing full mesh morphing for isotropic and anisotropic mesh adaptation based on the library MOM3D, was reported in [58]. While the results were promising, the approach lacked the ability to adapt the periodicities, which is an important limitation in turbomachinery simulations. Mesh adaptation for structured meshes has also been proposed, notably in [63] and more recently in [67, 68]. These methods, which combine mesh movement and refinement, can handle the periodicities more efficiently and allow anisotropic mesh adaptation. However, they have the drawback of requiring an initial structured mesh and the node movement can lead to poor quality cells.

In this work, we propose to develop a mesh-adaptive solution platform for turbomachinery applications and to analyze the obtained results on two well-documented cases: the LS89 high pressure axial turbine vane and the NASA Rotor 37. Turbomachinery applications present specific characteristics which make them a challenging topic for mesh adaptation:

- Complex enclosed geometries (small gaps, cavities, seals, ...),
- Complex physical interactions (rotating frames, shocks, boundary layers, ...),
- Periodic resolution,
- Moving geometries.

The following choices have been made.

Reynolds-Averaged Navier-Stokes (RANS) numerical simulations are considered with the Spalart-Allmaras turbulence model. For rotating machines, we are solving the equations of movement in the rotating frame instead of the absolute frame.

The considered flow solver is a vertex-centered (flow variables are stored at vertices of the mesh), it uses a Finite Volume discretization for the convective and the source terms and a Finite Element discretization for the viscous terms on unstructured meshes composed of triangles in 2D and tetrahedra in 3D. The time integration considers an implicit temporal discretization. At each time step, the linear system of equations is approximately solved using a Symmetric Gauss-Seidel (SGS) implicit solver and local time stepping to accelerate the convergence to steady state. It is very important to exactly differentiate all the terms as it greatly improves the flow solver convergence. This choice for the flow solver is a major difference with what is encountered in the community as most of the flow solver are cell-centered on multi-block structured meshes composed of hexahedra [7, 8, 12, 13, 22, 27, 32, 61]. Its description is provided as this numerical method has demonstrated to be able to obtain high-fidelity prediction for viscous flows using unstructured adapted meshes. The periodicity is strongly enforced in the flow solver using ghost entities to minimize the impact on the source code. In that case, there is a little memory overhead but periodic vertices are computed similarly to inner vertices. This clearly facilitates the implementation and the impact of the periodicity in the code is localized in a few number of routines.

As regards the adaptation of the mesh, we developed an approach inspired by [23] which is one of the rare paper on local remeshing for periodic domain. The main difference between the proposed approach and [23] is that we require to keep the initial periodic domain geometry while in [23] the domain shape can change drastically. We add this constraint because it eases the development of other parts of the adaptive process (flow solver, error estimate, solution interpolation) and facilitates any solution
2.1. The Navier-Stokes equations

The compressible Navier-Stokes equations for mass, momentum and energy conservation read:

\[
\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\
\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) + \nabla p &= \nabla \cdot \mathbf{T}, \\
\frac{\partial (\rho E)}{\partial t} + \nabla \cdot ((\rho E + p) \mathbf{u}) &= \nabla \cdot (\mathbf{T} \cdot \mathbf{u}) + \nabla \cdot (\lambda \nabla T),
\end{aligned}
\]
where \( \rho \) denotes the density (kg/m\(^3\)), \( \mathbf{u} \) the velocity (m/s), \( E \) the total energy per mass (m\(^2\)·s\(^{-2}\)), \( p \) the pressure (N/m\(^2\)), \( T \) the temperature (K), \( \mu \) the laminar dynamic viscosity (kg/(m·s)) and \( \lambda \) the laminar conductivity. \( \mathbf{T} \) the stress tensor:

\[
\mathbf{T} = \mu \left( \nabla \otimes \mathbf{u} + \nabla \otimes \mathbf{u}^\ast \right) - \frac{2}{3} \nabla \cdot \mathbf{u} = \mu \mathbf{\tau},
\]

and, in 3D, \( \mathbf{u} = (u, v, w) \) and

\[
\nabla \cdot \mathbf{u} = \begin{bmatrix} u_x + v_y + w_z & 0 & 0 \\
0 & u_x + v_y + w_z & 0 \\
0 & 0 & u_x + v_y + w_z \end{bmatrix},
\]

where \( u_x = \frac{\partial u}{\partial x} \), \( u_y = \frac{\partial u}{\partial y} \), \( u_z = \frac{\partial u}{\partial z} \) (idem for \( v \) and \( w \)). The variation of nondimensionalized laminar dynamic viscosity and conductivity coefficients \( \mu \) and \( \lambda \) as a function of the dimensional temperature \( T \) is defined by Sutherland’s law:

\[
\mu = \mu_\infty \left( \frac{T}{T_\infty} \right)^{3/4} \left( \frac{T_\infty + Su}{T + Su} \right) \quad \text{and} \quad \lambda = \lambda_\infty \left( \frac{T}{T_\infty} \right)^{3/4} \left( \frac{T_\infty + Su}{T + Su} \right),
\]

where \( Su = 110 \) is the Sutherland temperature and the index \( \infty \) denotes reference quantities. The relation linking \( \mu \) and \( \lambda \) is expressed from the Prandtl laminar number:

\[
\Pr = \frac{\mu C_p}{\lambda} \quad \text{with} \quad \Pr = 0.72 \quad \text{for (dry) air},
\]

where \( C_p \) is the specific heat at constant pressure.

In the case of the Reynolds Average Navier-Stokes (RANS) numerical simulations, the Navier-Stokes equations are completed by a turbulence model defined by one or more equations (here the Spalart-Allmaras model given in Section 2.3), and the laminar dynamic viscosity \( \mu \) (resp. the laminar conductivity \( \lambda \)) are replaced in the above equations by the sum between the laminar and the turbulent dynamic viscosity (resp. conductivity): \( \mu + \mu_t \) (resp. \( \lambda + \lambda_t \)). The turbulent dynamic viscosity \( \mu_t \) is given by the turbulence model and the turbulent conductivity \( \lambda_t \) is expressed from the Prandtl turbulent number:

\[
\Pr_t = \frac{\mu_t C_p}{\lambda_t} \quad \text{with} \quad \Pr_t = 0.9 \quad \text{for (dry) air}.
\]

### 2.2. The Navier-Stokes equations in rotating frame

In the case of rotating machines (such as a turbomachine), we choose to solve the equations of movement in the rotating frame (relative frame) instead of the absolute frame. Let \( \mathbf{u}_A \) be the fluid velocity in the absolute frame, \( \mathbf{u}_R \) the fluid velocity in the rotating frame, and \( \mathbf{\Omega} \) the rotating velocity of the machine. The composition of the velocities gives:

\[
\mathbf{u}_A = \mathbf{u}_R + \mathbf{\Omega} \times \mathbf{r} = \mathbf{u}_R + \mathbf{R},
\]

where \( \mathbf{r} \) is the radius at which the considered point is located. If we denote by \( \mathbf{x} \) the position vector of the considered point w.r.t the rotation center, \( \text{i.e.}, \mathbf{x} = OP \), and by \( \mathbf{e}_\Omega \) the rotation axis, \( \text{i.e.}, \mathbf{\Omega} = \|\mathbf{\Omega}\| \mathbf{e}_\Omega = \omega \mathbf{e}_\Omega \), then \( \mathbf{r} \) is given by (see Figure 1):

\[
\mathbf{r} = \mathbf{x} - \alpha \mathbf{e}_\Omega \quad \text{with} \quad \alpha = \mathbf{x} \cdot \mathbf{e}_\Omega.
\]

As the density and pressure are left unchanged by the change of frame, we thus have the following transformation functions for primitive (physical) variables:

\[
\begin{bmatrix}
\rho_A \\
\mathbf{u}_A \\
\rho A
\end{bmatrix} = \mathbf{U}_A = f(U_R) = \begin{bmatrix}
\rho_R \\
\mathbf{u}_R + \mathbf{R} \\
\rho_R
\end{bmatrix},
\]

and for the conservative variables:

\[
\begin{bmatrix}
\rho_A \\
\rho \mathbf{u}_A \\
\rho E_A
\end{bmatrix} = \mathbf{W}_A = f(W_R) = \begin{bmatrix}
\rho_R \\
\rho \mathbf{u}_R + \rho R \\
\rho E_R + \rho \mathbf{u}_R \cdot \mathbf{R} + \frac{1}{2} \rho_R \|\mathbf{R}\|^2
\end{bmatrix},
\]

as

\[
(\rho E)_A = \frac{\rho_R}{\gamma - 1} + \frac{1}{2} \rho_R \|\mathbf{u}_R + \mathbf{R}\|^2 = \frac{\rho_R}{\gamma - 1} + \frac{1}{2} \rho_R \|\mathbf{u}_R\|^2 + 2 \rho_R \cdot \mathbf{R} + \|\mathbf{R}\|^2.
\]
Solving the Navier-Stokes equations in a rotating frame at constant rate introduces the Coriolis and centrifugal forces,

\[
F_{\text{Coriolis}} = -2\rho R \Omega \times u_R \tag{5}
\]

\[
F_{\text{Centrifugal}} = -\rho R \Omega \times (\Omega \times r) = \rho R \omega^2 r, \tag{6}
\]

with \(\omega = \|\Omega\|\). The Coriolis force being by definition orthogonal to the fluid velocity, it does not produce work. The energy equation is thus modified by adding the work of the centrifugal force:

\[
E_{\text{Centrifugal}} = \rho R \Omega \times (\Omega \times r) \cdot u_R = \rho R \omega^2 r \cdot u_R. \tag{7}
\]

2.3. Turbulence modeling: the Spalart-Allmaras one equation model

According to the standard approach to turbulence modeling based upon the Boussinesq hypothesis, the turbulence is modeled with an eddy viscosity \(\mu_t\), which is added to the laminar (or dynamic) viscosity \(\mu\). The dynamic viscosity is usually taken to be a function of the temperature, whereas \(\mu_t\) is obtained using a turbulence model. Here we choose the Spalart-Allmaras one equation turbulence model [65] given by the following equation:

\[
\frac{\partial \tilde{v}}{\partial t} + u \cdot \nabla \tilde{v} = c_{t1}[1 - f_{t2}]\tilde{S} \tilde{v} - \left[ c_{w1}f_{u} - \frac{c_{b1}}{k^2}f_{z1} \right] \left( \frac{\tilde{v}}{\tilde{d}} \right)^2 + \frac{1}{\sigma} \left[ \nabla \cdot ((v + \tilde{v})\nabla \tilde{v}) + c_{b2}\|\nabla \tilde{v}\|^2 \right] + f_{t1} \Delta u^2, \tag{8}
\]

where \(\tilde{v}\) is the turbulent kinematic viscosity and all the constants are defined below. In the standard model the trip term is being left out, i.e., \(f_{t1} = 0\). Moreover, some implementations also ignore the \(f_{t2}\) term as it is argued that if the trip is not included, then \(f_{t2}\) is not necessary [25]. This simplified version has been considered and we prefer to write it under the following form, which is more appropriate for its discretization with the Finite Volume - Finite Element method. Indeed, Equation (8) can be decomposed into the following terms:

\[
\frac{\partial \tilde{v}}{\partial t} + u \cdot \nabla \tilde{v} = c_{b1}\tilde{S}\frac{\tilde{v}}{d} - \frac{c_{w1}f_{u} - c_{b1}}{k^2}f_{z1} \left( \frac{\tilde{v}}{\tilde{d}} \right)^2 + \frac{\rho}{\sigma} \nabla \cdot ((v + \tilde{v})\nabla \tilde{v}) + \frac{c_{b2}}{\sigma}\|\nabla \tilde{v}\|^2. \tag{9}
\]

Notice that this is not a conservative model. If a conservative form of the Spalart-Allmaras is foreseen, we have to consider the variation proposed by Catris and Aupoix [17]. The turbulent eddy viscosity is computed from:

\[ \mu_t = \rho \tilde{v} f_{t1}, \quad \text{where} \quad f_{t1} = \frac{\chi^3}{\chi^3 + c_{v1}} \text{ and } \chi = \frac{\tilde{v}}{v} \quad \text{with} \quad v = \frac{\mu}{\rho}. \]

Additional definitions are given by the following equations:

\[ f_{t2} = 1 - \frac{\chi}{1 + \chi f_{t1}} \quad \text{and} \quad \tilde{S} = \|\nabla \times u\| + \frac{\tilde{v}}{k^2 c_{v2}} f_{t2} \]

\(d\) is the distance to nearest wall which is computed for each vertex at the beginning of the simulation. The set of closure constants for the model is given by

\[ \sigma = \frac{2}{3}, \quad c_{b1} = 0.1355, \quad c_{b2} = 0.622, \quad \kappa = 0.41, \quad c_{w1} = \frac{c_{b1}}{\kappa} + \frac{1 + c_{b2}}{\sigma}, \quad c_{w2} = 0.3, \quad c_{v3} = 2, \quad c_{v1} = 7.1. \]

![Figure 1: Description of all the vectors involved in the rotating frame source term.](image)
Finally, the function \( f_w \) is computed as:

\[
  f_w = \frac{1}{\sqrt[6]{g^6 + c_{w3}^6}} \quad \text{with} \quad g = r + c_{w2} \left( \rho^6 - r \right) \quad \text{and} \quad r = \min \left( \frac{\tilde{v}}{S_k^2 z^2}, 10 \right).
\]

For the NASA Rotor 37, we consider the QCR version of the Spalart-Allmaras turbulence model, described in [64], which modified the turbulent stress tensor:

\[
  \mathcal{T} = \mu \tau + \mu_i (\tau - \tau_{\text{QCR}}), \quad (10)
\]

with the \( \tau_{\text{QCR}} \) given in [64]. According to the NASA Turbulence Modeling Resource, the considered version is the SA-noft2-QCR2000.

2.4. Vector form of the Reynolds-averaged Navier-Stokes system in rotating frame

The considered Reynolds-averaged Navier-Stokes (RANS) system is rewritten under a vector form:

\[
  W_t + \nabla \cdot \mathbf{F}^E = \nabla \cdot \mathbf{F}^V + \mathbf{F}^S,
\]

where \( W \) is the nondimensionalized conservative variables vector:

\[
  W = (\rho, \rho u, \rho v, \rho w, \rho E, \rho \tilde{v})^T.
\]

\( \mathbf{F}^E \) is the convective (Euler) fluxes vector:

\[
  \mathbf{F}^E(W) = \begin{pmatrix} \mathcal{F}^E_1(W), \mathcal{F}^E_2(W), \mathcal{F}^E_3(W) \end{pmatrix} = (\rho u, \rho u u + pe_1, \rho v u + pe_2, \rho w u + pe_3, \mathbf{u} (\rho E + p), \rho \tilde{v} \mathbf{u})^T, \quad (11)
\]

with \( (e_1, e_2, e_3) \) the canonical basis. \( \mathbf{F}^V \) is the viscous fluxes vector:

\[
  \mathbf{F}^V(W) = \begin{pmatrix} \mathcal{F}^V_1(W), \mathcal{F}^V_2(W), \mathcal{F}^V_3(W) \end{pmatrix} = \begin{pmatrix} 0, \mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathbf{T} \cdot \mathbf{u} + \lambda \nabla T, \frac{\mu}{\sigma} (\gamma + \tilde{v}) \nabla \tilde{v} \end{pmatrix}^T \quad (12)
\]

where the stress tensor \( \mathcal{T} = (\mu + \mu_i) \left[ (\nabla \otimes \mathbf{u} + \mathbf{u} \otimes \nabla \mathbf{u}) - \frac{2}{3} \nabla \cdot \mathbf{u} \right] \) has for components:

\[
  \mathcal{T}_{11} = (\mu + \mu_i) \frac{2}{3} (2u_x - v_y - w_z), \quad \mathcal{T}_{12} = (\mu + \mu_i) (u_y + v_z), \quad \mathcal{T}_{13} = (\mu + \mu_i) (u_z + w_x), \quad \ldots
\]

or its modified version given by the Spalart-Allmaras QCR model, see Relation (10). \( \mathbf{F}^S(W) \) is the source terms fluxes, i.e., the Coriolis and centrifugal forces, and the diffusion, production and destruction terms from the Spalart-Allmaras turbulence model:

\[
  \mathbf{F}^S(W) = \begin{pmatrix} 0 \\
  -2(\Omega_x \rho w - \Omega_y \rho v) + \rho r_z \omega^2 \\
  -2(\Omega_x \rho u - \Omega_z \rho w) + \rho r_y \omega^2 \\
  -2(\Omega_y \rho v - \Omega_z \rho u) + \rho r_x \omega^2 \\
  \omega^2 \mathbf{r} \cdot (\rho \mathbf{u}) \\
  \omega^2 \mathbf{r} \cdot (\rho \mathbf{u}) + \rho c_{w1} \tilde{S} \tilde{v} + c_{w3} f_w \rho \left( \frac{\tilde{v}}{\tilde{S}} \right)^2 \end{pmatrix}. \quad (13)
\]

2.5. Turbomachinery cost functional

In this work, we will analyse five turbomachinery cost functions to evaluate the mesh convergence of the mesh adaptation process. These coefficients can also be used as functional of interest for the goal-oriented mesh adaptation. They will be differentiated in Section 5.1 to define the right-hand side of the adjoint state system. First, let us recall the expression of the the total pressure \( p_t \) and total temperature \( T_t \) :

\[
  p_t = p \left( 1 + \frac{1}{2} (\gamma - 1) M^2 \right)^{\frac{\gamma}{\gamma - 1}},
\]

\[
  T_t = T \left( 1 + \frac{1}{2} (\gamma - 1) M^2 \right).
\]

The mass flow is expressed as

\[
  \mathcal{D} = \int_{\Gamma} \rho \mathbf{u} \cdot \mathbf{n} \, d\Gamma = \sum_{F \in \Gamma} A_F \rho \mathbf{u}_F \cdot \mathbf{n}_F, \quad \text{(14)}
\]
where $|A_i|$ and $n_i$ are the area and the normal of the boundary face associated with $P_i$ (see Section 4.1 for the spatial discretization of the equations). From which, we can deduce "the pressure mass flow" (where $p$ is either the static pressure $p_s$ or the total pressure $p_t$):

$$D_p = \int_{\Gamma} p \rho u \cdot n \, d\Gamma = \sum_{P_i \in \Gamma} |A_i| p_i \rho_i u_i \cdot n_i,$$

and "the temperature mass flow" (where $T$ is either the static temperature $T_s$ or the total temperature $T_t$):

$$D_T = \int_{\Gamma} T \rho u \cdot n \, d\Gamma = \sum_{P_i \in \Gamma} |A_i| T_i \rho_i u_i \cdot n_i.$$

The total pressure ratio is expressed as

$$J_{p_t} = \frac{p_{t_{out}}}{p_{t_{in}}} \quad \text{where} \quad \mathcal{P}_t = \frac{D_{p_t}}{D} = \frac{\int_{\Gamma} p_1 \rho u \cdot n \, d\Gamma}{\int_{\Gamma} \rho u \cdot n \, d\Gamma} = \sum_{P_i \in \Gamma} \frac{|A_i| p_i \rho_i u_i \cdot n_i}{\sum_{P_i \in \Gamma} |A_i| \rho_i u_i \cdot n_i},$$

where integrals are either on the inlet surface $\Gamma_{in}$ or on the outlet surface $\Gamma_{out}$. The total temperature ratio is expressed as

$$J_{T_t} = \frac{T_{t_{out}}}{T_{t_{in}}} \quad \text{where} \quad \mathcal{T}_t = \frac{D_{T_t}}{D} = \frac{\int_{\Gamma} T_1 \rho u \cdot n \, d\Gamma}{\int_{\Gamma} \rho u \cdot n \, d\Gamma} = \sum_{P_i \in \Gamma} \frac{|A_i| T_i \rho_i u_i \cdot n_i}{\sum_{P_i \in \Gamma} |A_i| \rho_i u_i \cdot n_i},$$

where integrals are either on $\Gamma_{in}$ or on $\Gamma_{out}$. The isentropic efficiency is expressed as

$$\eta = \frac{J_{p_{t_{in}}} - 1}{J_{T_{t_{in}}} - 1},$$

and, finally, the loss coefficient is expressed as

$$\omega = \frac{q_{t_{out}} - q_{t_{in}}}{q_{t_{in}} - q_{t_{in}}}.$$

3. Anisotropic mesh adaptation algorithm with mesh-convergence analysis

Mesh adaptation is a non-linear problem where the couple formed by the mesh and the solution needs to be converged at the same time. Therefore an iterative process is required which is usually achieved by means of a mesh adaptation loop starting from an initial mesh $\mathcal{H}_0$, an initial solution $W_0$, an initial adjoint state $W_0^\ast$ if goal-oriented mesh adaptation is considered, and a given mesh complexity $C$ (the continuous counterpart of the mesh size, see Section 6).

At each step of the mesh adaptation loop, a metric tensor $M_i$ is computed from the triple ($\mathcal{H}_i$, $W_i$, $W_i^\ast$) and the given mesh complexity $C$, using the selected error estimate, see Section 6. Metric tensor field $M_i$ contains information on sizes and directions of the elements of the adapted mesh we seek. This information is then used by the remesher to generate a new adapted mesh $\mathcal{H}_{i+1}$ [44]. Then $W_i$ is interpolated on $\mathcal{H}_{i+1}$ to obtain $(W^0)_{i+1}$ which is then used as a restart solution for the next flow solution of the mesh adaptation loop [1]. In the case of goal-oriented mesh adaptation, the adjoint state $W_i^\ast$ can be also interpolated on the new mesh $\mathcal{H}_{i+1}$ to obtain $(W^{a,0})_{i+1}$ which is used as a restart for the next adjoint solution. Restart solutions are important to not waste time in the adaptive process and reuse at maximum the previous work done. This iterative process is depicted by the step 1 while loop in Algorithm 1.

The convergence criteria of step 1(f) is up to the expectations of the user, it specifies when the couple mesh/solution is considered as converged for the current complexity in the process. In this work, for turbomachinery applications, we consider that the solution is converged at the given complexity if the mass flow, the pressure ratio and temperature ratio are not varying by a given percentage $\epsilon$ on three consecutive iterations. Here, we choose $\epsilon = 0.01$, i.e., 1%.

In the context of a mesh convergence analysis this adaptation loop has to be repeated for several increasing mesh complexities $\{C_j\}$. An efficient strategy consists in converging the couple mesh/solution for a given complexity and reuse the final mesh, solution and adjoint state to initialize the next computations at an increased mesh complexity. Such a process enables a multiscale resolution of the flow by solving large scale features on coarse adapted meshes (at the smallest complexities) and the fine scale
Algorithm 1 General mesh adaptation algorithm with mesh-convergence analysis

Initial mesh \( \mathcal{H}_0 \), solution \( W_0 \), adjoint \( W^\ast_0 \), and complexity \( C^0 \)

//--- Outer loop to perform the convergence study

while \( C^j \leq C^{\text{max}} \) do

//--- Inner loop to converge the mesh adaptation at fixed complexity

1. while \( i \leq n_{\text{adap}} \) do

   (a) Compute optimal metric for the considered error estimate and complexity \( \Rightarrow \mathcal{M}^i_{j-1} \)
   (b) Generate new adapted mesh \( \Rightarrow \mathcal{H}^i_j \)
   (c) Interpolate primal and adjoint states on the new mesh \( \Rightarrow (W^i_0)_j \) and \( (W^\ast_0)_j \)
   (d) Compute primal state \( \Rightarrow W^i_j \)
   (e) Compute adjoint state \( \Rightarrow (W^\ast)_{j}^i \)
   (f) if (convergence check) then
       \( i = n_{\text{adap}} + 1 \)
   else
       \( i = i + 1 \)
   fi

done

2. \( \mathcal{H}^{i+1}_{n_{\text{adap}}} = \mathcal{H}^i_{n_{\text{adap}}} ; \ W^{i+1}_0 = W^i_{n_{\text{adap}}} ; \ (W^\ast)_{0}^{i+1} = (W^\ast)_{n_{\text{adap}}}^i ; \ C^{i+1} = \alpha \cdot C^i \)

done

features of the flow on fine adapted meshes (at the largest complexities). This acts like a "multigrid effect" and enables faster convergence on fine adapted meshes. This process is represented by the outer while loop in Algorithm 1. At each outer loop iteration, the complexity is increased by a factor \( \alpha \). In this work, we have set \( \alpha = 2 \) to multiply the mesh size by a factor 2 when increasing the complexity. We have found that it is very advantageous to converge on the smallest complexities because of lot of work is done in converging the solution and these iterations are inexpensive in comparison to the largest complexities.

Algorithm 1 can be used as it in most of the cases. In this work, it has been used for all LS89 high pressure axial turbine vane computations (see Section 8.1). For the NASA Rotor 37 compressor (see Section 8.2), we simulate the whole characteristic until the compressor stall by increasing the applied outlet/inlet pressure ratio. For high mass flow rate - i.e., up to 98% of the choke mass flow - we can also use it as it. In such cases, each pressure ratio configuration can be run independently in parallel which is very attractive.

However, when a rotor’s full characteristics is studied, a new difficulty occurs when we increase the pressure ratio in the flow solver boundary conditions, in other words, when we simulate low mass flow rate functioning point (below 98% of the choke mass flow). For these cases, this algorithm cannot be applied directly. Indeed, as one can see in the results section, for a given pressure ratio boundary condition prescription, the simulated operating point change with the mesh size until mesh convergence is reached. For the NASA Rotor 37 compressor, the larger the mesh size the higher the mass flow rate. We immediately see that the smaller the mesh size, the earlier the stall occurs for the pressure ratio boundary conditions. We are not thus able to run high pressure ratio prescription on the coarser adapted meshes (lower complexity). To deal with such cases, at a given complexity, we have to start from the previous pressure ratio to compute a higher boundary conditions pressure ratio until stall occurs. In this case, the global algorithm is modified as described in Algorithm 2 where \( \{\mathcal{P}^k\}_k \) are the prescribed flow solver boundary conditions pressure ratio. This time each complexity can be run independently in parallel but not each pressure ratio prescription which is less efficient. The modified initialization using a previous ratio final mesh, solution and adjoint state is done at step 1(b).
Algorithm 2 General mesh adaptation algorithm with mesh convergence analysis and pressure ratio study

//--- Outer loop to perform the convergence study
while \( C^j \leq C_{\text{max}} \) do

Initial mesh \( \mathcal{H}_0 \), solution \( W_0 \), adjoint \( W^\ast_0 \), and pressure ratio \( p^0 \); set complexity \( C^j \)

//--- Second loop to compute all the pressure ratio
1. while \( p^k \leq p^{\text{max}} \) do

//--- Inner loop to converge the mesh adaptation at fixed complexity
(a) while \( i \leq n_{\text{adap}} \) do

i. Compute optimal metric for the considered error estimate and complexity \( = \mathcal{M}_{i-1} \)
ii. Generate new adapted mesh \( = \mathcal{H}_i \)
iii. Interpolate primal and adjoint states on the new mesh \( (W^0)_i \) and \( (W^\ast)_i \)
iv. Compute primal state \( = W_i \)
v. Compute adjoint state \( = W^\ast_i \)
vi. if (convergence check) then

\( i = n_{\text{adap}} + 1 \)
else
\( i = i + 1 \)
fi
done

(b) if (stall occurs) then

break
else

\( \mathcal{H}_{i+1} = \mathcal{H}_{n_{\text{adap}}} \); \( W_{i+1} = W_{n_{\text{adap}}} \); \( (W^0)_{i+1} = (W^0)_{n_{\text{adap}}} \);
set \( p^{k+1} \)
fi
done

2. \( C^{j+1} = \alpha \cdot C^j \)
done

4. RANS flow solver in rotating frame

Reynolds-Averaged Navier-Stokes (RANS) numerical simulations are considered with the Spalart-Allmaras turbulence model. \textsc{Wolf} is a vertex-centered (flow variables are stored at vertices of the mesh) mixed Finite Volume - Finite Element Navier-Stokes solver on unstructured meshes composed of triangles in 2D and tetrahedra in 3D [4, 19, 21, 46]. The time integration considers an implicit temporal discretization. At each time step, the linear system of equations is approximately solved using a Symmetric Gauss-Seidel (SGS) implicit solver, and local time stepping and local CFL to accelerate the convergence to steady state. For the turbulence model, the Spalart-Allmaras is loosely-coupled to the mean-flow equations, where the mean-flow and turbulence model equations are relaxed in an alternating sequence. This choice for the flow solver is a major difference with what is encountered in the community as most of the flow solver are cell-centered on multi-block structured meshes composed of hexahedra [7, 8, 12, 13, 22, 27, 32, 61].

The spatial and temporal discretization of \textsc{Wolf} has been thoroughly detailed in [2]. In the following, we will recall the main lines of the numerical scheme and we will detail specific features for turbomachinery such as rotating frame source terms, boundary conditions, and how periodicity is taken into account.

4.1. Spatial discretization

The spatial discretization of the fluid Equations (1) and (8) is based on an hybrid Finite Volume - Finite Element formulation on unstructured meshes. The convective terms are solved by a second order Finite Volume method on the dual mesh composed of median cells. The viscous terms are solved by the \( P^1 \) Galerkin Finite Element Method (FEM) which provides second order accuracy.
Let \( \mathcal{H} \) be a mesh of domain \( \Omega \), the vertex-centered Finite Volume formulation consists in associating with each vertex \( P_i \) of the mesh a control volume or finite volume cell, denoted \( C_i \). Discretized domain \( \Omega_b \) can be written as the union of the elements or the union of the finite volume cells:

\[
\Omega_b = \bigcup_{i=1}^{N_K} K_i = \bigcup_{i=1}^{N_V} C_i,
\]

where \( N_K \) is the number of elements and \( N_V \) the number of vertices. Note that the dual mesh (composed of cells) is built in a preprocessing step. Consequently, only a simplicial mesh is needed in the input. Several choices are possible to build finite volume cells. In this work, the median cells are considered. The equations are integrated on each cell \( C_i \) (using the Green formula):

\[
|C_i| \frac{dW_i}{dt} + \mathbf{F}_i = \mathbf{S}_i + \mathbf{Q}_i + \mathbf{\Gamma}_i,
\]

where \( W_i \) is the mean value of the solution \( W \) on cell \( C_i \), \( \mathbf{F}_i \), \( \mathbf{S}_i \), \( \mathbf{Q}_i \), and \( \mathbf{\Gamma}_i \) are respectively the numerical convective, viscous, source flux and boundary terms:

\[
\mathbf{F}_i = \int_{\partial C_i} F(W_i) \cdot \mathbf{n}_i \, dy,
\]

\[
\mathbf{S}_i = \int_{\partial C_i} S(W_i) \cdot \mathbf{n}_i \, dy,
\]

\[
\mathbf{Q}_i = \int_{\Omega_i} Q(W_i) \, d\Omega,
\]

\[
\mathbf{\Gamma}_i = \int_{\partial \Omega_i \cap \partial \Omega} G(W_i) \, dy,
\]

where \( \mathbf{n}_i \) is the outer normal to the cell surface \( \partial C_i \), and \( F \), \( S \) and \( Q \) are respectively the convective, viscous and source terms flux functions as defined previously in Relations (11), (12) and (13). \( G \) holds for the boundary flux function specified to impose the desired boundary conditions.

### 4.1.1. Convective fluxes discretization

The integration of the convective fluxes \( \mathbf{F} \) of Equation (15) is done by decomposing the cell boundary \( \partial C_i \) into many facets. These facets are grouped and associated to the edges joining vertex \( P_i \) to its neighbors. The facets associated to each edge are averaged to yield a single mean face per edge:

\[
\mathbf{F}_i = \int_{\partial C_i} F(W_i) \cdot \mathbf{n}_i \, dy = \sum_{p \in \mathcal{V}(P_i)} F_{|\partial C_{ij}} \cdot \int_{\partial C_{ij}} \mathbf{n}_i \, dy,
\]

where \( \mathcal{V}(P_i) \) is the set of all neighboring vertices linked by an edge to \( P_i \) and \( F_{|\partial C_{ij}} \) represents the constant value of \( F(W) \) at interface \( \partial C_{ij} \). We notice that the computation of the convective fluxes is performed mono-dimensionally in the direction normal to the boundary of the finite volume cell. Therefore, the numerical calculation of the flux function \( \Phi_{ij} \) at the interface \( \partial C_{ij} \) is achieved by the resolution of a one-dimensional Riemann problem in the direction of the normal \( \mathbf{n}_j \) by means of an approximate Riemann solver. Here, we consider HLLC approximate Riemann solver proposed by Batten and al. [10]:

\[
\Phi_{ij}^{\text{convecive}} = \Phi_{ij}^{\text{HLLC}}(W_i, W_j, \mathbf{n}_j) = F_{|\partial C_{ij}} \cdot \int_{\partial C_{ij}} \mathbf{n}_i \, dy \quad \text{where} \quad \mathbf{n}_j = \int_{\partial C_{ij}} \mathbf{n}_i \, dy.
\]

**Second order accurate version.** The MUSCL type reconstruction method has been designed to increase the order of accuracy of the scheme [35]. The idea is to use extrapolated values \( W_{ij} \) and \( W_{ji} \) instead of \( W_i \) and \( W_j \) at the interface \( \partial C_{ij} \) to evaluate the flux with the approximate Riemann solver. Note that, in the implementation, the primitive variables \( \rho, \mathbf{u}, p \) are extrapolated to guarantee the positivity of the density and the pressure, then the conservative variables are reconstructed from these values. Thus, the gradients of the primitive variables are evaluated. However, in the following, to simplify the notation, we still denote by \( W \) the primitive variables vector. The numerical flux becomes:

\[
\Phi_{ij}^{\text{convecive}} = \Phi_{ij}^{\text{HLLC}}(W_{ij}, W_{ji}, \mathbf{n}_{ij}),
\]

where \( W_{ij} \) and \( W_{ji} \) are linearly extrapolated as:

\[
W_{ij} = W_i + \frac{1}{2} (\nabla W)_i \cdot \frac{P_i - P_j}{P_j} \quad \text{and} \quad W_{ji} = W_j + \frac{1}{2} (\nabla W)_j \cdot \frac{P_i - P_j}{P_i}.
\]

In contrast to the original MUSCL approach and most of the vertex-centered finite volume scheme, the approximate "slopes" \( (\nabla W)_{ij} \) and \( (\nabla W)_{ji} \) are defined for each edge. In other words, we can qualify this numerical scheme as a multi-slope finite-volume scheme unlike classical single-slope finite-volume schemes, which results in a more accurate scheme because it uses more directional information. This scheme is obtained using a combination of centered and upwind gradients in order to build low dissipation second order numerical scheme [33] that are called \( \beta \)-scheme. Even lower dissipation schemes can be obtained using also nodal gradients [21].
4.1.3. Source terms discretization

with the FEM:

\[ \phi_{\text{source}} = \int_{C_i} \left( \mathcal{Q}_{\text{Coriolis}}(W_i) + \mathcal{Q}_{\text{centrifugal}}(W_i) \right) \, d\Omega = |C_i| \begin{pmatrix} 0 & -2(\Omega \times (\rho \mathbf{u}))_i + \rho_i r_i \omega_i^2 \\ \omega_i^2 \mathbf{r} \cdot (\rho \mathbf{u})_i \\ 0 \end{pmatrix} \right) \]

with \( \omega = |\Omega| \). (17)

**Limiter function.** MUSCL schemes are not monotone and can be a source of spurious oscillations especially in the vicinity of discontinuities [19]. These oscillations can affect the accuracy of the final solution or simply end the computation because (for instance) of negative pressures. A widely used technique for addressing this issue is to guarantee the TVD property in 1D [28] or the LED property in 2D/3D of the scheme, which ensures that the extrapolated values \( W_{ij} \) and \( W_{ij}^{\text{lim}} \) are not invalid. To guarantee the TVD or the LED properties, limiting functions are coupled with the previous high-order gradient evaluations. The gradient is substituted by a limited gradient denoted \( \nabla W_{ij}^{\text{lim}} \). The choice of the limiting function is crucial as it directly affects the numerical dissipation of the scheme and the convergence of the simulation.

\( \beta \)-scheme requires specific limiter functions with three entries because we have at hand three different gradient: the centered, the upwind and the low dissipation gradients. Piperno et al. have extended the Van Albada limiter to \( \beta \)-scheme [56, 55]. The superbee limiter has been extended by Koren-Dervieux in [33, 19]. In [2], a new limiter has been proposed which is as smooth as the Piperno limiter and as low dissipative as the Koren-Dervieux limiter.

**Discretization of the Spalart-Allmaras convection term.** The convective term of Equation (9) can be discretized using a classical linear advection of the turbulent variable \( \rho \tilde{v} \). But, here we consider a nonlinear approach that has been proposed in [34]:

\[ \Phi_{\rho \text{convective}}(W_{ij}, W_{ij}, n_{ij}) = \Phi_{\rho \text{convective}}(W_{ij}, W_{ij}, n_{ij}) \ \begin{cases} \tilde{v}_i & \text{if } \Phi_{\rho \text{convective}}(W_{ij}, W_{ij}, n_{ij}) > 0 \\ \tilde{v}_j & \text{otherwise} \end{cases} \] (16)

where the flux \( \Phi_{\rho \text{convective}} \) of the density variable is computed with the HLLC approximate Riemann solver. It has been proven that this scheme preserve the maximum principle for the convected turbulent variable.

4.1.2. Viscous terms discretization

The viscous terms are discretized using the \( P_1 \) Finite Element method (FEM) which is second order accurate. We need to evaluate

\[ S_i = \sum_{p \in \mathcal{P}(P_1)} \int_{\partial C_{ij}} S_i(W) \cdot n \, d\gamma, \]

where \( \partial C_{ij} \) is the common interface between cells \( C_i \) and \( C_j \). Let \( \varphi_i \) be the \( P_1 \) Finite Element basis function associated with vertex \( P_i \), we have: \( \int_K \nabla \varphi_i \cdot d\Omega = - \int_{\partial C_{ij} \cap K} n \, d\gamma \), which states the equivalence between the Finite Element and the Finite Volume for linear solutions. As the solution is represented on the \( P_1 \) basis, \( S_i(W) \) (which comes from a gradient) is assumed constant by parts on each element \( K \). Then, we obtain

\[ S_i = \sum_{p \in \mathcal{P}(P_1)} \int_{\partial C_{ij}} S_i(W) \cdot n \, d\gamma = \sum_{K \cap \partial C_{ij}} S_i(W)|_{K} \cdot \int_{\partial C_{ij} \cap K} n \, d\gamma = - \sum_{K \cap \partial C_{ij}} \int_{K} S_i(W)|_{K} \cdot \nabla \varphi_i \, d\Omega. \]

In practice, as the nodal viscous term is decomposed as the sum of the contribution of each element containing this node, we cycle on the elements instead of the vertices to assemble them. Given an element \( K = (P_i, P_j, P_k, P_l) \) we have the partial flux associated with each vertex

\[ \Phi_{\text{viscous}}|_{k}(W_r, W_j, W_k, W_l) = \int_{\partial C_{ij} \cap K} S_i(W)|_{k} \cdot n \, d\gamma = - \int_{K} S_i(W)|_{K} \cdot \nabla \varphi_i \, d\Omega. \]

If the QCR version of the Spalart-Allmaras turbulence model, then the stress tensor \( \mathcal{T} \) should be modified accordingly, see Equation (10).

**Discretization of the Spalart-Allmaras dissipation term.** The Spalart-Allmaras dissipation term of Equation (9) is also discretized with the FEM:

\[ \Phi_{\text{viscous}}|_{k}(W_r, W_j, W_k, W_l) = |K| \frac{1}{\sigma^2 \rho_k} \left( \nu_k |v_k| + \rho_k \right) \nabla v_k \cdot \nabla - \mathbf{r} \cdot (\rho_k |v_k|). \]

4.1.3. Source terms discretization

In the case where we solve the equations in the rotating frame, we add the Coriolis and centrifugal forces given by Equation (13) as source terms by integration on each vertex cell :

\[ \Phi_{\text{source}} = \int_{C_i} \left( \mathcal{Q}_{\text{Coriolis}}(W_i) + \mathcal{Q}_{\text{centrifugal}}(W_i) \right) \, d\Omega = |C_i| \begin{pmatrix} 0 & -2(\Omega \times (\rho \mathbf{u}))_i + \rho_i r_i \omega_i^2 \\ \omega_i^2 \mathbf{r} \cdot (\rho \mathbf{u})_i \\ 0 \end{pmatrix} \]

with \( \omega = |\Omega| \). (17)
Discretization of the Spalart-Allmaras source term. The Spalart-Allmaras source terms (diffusion, production and destruction) are discretized by simple integration on each vertex cell:

$$\Phi_{\text{source}}(p_i) = \int_{C_i} Q \rho_i (W_i) d\Omega = |C_i| Q \rho_i (W_i),$$

where $|C_i|$ is the volume of cell associated to vertex $P_i$ and the source term $Q \rho_i (W_i)$ is given by the last line of the source term vector of Relation (13).

4.1.4. Boundary conditions discretization

Boundary conditions are imposed boundary element-wise, e.g. a boundary flux is computed for each boundary edge/face and assembled to each of its vertices. Consequently, even if a vertex-center scheme is considered, consistent boundary conditions are obtained because the type of conditions can be multiple around a vertex. For turbomachinery computations, three different boundary conditions are involved: no-slip boundary condition for walls, subsonic inlet and subsonic outlet boundary conditions.

For the subsonic inlet and outlet boundary conditions, we consider an approach similar to the one proposed in [15].

No-slip boundary condition. For no-slip boundary conditions, a null velocity $u = 0$ and a turbulent variable equal to $\tilde{y} = 0$ are strongly enforced at each iteration. Consistently, we impose $\Phi_\rho = 0$, $\Phi_{\rho u} = 0$, and $\Phi_\rho = 0$ at the boundary. The energy flux is fixed according to the desired temperature behavior: for an adiabatic wall it is null and for an isothermal wall the energy variable is enforced similarly to the velocity.

In the case of rotating machine, the Navier-Stokes equations are solved in the rotating frame. Therefore, the moving walls are considered fix and the fix wall are moving with as imposed velocity minus the rotating velocity: $u = -\Omega \times r$ (see Section 2.2) which is again enforced strongly.

Subsonic Inlet. Subsonic inflow boundary condition is used to prescribe a consistent physical incoming flow with a given total pressure $p_{\text{ext}}$, total temperature $T_{\text{ext}}$, and flow direction $n$ which is the normal of the considered boundary face. The variables $p$, $R$ and $H$ are extrapolated, and we update $p$ and $u$. In order to provide a numerically and physically consistent boundary condition, we rely on Riemann invariants across the boundary surface to compute an appropriate external state. The boundary flux is then computed with the HLLC approximate Riemann solver using the external state (denoted with the subscript $\text{ext}$) and the inner state.

In the subsonic case, sound wave travels upstream thus the negative Riemann invariant is used to compute a consistent external state. Given an imposed total pressure $p_{\text{ext}}$ and temperature $T_{\text{ext}}$, the inner state $W = (p, \rho u, \rho e)$ and the normal to the face $n$, we can compute the normal velocity $u_n = u \cdot n$, the boundary sound velocity $c = \sqrt{\gamma p}/\rho$, the outgoing characteristic that must be conserved

$$R^- = u_n - \frac{2c}{\gamma - 1},$$

and the ingoing enthalpy defined by our desired state

$$H_t = \frac{\gamma}{\gamma - 1} R_t, \quad c_{\text{set}} = \gamma R_t, T_{\text{set}}.$$  

These quantities are conserved across the boundary and are thus related to external state by:

$$H_t = \frac{c_{\text{ext}}^2}{\gamma - 1} + \frac{u_{\text{ext}}^2}{2} \quad \text{and} \quad R^- = u_{\text{ext}} - \frac{2c_{\text{ext}}}{\gamma - 1},$$

which forms a set of coupled equation. We solve them by replacing $u_{\text{ext}}$ or $c_{\text{ext}}$ in the first equation by its expression obtained from the second equation. We are left with a quadratic equation in $u_{\text{ext}}$ or $c_{\text{ext}}$ to solve that has two solutions. The physical root is the largest one (positive). We deduce the other one by using the $R^-$ relation. For instance, we set:

$$c_{\text{ext}} = \frac{\gamma - 1}{2} (u_{\text{ext}} - R^-),$$

we replace it in the $H_t$ relation and we end-up with the following quadratic equation in $u_{\text{ext}}$:

$$(1 + \frac{\gamma - 1}{2})u_{\text{ext}}^2 - (\gamma - 1)R^- u_{\text{ext}} + \frac{\gamma - 1}{2} (R^-)^2 - 2H_t = 0.$$  

The physical root $u_{\text{ext}}$ is the largest of the two solutions and $c_{\text{ext}}$ is deduced using the $R^-$ relation. External pressure and temperature are then deduced from isentropic relations:

$$\frac{p_{\text{ext}}}{p} = \left(1 + \frac{\gamma - 1}{2} M^2\right)^{\frac{\gamma}{\gamma - 1}} \quad \text{and} \quad \frac{p_{\text{ext}}}{p} = \left(\frac{T_{\text{ext}}}{T}\right)^{\frac{\gamma}{\gamma - 1}},$$
leading to
\[ p_{\text{ext}} = p_{\text{t,ref}} \left( 1 + \frac{\gamma - 1}{2} \frac{u_{\text{ext}}^2}{c_{\text{ext}}^2} \right)^{-\frac{\gamma}{\gamma - 1}}, \quad T_{\text{ext}} = T_{\text{t,ref}} \left( \frac{p_{\text{ext}}}{p_{\text{t,ref}}} \right)^{\frac{\gamma}{\gamma - 1}} = T_{\text{t,ref}} \left( 1 + \frac{\gamma - 1}{2} \frac{u_{\text{ext}}^2}{c_{\text{ext}}^2} \right)^{-1}, \]
and finally, external state reads
\[ W_{\text{ext}} = \left( \frac{p_{\text{ext}}}{R_{\text{t,ref}}}, u_{\text{ext}} n, p_{\text{ext}} \right). \]

Finally the boundary flux is given by:
\[ \Phi_{\text{outlet}} = \sum_{F \in P_i} \frac{1}{3} \Phi_{\text{HLLC}}(W_i, W_{\text{ext}}, n_F), \]
where \( \{F\}_j \) is the set of boundary faces containing vertex \( P_j \).

**Subsonic Outlet.** Subsonic outflow boundary condition is used to prescribe a consistent physical outgoing flow with a given pressure. As for the inflow boundary conditions we rely on Riemann invariants. In the subsonic case, outflow pressure is imposed by upwind going sound waves. Downstream traveling Riemann invariant and entropy:
\[ R^+ = \frac{2c}{\gamma - 1} + u \cdot n \quad \text{and} \quad s = \frac{p}{\rho^\gamma}, \]
are computed from inside the domain and assumed constant across the frontier. As we want to impose a given pressure \( p_{\text{ext}} = p_{\text{t,ref}} = \beta p_{\text{t,ref}} \), we can deduce
\[ p_{\text{ext}} = \left( \frac{p_{\text{t,ref}}}{s} \right)^{\frac{s}{\gamma}}, \quad c_{\text{ext}} = \sqrt{\gamma p_{\text{ext}}/p_{\text{ext}}}, \]
and the normal velocity from the Riemann invariant
\[ u_{\text{ext}} = R^+ = \frac{2c_{\text{ext}}}{\gamma - 1}. \]

As tangential velocity is advected by the flow and assumed constant, external state is defined as
\[ W_{\text{ext}} = (p_{\text{ext}}, u_{\text{ext}} n + u_t, p_{\text{ext}}), \]
with \( u_t = u - (u \cdot n) n \). The boundary flux is again given by:
\[ \Phi_{\text{outlet}} = \sum_{F \in P_i} \frac{1}{3} \Phi_{\text{HLLC}}(W_i, W_{\text{ext}}, n_F), \]
where \( \{F\}_j \) is the set of boundary faces containing vertex \( P_j \).

**Taking into account rotating frame.** Subsonic inlet and outlet boundary conditions are defined in the reference frame, boundary velocity must thus be transformed into the reference frame in order to compute the new boundary velocity which is then transformed back into the rotating frame.

### 4.2. Implicit time integration

There are two iterative strategies to approach a solution of a nonlinear system: i) the Newton method and ii) pseudo-transient continuation method. The Newton method does not use pseudo-time terms (i.e., the mass matrix), which is equivalent to using an infinite CFL number. It makes it very difficult to use for ill-conditioned linear system. The pseudo-transient continuation method using pseudo-time marching is assumed to be more robust, because it increases the diagonal dominance of the matrix. A nice discussion about these methods is given in [49]. Because we deal with complex 3D geometries and highly anisotropic adapted mesh, the convergence to steady state is achieved using the pseudo-transient continuation method with an implicit time integration. The implicit temporal discretization considers the backward Euler time-integration scheme as high-order accuracy in time is not required for steady flows. At each time step, the linear system of equations is approximately solved using a Symmetric Gauss-Seidel (SGS) implicit solver, and local time stepping and local CFL to accelerate the convergence to steady state. From our experience, we have made the following - crucial - choices to solve the compressible Navier-Stokes equations:

- the SGS relaxation iterates until the residual of the linear system is reduced by one or two orders of magnitude
the choice of the renumbering also impacts strongly the convergence of the non-linear system. While Hilbert-type (space filling curve) renumbering is very efficient for cache misses and memory contention [3, 62], Breadth-first search renumbering proves to be more effective for the convergence of the implicit method and the overall efficiency

- we found very advantageous to exactly differentiate the convective terms, the viscous terms, the source terms and the boundary conditions. Exact differentiation greatly improves the flow solver convergence. The only (unfortunate) approximation is the MUSCL part which is not differentiated because it changes the matrix pattern

- to achieve high efficiency, automation and robustness in the resolution of the non-linear system of algebraic equations to steady-state, it is mandatory to have a clever strategy to specify the CFL. In WOLF, the CFL evolution depends on the non-linear convergence and on the evolution of the solution at each time step. Like this there is no adhoc CFL law prescribed by the user, the evolution of the CFL is automatically managed by the flow solver. Furthermore, we have found very advantageous to use a local CFL for each vertex. This approach is detailed in [2]. A somehow similar strategy has been developed in USM3D [49] (NASA).

4.2.1. Implicit system

Once the equations have been discretized in space, a set of ordinary differential equations in time is obtained. For an implicit time integration, the semi-discretized RANS system becomes:

\[
\frac{\partial C_i}{\partial t^n} \delta W_i = - \mathbf{F}_i^{n+1} + \mathbf{S}_i^{n+1} + \mathbf{Q}_i^{n+1} + \mathbf{\Gamma}_i^{n+1},
\]  

where \( \delta W_i = W_i^{n+1} - W_i^n \), \(|C_i|\) is the area/volume of the finite volume cell associated with vertex \( P_i \) and \( \delta t^n \) is the local time step at iteration \( n \) for vertex \( P_i \) given by:

\[
\delta t = \text{CFL} \frac{h^2}{h(c + |\mathbf{u}|)} \left( \frac{C_p}{\rho} \left( \frac{\mu}{Pr} + \frac{\mu_t}{Pr_t} \right) \right) = \text{CFL} \frac{h^2}{h(c + |\mathbf{u}|)} \left( \frac{\lambda + \lambda_t}{\rho} \right)
\]

where \( C_p \) is the specific heat at constant pressure, \( Pr \) and \( Pr_t \) are the laminar and turbulent Prandtl numbers, \( \mu \) and \( \mu_t \) are the laminar and turbulent dynamic viscosities and, \( \lambda \) and \( \lambda_t \) are the laminar and turbulent dynamic conductivity. In the above relation, \( h \) is the representative mesh size for vertex \( P_i \) which is taken as the smallest height of all the elements surrounding \( P_i \).

The implicit system is obtained by linearization of the semi-discretized RANS system with respect to the conservative variables \( W \). However, the computation of the Jacobian of the second order convective flux \( \frac{\partial \mathbf{F}_i^n}{\partial W_j} \) introduces an extra-difficulty as it involves the second order ball of vertex \( P_i \), as seen above, while the linearization of the other terms only involve the first order ball. This will enlarge drastically the pattern of the matrix\(^1\) which will not be edge-based anymore. It will lead to a large memory and CPU overhead. For all these reasons, it is approximated by the Jacobian of the first order convective flux \( \frac{\partial \mathbf{F}_i^n}{\partial W_j} \) which only depends on the first order ball (the other Jacobian terms are kept unchanged). After linearization of the RHS, it becomes:

\[
\left( \frac{|C_i|}{\delta t^n} L \mathbf{I} + \frac{\partial \mathbf{F}_i^n}{\partial W_i} - \frac{\partial \mathbf{S}_i^n}{\partial W_i} - \frac{\partial \mathbf{Q}_i^n}{\partial W_i} - \frac{\partial \mathbf{\Gamma}_i^n}{\partial W_i} \right) \delta W_i + \sum_{j \in V(i)} \left( \frac{\partial \mathbf{F}_j^n}{\partial W_j} - \frac{\partial \mathbf{S}_j^n}{\partial W_j} - \frac{\partial \mathbf{Q}_j^n}{\partial W_j} - \frac{\partial \mathbf{\Gamma}_j^n}{\partial W_j} \right) \delta W_j = - \mathbf{F}_i^n + \mathbf{S}_i^n + \mathbf{\Gamma}_i^n.
\]

where \( P_j \in V(i) \) is the set of vertices connected to vertex \( P_i \) by an edge. As the RHS still consider the second order convective flux, this acts as an approximated Jacobian but it is independent of the residual and thus does not affect the spatial order of the scheme. The first term of the LHS contributes to the diagonal of the matrix and the second term of the LHS (i.e., the sum) contributes to extra-diagonal terms on line \( i \) of the matrix. This linearized system can be written in vector form:

\[
\mathbf{A}^n \delta \mathbf{W}^n = \mathbf{R}^n \quad \text{with} \quad \mathbf{A}^n = \frac{|C_i|}{\delta t^n} \mathbf{I} - \frac{\partial \mathbf{R}^n}{\partial \mathbf{W}} \quad \text{and} \quad \delta \mathbf{W}^n = \mathbf{W}^{n+1} - \mathbf{W}^n,
\]

where \( \mathbf{R}^n = -\mathbf{F}^n + \mathbf{S}^n + \mathbf{Q}^n + \mathbf{\Gamma}^n \) and \( \mathbf{\tilde{R}}^n = -\mathbf{\tilde{F}}^n + \mathbf{S}^n + \mathbf{Q}^n + \mathbf{\Gamma}^n \).

When solving the RANS equations, the following choice has been made. In the linear system, the Spalart-Allmaras is loosely-coupled to the mean-flow equations, where the mean-flow and turbulence model equations are relaxed in an alternating sequence.

\(^1\)In 3D, instead of an average of twenty entries per matrix line, it will increase to an average of a hundred.
In other words, at each solver iteration, we are solving independently two linear systems, one for the mean flow where each entry is a block of size $5 \times 5$ in 3D and one for the turbulence model where each entry is a single value when considering the Spalart-Allmaras model. Because of this choice, we have chosen to differentiate the turbulent system with respect to $\tilde{v}$ and not with respect to $\rho \tilde{v}$ according to Equation (9). Thus the turbulent linear system is computing $\delta \tilde{v}^\rho$ and the mass matrix diagonal term is multiplied by $\rho^\rho$ (in other word we consider $\rho$ constant in the turbulent system).

The computation of exact Jacobians for the convective, viscous terms, and turbulent terms have been given in [2]. Here, we only give the differentiation of the rotating frame source terms and the turbomachinery boundary conditions.

### 4.2.2. Rotating frame source terms Jacobians

Taking into account rotating frames leads to add a source term given by Relations (5), (6) and (7) which are discretized as in Relation (17). For the implicit time integration, it is differentiated as follow:

$$
\frac{\partial \Phi_{\text{source}}}{\partial \tilde{W}} |_i = \left[ \begin{array}{cccc}
0 & 0 & 0 & 0 \\
\Omega_z & 0 & -\Omega_y & 0 \\
-\Omega_y & 2\Omega_x & 0 & 0 \\
\Omega_y & -2\Omega_x & 0 & 0 \\
0 & \Omega_z^2 & \Omega_x^2 & \Omega_y^2 \\
\end{array} \right],
$$

with $\Omega = (\Omega_x, \Omega_y, \Omega_z)$, $\omega = ||\Omega||$, and $\mathbf{r} = (r_x, r_y, r_z)$.

### 4.2.3. Boundary conditions Jacobian

Similarly to the other terms, the boundary conditions are exactly differentiated.

**No-slip boundary condition.** For no-slip boundary conditions, nothing is done for $\rho$ and $\rho E$ as there is no flux. Dirichlet (enforced variables) boundary conditions, such as the velocity in no-slip boundary conditions are enforced strongly in the matrix. To avoid renumbering and matrix reshaping, we do not separate imposed nodes from degrees of freedom. Instead, the nodal value of the variable is enforced, the corresponding line in the Jacobian is set to identity and the residual of the variable is set to zero so that the linear system looks like

$$
\text{with } \mathbf{0} = (\Omega_x, \Omega_y, \Omega_z), \quad \mathbf{r} = (r_x, r_y, r_z).
$$

**Inhomogeneous Dirichlet conditions.** The above method works well in the case we enforce $\mathbf{u} = 0$ on $\Gamma$ because we also have $\rho \mathbf{u} = 0$. But, having different Dirichlet boundary conditions (such as in the case of rotating machine) impacts the matrix on how to enforce strongly the boundary conditions.

In the case with only $\mathbf{u} = 0$ on $\Gamma$, for each vertex of $\Gamma$, the matrix lines for the momentum are set to 0, the diagonal to identity, and the RHS to 0. Showing only the diagonal block, for such vertices we will have:

$$
\begin{bmatrix}
* & * & * & * & * \\
* & * & * & * & * \\
* & * & * & * & * \\
0 & 0 & 0 & 0 & 0 \\
* & * & * & * & * \\
\end{bmatrix}
\begin{bmatrix}
\delta \mathbf{W} \\
\delta \mathbf{W}_m \\
\delta \mathbf{W}_n \\
\delta \mathbf{W}_o \\
\delta \mathbf{W}_p \\
\end{bmatrix}
= \begin{bmatrix}
* \\
0 \\
0 \\
0 \\
0 \\
0 \\
\end{bmatrix},
$$

which enforces the increments $\delta \rho \mathbf{u} = 0$, and thus the solution $\mathbf{u} = 0$.

In rotating frame, as we solve the equations in the relative frame, we have

$$
\mathbf{u} = 0 \text{ on rotating surfaces } \Gamma_{rot} \quad \text{and} \quad \mathbf{u} = -\Omega \times \mathbf{r} \text{ on fixed surfaces } \Gamma_{fix}.
$$

On the rotating surface $\Gamma_{rot}$, we do the same as above. For the fixed surface, we want:

$$
\mathbf{u}_{\text{fix}} = -\Omega \times \mathbf{r} \quad \Leftrightarrow \quad -\rho^{\rho+1} \mathbf{u}_{\text{fix}} + (\rho \mathbf{u})^{\rho+1} = 0.
$$

Let us write it in increments of the solution:

$$
-\rho^{\rho+1} \mathbf{u}_{\text{fix}} + (\rho \mathbf{u})^{\rho+1} + \delta (\rho \mathbf{u}) = 0
$$

$$
\Leftrightarrow \quad -\delta \rho \mathbf{u}_{\text{fix}} + \delta (\rho \mathbf{u}) = 0.
$$
because we have $\rho^u u_{xi} = (\rho^u)^6$ as it is enforce strictly. Again, showing only the diagonal block, this is enforced in the linear system by:

$$
\begin{bmatrix}
* & * & * & \delta \rho & \Phi_\rho \\
-u_{xi} & 1 & 0 & \delta \rho u & 0 \\
* & * & * & \delta \rho E & \Phi_{\rho E}
\end{bmatrix}
\begin{bmatrix}
* \\
* \\
* \\
* \\
* 
\end{bmatrix}
= 
\begin{bmatrix}
* \\
* \\
* \\
* \\
* 
\end{bmatrix}
+ 
\begin{bmatrix}
(\Omega \times \mathbf{r}) & 1 & 0 \\
* & * & * \\
* & * & * \\
* & * & * \\
* & * & *
\end{bmatrix}
\begin{bmatrix}
\delta \rho u \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
= 
\begin{bmatrix}
* \\
* \\
* \\
* \\
* 
\end{bmatrix}.
$$

This is an important difference that improves the convergence and the prediction of the flow solver (and also of the adjoint solver). We illustrate its impact on the NASA Rotor 37 case where the solution is computed on an initial uniform mesh with the pressure ratio of 1. Figure 2 (left) shows the convergence of the density residual with (in red) and without (in green) proper differentiation of the Dirichlet boundary condition. A non-exact differentiation will stall the convergence even if the limiter are frozen (after iteration 800). Similarly, we can see (right picture) the impact on the prediction of the pressure ratio where we observe oscillation and a decrease when limiter are frozen. Again, this emphasizes the fundamental importance of exactly differentiating the terms for the implicit matrix. No approximations are tolerated.

\[\text{Figure 2: Convergence of the residual (left) and the pressure ratio functional (right) with exact differentiation of the inhomogeneous Dirichlet boundary condition with respect to approximated one. The limiter is frozen after 800 iterations. We observe oscillations when an incorrect matrix is considered and a drop of the functional when the limiter is frozen.}\]

\[\text{Subsonic inlet and outlet.}\] The fluxes for the subsonic inlet and outlet has been detailed in Section 4.1.4. Both have exactly the same form and depends on the definition of an external state $W_{\text{ext}}$ which is defined from the interior state $W_i$, and the use of the HLLC approximate Riemann solver:

$$
\Phi_{i \text{out}} = \sum_{F \in F_i} \frac{|F|}{3} \Phi^{\text{HLLC}}(W_i, W_{\text{ext}}(W_i), n_F),
$$

Using the chain rule, the inlet/outlet boundary conditions are differentiated for each face as:

$$
\frac{\partial \Phi^{\text{HLLC}}(W_i, W_{\text{ext}}(W_i), n_F)}{\partial W_i} = \frac{\partial \Phi^{\text{HLLC}}}{\partial W_i} \frac{\partial W_i}{\partial W_i} + \frac{\partial \Phi^{\text{HLLC}}}{\partial W_{\text{ext}}} \frac{\partial W_{\text{ext}}}{\partial W_i} = \frac{\partial \Phi^{\text{HLLC}}}{\partial W_i} + \frac{\partial \Phi^{\text{HLLC}}}{\partial W_{\text{ext}}} \frac{\partial W_{\text{ext}}}{\partial W_i}.
$$

The differentiation of the approximate Riemann solver has been used above for the convective fluxes. It remains to evaluate the term $\frac{\partial W_{\text{ext}}}{\partial W_i}$ which is quite difficult, in particular because there is a resolution of a second order polynomial. This term is differentiated exactly by using automatic differentiation. We used TAPENADE from Inria which is freely available [29].

\[\text{4.2.4. Solving the linear system: SGS relaxation}\]

The linearized system obtained in the previous sections and given by Relation (19) is solved at each flow solver iteration. In practice, we ask the user to provide a maximal number of iterations $k_{\text{max}}$ (usually 20) and a targeted order of magnitude by which the relative residual of the system must be decreased (usually 0.01). The iteration is stopped when this targeted relative residual or the maximal iteration is reached. To solve the linear system, the considered method is the SGS relaxation (Symmetric Gauss-Seidel) based on the Lower-Upper Symmetric Gauss-Seidel (LU-SGS) implicit solver initially introduced by Jameson [30] and fully developed by Sharov et al. [62]. The SGS relaxation is very attractive because it uses an edge-based data structure which can be efficiently parallelized with p-threads [3, 62]. The linear system solving is detailed in [2] with the automatic local CFL law evolution.
4.3. Periodicity implementation

In the case of periodic flows, we assume that the solution flow has a given periodicity (i.e., it is invariant by a given geometrical transformation \( u(T(x)) = u(x) \)) due to periodic physical properties. We will consider translation and rotation periodicities. We can then restrain computation to (at least) a single period and emulate the rest of the domain with periodic boundary conditions. The domain is thus delimited by an arbitrary surface \( \Gamma_{\text{per}} \) and its periodic mapping \( \Gamma_{\text{per}}' = T(\Gamma_{\text{per}}) \). In the periodic mesh, surfaces \( \Gamma_{\text{per}} \) and \( \Gamma_{\text{per}}' \) are meshed identically, thus there is a one-to-one mapping between their vertices and faces. As the periodicity is enforced in the discretization so that each vertex on \( \Gamma_{\text{per}} \) has a linked vertex on \( \Gamma_{\text{per}}' \), there is no solution interpolation on the boundaries.

To enforce strongly the periodicity, we use ghost or virtual entities (vertices and elements). Indeed, vertices on periodic boundaries have initially a part of their geometric stencil (i.e., a part of their geometric entities ball) represented inside the mesh of the domain. The other part of the geometric stencil is on the other side of the domain around the linked vertices. As these vertices have to be treated like inner vertices, their geometric stencil are completed with ghost or virtual entities (vertices and elements) using the geometric stencil of their linked periodic vertices. Two choices in the implementation are possible:

- virtual entities are linked on existing entities on the other side of the domain using the linked vertex. This is memory efficient because no extra-entities are created, just a memory link is used. However, such an implementation impacts all the source code. Thus, each new implemented functionality should take care of the periodicity
- ghost entities are created inside the mesh. Then, there is a little memory overhead but in that case periodic vertices are computed similarly to inner vertices. This clearly facilitates the implementation and the impact of the periodicity on the source code, the periodicity is localized in a few number of functions.

In Wolf, we choose the second method. Therefore, the intial mesh (in red) - Figure 3 (left) - is completed in the pre-processing step before the beginning of the simulation with ghost entities (in yellow) as can be seen in Figure 3 (right).

Creating ghost entities. The creation of the ghost entities (vertices, elements, boundary faces) consists in duplicating one layer of elements on the other side of the periodic frontiers, and this for all periodic frontiers. To this end, any volume element which is connected to a periodic vertex (a vertex on a periodic surface) is duplicated and mapped with the periodic mapping (translation and/or rotation) on the other side of the domain. At the same time, ghost vertices are created using a tag array and the periodic vertex link array is updated.

Then, the more complex phase consists in updating the boundary faces. First, the original periodic faces are suppressed. Second, we have to create the new periodic faces composed only of ghost vertices and to create the new boundary faces (such as inlet and outlet faces) that can be composed of ghost and periodic vertices or only of ghost vertices. This should be done with care using hash table as complex situations can occur. It is important to note that no a priori (human) assumption can be made to create ghost entities as complex situations can occur, even more in 3D and even more with highly anisotropic meshes. Such incorrect hypothesis are generally founded on the vision of structured or isotropic meshes. For instance, a face composed only of ghost vertices can be in fact an internal face. In Figure 4 (which is just a 2D case), we illustrate that certain hypotheses which seem obvious to us a priori prove in fact immediately incorrect. For instance in 2D, there is not only one ghost outlet edge which is duplicated on both side of the domain as we can see in Figure 3 (top right). But, with the anisotropy of the mesh and in particular in the wake, many outlet boundary edges are created, see Figure 4 (right picture). Moreover, ghost boundary edges are not only composed of a periodic vertex and a ghost vertex but can be composed of two ghost vertices. Thus, a ghost edge composed of two ghost vertices can be either a ghost periodic edge or a ghost boundary edge or an internal edge. Another one is that two ghost vertices issued from two vertices on the outlet surface do not necessarily defined a ghost outlet edge. In the exemple Figure 4 (right picture), we observe that hole can exist in the ghost outlet surface depending on the connectivity of the mesh. These are just two examples and many other tricky configuration exists because we deal with unstructured meshes and highly anisotropic elements.

Updating data at ghost entities. Similarly to distributed memory parallelization (MPI) implementation, data at ghost entities are updated at key points in the algorithm to always have a correct value when needed. Scalar data are simply copied but vectors (velocity, gradients, ...) need to be rotated in the case of a periodicity by rotation. In Wolf, two reductions are done: one at the end of the fluxes computations and one after the solution under-relaxation. As one can see, the impact on the code is minimal. Note that edges connecting two ghost vertices are skipped to reduce CPU time overhead.

Implicit time advancing. The implicit scheme requires an appropriate definition of the matrix such that the system is not over-constraint. For sake of simplicity, we choose to not take into account directly the periodicity in the linear system. To minimize the impact on the source code, the periodicity is enforced through the ghost vertices:  

\footnote{The periodicity can be enforced directly by the periodic vertices but it requires in the matrix connectivity between vertices that are not mesh edges which is not compatible with our edge-based data structure where the matrix entries are given by the edges of the mesh plus the diagonal. Such a choice will have a lot more impact on the source code than the chosen approach.}

In WOLF, we choose the second method. Therefore, the initial mesh (in red) - Figure 3 (left) - is completed in the pre-processing step before the beginning of the simulation with ghost entities (in yellow) as can be seen in Figure 3 (right).
Figure 3: Periodic domains without (left) and with (right) ghost entities. Inner elements are colored in red and ghost elements are colored in yellow. Example on the 2D LS89 blade case with vertical translation periodicity (top) and on the 3D NASA Rotor 37 case with axial rotation periodicity (bottom).

Figure 4: Illustration of the complex topology of the mesh obtained when creating the ghost entities on the LS89 geometry with an adapted anisotropic mesh. In this case, we have a vertical translation periodicity. Left, view of the initial mesh without ghost entities in the top right corner where the periodic boundary surface (in red) and the outlet boundary surface (in black) merge. The future ghost entities on the other side of the domain will be the first layer of elements. Right, view of the new mesh with ghost entities (in yellow) in the bottom right corner where the periodic boundary surface (in red) and the outlet boundary surface (in black) merge. In black dotted line is represented the ghost outlet boundary surface.
• The matrix size is the initial number of vertices plus the ghost vertices. Ghost vertices are put at the end of the list.
• Ghost vertices are treated as Dirichlet points with an imposed variation of the solution. Thus, ghost vertices lines are set to zero everywhere except on the diagonal where we set the identity matrix.
• To reduce the CPU time overhead, we don’t solve the ghost vertices in the linear system, we just use their values during the SGS passes.
• After each pass (backward or forward), we update (copy) linear system solution ($\delta W$) from the inner vertices to ghost vertices to enforce the periodicity. The ghost vertices solutions are taken into account through the matrix column.

5. Adjoint solver

The error estimate in the goal-oriented mesh adaptation presented in Section 6 requires the computation of the numerical adjoint to obtain the sensitivity of the considered output functional $J$ to the flow. We denote by $R$ the nonlinear residual operator representing the system of equations and $W$ the exact solution such that $R(W) = 0$. The adjoint state $W^*$ is solution of the following system:

$$
\left( \frac{\partial R}{\partial W} \right)^T W^* = \left( \frac{\partial J}{\partial W} \right)^T.
$$

Numerically this consists in computing the sensitivity of a given functional $J(W)$ to the local nodal residual $R_i(W)$, through the linearization of the numerical problem:

$$
\frac{\partial J}{\partial R} = \frac{\partial J}{\partial W} \left( \frac{\partial R}{\partial W} \right)^{-1} = \left( \frac{\partial R}{\partial W} \right)^{-T} \cdot \frac{\partial J}{\partial W}.
$$

Therefore, the adjoint matrix is the transpose of the implicit matrix without the mass matrix, see Section 4.2. Additionally, in order to improve the differentiation of inviscid fluxes, we consider the extrapolated variables to compute the inviscid jacobians even if only the first order terms are differentiated. We are thus solving the following system:

$$
\left( \frac{\partial R}{\partial W} \right)^T W^* = \left( \frac{\partial J}{\partial W} \right)^T \iff A^* W^* = \left( \frac{\partial J}{\partial W} \right)^T. \quad (20)
$$

In the following sections, we provide the differentiation of several turbomachinery output functional that can be used to define the right-hand side (RHS) of the above linear system. Then, we give the choice made to solve the adjoint linear system. And finally, we describe how the periodicity is taken into account in the adjoint matrix.

5.1. Turbomachinery cost function for goal-oriented mesh adaptation

In Section 2.5, we have provided several output functionals used to evaluate a turbomachinery design. Each of these functionals can be used to define the RHS of the adjoint System (20). The differentiation of these functionals relies on the differentiation of the following elementary variables:

$$
\frac{\partial p}{\partial W} = (\gamma - 1) \left( \frac{||u||^2}{2}, -u, 1 \right)^T, \quad (21)
$$

$$
\frac{\partial T}{\partial W} = \frac{1}{\rho c_v} \left( ||u||^2 - E, -u, 1 \right)^T, \quad (22)
$$

$$
\frac{\partial M^2}{\partial W} = \left( -\frac{M^2}{\rho} \left( 1 + \frac{\gamma (\gamma - 1)}{2} M^2 \right), \frac{2u}{\gamma p} \left( 1 + \frac{\gamma (\gamma - 1)}{2} M^2 \right), -\frac{(\gamma - 1) M^2}{\rho} \right)^T. \quad (23)
$$

with $p = p_s$ the static pressure, $T = T_s$ the static temperature and $M$ the local Mach number. From these, we can obtain expressions of the total pressure $p_t$ and total temperature $T_t$ differentiation in function of the differentiation of $p$, $T$ and $M$. The total pressure and temperature read:

$$
p_t = p \left( 1 + \frac{1}{2} (\gamma - 1) M^2 \right)^{\frac{\gamma T}{\gamma - 1}} = p N^{\frac{T}{\gamma - 1}} \quad \text{and} \quad T_t = T \left( 1 + \frac{1}{2} (\gamma - 1) M^2 \right) = T N,
$$

where $N = \left( 1 + \frac{1}{2} (\gamma - 1) M^2 \right)^{\frac{\gamma - 1}{2}}$. The turbomachinery cost function can then be expressed in terms of the desired output functional $J$ as

$$
J = J(p_t, T_t, M).$$

The adjoint matrix $A^*$ allows to compute the sensitivity of $J$ to the control parameter $Y$ as

$$
\frac{\partial J}{\partial Y} = A^* \left( \frac{\partial J}{\partial W} \right) = A^* \left( \frac{\partial J}{\partial W} \right)^T.
$$

This is particularly useful for goal-oriented mesh adaptation where the adjoint $A^*$ leads to a system of equations that can be solved for the sensitivity of $J$ to the control parameter $Y$. The adjoint matrix $A^*$ is thus the key to the efficiency of the mesh adaptation process, allowing for a rapid computation of the sensitivity of the output functional to changes in the control parameter $Y$.
where we have denoted by $N$ the ratio between the total temperature and the static temperature: $N = \left(1 + \frac{1}{2}(\gamma - 1)M^2\right)$ which will simplify the derivative expressions. The total pressure is differentiated as

$$\frac{\partial p_t}{\partial W} = \frac{\partial p}{\partial W} \left(1 + \frac{1}{2}(\gamma - 1)M^2\right)^{\frac{\gamma}{\gamma - 1}} + p \left(\frac{\gamma}{\gamma - 1}\right) \frac{1}{2}(\gamma - 1)2M \frac{\partial M}{\partial W} \left(1 + \frac{1}{2}(\gamma - 1)M^2\right)^{\frac{\gamma - 1}{\gamma}}.$$

$$= N \frac{\partial p}{\partial W} + \gamma pMN \frac{\partial M}{\partial W}. \quad (24)$$

The total temperature is differentiated as

$$\frac{\partial T_t}{\partial W} = \frac{\partial T}{\partial W} \left(1 + \frac{1}{2}(\gamma - 1)M^2\right) + T \frac{1}{2}(\gamma - 1)2M \frac{\partial M}{\partial W} = N \frac{\partial T}{\partial W} + (\gamma - 1)TM \frac{\partial M}{\partial W}.$$

Now, we can give the differentiation of turbomachinery cost functions.

**Remark 5.1.** The error estimate proposed in Section 6 does not consider the turbulent variable $\tilde{v}$ in its formulation (it is implicitly taken into account via $\mu_i$), but if the error estimation involves the turbulent variable then it will require to solve the fully coupled system and to differentiate the cost functions w.r.t to $\tilde{v}$.

### 5.1.1. Mass flow

The mass flow is expressed as

$$\mathcal{D} = \int_{\Gamma} \rho \mathbf{u} \cdot \mathbf{n} \, d\Gamma = \sum_{P \in \Gamma} |A_i| \rho_i \mathbf{u}_i \cdot \mathbf{n}_i, \quad (25)$$

where $|A_i|$ and $\mathbf{n}_i$ are the area and the normal of the boundary face associated with $P_i$ from which we deduce

$$\frac{\partial \mathcal{D}}{\partial W_i} = |A_i| (0, \mathbf{n}_i, 0)^T.$$

If we look at the pressure mass flow (where $p$ is either the static pressure $p_i$ or the total pressure $p_t$):

$$\mathcal{D}_p = \int_{\Gamma} p \rho \mathbf{u} \cdot \mathbf{n} \, d\Gamma = \sum_{P \in \Gamma} |A_i| \rho \mathbf{u}_i \cdot \mathbf{n}_i,$$

from which we deduce

$$\frac{\partial \mathcal{D}_p}{\partial W_i} = |A_i| (\rho \mathbf{u} \cdot \mathbf{n}) \frac{\partial p}{\partial W_i} + |A_i| p (0, \mathbf{n}_i, 0)^T. \quad (26)$$

where the differentiation of $\frac{\partial p}{\partial W_i}$ can be either the static pressure given by Relation (21) or the total pressure given by Relation (24). And, if we look at the temperature mass flow (where $T$ is either the static temperature $T_i$ or the total temperature $T_t$):

$$\mathcal{D}_T = \int_{\Gamma} T \rho \mathbf{u} \cdot \mathbf{n} \, d\Gamma = \sum_{P \in \Gamma} |A_i| T \mathbf{u}_i \cdot \mathbf{n}_i,$$

from which we deduce

$$\frac{\partial \mathcal{D}_T}{\partial W_i} = |A_i| (\rho \mathbf{u} \cdot \mathbf{n}) \frac{\partial T}{\partial W_i} + |A_i| T (0, \mathbf{n}_i, 0)^T. \quad (27)$$

where the differentiation of $\frac{\partial T}{\partial W_i}$ can be either the static temperature given by Relation (22) or the total temperature given by Relation (25).

### 5.1.2. Pressure ratio

The pressure ratio is expressed as

$$J_{p_t} = \frac{p_{out}}{p_{in}} \quad \text{where} \quad p_{in} = \int_{\Gamma} p_i \rho \mathbf{u} \cdot \mathbf{n} \, d\Gamma = \sum_{P \in \Gamma} |A_i| \rho_i \mathbf{u}_i \cdot \mathbf{n} \quad \text{and} \quad \mathcal{D}_p = \int_{\Gamma} p \rho \mathbf{u} \cdot \mathbf{n} \, d\Gamma = \sum_{P \in \Gamma} |A_i| \rho \mathbf{u}_i \cdot \mathbf{n} = \frac{\partial \mathcal{D}_p}{\partial \mathcal{D}}.$$

20
where integrals are either on $\Gamma_{in}$ or on $\Gamma_{out}$. Using the above notations, let’s start with the differentiation of $P_i$:

$$\frac{\partial P_i}{\partial W_i} = \frac{\partial \mathcal{D}_P^i}{\partial W_i} - \mathcal{D}_P^i \frac{\partial \mathcal{D}_P^i}{\partial W_i},$$

with the differentiated terms are given by Relations (25) or (26). From this, we deduce:

$$\frac{\partial J_{P_i}}{\partial W_i} = \frac{\partial P_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial P_i} - \frac{\partial P_i^{\text{in}}}{\partial W_i} \frac{\partial T_i}{\partial P_i} = \frac{1}{(T_i^\text{in})^2} \left\{ -\frac{\partial P_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial P_i} \right\} \text{ if } P_i \in \Gamma_{in},$$

$$\frac{\partial J_{P_i}}{\partial W_i} = \frac{\partial P_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial P_i} - \frac{\partial P_i^{\text{in}}}{\partial W_i} \frac{\partial T_i}{\partial P_i} = \frac{1}{(T_i^\text{in})^2} \left\{ -\frac{\partial P_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial P_i} \right\} \text{ if } P_i \in \Gamma_{out}.$$

### 5.1.3 Temperature ratio

The temperature ratio is expressed as

$$J_{T_i} = \frac{T_i^{\text{out}}}{T_i^{\text{in}}} \quad \text{where} \quad T_i = \frac{\int_{\Gamma} T_i \rho \mathbf{u} \cdot \mathbf{n} \, d\Gamma}{\sum_{P_i \in \Gamma} \rho_i \mathbf{u}_i \cdot \mathbf{n}} = \frac{\mathcal{D}_{T_i}}{D}.$$

where integrals are either on $\Gamma_{in}$ or on $\Gamma_{out}$. Using the above notations, let’s start with the differentiation of $T_i$:

$$\frac{\partial T_i}{\partial W_i} = \frac{\partial \mathcal{D}_{T_i}}{\partial W_i} - \mathcal{D}_{T_i} \frac{\partial \mathcal{D}_{T_i}}{\partial W_i},$$

with the differentiated terms are given by Relations (25) or (27). From this, we deduce:

$$\frac{\partial J_{T_i}}{\partial W_i} = \frac{\partial T_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial T_i} - \frac{\partial T_i^{\text{in}}}{\partial W_i} \frac{\partial T_i}{\partial T_i} = \frac{1}{(T_i^\text{in})^2} \left\{ -\frac{\partial T_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial T_i} \right\} \text{ if } P_i \in \Gamma_{in},$$

$$\frac{\partial J_{T_i}}{\partial W_i} = \frac{\partial T_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial T_i} - \frac{\partial T_i^{\text{in}}}{\partial W_i} \frac{\partial T_i}{\partial T_i} = \frac{1}{(T_i^\text{in})^2} \left\{ -\frac{\partial T_i^{\text{out}}}{\partial W_i} \frac{\partial T_i}{\partial T_i} \right\} \text{ if } P_i \in \Gamma_{out}.$$

### 5.1.4 Isentropic efficiency

The isentropic efficiency is expressed as

$$\eta = \frac{J_{T_i}^{\text{isent}}} {J_{T_i}^{\text{out}}} - 1.$$

Using the above differentiations, for $\eta$ we get:

$$\frac{\partial \eta}{\partial W_i} = \frac{\gamma - 1}{\gamma} \frac{\partial J_{P_i}}{\partial W_i} \frac{J_{T_i}^{\text{isent}} - 1 - (J_{T_i}^{\text{isent}} - 1) \frac{\partial J_{T_i}}{\partial W_i}}{(J_{T_i}^{\text{out}} - 1)^2}.$$

### 5.1.5 Loss coefficient

The loss coefficient is expressed as

$$\omega = \frac{\phi^{\text{in}} - \phi^{\text{out}}}{\phi^{\text{in}} - \phi^{\text{out}}}.$$

Using the above differentiations, for $\omega$ we get:

$$\frac{\partial \omega}{\partial W_i} = \frac{(\frac{\partial \phi^{\text{in}}}{\partial W_i} - \frac{\partial \phi^{\text{out}}}{\partial W_i}) (\phi^{\text{in}} - \phi^{\text{in}}) - (\phi^{\text{out}} - \phi^{\text{out}}) (\frac{\partial \phi^{\text{in}}}{\partial W_i} - \frac{\partial \phi^{\text{out}}}{\partial W_i})}{(\phi^{\text{in}} - \phi^{\text{out}})^2} \left\{ -\frac{\partial \phi^{\text{in}}}{\partial W_i} \frac{\phi^{\text{in}} - \phi^{\text{in}}}{\phi^{\text{out}} - \phi^{\text{out}}} \right\} \text{ if } P_i \in \Gamma_{in},$$

$$\frac{\partial \omega}{\partial W_i} = \frac{(\frac{\partial \phi^{\text{in}}}{\partial W_i} - \frac{\partial \phi^{\text{out}}}{\partial W_i}) (\phi^{\text{in}} - \phi^{\text{in}}) - (\phi^{\text{out}} - \phi^{\text{out}}) (\frac{\partial \phi^{\text{in}}}{\partial W_i} - \frac{\partial \phi^{\text{out}}}{\partial W_i})}{(\phi^{\text{in}} - \phi^{\text{out}})^2} \left\{ -\frac{\partial \phi^{\text{in}}}{\partial W_i} \frac{\phi^{\text{in}} - \phi^{\text{in}}}{\phi^{\text{out}} - \phi^{\text{out}}} \right\} \text{ if } P_i \in \Gamma_{out}. $$
5.1.6. Dealing with rotating frame

In the case of rotating machines, we solve the equations of movement in the rotating frame (relative frame) instead of the absolute frame, see Section 2.2. But, the above turbomachinery output functionals should be evaluated in the absolute frame:

\[ j_A(W_A) = j_A(f(W_R)) = j_R(W_R), \]

where subscript \( A \) (resp. \( R \)) denotes function or variables in the absolute (resp. relative) frame. In other words, their expressions correspond to the functional \( j_A \). As the mean flow is evaluated in the relative frame, the adjoint state is also solved in the relative frame leading to:

\[ A^*_R W_R = \left( \frac{\partial j_A}{\partial W_R}(W_A) \right)^T. \]

But, in the above sections we have computed \( \frac{\partial j_A}{\partial W_A}(W_A) \). To get the correct right-hand side, we use the chain rule to differentiate the composite function:

\[ \frac{\partial j_A}{\partial W_R}(W_A) = \frac{\partial j_A}{\partial W_R}(f(W_R)) = \frac{\partial (j_A \circ f)}{\partial W_R}(W_R) = \frac{\partial j_A}{\partial (f(W_R))}(f(W_R)) \cdot \frac{\partial f}{\partial W_R}(W_R) = \frac{\partial j_A}{\partial W_A}(W_A) \cdot \frac{\partial f}{\partial W_R}(W_R). \]

Thus, the above cost function differentiation should be computed using the solution in the absolute frame and multiplied with the Jacobian of the transformation:

\[ \frac{\partial f}{\partial W_R}(W_R) = \begin{bmatrix} 1 & 0 & 0 \\ \Omega \times r & 1 & 0 \\ \frac{\|\Omega\times r\|^2}{2} & \Omega \times r & 1 \end{bmatrix}. \]

5.2. Boundary adjoint correction

As stated above, a formal description of the nodal value of the adjoint is the sensitivity of a given functional to a correction or perturbation of local nodal residual. In that respect, Dirichlet boundary conditions have a specific behavior. Indeed, as we impose a given field value at these nodes, their residual is discarded and plays no role in the computation. There is thus no dependency of the considered functional to these nodes and formally the adjoint should be zero at these boundaries. In practice, as the adjoint is computed with the transpose of the Jacobian matrix whose values have been corrected to impose the Dirichlet boundary conditions (see Section 4.2.3), the adjoint reflects the sensitivity of the functional to these boundary values. In consequence, the adjoint field is discontinuous at Dirichlet boundary conditions such as no-slip boundaries. However, in the context of error estimation for mesh adaptation, we assume the adjoint to be a continuous inner function. Therefore, at Dirichlet boundary conditions, we replace its value by an extrapolation from the interior of the domain using a quadratic least square fitting.

5.3. Solving the adjoint linear system

Goal-oriented error estimates rely on an accurate computation of the adjoint state that proves to be a stiff problem for RANS equations. Failing to converge the adjoint linear system to machine zero may impact negatively the adaptive process, eg. we observe noise in the adapted mesh. In our applications, we generally converge the linear system by twelve orders of magnitude to obtain an accurate adjoint state for mesh adaptation. But, this linear system is a lot more stiffer than the flow solver one because there is no mass matrix, i.e. it is equivalent to set a CFL equal to infinity. For inviscid flows, we didn’t see any issue in solving that linear system using a restarted GMRES preconditioned with LUSGS relaxation. However, we have observed that the adjoint linear system is a lot harder to converge for RANS. In fact, the finer is the adapted mesh the stiffer is the linear system.

At first, we have increased the Krylov space size but on highly-resolved anisotropic adapted meshes it was frequent that the linear system did not converge (eg. the residual was reduced by three to five orders of magnitude) after 10 000 iterations and a Krylov space size of 1 000 ! Such Krylov space sizes have stringent memory requirements which make them unusable in practice. The key to solve this issue was to consider a stronger preconditioner: a restarted GMRES preconditioned with SGS relaxation. For the preconditioner, we perform several passes of SGS (generally between 20 and 40 passes). In that case, we observe that the linear system is converging when considering Krylov space of size up to 200.

5.4. Periodic adjoint

We also have to take into account the periodicity in the adjoint system resolution. Like for the main flow solver (see Section 4.3), we are using ghost vertices and we rely on a direct modification of the matrix. The matrix size is the initial number of vertices plus the ghost vertices. Ghost vertices are put at the end of the list. First, the adjoint matrix is directly assembled that is to say the transposed block are transposed in the matrix at the assembling. Then, as ghost vertices are treated as Dirichlet points with an imposed variation of the solution, the ghost vertices lines are set to zero everywhere except on the diagonal where we set the identity matrix.
Similarly to the flow solver, we don’t solve the ghost vertices in the linear system, we just use their values during the GMRES and the SGS passes. After each pass (backward or forward) of SGS, we update (copy) linear system solution \(\delta \mathbf{W}\) from the inner vertices to ghost vertices to enforce the periodicity. The ghost vertices solutions are taken into account through the matrix column. Similarly, the ghost vertices values are updated throughout the GMRES.

6. RANS error estimates

In the mesh adaptation process, the metric field \((M(x))_{x \in \Omega}\) used to prescribe the new adapted mesh \(\mathcal{H}\) is automatically deduced from the actual solution or from the actual solution and adjoint state using different error estimates [5]. The goal is to find the optimal mesh \(\mathcal{H}_{Opt}\) which minimizes the given error model \(E\) for a fixed number of elements \(C(\mathcal{H}_{Opt}) = N\):

\[
\mathcal{H}_{Opt} = \arg \min_{C(\mathcal{H})=N} E(\mathcal{H}).
\]

This problem can be analytically solved by recasting it in the continuous mesh theoretical framework [39, 40]: the goal is to find the optimal continuous mesh \(M_{Opt}\) which minimizes the given continuous error model \(E\) for a fixed continuous mesh complexity \(C(M) = N\):

\[
M_{Opt} = \arg \min_{C(M)=N} E(M).
\]

The continuous mesh complexity is the continuous counterpart of the discrete mesh size (number of points or elements) and is used to prescribed the mesh size during the adaptation process. It is given by relation [38, 5]

\[
C(M) = \int_{\Omega} \sqrt{\det M} \, d\Omega.
\]

There is a direct relationship between the prescribed metric complexity and the number of elements of the generated mesh. If a unit mesh \(\mathcal{H} = \cup K_i\) (where the \(K_i\) are the tetrahedra of mesh \(\mathcal{H}\)) is generated with respect to \((M(x))_{x \in \Omega}\) [38, 5], then the continuous mesh complexity and the mesh size are linked by:

\[
C(M) \approx \sum_k \sqrt{\det M_k} |K_i| \approx \sum_k \frac{\sqrt{2}}{12} = \frac{\sqrt{2}}{12} \times nt,
\]

where \(|K|\) is the volume of \(K\), \(M_K\) is the average metric at element \(K\), and \(nt\) is the number of elements of the mesh. In this work, we consider feature-based error estimates based on a control of the interpolation error in \(L^p\)-norm and goal-oriented error estimate based on the \textit{a priori} analysis of [2, 41].

6.1. Feature-based error estimate

The most natural and straight forward approach is to control the interpolation error of a sensor field \(u = f(W)\) [16, 24, 38] which is defined from solution field \(W\). Given a continuous sensor \(u\), it is represented by its discrete nodal values on the mesh \(u_i = u(x_i)\) and its piecewise linear representation \(\Pi_h u\) on mesh \(\mathcal{H}\). The \(L^p\)-norm of the interpolation error of the sensor field \(u\) is stated as

\[
E_{L^p}(\mathcal{H}) = \left( \int_{\Omega} |u - \Pi_h u|^p \right)^{1/p}.
\]

Feature-based mesh adaptation generates discrete adapted meshes that minimizes the global interpolation error of the given sensor field \(u\) for the considered number of elements. Under certain assumptions, we can prove that this approach also controls the approximation error [43]. The analytical expression of the optimal continuous mesh \(M_{Opt}\) that minimizes the interpolation error in \(L^p\)-norm of sensor \(u\) for a given complexity \(N\) is [4]:

\[
M_{Opt}(x) = N^{\frac{1}{2}} \left( \int_{\Omega} |\det(\Pi_h(x))| \, d\Omega \right)^{-\frac{1}{2}} \det(\Pi_h(x))^{-\frac{1}{2p}} |\Pi_h(x)|,
\]

where \(d\) is the space dimension and \(H_u\) is the Hessian of the sensor \(u\) computed using a double \(L^2\)-projection method [18, 4]. Then, to generate the associated adapted mesh, a unit mesh is generated with respect to this metric field.

For inviscid flows, many studies have pointed out that controlling the \(L^2\)-norm of the interpolation error of the chosen sensor is the most appropriate choice [5, 42]. But recently, Park and Balan [53] have pointed out benefits of controlling the interpolation error in \(L^4\)-norm for the ONERA M6 RANS case instead of using the \(L^2\)-norm because this norm choice captures and adapts quicker the boundary layer region. Therefore, in this paper, we consider the \(L^4\)-norm of the interpolation error of the Mach sensor field as feature-based error estimates. Controlling the interpolation error of the Mach sensor field will tend to also control the interpolation error of all primitive variables.
6.2. Goal-oriented error estimate

Feature-based adaptation is efficient to control the interpolation error of a sensor field but it ignores the non-linear dependency of the solution \( W \) to the mesh and the considered system of equations. It improves the overall solution but ignores the ultimate goal of the simulation. In our case, we are interested in computing the main turbomachinery coefficients (mass flow, pressure ratio, temperature ratio, isentropic efficiency, loss coefficients, ...). This is achieved by considering goal-oriented error estimates where the non-linear dependency is taken into account by using the adjoint state \( W^\star \). The goal is to minimize the approximation error in the computation of a given scalar functional \( J(W) \):

\[
E_{go}(H) = |J(W) - J(W_h)| = |\delta J|.
\]

To do so, various approaches have been proposed to take into account the sensitivity of the functional \( J \) to the local residual with the discrete adjoint. Namely, as the discrete numerical solution \( W_h \) is sought to cancel the residual \( R(W_h) = 0 \), any error in the computation of the residual \( \delta R \) will lead to an error in the computed solution. This error in the solution leads to an error in the computation of the output functional

\[
\delta J \approx \frac{\partial J}{\partial W} \cdot \delta W \approx \left( \frac{\partial J}{\partial W} \right)^{-1} \delta R, \quad \text{where} \; W^\star = \left( \frac{\partial R}{\partial W} \right)^T \cdot \frac{\partial J}{\partial W},
\]

is the discrete adjoint. The local error in the residual \( \delta R \) is then related to the local mesh size by different means [26, 31, 41, 36, 52, 54, 72].

We use here a different goal-oriented error analysis to analytically derive optimal metric field from the adjoint and primal fields. This \textit{a priori} analysis has been done for inviscid flows and has shown excellent results [41]. The same analysis has been extended to laminar viscous flows by taking into account viscous terms and shown very good results [11]. The main advantage of these estimates in comparison to other goal-oriented error estimates is that the anisotropy of the mesh appears naturally using the continuous mesh framework [39, 40] and they provide directly the analytical expression of the optimal mesh. In our error estimation problem, the approximation error on the functional can be decomposed into an implicit error term and an interpolation error term:

\[
|\delta J| = |J(W) - J(W_h)| = |\left( \frac{\partial J}{\partial W} \right)(W - \Pi_h W)| \leq |\left( \frac{\partial J}{\partial W} \right), W - \Pi_h W| + |\left( \frac{\partial J}{\partial W} \right), \Pi_h W - W_h|.
\]

The interpolation error term is easily estimated using Relation (28). For the implicit term we demonstrate [11, 41] that:

\[
H \left( \frac{\partial J}{\partial W} \right), W_h - \Pi_h W | \approx (\Psi(W) - \Psi_h(W), \Pi_h W + W^\star), \quad \text{where} \; \Psi \; \text{and} \; \Psi_h \; \text{are the continuous and discrete operators of the considered problem.}
\]

We use the Finite Element framework to accurately analyze the implicit error term involved in the Navier-Stokes equations. We denote by \( F_E(W) \) the convective fluxes given by Relations (11) and \( F^V(W) \) the viscous fluxes deduced from Relations (12), e.g. \( S(W) = \nabla \cdot F^V(W) \). Starting from Relation (30), both inviscid and viscous fluxes are linearized with respect to the solution \( W \) and its gradients \( \nabla W \) and integrating by parts (and omitting the boundary terms) we obtain [2]:

\[
\left( \frac{\partial J}{\partial W} \right) W_h - \Pi_h W \approx \int_{\Omega} \left( \nabla \cdot (F_E(W) - F_E(W_h)) - \nabla \cdot (F^V(W) - F^V(W_h)) \right) \cdot W^\star \; d\Omega
\]

\[
\approx \int_{\Omega} \left( \sum_{i} \nabla_x \left[ \left( \frac{\partial F_E}{\partial W} \right)^T \right] (W - \Pi_h W) \right) \cdot W^\star + \sum_{i,j} \nabla_{x,i} \left[ \left( \frac{\partial F^V}{\partial W} \right)^T \nabla_{x,j} (W - \Pi_h W) \right] \cdot W^\star \; d\Omega
\]

\[
\approx - \int_{\Omega} \left( \sum_{i} \left( \frac{\partial F_E}{\partial W} \right)^T \nabla_{x,i} W^\star + \sum_{i,j} \left( \frac{\partial F^V}{\partial W} \right)^T \nabla_{x,i,j} W^\star \right) (W - \Pi_h W) \; d\Omega.
\]

This implicit error estimate is a weighted sum of interpolation errors in \( L^1 \)-norm on the conservative variables where the weights depend on the gradient and the hessian of the adjoint state and on the convective and viscous fluxes. From Relation (29), we just have to add the interpolation error term to have the approximation error estimate:

\[
|J(W) - J(W_h)| \leq \int_{\Omega} \left| \left( \frac{\partial J}{\partial W} \right) + \sum_{i} \left( \frac{\partial F_E}{\partial W} \right)^T \nabla_{x,i} W^\star + \sum_{i,j} \left( \frac{\partial F^V}{\partial W} \right)^T \nabla_{x,i,j} W^\star \right| |W - \Pi_h W| \; d\Omega = \int_{\Omega} \sum_{i} |G(W)| |W_i - \Pi_h W_i| \; d\Omega.
\]
Thus, we can directly apply to this sum the formulation of the optimal continuous mesh, Relation (28), to find the analytical expression of the optimal goal-oriented continuous mesh. For practical implementation, we give the expression of the weights in 3D for each conservative variable:

\[
G(\rho) = -\frac{\partial j}{\partial \rho} + \sum_i \frac{\partial f_i}{\partial \rho} \cdot \nabla_x \mathbf{W}^r - u f_{\rho u}(W, H_W) - v f_{\rho v}(W, H_W) - w f_{\rho w}(W, H_W) - (E - u^2 - v^2 - w^2) f_{\rho E}(W, H_W)
\]

\[
G(\rho u) = -\frac{\partial j}{\partial \rho u} + \sum_i \frac{\partial f_i}{\partial \rho u} \cdot \nabla_x \mathbf{W}^r + f_{\rho u}(W, H_W) - u f_{\rho E}(W, H_W)
\]

\[
G(\rho v) = -\frac{\partial j}{\partial \rho v} + \sum_i \frac{\partial f_i}{\partial \rho v} \cdot \nabla_x \mathbf{W}^r + f_{\rho v}(W, H_W) - v f_{\rho E}(W, H_W)
\]

\[
G(\rho w) = -\frac{\partial j}{\partial \rho w} + \sum_i \frac{\partial f_i}{\partial \rho w} \cdot \nabla_x \mathbf{W}^r + f_{\rho w}(W, H_W) - w f_{\rho E}(W, H_W)
\]

\[
G(\rho E) = -\frac{\partial j}{\partial \rho E} + \sum_i \frac{\partial f_i}{\partial \rho E} \cdot \nabla_x \mathbf{W}^r + f_{\rho E}(W, H_W)
\]

where we have

\[
f_{\rho u}(W, H_W) = \frac{1}{3} \frac{(\mu + \mu_\nu)}{\rho} \left( 4 (\rho u')_{xx} + 3 (\rho u')_{yy} + 3 (\rho u')_{zz} + (\rho v')_{xy} + (\rho w')_{xz} \right.
\]

\[
+ 4 u (\rho E')_{xx} + v (\rho E')_{xy} + 3 u (\rho E')_{yz} + 3 u (\rho E')_{xz} - 5 \omega_{x,y} + 5 \omega_{y,z} \right)
\]

\[
f_{\rho v}(W, H_W) = \frac{1}{3} \frac{(\mu + \mu_\nu)}{\rho} \left( 3 (\rho v')_{xx} + 4 (\rho v')_{yy} + 3 (\rho v')_{zz} + (\rho u')_{xyc} + (\rho w')_{yz} \right.
\]

\[
+ 3 v (\rho E')_{xx} + u (\rho E')_{xy} + 4 v (\rho E')_{yz} + 3 v (\rho E')_{xz} + 5 \omega_{x,y} - 5 \omega_{y,z} \right)
\]

\[
f_{\rho w}(W, H_W) = \frac{1}{3} \frac{(\mu + \mu_\nu)}{\rho} \left( 3 (\rho w')_{xx} + 3 (\rho w')_{yy} + 4 (\rho w')_{zz} + (\rho u')_{xy} + (\rho v')_{yz} \right.
\]

\[
+ 3 w (\rho E')_{xx} + u (\rho E')_{xx} + 3 w (\rho E')_{yz} + v (\rho E')_{yz} + 4 w (\rho E')_{xz} - 5 \omega_{x,y} + 5 \omega_{x,z} \right)
\]

\[
f_{\rho E}(W, H_W) = \frac{(\lambda + \lambda_\nu)}{\rho} \left( (\rho E')_{xx} + (\rho E')_{yy} + (\rho E')_{zz} \right)
\]

where the terms \( \frac{\partial j}{\partial \mathbf{W}} \) have been provided in Section 5.1, and \( \omega \) terms are given by

\[
\omega_{x} = (\omega_{x,x}, \omega_{x,y}, \omega_{x,z}) = \nabla u \times \nabla \rho E', \quad \omega_{y} = (\omega_{y,x}, \omega_{y,y}, \omega_{y,z}) = \nabla v \times \nabla \rho E', \quad \omega_{z} = (\omega_{z,x}, \omega_{z,y}, \omega_{z,z}) = \nabla w \times \nabla \rho E'.
\]

Remark 6.1. In this work, we do not have considered the turbulent equation in the goal-oriented error estimate. Note that in this simplified version, the turbulence model is taken into account implicitly throughout \( \mu \) and \( \lambda \).

7. Periodic mesh adaptation

As we choose to enforce strong geometric periodicity in the flow solver, it has to be enforced in the mesh at each adaptation. Theoretically it simply requires to simultaneously perform the local mesh operations on both sides of the domain for periodic surfaces. But, implementing such a process in a robust manner for industrial geometries is very technical and difficult. Three strategies can be envisioned to deal with periodic surfaces in local remeshing algorithm, each of these strategies has its own pros and cons.

For the first strategy, the periodicity is directly managed in the local remesher, thus when a local modification is performed on a periodic surface it has to be performed at the same time on the other periodic surface and the results depends on both configurations, i.e., the mesh modification is accepted if the operations on both sides are accepted. The pros are this method is efficient because it requires only one call of the local remesher and it preserves the original domain geometry. But, this method is hard to implement as it requires several structural modifications in the mesh adaptation tool, it makes the maintenance of the code more difficult and it requires to design a cavity operator with double constraints.

For the second strategy, the periodicity can be managed by adding a migration step and then another remeshing step to remesh the periodic surface as proposed in [23]. The idea is that the periodic frontier has been chosen arbitrarily and is not different from any other separation in the domain and could be adapted in the volume. To do so, after a first mesh adaptation pass where
the periodic boundaries have been left unchanged, elements are migrated from one side of the domain to the other side such that the periodic surface is now embedded inside the new domain. Then, the region around the periodic surface can be adapted in the second pass. In that approach, the original periodic surface is lost. This method is easier to implement because it just requires to implement the migration step and re-use the same local remesher. Here, the local remesher is not aware of the periodicity. As regards the cons, this approach is more expensive than the first method because we have two remeshing steps and one migration step. But, the main problem with this method is that it does not preserve the original domain geometry which is cumbersome in an industrial context.

The third strategy, that we propose in this paper, is to add one more migration step to the second method and to preserve the periodic surface when the region around it is adapted. This method will preserve the original domain geometry. Again, the pros are: it is easier to implement, we use the same local remesher blind of the periodicity and we preserve the original domain shape. The cons are that it is more expensive than the first method because we have two remeshing steps and two migration steps. The main difficulty of this method is the following: to preserve the shape of the original domain, we have to preserve the shape of the embedded surface during the second remeshing step which requires a local remesher handling no-manifold geometries.

In this work, we use the remesher Feflo.a which is able to remesh no-manifold geometries [44]. It is combined with a migration tool.

7.1. The local adaptive remesher: Feflo.a

We give a brief overview of the AMG/Feflo.a meshing algorithm that is used as the local adaptive remesher. The main features are the followings:

- it is metric-based and uses the concept of unit mesh,
- the volume and the surface meshes are adapted simultaneously in order to keep a valid 3D mesh throughout the entire process. This guarantees the robustness of the complete remeshing step,
- it relies on a single cavity operator capable of automatically managing a combination of generalized standard operators in one go (insertion, collapse, swap of edges and faces) at once,
- it is capable of handling extremely anisotropic meshes,
- the surface geometry can be either represented by the CAD or a $P^3$ surface mesh model. Each time a vertex is inserted or moved on the surface then it is projected on it using one of these two geometric models.

For a complete description, we refer to [45, 37]. Other local adaptive remesher exist such as EPIC [48] or Refine [31].

7.2. Periodic mesh adaptation algorithm

Following the third strategy, the main steps of the periodic mesh adaptation process are:

1. Adapt the surface and volume meshes leaving periodic surface meshes (periodic boundaries) unchanged
2. Migrate $N$ layers of elements from one side of the periodic boundary to the other side. Transport the metric field accounting for rotations
3. Adapt the new domain with the periodic surface as constraint (i.e., as a no-manifold surface) inside the volume
4. Migrate back the elements to recover the initial geometry of the domain.

In the following, we describe in more details all these steps.

In the first step, the local remesher adapts the surface and the volume meshes but no mesh modifications are allowed on periodic surfaces. This imposes constraint to the mesh adaptation process in the neighborhood of the periodic surface, it thus may lead to the generation of bad quality elements in this region. This bad elements will be removed later in step 3.

The second step consists in migrating elements from one side of the periodic boundary to the other side in order to embed the periodic surface inside the volume and to be able to adapt it. We provide in Figure 5 the schematic process of the migration step. We start from a periodic mesh in the vertical direction, lines in green represent both sides of the periodic domain. Lines in blue and red define elements that are going to be migrated on the other side of the domain. Vertices of the mesh are numbered for clarity and we use '$$'$ to state periodic of, for instance, $1'$ is the periodic vertex associated with vertex 1.

Starting from the initial mesh, Figure 5 (a), we go through the elements and wrap the mesh on itself, replacing the vertices on the periodic surface of one side of the domain by their periodic counterpart, see Figure 5 (b). This step affects blue volume elements and the non-periodic frontiers (thicker blue lines). This leads to a topologically periodic mesh with elements of negative volume (at that point the blue ones). The vertices (here 16, 17, 18) and the periodic edges (dot blue lines) that have been disconnected from the rest of the domain are removed.

26
Now, vertices can be moved independently, keeping the topology of the mesh. We simply migrate (by rolling the mesh on itself) a few layers (any number) of elements to their periodic location without changing the connectivity. Figure 5 (c) shows the resulting mesh after moving a first layer of elements, those on the other side of the periodic domain (the blue boundary and volume elements). Then, in Figure 5 (d), the resulting mesh after migrating a second layer of elements (the red boundary and volume elements). Finally, vertices on the new edges forming the new periodic surface (here 10, 11, 12) are identified and duplicated (into 16, 17, 18) in order to recreate their periodic counterpart and the mesh is finally un-wrap, i.e., elements are then reconnected to the proper vertices in order to recover a mesh with elements of positive volume. The resulting mesh after migrating two layers of elements is shown in Figure 5 (e), and now the periodic surface (in green) is embedded inside the domain.

At the end of the migration, the elements that have been displaced in this process are flagged in order to be identified later. The new periodic frontiers are identified and set in the final migrated mesh (here red edges 10-11 and 11-12 and the grey edges 16-17 and 17-18).

Note that displacing more than one layer of elements gives more space to the local remesher tool to perform mesh modification, thus it is important for better quality meshes. Indeed, this is helpful in presence of strong anisotropy such as in shocks or in boundary layers regions. In practice, we migrate 5 layers of elements. However this imposes a constraint on the maximal size for the elements. If we choose to migrate $N$ layers of elements, then it is natural to have at least $2N$ layer of elements in the smaller diameter between both periodic surfaces.

Another important remark is that in 3D with highly anisotropic meshes, the migration process may create unconnected components that will constrain the third step. If such unconnected components are detected, then they are also migrated on the other side to keep a connected domain.

In the third step, the mesh in the neighborhood of the original periodic surface (green edges) can now be adapted. In the meshing process, it is treated as non-manifold surface. In other word, it is considered as a surface inside the volume, this surface can be adapted but its geometry must be preserved during the mesh adaptation process. The new periodic surfaces (here

![Figure 5: Illustration of the different steps in the migration process.](image)

![Figure 6: Adaptation of the periodic surface after the migration process.](image)
the horizontal red and grey edges) are constrained and are left unchanged during this second remeshing pass, guaranteeing the periodicity of the mesh. In Figure 6, starting from the migrated mesh (picture (a)) we then obtain the migrated adapted mesh (picture (b)) where the original periodic surface (green edges) has been adapted. Note that in this second remeshing pass, the mesh can be modified everywhere even if it is not on or close to the original periodic surface. This will get rid of bad quality elements that may have been created in the first remeshing pass.

In order to adapt the mesh in the transformed configuration, we need the corresponding metric. Hence, it has to be transformed consistently with the mesh, in particular, a unit edge in the initial configuration must remain unit in the transformed configuration. Obviously, the metric is the same in the part of the domain that is not displaced. For translation periodicity, the metric space is unchanged, so that copying the metric from the initial node to the translated node is sufficient. This is not the case for rotation periodicity. A rotation changes the directions of the metric, hence the metric has to be copied and rotated from the initial node to the translated node. Let us denote by $e$ a unit vector in the metric $\mathcal{M}$, representing an edge:

$$e^T \mathcal{M} e = 1.$$ 

The edge will be moved across the domain by the rotation and its orientation will change. We thus assimilate here the edge to a vector determining its orientation. This vector becomes in the rotation process

$$\bar{e} = Re,$$

where $R$ is the matrix of the rotation. As this new edge $\bar{e}$ has to remain unit in the new metric $\bar{\mathcal{M}}$, we deduce

$$1 = \bar{e}^T \bar{\mathcal{M}} \bar{e} = e^T (R^T \bar{\mathcal{M}} R)e.$$ 

Thus, we deduce

$$\bar{\mathcal{M}} = RMR^T,$$

where we use that for a rotation $RR^T = Id$.

In a fourth step, the mesh is migrated back using exactly the same process as above to obtain the final adapted mesh, see Figure 6 (c). The surface and the volume meshes have been adapted while preserving the periodicity and the original domain geometry.

7.3. Some exemples

The migration process is illustrated on two practical cases in Figure 7. Several layers of elements have been displaced in both cases, these elements are colored in green. After the migration process, the original periodic surface becomes the separation between the grey and the green elements. As it is now embedded in the volume its neighborhood can be adapted.

Figures 8, 9 and 10 demonstrate the importance of periodic mesh adaptation where we compare on the NASA Rotor 37 case simulations without (on the left) and with (on the right) periodic mesh adaptation. As expected, the original shape of the domain is preserve as any mesh modifications on the periodic surface preserves the periodic surface.
When the periodic frontiers are appropriately adapted (right pictures), we can see many shocks crossing the frontier without being affected. We do not see any numerical artifact in the mesh due to the periodic frontier. This is what we expect from a numerical periodic frontier, to behave as another domain. In Figure 10 (right), one can see the curve corresponding to the periodic surface that has been preserve and adapted.

When the periodic frontiers are left unadapted (left pictures), as the inner mesh is continuously refined, as fine as the initial mesh can be on the boundary, it inevitably ends to be too coarse compared to the required mesh size at a point of the mesh adaptation process. This leads to an inversion of the local anisotropy which deteriorates the quality of the solution and finally generates instabilities leading to unphysical results and finally the break down of the solver. This exercise just points out that periodic mesh adaptation is mandatory.

![Figure 8: NASA Rotor 37 case. Closeup view on the shock crossing the periodic frontier, without (left) and with (right) periodic mesh adaptation.](image)

![Figure 9: NASA Rotor 37 case. Cut in the adapted mesh, without (left) and with (right) periodic mesh adaptation.](image)

![Figure 10: NASA Rotor 37 case. Close-up view of the cut in the adapted mesh, without (left) and with (right) periodic mesh adaptation.](image)
8. Numerical results

In this numerical section, we consider turbomachinery applications. The first case is a well-know 2D case, the LS89 high pressure axial turbine vane, to emphasize which benefits can be expected in three-dimensions. The second case is the NASA Rotor 37.

8.1. LS89 blade

8.1.1. Description of the test case

The first test case selected to demonstrate the developed mesh adaptation process in 2D is the transonic LS89 linear cascade (2D profile linearly extruded in the spanwise direction) \[9\]. This test case is representative of a high-pressure turbine stator (also called Nozzle Guide Vane) and is considered as one of the most classic openly-available turbomachinery test cases. Several experimental campaigns of this blade have been performed, most focusing on the impact of shocks and freestream turbulence on the laminar-to-turbulent transition, and by extension to the heat transfer, occurring on the suction side of the blade. In addition, several publications of numerical simulations, ranging from 2D RANS to LES and even DNS, are available (for example \[60, 70\]). In this study, laminar-to-turbulent transition is not taken into account while the walls are considered adiabatic. However, the high Reynolds number, the transonic nature of the test case and the lack of more relevant open 2D turbomachinery test cases with engine-representative Reynolds render this case the best candidate for the first demonstration of periodic mesh adaptation in 2D.

The following Table summarizes the main characteristics of the LS89 test case and the experimental operating conditions:

<table>
<thead>
<tr>
<th>Test case</th>
<th>Chord</th>
<th>(p_i)</th>
<th>Re</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS89</td>
<td>67.647mm</td>
<td>(1.87 \times 10^5) Pa</td>
<td>(1.13 \times 10^5)</td>
</tr>
</tbody>
</table>

8.1.2. Study parameters

For the numerical simulations, the reference quantities are:

\[
\rho_{\text{ref}} = 1.275349 \text{ kg.m}^{-3} , \quad \rho_{i,\text{ref}} = 10^5 \text{ Pa}, \quad \rho_{i} = 1.828 \times 10^5 \text{Pa},
\]

\[
u_{\text{ref}} = (400, 0, 0) \text{ m.s}^{-1}, \quad \mu_{\text{ref}} = 1.716 \times 10^{-5} \text{ Pa.s}^{-1}, \quad T_{\text{in}} = 413.3 \text{K}.
\]

In order to illustrate the versatility of mesh adaptation process, we choose four different configurations for the outflow pressure:

<table>
<thead>
<tr>
<th>(p_{\text{ratio}})</th>
<th>0.5908</th>
<th>0.5470</th>
<th>0.4923</th>
<th>0.3604</th>
</tr>
</thead>
<tbody>
<tr>
<td>(p_{\text{out}})</td>
<td>(1.08 \times 10^5)</td>
<td>(1.00 \times 10^5)</td>
<td>(0.90 \times 10^5)</td>
<td>(0.66 \times 10^5)</td>
</tr>
</tbody>
</table>

The first case is a smooth case, but for the second and third cases a shock appears on the suction side in different places impacting the boundary layer and the wake. For the forth case, a complex shock-wake interaction occurs.

In order to illustrate the independence of the mesh adaptation process to the initial mesh, the computation starts from a non-adapted (uniform) mesh only composed of 2 239 vertices and 4 098 triangles. As one can see in Figure 11, no hypothesis on the boundary layer, wake or shock wave regions has been made in the initial mesh generation process. The simplest quasi-uniform inviscid-like mesh has been generated.

For the mesh adaptation process, we have considered three different error estimates:

- the feature-based approach where we control the interpolation error on the local Mach number in \(L^4\)-norm. The local Mach number has been chosen as sensor because it automatically enables the capture of boundary layer, wake and shock wave. This guaranties the convergence of the solution and the proper representation of any physical phenomenon of interest
- the goal-oriented approach where the output functional is the loss coefficient as the loss coefficient is the main functional of interest in this case
- the goal-oriented approach where the output functional is the pressure ratio in order to analyse the impact of a different functional choice.

For the mesh convergence study, we have considered seven complexities:

\{4 000, 8 000, 16 000, 32 000, 64 000, 128 000, 256 000, 512 000\},

leading to adapted meshes the number of vertices of which is approximatively

\{5 000, 10 000, 20 000, 40 000, 75 000, 150 000, 300 000, 550 000\}. 

30
For each complexity, we can perform up to 20 mesh adaptation iterations. After each iteration, we analyze the convergence of debit ratio, total pressure ratio and loss coefficient on the last 3 adapted meshes. If the variations of these 3 functionals is less than 1% on the last 3 adapted meshes, then we assume we are mesh converged for the current complexity, we break the mesh adaptation loop, and run the next mesh adaptation loop with an increased complexity. In practice, we observe that for the majority of the cases, we only perform 4 mesh adaptations iterations (mainly because we analyse the convergence on the last 3 meshes). Obviously, two mesh adaptations at large complexity seems to be sufficient.

The periodic domain is adapted using the method described above, five layers of elements close to one side of the periodic domain are moved to the other side of the domain to perform the adaptation of the periodic surface.

8.1.3. Results analysis

Figures 14, 15 and 16 show for all cases the adapted meshes composed of approximately 150,000 vertices with the associated solutions (the local Mach number field). These figures point out the proper adaptation in the wake of the blade, in the shocks and boundary layers regions by the solution mesh-adaptive process. The periodic adaptation is clearly illustrated by the adaptation of the wake of the blade. These cases already show the versatility and efficiency of mesh adaptation for turbomachinery applications and in this process the importance of periodic adaptation.

As the pressure difference increases, the initial subsonic flow becomes transonic and a shock appears on the suction side of the blade. This shock is initially weak and requires an appropriate fine discretization in the proper position to be computed. Then, this shock moves toward the trailing edge and becomes stronger. Similarly, the wake is subject to a strong diffusion if not discretized properly. Figure 12 shows the shock-boundary layer interaction region for the case $p_{ratio} = 0.547$ and again points out the high accuracy obtained thanks to anisotropic mesh adaptation.

In Figure 17, we compare, on the $p_{ratio} = 0.547$ case, the adapted meshes obtained with the goal-oriented mesh adaptation based on the loss coefficient functional (left pictures) and adapted meshes obtained with the feature-based approach (right pictures). These meshes contain approximately 40,000 vertices. We observe that the goal-oriented method puts less vertices in the shock and wake regions. It may increase or reduce the resolution in the boundary layer region depending on its impact on the functional. Finally, it focuses more at the inlet and outlet boundaries. The adjoint of $\rho$ and $\rho u$ are shown in Figure 13. We notice a discontinuity of the adjoint at the inlet. This will be investigated in a near future.

Now, we analyse the convergence of the debit ratio and the loss coefficient for each case. Figure 18 depicts the convergence of these two output functionals for the three considered error estimates. All approaches are converging toward the same solution, except for the last case where the feature-based approach gives a different answer from the goal-oriented one. Overall, the goal-oriented mesh adaptation provides a smoother convergence than the feature-based one.

For a deeper analysis, we plot the convergence history of the functionals for two error estimates. In these plots, we have in red the convergence at each complexity and, in blue the global convergence by retaining the final value for each complexity. Figures 19 and 20 show it for the debit ratio and the loss coefficient, respectively. Figure 21 is a zoom on the highest complexity for the loss coefficient. Overall, a lot is done on the coarser adapted meshes. Thus, converging on the coarse adapted meshes fasten the convergence. We observe that mesh convergence is achieved for all cases. Note that on the finest adapted meshes, $i.e.$, the largest complexities, all meshes (except the first one) gives very close answers. This emphasizes the stability of the process even if each time a new adapted mesh with no link to the previous one is generated.
Figure 12: LS89 blade case $p_{\text{ratio}} = 0.547$. Close-up view on the shock boundary layer interaction region to emphasize the high resolution thanks to anisotropic mesh adaptation.

Figure 13: LS89 blade case $p_{\text{ratio}} = 0.547$. Adjoint of the density $\rho$ (left) and adjoint of the first component of the momentum $\rho u$ (right).
Figure 14: LS89 blade cases. Mach isovalues (left) and feature-based adapted meshes (right) for the four pressure ratios $p_{\text{ratio}} = 0.5908$, $p_{\text{ratio}} = 0.5470$, $p_{\text{ratio}} = 0.4923$, $p_{\text{ratio}} = 0.3604$. Mach isovalues are calculated as $M = \sqrt{\frac{2}{\gamma - 1}(p_{\text{out}} - p_{\text{in}})}$.
Figure 15: LS89 blade cases. Mach isovalues (left) and feature-based adapted meshes (right) for the four pressure ratios $p_{\text{ratio}} = p_{\text{out}} / p_{\text{in}}$. Close-up view in the wake region.
Figure 16: L589 blade cases. Mach isovalues (left) and feature-based adapted meshes (right) for the four pressure ratios $p_{ratio}$ = $p_{ratio} \times p_{in}$. Close-up view on the blade.
Feature-based $L^4$ on the Mach  

Goal-oriented on the loss coefficient

Figure 17: LS89 blade case $p_{\text{ratio}} = 0.547$. Comparison of adapted meshes obtained with the feature-based error estimate (left picture) and the goal-oriented error estimates based on the loss coefficient functional (right pictures). These adapted meshes are composed of approximatively 40000 vertices.
Figure 18: LS89 blade cases. Mesh convergence of the debit ratio and loss coefficient for all $p_{\text{ratio}}$ cases and all considered error estimates.
Figure 19: LS89 blade cases. Convergence history of the debit ratio for the feature-based error estimate (left) and the goal-oriented error estimate (right) throughout the whole mesh convergence analysis. In red, the convergence of the debit ratio at each complexity and, in blue, the global convergence of the debit ratio by retaining the final debit ratio value for each complexity.
Figure 20: LS89 blade cases. Convergence history of the loss coefficient for the feature-based error estimate (left) and the goal-oriented error estimate (right) throughout the whole mesh convergence analysis. In red, the convergence of the loss coefficient at each complexity and, in blue, the global convergence of the loss coefficient by retaining the final loss coefficient value for each complexity.
Figure 21: LS89 blade cases. Convergence history of the loss coefficient for the feature-based error estimate (left) and the goal-oriented error estimate (right) throughout the whole mesh convergence analysis. In red, the convergence of the loss coefficient at each complexity and, in blue, the global convergence of the loss coefficient by retaining the final loss coefficient value for each complexity.
8.2. NASA Rotor 37

In this section, the developed mesh adaptation process is evaluated on a realistic 3D test case, the NASA Rotor 37. This test case is particularly challenging as it is a transonic compressor with complex phenomena such as shocks, shock/boundary layer interaction and flow separation. Additionally, LES results have shown that transition occurs almost immediately at the leading edge due to the leading edge shocks \(2^\circ\), hence removing the need for RANS transition modeling. The results are obtained for different operating points and are compared to available experimental data as well as with standard RANS simulations on structured meshes. The full compressor characteristic has been simulated. Two operating points are of particular interest: the experimental near peak efficiency point where mass flow is 98% of the value at choke conditions, as most experimental measurements are at this operating point, and the experimental near-stall point where mass flow is 93% of the value at choke conditions, as it is the last stable point before surge occurs in the experiment.

8.2.1. Description of the test case

The 3D turbomachinery test case selected is the transonic Rotor 37 from NASA, designed in 70’s as part of the Stage 37 [57] and retested in the 90’s with the rotor alone [66]. It is a configuration that is openly available and has been extensively investigated numerically in the past. It is also the reference test case for most mesh adaptation efforts for turbomachinery that are found in the literature [58, 67, 68]. The full experimental annulus contains 36 blades. In this work, a single blade sector is simulated (10 degrees) with periodicity imposed in the azimuthal direction. The rotational speed is 17188.7 rpm, leading to relative tip and hub Mach numbers of 1.48 and 1.13 respectively. The experimental measurements consisted of radial or pitch profiles at 4 stations, the first being at the domain inlet, the second and the third before and after the blade leading and trailing edges and the fourth further downstream the rotor. Laser Doppler Anemometry was also performed, providing contours of the relative Mach number at different blade heights.

To generate the CAD of the blade, instead of relying on the coordinates file available in the literature, it was chosen to regenerate it using the original blade design code employed at the time by NASA, which is openly available. The hub fillet, whose characteristics are provided in the blind test case description [66], is also included. This was done in an effort to have higher CAD fidelity, as the blade coordinates in the literature include few significant digits, leading to non-physical oscillations and curvature discontinuities on the blade surface. Such oscillations can considerably hinder the mesh adaptation, as the process will naturally detect them and refine the mesh accordingly, hence exacerbating the CAD inaccuracy. The drawback of this CAD regeneration is that the produced geometry is the cold one (no displacements due to the mechanical forces on the blades when running), while the coordinates reported in the literature are for the hot geometry (produced using a FEM simulation of the test bench). As rerunning a FEM analysis of the test bench with the new CAD is impossible, some differences with the experimental measurements are expected. It was reported, notably, in [66] that the cold geometry is expected to have a reduced choked mass flow, as the impact of the centrifugal forces tends to untwist the blade at high radius and increase the effective flow passage.

8.2.2. Simulation set-up for the structured RANS

As structured meshes are the industry-standard for RANS simulations of turbomachinery components with limited technological effects, such as this case, the same configuration was simulated using a solver that runs on structured multi-block meshes. The objective is to evaluate how the results produced via mesh adaptation compare to the standard approach. Comparisons are performed both in terms of compressor performance, radial profiles as well as on local flow phenomena, notably the tip clearance flow.

The solver employed for the structured simulations is the elsA solver, developed by Onera [14]. It is a Finite-Volume, cell-centered solver. The spatial numerical scheme used in this work is the Roe scheme [59], in conjunction with the Van Albada limiter. For the temporal integration, a simple implicit Backward Euler temporal integration scheme is employed with a CFL ramp. The simulations start with an initial CFL value of 0.95 which is gradually increasing up to 10 to accelerate the convergence as the simulation progresses. The turbulence model employed is the standard Spalart-Allmaras model, in order to ensure the maximum coherence possible with the unstructured WOLF simulations. The solver runs in the relative frame of reference, similarly to WOLF. For the boundary conditions, at the inlet a total pressure, a total enthalpy and the flow angles were set, hence removing the need for RANS transition modeling. The results are obtained for different operating points and are compared to available experimental data as well as with standard RANS simulations on structured meshes. The full compressor characteristic has been simulated. Two operating points are of particular interest: the experimental near peak efficiency point where mass flow is 98% of the value at choke conditions, as most experimental measurements are at this operating point, and the experimental near-stall point where mass flow is 93% of the value at choke conditions, as it is the last stable point before surge occurs in the experiment.

For all meshes and the given rotation speed, the entire compressor characteristic has been simulated. This was achieved by changing the outlet static pressure. In total, approximately 40 simulations were performed.
8.2.3. Study parameters for the mesh-adaptive solution process

We present the results obtained with the mesh-adaptive solution platform on the NASA Rotor 37 cold geometry. The rotation speed is 1800 rad s\(^{-1}\) and we study a sector of 10\(^{\circ}\). We choose the following reference state to initialize the computation:

\[
\rho_{\text{ref}} = 1.225 \text{ kg m}^{-3}, \quad p_{\text{ref}} = 101.325 \text{ Pa}, \quad T^{\text{ref}} = 288.15 \text{ K},
\]

\[
\mathbf{u}_{\text{ref}} = (400, 0, 0) \text{ m s}^{-1}, \quad \mu_{\text{ref}} = 1.789 \times 10^{-5} \text{ Pa s kg}^{-1} \text{ m}^{-1} \text{s}^{-1}.
\]

These flow conditions provide a computational equivalent to the ones already performed at Safran Tech with the elsA flow solver. For the inlet boundary condition, we impose weakly \(p_{\text{in}}^\text{ref} = 101.325 \text{ Pa}, T^\text{ref} = 288.15 \text{ K}\) using Riemann invariant. For the outlet boundary condition, we impose weakly \(p_{\text{out}} = \beta p_{\text{in}}^\text{ref}\) using Riemann invariant.

To study the whole characteristic of this compressor rotor, we have run 16 operating points with the following \(\beta\) values :

\[\{1, 1.025, 1.05, 1.075, 1.1, 1.125, 1.15, 1.175, 1.2, 1.225, 1.24, 1.25, 1.26, 1.27, 1.28, 1.29\}.\]

The mass flow rate is varying from the choke condition \(\beta = 1\) to the near-stall condition for \(\beta = 1.29\) and the peak efficiency is close to \(\beta = 1.2\). For each operating point, we perform a mesh convergence study with the mesh-adaptive solution platform. Depending on the mesh size, more operating points having be added to enhance the characteristic representation.

For the mesh adaptation, we consider the feature-based approach controlling the \(L^1\) norm of the interpolation error on the local Mach number variable and the goal-oriented method focusing on the pressure ratio functional. The initial mesh is a uniform isotropic mesh composed of 158 390 vertices and 851 932 tetrahedra. The mesh convergence study is performed using 5 complexities:

\[\{160,000, 320,000, 640,000, 1,280,000, 2,560,000, 5,120,000\},\]

leading to adapted meshes the number of vertices of which is approximatively

\[\{360,000, 720,000, 1,400,000, 2,800,000, 5,400,000, 11,000,000\}.\]

For each complexity, we can perform up to 20 mesh adaptation iterations. After each iteration, we analyze the convergence of debit ratio, total pressure ratio and total temperature ratio on the last 3 adapted meshes. If the variations of these 3 functionals is less than 1\% on the last 3 adapted meshes, then we assume we are mesh converged for the current complexity, we break the mesh adaptation loop, and run the next mesh adaptation loop with an increased complexity. In practice, we observe that for the majority of the cases, we only perform 4 mesh adaptations iterations (mainly because we analyse the convergence on the last 3 meshes). This analysis suggests that two mesh adaptations could be sufficient, except on very coarse meshes. It is worth noting that every CFD simulation in the mesh adaptation process runs until "full" convergence (when the residuals reach \(10^{-6}\) and the log-residuals reach \(10^{-12}\) see [2]).

Performing a mesh convergence study using the mesh adaptation loop for all the operating points and both error estimates represents more than 1 000 runs.

8.2.4. Plots results analysis

The statistics for the main operating points are given in Table 1 for the feature based mesh adaptation (top) and the goal-oriented mesh adaptation (middle). In both cases, we are giving the results obtained on adapted meshes composed of 5 millions vertices (i.e., a mesh theoretical complexity of 2 560 000). The statistics for the closest elsA simulations on structured mesh, with a mesh theoretical complexity of 2 560 000). The statistics for the closest elsA simulations on structured mesh, with 7M cells, are also provided for comparison (bottom). The particular operating points are the following:

- the choke point with the highest mass flow rate
- the experimental near peak efficiency point which was obtained for a mass flow rate ratio of 0.98. The mass flow rate ratio being the ratio between the mass flow rate and the choke mass flow rate
- the experimental near stall point which was obtained for a mass flow rate ratio of 0.93
- the CFD peak efficiency point which is given by the highest obtained isentropic efficiency
- the CFD near stall point, which is given by the highest obtained pressure ratio (frequently considered as the surge/stall point in RANS simulations)
- the CFD last stable point (stall) which is given by highest \(P_{\text{ratio}}\) prescription for which simulations converged.
Table 1: NASA Rotor 37. Computations statistics for the feature-based (top) and the goal-oriented (middle) mesh adaptation runs for adapted meshes composed of 5 millions vertices, and for the structured RANS meshes (bottom) runs composed of 7 millions cells. In this table, the mass flow rate is denoted MFR and the choke mass flow rate MFR°. In the experiments (XP), the near peak efficiency is considered for a mass flow rate ratio MFR/MFR° of 0.98 and the near stall condition for 0.93. For the numerical simulations, the peak efficiency is considered for the highest obtained isentropic efficiency and the near stall condition for the highest obtained pressure ratio with the flow solver.

Mesh adaptation strategy. The first observation is that for the lower \( P_{ratio} \) (i.e., \( P_{ratio} \leq 1.150 \)), we were able to use the classical mesh adaptation algorithm and start from the lower complexity without any issue. But, for very low complexities (40,000 and 80,000), the simulations start to diverge (typical indication of compressor instability/surge in steady-state simulations) at a relatively low \( P_{ratio} \) prescription \( (P_{ratio} \approx 1.175) \), thus impeding the increase of the \( P_{ratio} \). This is why the mesh adaptation algorithm has been modified and higher \( P_{ratio} \) prescriptions are applied, when the complexity is increased, starting from a lower \( P_{ratio} \). As the goal-oriented mesh adaptation provides higher accuracy than the feature-based mesh adaptation, we notice that higher \( P_{ratio} \) prescription can be achieved on coarser adapted meshes. For instance, the \( P_{ratio} \approx 1.280 \) prescription can be run with 720,000 vertices using the goal-oriented approach while meshes composed of 1,400,000 vertices are required for the feature-based one.

Mesh convergence analysis. We analyze the mesh convergence of each functional for each prescribed operating point in the flow solver. Figures 22, 23, 24 and 25 show respectively the mesh convergence of the outlet mass flow, total pressure ratio, total temperature ratio and isentropic efficiency functionals. For each figure, the left picture gives the results obtained with the feature-based mesh adaptation and the right picture with the goal-oriented approach. The top picture gives the results for the lower \( P_{ratio} \) (from 1.000 to 1.175) and the bottom picture for the higher \( P_{ratio} \) (from 1.200 to 1.290).

Similar conclusions arise for the outlet mass flow, the total pressure ratio and the isentropic efficiency. We observe that mesh convergence occurs quickly, with only a few hundred thousands of vertices in the mesh, for the operating points between the choke and the near peak efficiency point \( (P_{ratio} \approx 1.200) \). However, as we move towards higher pressure ratios, mesh convergence occurs for a higher number of vertices and at the highest \( P_{ratio} \), further increasing the number of points appears necessary for full mesh convergence. In other words, the lower the \( P_{ratio} \) the faster the mesh convergence. The potential reasons for this will be analyzed in a following section.

We also observe that the mesh convergence occurs earlier (early capturing) with the goal-oriented mesh adaptation than the feature-based mesh adaptation pointing out the benefits of using the adjoint to adapt the mesh. Note that, nonetheless, both approaches are converging toward the same values.

With regards to the total temperature ratio, we achieve mesh convergence for all operating points very quickly, with only half a million vertices. This suggests that this functional is easier to converge. However, it is worth reminding that for the goal-oriented approach, the objective function is the total pressure ratio. Switching to an alternative objective function that is more
directly dependent on the total temperature ratio, such as the isentropic efficiency, could potentially alter this behavior.

In conclusion, we achieve mesh convergence for at least half of the operating points (lower pressure ratios) and we are almost mesh converged for remaining ones. Therefore, we are pretty confident on the obtained results as the mesh-adaptive solution process has reduced drastically the discretization error uncertainty\(^1\) on the obtained numerical solutions. The predictions are the ones given purely by the considered geometric and physical models: the cold geometry and the RANS equations with the Spalart-Allmaras turbulence model.

**Characteristics analysis.** Figures 26, 27 and 28 show the total pressure ratio, isentropic efficiency and total temperature ratio characteristics, their convergence with respect to the prescribed complexity, and the comparison with the experimental data on the hot geometry. To better compare with experimental data on the hot geometry, we also provide the normalized characteristic using the ratio between the mass flow and the choke mass flow. For each figure, the feature-based mesh adaptation is presented on the left-hand side and the goal-oriented mesh adaptation on the right-hand side. The top picture shows the characteristic with respect to the mass flow rate and the bottom picture the characteristic with respect to the normalized mass flow i.e., the mass flow rate divided by the choke mass flow rate. For better characteristic representation in the near stall region, at higher mesh complexities \( C \geq 1\,280\,000 \), in addition, we have run 7 operating points with the following \( \beta \) values:

\[ \{1.285, 1.295, 1.296, 1.297, 1.298, 1.299, 1.300\} \]

This represents a total of 23 operating points simulated at the highest complexities.

The plots of the characteristics for all the considered mesh complexities clearly highlight that the numerical solutions are mesh converged. Indeed, we note that the curves are superimposed for high mass flow rate illustrating the early mesh convergence for low \( P_{\text{ratio}} \). For low mass flow rates, the mesh convergence is visible with the reduction of the gap between two curves when the mesh complexity increases. We are not perfectly mesh converged but we are very close.

The normalized mass flow total pressure and isentropic efficiency characteristics for the different elsA structured simulations are also presented in Figure 29. Similar tendencies as with the mesh-adapted simulations is observed: increasing the resolution pushes both the pressure ratio and the isentropic efficiency higher, while the choked mass flow remains similar. The higher resolutions appear to be relatively mesh converged, though at a much higher resolution than the mesh-adapted ones. However, differences at lower mass flows (near stall) are still present. Finally, the 5M mesh-adapted pressure ratio and isentropic efficiency characteristics are compared to the 7M and 32M structured mesh predictions together in Figure 30. It is readily observed that at most mass flows the mesh-adapted solutions are higher both for the pressure ratio and, most importantly, for the isentropic efficiency.

We observe that for the considered geometric and physical models:

- Our results are not in accordance with the literature [7, 8, 12, 13, 22, 27, 32, 61] which generally shows curves below the data. The total pressure ratio characteristic is above the experimental data both for the mesh-adapted and structured meshes (partially). We think this is mainly due to the fact that we are running the cold geometry instead of the hot geometry. Having the hot geometry and comparing the results to the cold one could be of main interest to answer any remaining questions.

- the isentropic efficiency characteristic is below the experiment data for high mass flow rate ratio but fits with the experiments in the near stall region for the mesh-adapted predictions. This means that we don’t have the same slope. Note that the goal-oriented adaptation predicts higher efficiency than the feature-based one, and appears to fit better to the experiments. When we compare our results to the literature and the structured elsA simulations, we observe that the predicted isentropic efficiency is a lot higher than the literature and elsA results [7, 8, 12, 13, 22, 27, 32, 61]. As will be shown in the radial profiles, this is potentially due to a lack of resolution of the non-adapted meshes.

- for the total temperature ratio characteristic the same conclusions as for the total pressure ratio characteristic arise.

**What is the error variation?** To better evaluate the convergence of the performances, the total pressure ratio w.r.t the mesh size for the both mesh adaptation approaches for 3 mass flow rate ratios \((\text{MFR}/\text{MFR}^\ast = 1.00, \text{MFR}/\text{MFR}^\ast = 0.98, \text{and MFR}/\text{MFR}^\ast = 0.93)\) are represented in Figure 32. In all plots, the goal-oriented prediction at the highest complexity is used as the reference. Three level of error are represented: 0.1 % in black, 0.2 % in brown and 0.5 % in red.

For \( \text{MFR}/\text{MFR}^\ast = 1.00 \), we note that the gap between the reference solution and all solution obtained with the goal-oriented mesh adaptation is less than 0.1 % after half millions vertices. For the feature-based approach the gap is between 0.2 and 0.5 %.

---

\(^1\)No more effect of the discretization on the obtained numerical solution.
For MFR/MFR\textsuperscript{*} = 0.98, the variation is larger. For the goal-oriented mesh adaptation, the gap of 0.5 \% is obtained with a million vertices and less than 0.2 \% above. The gap is even larger for the feature-based approach but less than 0.2 \% on the five millions vertices mesh.

For MFR/MFR\textsuperscript{*} = 0.93, solution are closer and all of them are in the 0.5 \% gap frame.

These results illustrate that goal-oriented mesh adaptation is likely the better candidate from an aerodynamic engineer’s perspective who designs a compressor. However, the performance of the feature-based adaptation remains close to the goal-oriented one. This is also desirable due to the wide range of turbomachinery RANS simulations an engine manufacturer needs to perform, on a multitude of engine components and with different simulation objectives. This implies that with a purely goal-oriented approach, a very wide range of objective functions would need to be available for goal-oriented mesh adaptation. As a result, a combination of goal-oriented and feature-based mesh adaptation suggests that mesh adaptation will be applicable for practically all RANS simulations performed for the design of an engine.

### Radial Profiles

To further validate the predicted flow fields with mesh adaptation, the radial profiles of the total pressure and total temperature ratios downstream the rotor are plotted in Figure 31 for the two main operating points: near peak efficiency (top) and near stall (bottom). They are compared with the available experimental measurements. Only the results from the mesh-adaptive simulations with adapted meshes composed of 5M vertices are presented as it was previously shown that at this complexity the predictions are practically mesh converged. They are also compared to the structured elsA simulations with 7M (closest structured simulation to the mesh-adapted solutions with the chosen complexity) and 32M cells. It is readily observed that both feature-based and goal-oriented simulations have a very good qualitative agreement with the experimental results, with the curves following very accurately the shape of the experimental profiles. This indicates that the main flow phenomena, notably the secondary flows, are well predicted. The quantitative agreement is also reasonable. Some discrepancies on the levels are present but that is to be expected due to the mass flow discrepancy between the cold geometry and the hot geometry that was measured experimentally. Finally, it is well established in the literature [61] that the cavity under the rotor, not modeled here as its actual geometry is not available, impacts the aerodynamic losses near the hub, explaining the slight over prediction observed in that area. Finally, comparing with the elsA predictions, a good agreement is found between the two solvers. The exception is near the endwalls (hub and casing), particularly for the total temperature ratio. In these regions, the mesh adaptation appears to be closer to the experimental measurements. Furthermore, it is in those areas that adding extra cells in the structured simulations produces the biggest differences, notably by reducing the total temperature peaks and pushing the structured results towards the unstructured mesh-adapted predictions. These results clearly point to an improved secondary flows prediction found near the endwalls thanks to the mesh adaptation, as the process will naturally refine the vortices in those regions. It also explains the differences observed in the global isentropic efficiency between the solvers, as the over prediction of total temperature near the endwalls should lead to a decrease in the isentropic efficiency.

#### 8.2.5. Mesh and results analysis

**Near peak efficiency.** We now present solutions and adapted meshes obtained in this study to point out the high-fidelity obtained thanks to anisotropic unstructured mesh adaptation. We show results for the feature-based and the goal-oriented methods on the five millions vertices adapted mesh obtained for $P_{\text{ratio}} = 1.225$ corresponding (see Table 1) to a mass flow rate ratio equal to 0.98 (the experimental near peak efficiency operating point). For the associated solutions, we show the relative Mach number (i.e., in the rotating frame) because it provides more information about the physics.

Figure 33 presents the adapted surface mesh and the associated solution. These figures emphasize the adaptation of the periodic surface and the numerous shock waves that are crossing this surface leading to complex mesh adaptation patterns. Periodic mesh adaptation is critical to capture properly the physics involved in this test case. Mesh adaptation that cannot handle the periodicities in this transonic case can, for example, lead to unphysical shock reflections that pollute the flow in the passage or simply lead to excessive smearing of the shocks. We also notice the high resolution of the surface mesh on the blade and, in particular, where the shock boundary layer interaction occurs.

The feature-based and the goal oriented mesh adaptations have a different behavior in certain areas. The feature-based adaptation puts more refinement near the inflow and in the shock regions. The goal oriented adaptation, on the other hand, focuses more on no-slip (rotating or not) surfaces.

Now, we look inside the volume. To this end five different cut planes are presented. Three cut planes along the passage are shown: one close to the hubwall in Figure 34 ($y = -0.195$), one at the blade midspan in Figure 35 ($y = -0.213$) and one near the tipwall in Figure 36 ($y = -0.230$). And, two cut planes along the blade are shown: one at mid blade in Figure 37 ($x = 0.009$) and one going at the same time through the leading edge and the trailing edge of the blade in Figure 38 ($0.330057x + 0.0152761y + 0.943837z + 0.009 = 0$).

First, we notice the great complexity of the physics for this transonic case with many shock waves, shock wave - shock wave interactions, shock boundary layer interaction with separated boundary layer, tip vortices, wakes and so on. These rich physics are automatically captured by the mesh adaptation process leading to a very accurate numerical solution with high mesh...
resolution around all flow phenomena of importance. It is impossible to manually design such meshes and we can understand why manual meshing will lead to discretization errors and non mesh-converged solutions. Moreover, in the close-up views, we notice no influence of the periodic surface in the mesh adaptation showing that the periodic adaptation is working remarkably well. The mesh adaptation of the tip vortex near the tip wall is emphasized in Figure 39 where we show the refinements along the blade.

Like for the surface, the feature-based and the goal oriented mesh adaptations behave slightly differently with minor modifications in the mesh resolution of the different vortices, as well as the mesh density on the shocks near the tip. The anisotropy of the mesh in the vortex regions is, as expected, relatively limited but still enables a fine capturing of their motion. To analyze the impact of this small differences on the tip clearance structures, helicity thresholds near the tip region are plotted for the two different mesh adaptation approaches in Figure 42 (feature-based on the right and goal-oriented on the left). Two structured elsA predictions are also depicted, with 7M and 32M cells, for comparisons. Positive helicity (red colored) indicates a clockwise vortex, while negative helicity (blue colored) indicates a counter clockwise vortex, thus allowing to finely determine the origin of each observed vortex. For both mesh adaptation approaches, the flow picture is similar. Near the leading edge two main vortices are generated: the tip leakage vortex (positive helicity), due to the pressure gradient across the tip, and an induced vortex that rotates the opposite way. Further downstream, the main tip leakage vortex is pushed towards the neighboring blade, while the induced vortex rolls around it. However, leakage flow continues, forming the tip separation vortices and a second induced vortex appears. Finally, towards the end of the blade and after the shock, most of the vortices have dissipated. Between the two approaches, we can see that the goal-oriented solution focuses a bit more on the induced vortices, in agreement with the mesh views in Figure 39, highlighting the importance of the secondary flows for accurate predictions of compressor performances.

Comparing with the elsA simulations, it is readily observed that there is a reasonable agreement between Wolf and elsA near the leading-edge, irrespective of the cell counts in elsA. The main difference is the reduced induced vortex predicted by elsA. Further downstream, both elsA simulations indicate a completely dissipated induced vortex while the main tip leakage vortex appears to have also lost in intensity. The tip leakage vortex trajectory, however, is similar for all predictions. The dissipation of secondary flows in the structured meshes, compared to the mesh-adapted ones, is not surprising, as the meshing approach allows very limited control on where the refinement is applied. As a result, in this case, where only a global refinement of the structured mesh was performed, thus increasing the amount of cells around the field in a proportionate way, the tip leakage predictions do not significantly alter. An expert in structured mesh generation could potentially improve those results but this only highlights the difficulties in obtaining mesh converged results with structured meshes and the reliance in empirical meshing techniques.

One of the very interesting feature of mesh adaptation is its consistency with the boundary layer physics. Figure 40 depicts the $y^+$ value on the rotating blade for four adapted mesh sizes from 360 354 to 2 820 611 vertices obtained with the feature-based mesh adaptation. We immediately notice that, on the coarsest adapted mesh, we reach $y^+ = 1$ before the shock. This is a typical meshing best practice in the industry and the mesh adaptation process appears to automatically respect it. After the shock where the boundary layer is detached, mesh adaptation sets a $y^+$ around 2. We only notice larger $y^+$ value at the corner between the trailing edge and the hub, which is normal as the corner secondary flows in the area render the $y^+$ criterion and computation unreliable. The impressive point is that the same $y^+$ before and after the shock is prescribed on the finest adapted mesh stating clearly that the mesh adaptation converge in that region and do not over refine the boundary layer. The only region where the predicted physics are changing is where vortices interact with the boundary layer near the tip passage and in the region mention above.

Near stall operating point. Having looked at the flow field at the near-peak efficiency operating point, we now switch our attention to the experimental near-stall operating point, defined as 93% of the choked mass flow. A compressor rotor will typically go into stall either through an instability occurring near the tip (spike, rotating instabilities, ...) or through a corner separation, occurring at the junction between the hub and the blade. However, these phenomena are highly unsteady, hence predicting the exact entry of the rotor into stall with steady-state RANS is impossible. Nonetheless, steady RANS should be able to highlight the flow areas that are the more critical to lead to stall. The objective here is to evaluate how the compressor is more prone to enter in stall and potentially draw some conclusions on why mesh convergence is more difficult for this operating point.

Starting with the tip clearance vortices, Figure 43 depicts, for the near-stall operating point, the helicity plots at the tip clearance region for the two mesh adaptation approaches and the two structured elsA simulations with 7M and 32M cells (for consistency with the previous sections). Comparing to the near peak efficiency point (Figure 42), it is observed that all vortices appear to become more important in size and strength, occupying a larger part of the passage. Their position, however, remains similar and there are no particular signs on the vortices of a developing instability, for example potential spillage of the tip leakage flow of the adjacent blade passage ahead of the leading edge. Furthermore, past the shock the structures appear more dissipated, similarly to the near peak-efficiency operating point. The same differences between mesh-adapted and structured predictions are also present, notably the quick dissipation of the induced tip vortices past the leading edge for the structured meshes.

---

4 This physical phenomena is more isotropic.
To get more insights on the flow past the shock, an axial cut of the relative Mach number for the two mesh adaptation approaches and the two operating points, near peak efficiency and near stall, are presented side by side in Figure 41. For both mesh adaptation methods, the conclusions are similar: between the two operating points, near the hub there are very limited differences and a similarly sized shock-induced separation is predicted. This highlights that hub corner separation is unlikely to be the factor that leads to stall. On the other hand, near the tip, we can see that at the near-stall operating point (right blade on the two images), the size of the separation has increased and an additional separation appears at higher blade heights near the tip suction side. Additionally, the low velocity area around the tip, linked to the leakage flow, expands in the azimuthal direction while a small area low momentum fluid appears also on the pressure side at approximately 90% height. These findings hint that the stall comes near the tip, potentially through a tip clearance flow spillage towards the trailing edge of the neighboring blade [69]. The increased flow complexity and surface covered by the leakage flow implies that an increased mesh density is likely necessary to cover the area, even with mesh adaptation. Furthermore, as the flow in that area does not have strong privileged directions the algorithm has to refine this region with more isotropic elements, hence explaining the increased difficulty in getting full mesh convergence for these operating points.

Figure 22: NASA Rotor 37. Mesh convergence of the debit at the outlet for the feature-based (left) and the goal-oriented (right) mesh adaptation processes for all the imposed $P_{ratio}$ in the flow solver.
Figure 23: NASA Rotor 37. Mesh convergence of the pressure ratio during for the feature-based (left) and the goal-oriented (right) mesh adaptation processes for all the imposed $P_{ratio}$ in the flow solver.
Figure 24: NASA Rotor 37. Mesh convergence of the temperature ratio during the feature-based (left) and the goal-oriented (right) mesh adaptation processes for all the imposed $P_{ratio}$ in the flow solver.
Figure 25: NASA Rotor 37. Mesh convergence of the isentropic efficiency $\eta$ during the feature-based (left) and the goal-oriented (right) mesh adaptation processes for all the imposed $P_{\text{ratio}}$ in the flow solver.
Figure 26: NASA Rotor 37. Mesh convergence of the pressure ratio characteristic (top) and “normalized” pressure ratio characteristic (bottom) during the feature-based (left) and the goal-oriented (right) mesh adaptation processes while increasing the adaptive meshes complexity.
Figure 27: NASA Rotor 37. Mesh convergence of the isentropic efficiency characteristic (top) and "normalized" isentropic efficiency characteristic (bottom) during the feature-based (left) and the goal-oriented (right) mesh adaptation processes while increasing the adaptive meshes complexity.
Figure 28: NASA Rotor 37. Mesh convergence of the temperature ratio characteristic (top) and "normalized" temperature ratio characteristic (bottom) during the feature-based (left) and the goal-oriented (right) mesh adaptation processes while increasing the adaptive meshes complexity.
Figure 29: NASA Rotor 37. Mesh convergence of the pressure ratio characteristic (left) and the isentropic efficiency characteristic (right) for the structured elsA simulations.

Figure 30: NASA Rotor 37. Comparison of the feature-based and goal-oriented mesh adaptation results for an adapted mesh composed of 5M vertices with the elsA results for structured meshes composed of 7M and 32M cells. Pressure ratio (left) and isentropic efficiency (right) characteristics.
Figure 31: NASA Rotor 37. Radial profiles of the total pressure and total temperature ratio downstream of the rotor for the two main operating point: near peak efficiency (top) and near-stall (bottom). Both feature-based and goal-oriented results are for an adapted mesh composed of 5M vertices. Two different elsA predictions, with 7M and 32M cells, are presented.
Figure 32: NASA Rotor 37. Total pressure ratio for the both mesh adaptation approaches for 3 mass flow rates: MFR = 1.00, MFR = 0.98, and MFR = 0.93 (from top to bottom). For each picture, we plot the gap to finest solution obtained with the goal-oriented mesh adaptation. Three level of error are represented: 0.1% in black, 0.2% in brown and 0.5% in red.
Figure 33: NASA Rotor 37. Adapted surface meshes and associated solution (Mach number in rotating frame) for the feature-based (left) and the goal-oriented (right) mesh adaptation processes. Top, view of the top periodic surface. Bottom, view of the blade and the bottom periodic surface.
Figure 34: NASA Rotor 37. Adapted volume meshes and associated solution (Mach number in rotating frame) for the feature-based (left) and the goal-oriented (right) mesh adaptation processes with a cut plane close to the hubwall (plane equation $y = -0.195$). The domain is duplicated to highlight the effectiveness of the periodic adaptation. Top, global view, and bottom, zoom on the blades.
Figure 35: NASA Rotor 37. Adapted volume meshes and associated solution (Mach number in rotating frame) for the feature-based (left) and the goal-oriented (right) mesh adaptation processes with a cut plane at midspan (plane equation $y = -0.213$). The domain is duplicated to highlight the effectiveness of the periodic adaptation. Top, global view, and bottom, zoom on the blades.
Figure 36: NASA Rotor 37. Adapted volume meshes and associated solution (Mach number in rotating frame) for the feature-based (left) and the goal-oriented (right) mesh adaptation processes with a cut plane close to the tipwall (plane equation $y = -0.230$). The domain is duplicated to highlight the effectiveness of the periodic adaptation. Top, global view, and bottom, zoom on the blades.
Figure 37: NASA Rotor 37. Adapted volume meshes and associated solution (Mach number in rotating frame) for the feature-based (left) and the goal-oriented (right) mesh adaptation processes with a cut plane along the blade (plane equation $x = 0.009$). The domain is duplicated to highlight the effectiveness of the periodic adaptation.
Figure 38: NASA Rotor 37. Adapted volume meshes and associated solution (Mach number in rotating frame) for the feature-based (left) and the goal-oriented (right) mesh adaptation processes with a cut plane along the blade near the leading and the trailing edge (plane equation $0.330057x + 0.0152761y + 0.943837z + 0.009 = 0$). The domain is duplicated to highlight the effectiveness of the periodic adaptation.
Figure 39: NASA Rotor 37. Adapted volume meshes for the feature-based (left) and the goal-oriented (right) mesh adaptation processes. Cut planes orthogonal to the blade in order to emphasize the high mesh density to capture accurately the tip vortices.
Figure 40: NASA Rotor 37. $y^*$ provided by the feature-based mesh adaptation processes for different mesh sizes.

Figure 41: NASA Rotor 37. Relative mach number at an axial cut close to the leading edge for the near peak efficiency (left blade) and near stall point (right). Feature-based (left image) and goal-oriented (right image) approaches.
Figure 42: NASA Rotor 37. Helicity at axial slices near the tip clearance region for the two mesh adaptation approaches and two elsA simulations, near-peak efficiency. Positive (red-coloured) helicity indicates a vortex rotating clockwise while negative helicity (blue-coloured) indicates a counter-clockwise vortex.

Figure 43: NASA Rotor 37. Helicity at axial slices near the tip clearance region for the two mesh adaptation approaches and two elsA simulations, near-stall. Positive (red-coloured) helicity indicates a vortex rotating clockwise while negative helicity (blue-coloured) indicates a counter-clockwise vortex.
9. Conclusion and Future Works

This paper brings several major contributions in the field of CFD for turbomachinery applications. It has demonstrated that high-fidelity predictions can be obtained with unstructured meshes composed only of tetrahedra. It is not absolutely necessary to generate boundary layer meshes composed of prisms and/or hexahedra. The key point is to use unstructured anisotropic mesh adaptation where the error in the numerical solution is controlled by an error estimate. For the solution-adaptive process, it has been demonstrated that the final results are independent of the initial mesh, different error estimates converge toward the same solution, and that we can achieve mesh-converge results even in 3D for complex flows. Thanks to mesh adaptation the meshing process is greatly simplified, optimal and fully automatic.

In this regards, this paper brought several novelties and improvements in the design of solution-adaptive process for the RANS equations in turbomachinery.

On the solver side, we have presented a mixed Finite Element - Finite Volume formulation which is robust and accurate on highly anisotropic adapted meshes. We have presented our strategy to take into account periodicity which is treated through ghost entities to minimize the impact on the source code, and rotating frame. We have pointed out the importance of exact differentiation for the solver convergence when an implicit scheme is designed. And, we have proposed an efficient strategy to solve the linear system at each iteration which is based on Symmetric Gauss-Seidel (SGS) relaxation and an automatic CFL law control.

We have also presented our adjoint solver and how it takes into account periodicity. The adjoint problem turns out to be a stiff problem for RANS such that a GMRES solver with a weak preconditioner was not able to converge whatever the Krylov space size. To solve this issue, we proposed to use a preconditioner based on SGS relaxation which proved to be efficient if a large number of sweeps is performed. Several usual turbomachinery output functional have been differentiated as they can be used to govern the mesh adaptation process.

On the mesh adaptation side, we have recalled the two error estimates considered in this work: the feature-based and the goal-oriented ones. We have proposed a new mesh adaptation strategy to manage periodic domain. This choice has been made in order to preserve the geometry of the initial periodic domain and to minimize the impact on the local remesher software. It relies on two ingredients: i) local remesher which is able to handle no-manifold geometries and ii) two migration steps which can deal efficiently with highly anisotropic meshes. We have also proposed a mesh adaptation algorithm enabling mesh convergence study. In this algorithm, previous solutions are interpolated and used as restart to speed-up the overall process and make it efficient. A modified algorithm has been proposed when the complete characteristic is under study which manages automatically the increase in pressure ratio of the boundary conditions until stall.

This solution-adaptive process has been successfully applied to the LS89 blade and the NASA Rotor 37 cases. Both error estimates have been compared. For both cases, both on and off-design operating points were evaluated. A particular emphasis was put in the Rotor 37, as it is a compressor and thus naturally more prone of precision and robustness issues at off-design conditions. Its entire characteristic was simulated with the solution-adaptive process and was compared with predictions using standard structured hexahedral meshes. The results and comparisons to experimental measurements showed mesh convergence for the solution-adaptive process at limited vertex counts and highlighted that it provides an increased accuracy for a given number of DoFs compared to the standard structured meshes. The mesh adaptation refined around all relevant flow phenomena, notably the boundary layers, shocks, wakes and the secondary flows, providing a more accurate flow picture that can be of great help to the design engineer. Predictions at near stall operating points were also analyzed, with the results allowing to deduce a likely path towards stall for this particular compressor. Overall the viscous goal-oriented error estimate is a better choice but it requires a strong adjoint solver as the one presented in this work. If no adjoint is available, then the feature-based error estimate in L4-norm is still a good choice.

Regarding the perspectives of this work, one aim is to handle increased geometric complexity while another one is to handle increased physical complexity.

It is of great interest to analyze the impact of technological effects on the flow and to simulate more complex configurations such as a cooled turbine blade. Such geometries are very difficult (or impossible) to mesh with structured meshes while non-adapted unstructured meshes may suffer from precision issues. Another increased complexity can be to deal with a multi-stage turbomachinery configuration with several blade rows and how to couple it with mesh adaptation.

As mesh-convergence is achieved for a given turbulence model, an additional perspective is to compare the prediction of different turbulence models as no discretization error will impact the predictions. For instance, it will be interesting to compare the Spalart-Allmaras model with the Menter SST $k-\omega$ model. Seeing the efficiency of the actual process, the extension to URANS can be foreseen following the time-accurate mesh adaptation process presented in [6]. In a longer term, we need to extend this work to higher-order flow solvers and to tackle higher physical fidelity with DES or LES problems.
Acknowledgements

The authors want to acknowledge Adrien Loseille for providing software Feflo.a. This work was partially supported by funding from Safran Tech with technical monitors Frédéric Feyel and Vincent Brunet.

References


[34] B. Larroutourou. How to preserve the mass fractions positivity when computing compressible multi-component flows. RR-1080, INRIA, August 1989.


