Unstructured Mesh Generation Using Advancing Layers and Metric-Based Transition

David Marcum* and Frédéric Alauzet†

Center for Advanced Vehicular Systems (CAVS), Mississippi State University, MS 39762 USA
GAMMA3 Team, INRIA Paris-Rocquencourt, 78153, Le Chesnay, France

Keywords: viscous flow, unstructured mesh generation, anisotropic mesh, mesh displacement.

I. Introduction

Large-scale computational field simulation applications with unstructured meshes are widely used to help solve real world problems found in industry and government. In many of these cases the physics involved includes widely varying gradients. In particular, computational fluid dynamic (CFD) applications often involve significant flow field regions where viscous effects are dominant. These regions have gradients that vary by orders of magnitude in different directions. While viscous regions are typically prominent at the vehicle surface in the boundary-layer (BL) region, they can also be significant in the field with shear layers and vortical flows. In addition, high-speed flow fields can also include discontinuous phenomena, with shock waves and contact surfaces. These features may also have significant interaction with viscous dominated regions. High-resolution simulation of such cases with ideal isotropic discretization is simply not possible. Discretization of the field using anisotropic elements with length scales that match the gradients of the physics is the only known way to numerically solve such problems with a high level of resolution.

There are two primary approaches to generating unstructured meshes with anisotropic elements. The most general involves anisotropic triangulation using generalized metric terms. If the metrics are based upon the flow field as it evolves in the simulation then the result is a mesh adapted to the physics. Such a mesh typically has length scales that are directionally optimal for the given flow field. However, in BL regions the characteristics of a mesh generated using a generalized approach are not always ideal. Viscous BL regions near a surface often have very stringent numerical requirements as they involve high-gradient and non-linear physics that usually includes turbulence. These regions are known a priori and ideally suited to a pseudo-structured approach that generates elements with an advancing layer/normal type process. The resulting mesh is highly aligned, precisely spaced and very structured in at least the normal direction. Often such regions are generated with pentahedral and hexahedral elements for optimal flow solver efficiency. The characteristics of a pseudo-structured type mesh are ideal for the BL regions. Combining an advancing layers approach with a high-quality tetrahedral mesh generator for the field region results in an ideal approach for many applications. However, if there are significant field features then this approach requires an aligning surface for off-body features that may not be know a priori and it may be difficult to smoothly blend between attached and detached regions.

In this work we describe an approach that blends the two very different processes for a more elegant and optimal solution to overall problem. A good example application is an aircraft in a landing or takeoff condition as shown in Figure 1. In such cases there are numerous regions of viscous dominated flow with fully extended flaps and slats that interact with the main wing in widely varying ways and produce substantial detached viscous shear layers and vortices. Also, small features can be significant contributors to overall performance and high-resolution discretization of all features is essential to accurate simulation. In addition, anisotropy is required in multiple directions, not just normal to the surface. A blended approach that smoothly blends

---

*Professor, marcum@cavs.msstate.edu
†Visiting Professor (MSU) and Researcher, Frederic.Alauzet@inria.fr

American Institute of Aeronautics and Astronautics
the near body pseudo-structured region into a generalized anisotropic field region to capture attached and detached shear layers considerably improves the knowledge of and ability to predict such flow fields.

The unified approach taken here utilizes the best characteristics of both near body BL and generalized approaches. Specifically, near-body physics with anisotropy are resolved using an a priori pseudo-structured process\textsuperscript{21, 24} and off-body or field features are resolved using an adaptive generalized approach.\textsuperscript{9, 11, 17, 19} In particular the solution based metric tensor of the adaptive approach is augmented with a derived metric that is derived from the BL region of the pseudo-structured approach. The derived metric is based on local aspect ratio and geometry. This metric is then blended from the BL region into the overall field to allow for a smooth transition to the generalized field. The result is a flexible and optimal overall mesh generation process that can be used with or without adaptation. Results are presented here without adaptation that demonstrate the overall approach in the context of blending between the near body pseudo-structured region and the outer tetrahedral field region.

![Figure 1. Typical unstructured mesh and solution for a high-lift aircraft configuration.](image)

**II. Mesh generation algorithms**

**II.A. Advancing-Front and Local Reconnection (AFLR) algorithm**

The volume meshing process used in the present work is based on the Advancing-Front and Local Reconnection (AFLR) algorithm.\textsuperscript{21, 24} It has proven to be capable of generating high-quality unstructured volume meshes suitable for large-scale high-resolution CFD applications and is widely used in aerospace and other engineering disciplines.

The overall grid generation procedure used in the present work is a combination of automatic point creation, advancing type ideal point placement, and connectivity optimization schemes. A valid grid is maintained throughout the grid generation process. This provides a framework for implementing efficient local search operations using a simple data structure. It also provides a means for smoothly distributing the
desired point spacing in the field using a point distribution function. This function is propagated through
the field by interpolation from the boundary point spacing or by specified growth normal to the boundaries.
Points are generated using either advancing-front type point placement for isotropic elements, advancing-
point type point placement for isotropic right angle elements, or advancing-normal type point placement
for high-aspect-ratio elements. The connectivity for new points is initially obtained by direct subdivision
of the elements that contain them. Connectivity is then optimized by local-reconnection with a combined
Delaunay/min-max type (minimize the maximum angle, maximize the edge weight, etc.) type criterion. The
overall procedure is applied repetitively until a complete field grid is obtained.

The basic steps in the procedure are briefly outlined below. More complete details and results are
presented in.\textsuperscript{21,24}

1) Specify point spacing on the boundary surface geometry definition.
2) Generate a boundary surface grid (using a surface meshing version of AFLR\textsuperscript{23} or other suitable tool).
3) Generate a valid initial triangulation of the boundary surface points only and recover all boundary
surface faces.
4) Assign a point distribution function to each boundary point based on the local point spacing (edge
lengths). A background function can optionally be used to define the point distribution function (length
scale variation in the field).
5) For isotropic elements, generate points using advancing-front type point placement. Points are gener-
ated by advancing from the face of elements that only satisfy the point distribution function on one face.
6) For right angle elements, generate points using advancing-point type point placement. Points are
generated by advancing as in step 5, except multiple points are created by advancing along face normals
from the three face points of the satisfied edge/face.
7) For high-aspect-ratio elements, generate points using advancing-normal type point placement. Points
are generated one layer at a time from the boundaries by advancing along normals dependent upon the
boundary surface geometry.
8) Interpolate the point distribution function for new points from the containing elements. If geometric
growth is specified then the distribution function is determined from an approximate distance to the nearest
boundary and the specified geometric growth from that boundary.
9) Reject new points that are too close to other new points.
10) Insert the accepted new points by directly subdividing the elements that contain them. The resulting
triangulation is of very poor quality and will need to be improved.
11) Optimize the connectivity using local-reconnection (face swapping). For each element pair, compare
the reconnection criterion for all possible configurations and reconnect using the most optimal one. Repeat
this local-reconnection process until no elements are reconnected. A combined Delaunay and min-max type
criterion is used in three dimensions. This improves the overall grid quality substantially and overcomes most
of the problems associated with optimum local states that prohibit a global optimum from being obtained
with non-Delaunay criteria. The only characteristic required of the starting triangulation is that it be valid.
12) Repeat the point generation and local-reconnection process, steps 5 through 11, until no new points
are generated.
13) Smooth the coordinates of the field grid.
14) Optimize the connectivity using the local-reconnection process (step 11).

The algorithm above utilizes an isotropic length scale defined by the distribution function. For generalized
anisotropic elements (not pseudo-structured BL elements) an anisotropic length scale definition is required.
For the present work a generalized metric approach is used to generate anisotropic elements outside of the
BL region. Modifications to the original algorithm for anisotropy are accomplished by using a functional
approach for calculation of all geometric properties such as dot products, cross products, reconnection
criterion, etc. Those modifications are described in Section III.

II.B. Boundary layer meshing algorithm

Within the BL region a pseudo-structured mesh aligned with the boundary surface is optimal for accuracy
and performance of typical CFD solvers. This is the approach taken in the present work. While the standard
unstructured meshing procedure previously described can be utilized to generate pseudo-structured elements
in the BL region, an open (extrusion) or closed (displacement) approach is far more efficient. A modified
procedure using an advancing-normal/layers approach\textsuperscript{21,22} is used for volume mesh generation. In this
approach, the element connectivity is generated along with new points in high-aspect-ratio regions. Local-reconnection is not used to determine the connectivity in these regions. Instead, the connectivity is directly determined as each new point is generated. This produces a very structured connectivity and allows the tetrahedral elements to be easily combined into structured type elements. Typically, the majority of the tetrahedral elements within the high-aspect-ratio region can be combined into prismatic elements. If the surface mesh contains quad faces then hexahedral elements can be formed. The outer layer of this region may require some pyramid elements to match the outer tetrahedral element region. In all cases, the pentahedral elements have strict node, edge, and face matching to each other and to neighboring tetrahedral elements. Hexahedral elements may have either full matching with an attached pyramid element or split-face matching without.

The basic steps in the advancing-normal procedure are listed below.

1) Determine a normal vector at each active BL point. Initially the normals are based solely on the original boundary surface geometry. As the generation advances the normals are generated using the geometry of the outer layer of the BL grid.

2) Smooth the normal vectors with a weighting dependent upon the distance from the boundary. Initially the normals are not smoothed. At the estimated end of the boundary layer region full smoothing is applied.

3) Generate new points one layer at a time. For efficiency multiple layers can be generated in one step. Particularly early in the BL mesh generation process when the normal spacing is very small in comparison to the surface tangential spacing. New points are created along the normal vector with the normal spacing determined using geometric growth from the boundary surface.

4) Check distance between new points and surrounding element quality. The volume triangulation or oct-tree structure is used to efficiently check nearby points. As boundary layers merge new points may be too close and advancement should terminate locally. A new point is rejected if the distance between it and any nearby new (or existing) point is less than a preset fraction of the local element length scale. Boundary-layer advancement is terminated locally if a new point is rejected.

5) New points are also rejected if any of the surrounding elements that they may produce fail a quality check (e.g., maximum angle $< 160^\circ$). Boundary-layer advancement is terminated locally if a new point is rejected for quality.

6) Check element aspect ratio. As the grid advances and the normal spacing increases the element aspect ratio will eventually be isotropic. Boundary layer advancement is terminated locally when the aspect ratio on the next layer would be greater than a preset factor.

7) Attach accepted new points to the volume triangulation. New points are connected and attached to the existing element that contains them.

8) Generate a new boundary surface grid by inflating the previous surface at points that have continued to advance.

9) Repeat steps 1 through 8 until no new points are accepted.

10) Form BL elements and merge with the outer tetrahedral element mesh.

For the present work the above algorithm does not require a generalized anisotropic metric as the \textit{a priori} assumption of BL gradients is part of the process. In an adaptive process the tangential and normal spacing could be adapted using a generalized metric approach\textsuperscript{9,11,17,19} by treating each separately. In a typical complex aerospace configuration the elements at the edge of the BL region often have considerable anisotropy from the normal spacing not having reached outer region length scales. Further, optimal surface meshes for high-resolution of leading edge type or high-curvature regions require a structured or pseudo-structured mesh aligned with the curvature. Example surface meshes are shown for a wing-body in Figure 6 and for a nacelle in Figure 10. As with BL regions, aligned pseudo-structured quad-faces or right-angle-tria-faces are optimal. Surface aspect-ratios for realistic configurations often involve anisotropy similar in scale to BL regions. Unfortunately this results in issues with transition to isotropic elements that are more ideal in the outer field region. This creates practical limits on anisotropy of the surface faces to order ten. Consequently the combined pseudo-structured and generalized anisotropic meshing approach is advocated in this work to eliminate these limits. In the overall implementation a metric is required to accomplish the blending from BL to outer region. This metric is derived from the normal and tangential spacing of BL region elements in the outer layer. A discussion of this is provided in Section III.C.
III. Metric-based transition for viscous flowfields

This section deals with the blending or the transition of the mesh between the pseudo-structured BL region mesh and the generalized outer mesh region. We propose to use metric-based anisotropic mesh operators to generate a smooth anisotropic transition between the boundary layer and the rest of the flow field. These operators have been widely used successfully for anisotropic mesh adaptation.\(^4, 7, 8, 12, 14, 15, 19, 25\)

The blending requires to compute a metric field associated with the last layer of the boundary layer, to extrapolate this metric field into the flow field and to write the operators of Section II.A into the metric space, in other words, all geometric quantities thus quality function are computed in the given metric space. Moreover, the addition of metric-based operators provides the necessary framework and capability for adaptation with generalized anisotropic elements.

III.A. Basics of metric and notion of unit mesh

III.A.1. Euclidean metric space

For the sake of clarity, we recall the differential geometry notions that are used in the sequel. We use the following notations: bold face symbols, as \( \mathbf{a}, \mathbf{b}, \mathbf{u}, \mathbf{v}, \mathbf{x}, \ldots \), denote vectors or points of \( \mathbb{R}^3 \). The natural dot and cross product between two vectors \( \mathbf{u} \) and \( \mathbf{v} \) of \( \mathbb{R}^3 \) are denoted by: \( \mathbf{u} \cdot \mathbf{v} = \langle \mathbf{u}, \mathbf{v} \rangle \) and \( \mathbf{u} \times \mathbf{v} \).

An Euclidean metric space \((\mathbb{R}^3, \mathcal{M})\) is a vector space of finite dimension where the dot product is defined by means of a symmetric definite positive tensor \( \mathcal{M} \):

\[
\mathbf{u} \cdot \mathcal{M} \mathbf{v} = \langle \mathbf{u}, \mathbf{v} \rangle_{\mathcal{M}} = \langle \mathbf{u}, \mathcal{M} \mathbf{v} \rangle = \mathbf{u}^t \mathcal{M} \mathbf{v}, \quad \text{for } (\mathbf{u}, \mathbf{v}) \in \mathbb{R}^3 \times \mathbb{R}^3. \tag{1}
\]

In the following, the matrix \( \mathcal{M} \) is simply called a metric tensor or a metric. The dot product defined by \( \mathcal{M} \) makes \( \mathbb{R}^3 \) become a normed vector space \((\mathbb{R}^3, \|\cdot\|_{\mathcal{M}})\) and a metric vector space \((\mathbb{R}^3, d_{\mathcal{M}}(\cdot, \cdot))\) supplied by the following norm and distance definitions:

\[
\forall \mathbf{u} \in \mathbb{R}^3, \|\mathbf{u}\|_{\mathcal{M}} = \sqrt{\langle \mathbf{u}, \mathcal{M} \mathbf{u} \rangle} \quad \text{and} \quad \forall (\mathbf{a}, \mathbf{b}) \in \mathbb{R}^3 \times \mathbb{R}^3, \, d_{\mathcal{M}}(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|_{\mathcal{M}}. \tag{2}
\]

As metric tensor \( \mathcal{M} \) is symmetric, it is diagonalizable in an orthonormal basis:

\[
\mathcal{M} = \mathcal{T} \Lambda \Lambda^t \mathcal{T},
\]

where \( \mathcal{T} \) is an orthonormal matrix the lines of which are composed of the eigenvectors \((\mathbf{v}_i)_{i=1,3}\) of \( \mathcal{M} \) verifying \( \mathcal{T}^t \mathcal{R} \mathcal{T} = \mathcal{R}^t \mathcal{T} = \mathcal{I}_3 \). \( \Lambda = \text{diag}(\lambda_i) \) is a diagonal matrix composed of the eigenvalues of \( \mathcal{M} \), denoted \( (\lambda_i)_{i=1,3} \), which are strictly positive. From the previous definition, we deduce that application \( \Lambda^{\frac{1}{2}} \mathcal{R} \) where \( \Lambda^{\frac{1}{2}} = \text{diag}(\lambda_i^{\frac{1}{2}}) \) defines the mapping from the physical space \((\mathbb{R}^3, \mathcal{I}_3)\), where \( \mathcal{I}_3 \) is the identity matrix, to the Euclidean metric space \((\mathbb{R}^3, \mathcal{M})\):

\[
\Lambda^{\frac{1}{2}} \mathcal{R} : \quad (\mathbb{R}^3, \mathcal{I}_3) \quad \longrightarrow \quad (\mathbb{R}^3, \mathcal{M})
\]

\[
\mathbf{x} \quad \longrightarrow \quad (\Lambda^{\frac{1}{2}} \mathcal{R}) \mathbf{x}.
\]

And, we trivially recover: \( \mathbf{u} \cdot_{\mathcal{M}} \mathbf{v} = \mathcal{T} \left( (\Lambda^{\frac{1}{2}} \mathcal{R}) \mathbf{u} \right) \cdot \left( (\Lambda^{\frac{1}{2}} \mathcal{R}) \mathbf{v} \right) = \mathbf{u}^t \mathcal{M} \mathbf{v} \).

We are now able to deduce the following expression of the 3D cross product\(^a\) with respect to \( \mathcal{M} \):

\[
\mathbf{u} \times_{\mathcal{M}} \mathbf{v} = \left( \Lambda^{\frac{1}{2}} \mathcal{R} \right) \mathbf{u} \times \left( \Lambda^{\frac{1}{2}} \mathcal{R} \right) \mathbf{v} = \sqrt{\det \mathcal{M}} \left( \mathcal{R} \Lambda^{-\frac{1}{2}} \right) (\mathbf{u} \times \mathbf{v}). \tag{3}
\]

In an Euclidean metric space, volumes and angles are still well defined. These features are of main interest when dealing with meshing. For instance using Relations (1) and (3), given a parallelepiped \( K \) of \( \mathbb{R}^3 \), the volume of \( K \) computed with respect to metric tensor \( \mathcal{M} \) is:

\[
|K|_{\mathcal{M}} = \mathbf{u} \cdot_{\mathcal{M}} (\mathbf{v} \times_{\mathcal{M}} \mathbf{w}) = \sqrt{\det \mathcal{M}} |K|_{\mathcal{I}_3}, \tag{4}
\]

\(^a\)In 2D, we obtain the following relation: \( \mathbf{u} \times_{\mathcal{M}} \mathbf{v} = \sqrt{\det \mathcal{M}} (\mathbf{u} \times \mathbf{v}) \).
where $|K|_{I_2}$ is the Euclidean volume of $K$. The angle between two non-zero vectors $\mathbf{u}$ and $\mathbf{v}$ is defined by the unique real-value $\theta_M \in [0, \pi]$ verifying:

$$\cos(\theta_M) = \frac{\langle \mathbf{u}, \mathbf{v} \rangle_M}{\|\mathbf{u}\|_M \|\mathbf{v}\|_M}.$$

In three dimensions, dihedral angles\(^b\) of a tetrahedron can be computed in a given Euclidean space from this definition as it is the angle between the faces normal. These relations are very useful in 3D for computing the area of faces with respect to $M$, let $F$ be a triangle in $\mathbb{R}^3$:

$$2|F|_M = \| (\Lambda^\frac{1}{2} R) \mathbf{u} \times (\Lambda^\frac{1}{2} R) \mathbf{v} \| = \|\mathbf{u}\|_M \|\mathbf{v}\|_M \sin(\theta_M)$$

or

$$4|F|_M^2 = \| (\Lambda^\frac{1}{2} R) \mathbf{u} \times (\Lambda^\frac{1}{2} R) \mathbf{v} \|^2 = \|\mathbf{u}\|_M^2 \|\mathbf{v}\|_M^2 - \langle \mathbf{u}, \mathbf{M}\mathbf{v} \rangle^2.$$

**Geometric interpretation.** We will often refer to the geometric interpretation of a metric tensor. In the vicinity $V(a)$ of point $a$, the set of points that are at distance $\varepsilon$ of $a$, is given by:

$$\Phi_M(\varepsilon) = \{ \mathbf{x} \in V(a) \mid ^t(x - a) \mathbf{M} (x - a) = \varepsilon^2 \}.$$

The above relation defines an ellipsoid centered at $a$ with its axes aligned with the eigen directions of $\mathcal{M}$. Sizes along these directions are given by $h_i = \lambda_i^{-\frac{1}{2}}$. In the sequel, the set $\Phi_M(1)$ is called the **unit ball** of $\mathcal{M}$ and we denote by $B_M$. This ellipsoid depicted in Figure 2 (left). Notice that application $\Lambda^\frac{1}{2} R$ maps $B_M$ from physical space into the unit ball in the metric space and, conversely, application $^t R \Lambda^{-\frac{1}{2}}$ maps the unit ball into $B_M$, see Figure 2 (right).

**Figure 2.** Left, geometric interpretation of the unit ball $B_M$ where $\mathbf{v}_i$ are the eigenvectors of $\mathcal{M}$ and $\lambda_i = h_i^{-2}$ are the eigenvalues of $\mathcal{M}$. Right, mappings between physical space ($\mathbb{R}^3, I_2$) and Euclidean metric space ($\mathbb{R}^3, M$).

### III.A.2. Riemannian metric space

In differential geometry, a Riemannian manifold or Riemannian space $(\mathcal{M}, \mathcal{M})$ is a smooth manifold $M$ in which each tangent space is equipped with a dot product defined by a metric tensor $\mathcal{M}$, a Riemannian metric, in a manner which varies smoothly from point to point. Even if no global definition of the scalar product exists, various geometric notions can be defined on a Riemannian manifold such as angles, lengths of curves, areas (or volumes), curvatures, gradients of functions and divergences of vector fields. For instance, the distance between two points $\mathbf{x}$ and $\mathbf{y}$ is given by the length of the curve which locally joins these points along the shortest path: the geodesic.

In the context of mesh adaptation, we do not know any manifold, hence any Riemannian manifold. For our concern, we work with a simpler object called a **Riemannian metric space** defined by $M = (\mathcal{M}(\mathbf{x}))_{\mathbf{x} \in \Omega}$. In that specific case, we only know $M$ a Riemannian metric and $\Omega \subset \mathbb{R}^n$ a common space of parametrization which is our computational domain. There is still no global notion of scalar product. This mathematical object can be assimilated to a function that can represents a set of Cartesian surface (or graph surface):

\(^b\)The dihedral angle is the angle between two triangular faces of a tetrahedron.
Evaluating geometrical quantities in a Riemannian metric space is equivalent to evaluate these quantities on the underlying Cartesian surfaces. Broadly speaking, a Riemannian metric space curves the parametrization space, i.e., the computational domain.

Now let us extend the notions of length and volume defined in Section III.A.1 for Euclidean metric spaces to Riemannian metric spaces which will be the main operations performed by the mesh generator in such spaces. Fortunately, these notions can be easily derived in the context of meshing because we are not interested in evaluating these quantities on the Riemannian manifold. Indeed, as regards edge length computation, we do not want to compute the distance between two points which requires to find the shortest path on the curved manifold between these two points and to compute the length of the geodesic. We want to compute the length of the path between these two points defined by the straight line parameterization. To take into account the variation of the metric along the edge, the edge length is evaluated with an integral formula. Formally speaking, in Riemannian metric space $\mathcal{M} = (\mathcal{M}(x))_{x \in \Omega}$, the length of edge $ab$ is computed using the straight line parameterization $\gamma(t) = a + tab$, where $t \in [0,1]$:

$$
\ell_{\mathcal{M}}(ab) = \int_{0}^{1} \|\gamma'(t)\|_{\mathcal{M}} \, dt = \int_{0}^{1} \sqrt{\det \mathcal{M}(a + t ab) \, ab} \, dt.
$$

The notion of volume can also be extended to Riemannian metric spaces. Given a bounded subset $K$ of $\Omega$, the volume of $K$ computed with respect to $(\mathcal{M}(x))_{x \in \Omega}$ is:

$$
|K|_{\mathcal{M}} = \int_{K} \sqrt{\det \mathcal{M}(x)} \, dx.
$$

III.A.3. Metric-based mesh generation

To generate anisotropic meshes, one must be able to prescribe at each point of the domain privileged sizes and orientations for the elements. This information will be transmitted to the mesh generator which will try to best fit these demands. The use of Riemannian metric spaces is an elegant and efficient way to achieve this goal. Indeed, it is possible for a mesher to work in such spaces as geometric quantities are well-posed. The main idea of metric-based mesh adaptation, initially introduced in, is to generate a unit mesh in a prescribed Riemannian metric space.

UNIT ELEMENT. A tetrahedron $K$, defined by its list of edges $(e_{i})_{i=1,6}$, is unit with respect to a metric tensor $\mathcal{M}$ if the length of all its edges is unit in metric $\mathcal{M}$:

$$
\forall i = 1, \ldots, 6, \quad \ell_{\mathcal{M}}(e_{i}) = 1 \text{ with } \ell_{\mathcal{M}}(e_{i}) = \sqrt{\mathcal{M} e_{i} e_{i}}.
$$

If $K$ is composed only of unit length edges, then its volume $|K|_{\mathcal{M}}$ in $\mathcal{M}$ is constant equal to:

$$
|K|_{\mathcal{M}} = \frac{\sqrt{2}}{12} \text{ and } |K| = \frac{\sqrt{2}}{12} (\det(\mathcal{M}))^{-\frac{1}{2}},
$$

where $|K|$ is its Euclidean volume.

UNIT MESH. The notion of unit mesh is far more complicated than the notion of unit element as the existence of a mesh composed only of unit regular simplices with respect to a given Riemannian metric space is not guaranteed. For instance, if the Riemannian metric space is not compatible with the domain size, then, such a discrete mesh clearly does not exist. Another simple counter example is the following. Let us consider the canonical Euclidean space $(I^{3}(x))_{x \in \mathbb{R}^{3}}$. Then, it is well known that $\mathbb{R}^{3}$ cannot be filled only with the regular tetrahedron. Consequently, even for the simplest Riemannian metric space $(I^{3}(x))_{x \in \mathbb{R}^{3}}$, there is no discrete mesh composed only of the unit regular tetrahedron. Therefore, the notion of unit mesh has to be relaxed.

A discrete mesh $\mathcal{H}$ of a domain $\Omega \subset \mathbb{R}^{n}$ is a unit mesh with respect to Riemannian metric space $(\mathcal{M}(x))_{x \in \Omega}$ if all its elements are quasi-unit. The definition of unity for an element is relaxed by taking into account technical constraints imposed by mesh generators. To avoid cycling while analyzing edges lengths, a tetrahedron $K$ defined by its list of edges $(e_{i})_{i=1,6}$ is said to be quasi-unit if, $\forall i$, $\ell_{\mathcal{M}}(e_{i}) \in \left[\frac{1}{\sqrt{2}}, \sqrt{2}\right]$. The study in shows that several non-regular space filling tetrahedra verify this constraint, which guarantees...
the existence for constant Riemannian metric space. Unfortunately, this weaker constraint on edges lengths can lead to the generation of quasi-unit elements with a null volume. Consequently, controlling only the edges length is not sufficient, the volume must also be controlled to relax the notion of unit element which is practically achieved by managing a quality function.

**Generating adapted anisotropic meshes.** Using the previous notions, the problem of mesh generation can be considerably simplified. Indeed, whatever the kind of desired mesh (uniform, adapted isotropic, adapted anisotropic), the mesh generator will always generate a unit mesh in the prescribed Riemannian metric space.\(^{11}\) Consequently, the generated mesh is uniform and isotropic in the Riemannian metric space while it is adapted and anisotropic in the Euclidean space. This is illustrated in Figure 3. This idea has turned out to be a huge breakthrough in the generation of anisotropic adapted meshes.

\[
\begin{align*}
\text{Inputs} &\quad (\mathcal{H}_0, \mathcal{M}_i)_{i \in \mathcal{H}} \\
\text{Output} &\quad \mathcal{H}
\end{align*}
\]

![Figure 3. Metric-based mesh generation. Left, specified Riemannian metric space. Right, unit mesh in the prescribed Riemannian metric space which becomes adapted anisotropic in the Euclidean space.](image)

**III.B. Practical use of metrics**

The main advantage when working with metric spaces is the well-posedness of operations on metric tensors which enable us to manage directional sizes. These operations have a straightforward geometric interpretation when considering the ellipsoid associated with a metric. In this section, the practical uses of metrics inside the mesh generator is described.

**III.B.1. Metric Interpolation**

In practice, the Riemannian metric space or metric field is only known discretely at mesh vertices. The definition of an interpolation procedure on metrics is therefore mandatory to be able to compute the metric at any point of the domain. For instance, the computation of the volume of an element in a Riemannian metric space given by Relation (7) can be done using quadrature formula which require the computation of some interpolated metrics inside the considered element. Figure 4 illustrates metric interpolation along a segment, for which the initial data are the endpoints metrics.

Several interpolation schemes have been proposed in which are based on the simultaneous reduction. The main drawback of these approaches is that the interpolation operation is not commutative. Hence, the result depends on the order in which the operations are performed when more than two metrics are involved. Moreover, such interpolation schemes do not satisfy useful properties such as the maximum principle. Consequently, to design an interpolation scheme on these objects, one needs a consistent operational framework. We suggest to consider the log-Euclidean framework introduced in.\(^5\)

**Log-Euclidean framework.** We first define the notion of metric logarithm and exponential:

\[
\ln(\mathcal{M}) := \mathcal{R} \ln(\mathcal{A}) \mathcal{R} \quad \text{and} \quad \exp(\mathcal{M}) := \mathcal{R} \exp(\mathcal{A}) \mathcal{R},
\]
where $\ln(\Lambda) = \text{diag}(\ln(\lambda_i))$ and $\exp(\Lambda) = \text{diag}(\exp(\lambda_i))$. We can now define the logarithmic addition $\oplus$ and the logarithmic scalar multiplication $\odot$:

\[
\mathcal{M}_1 \oplus \mathcal{M}_2 := \exp(\ln(\mathcal{M}_1) + \ln(\mathcal{M}_2))
\]

\[
\alpha \odot \mathcal{M} := \exp(\alpha \ln(\mathcal{M})) = \mathcal{M}^\alpha.
\]

The logarithmic addition is commutative and coincides with matrix multiplication whenever the two tensors $\mathcal{M}_1$ and $\mathcal{M}_2$ commute in the matrix sense. The space of metric tensors, supplied with the logarithmic addition $\oplus$ and the logarithmic scalar multiplication $\odot$ is a vector space.

**Metric interpolation in the log-Euclidean framework.** Let $(x_i)_{i=1...k}$ be a set of vertices and $(\mathcal{M}(x_i))_{i=1...k}$ their associated metrics. Then, for a point $x$ of the domain such that:

\[
x = \sum_{i=1}^{k} \alpha_i x_i \quad \text{with} \quad \sum_{i=1}^{k} \alpha_i = 1,
\]

the interpolated metric is defined by:

\[
\mathcal{M}(x) = \bigoplus_{i=1}^{k} \alpha_i \odot \mathcal{M}(x_i) = \exp \left( \sum_{i=1}^{k} \alpha_i \ln(\mathcal{M}(x_i)) \right).
\]

(8)

This interpolation is commutative. Moreover, it has been demonstrated in\(^5\) that this interpolation preserves the maximum principle, i.e., for an edge $pq$ with endpoints metrics $\mathcal{M}(p)$ and $\mathcal{M}(q)$ such that $\det(\mathcal{M}(p)) < \det(\mathcal{M}(q))$ then we have $\det(\mathcal{M}(p)) < \det(\mathcal{M}(p + t pq)) < \det(\mathcal{M}(q))$ for all $t \in [0,1]$.

**Remark 1** The interpolation formulation (8) reduces to

\[
\mathcal{M}(x) = \prod_{i=1}^{k} \mathcal{M}(x_i)^{\alpha_i},
\]

if all the metrics commute. Therefore, an arithmetic mean in the log-Euclidean framework could be interpreted as a geometric mean in the space of metric tensors.

![Figure 4](image.png)

**Figure 4.** Metric interpolation along a segment where the endpoints metrics are the blue and violet ones.

**III.B.2. Numerical computation of geometric quantities in Riemannian metric spaces**

In this section, we describe how geometric quantities are computed numerically in Riemannian metric space as such quantities involve integrals. Let $\mathcal{M}$ be a discrete metric field defined at the vertices of a mesh $\mathcal{H}$ of a domain $\Omega_h$. Thanks to the interpolation operation, we have a continuous metric field in the whole domain, i.e., a Riemannian metric space $(\mathcal{M}(x))_{x \in \Omega_h}$. This representation of the metric field depends on $\mathcal{H}$ as the interpolation law is applied at the element level.
Computation of Edge Length. Approximation can be used to evaluate edge length in Riemannian metric space given by Relation (6). However, an analytical expression can be obtained if we consider that the metric field conforms to a geometric variation law as described in Section III.B.1.

Let \( \mathbf{e} = \mathbf{p}_1 \mathbf{p}_2 \) be an edge of the mesh of Euclidean length \( \| \mathbf{e} \|_2 \), and \( \mathcal{M}(\mathbf{p}_1) \) and \( \mathcal{M}(\mathbf{p}_2) \) be the metrics at the edge extremities \( \mathbf{p}_1 \) and \( \mathbf{p}_2 \). We denote by \( \ell_{\mathcal{M}}(\mathbf{e}) = \sqrt{\mathbf{e}^T \mathcal{M}(\mathbf{p}) \mathbf{e}} \) the length of the edge in metric \( \mathcal{M}(\mathbf{p}_1) \). We assume \( \ell_{\mathcal{M}_1}(\mathbf{e}) > \ell_{\mathcal{M}_2}(\mathbf{e}) \) and we set \( a = \frac{\ell_{\mathcal{M}_1}(\mathbf{e})}{\ell_{\mathcal{M}_2}(\mathbf{e})} \). The restriction of the (multi-dimensional) metric interpolation operator given by Relation (8) to an edge \( \mathbf{e} = \mathbf{p}_1 \mathbf{p}_2 \) leads to a geometric interpolation law on eigen values \( \lambda \) and size \( h \):

\[
\lambda(t) = \exp ((1-t) \ln(\lambda_1) + t \ln(\lambda_2)) = \lambda_1^{1-t} \lambda_2^t \quad \iff \quad h(t) = h_1^{1-t} h_2^t.
\]

Under these assumption, we deduce:

\[
\ell_{\mathcal{M}}(\mathbf{e}) = \ell_{\mathcal{M}_1}(\mathbf{e}) a - \frac{1}{a \ln(a)}.
\]

We also recall that the prescribed size in direction \( \mathbf{e} \) is

\[
h_{\mathcal{M}}(\mathbf{e}) = \frac{\| \mathbf{e} \|_2}{\ell_{\mathcal{M}}(\mathbf{e})}.
\]

Computation of Element Volume. The evaluation of a tetrahedron volume in a Riemannian metric space consists in computing numerically Integral (7). Higher order approximation can be obtained by using Gaussian quadrature and metric interpolation based on the Log-Euclidean framework. For instance, if one considers a \( k \)-point Gaussian quadrature with weights \( (\omega_j)_{j=1...k} \) and barycentric coefficients \( (\beta_{1}, \beta_{2}, \beta_{3}, \beta_{4})_{j=1...k} \), it yields:

\[
|K|_{\mathcal{M}} \approx |K|_{\mathcal{T}_3} \sum_{j=1}^{k} \omega_j \sqrt{\det \left( \exp \left( \sum_{i=1}^{4} \beta_{i}^j \log (\mathcal{M}_i) \right) \right)}.
\]

However, for faster volume evaluation, only a first order approximation is considered in this work:

\[
|K|_{\mathcal{M}} = \int_K \sqrt{\det \mathcal{M}} d\mathbf{x} \approx \sqrt{\det \mathcal{M}_K} |K|_{\mathcal{T}_3} = |K|_{\mathcal{M}_K} \text{ where } \mathcal{M}_K = \exp \left( \frac{1}{4} \sum_{i=1}^{4} \log (\mathcal{M}_{\mathbf{p}_i}) \right).
\]

III.B.3. Metric-based quality function

In this section, we propose the generalization of several quality functions to metric spaces. Two ways are possible to compute these quality functions in metric spaces.

The first approach consists in mapping all vectors in metric spaces thanks to mapping \( \Lambda^\frac{1}{2} \mathcal{R} \) and then compute quality functions using their classical formulation. This approach is easy to implement but requires to compute each time \( \Lambda^\frac{1}{2} \mathcal{R} \). It is also limited to evaluation in Euclidean metric spaces.

The second approach computes directly the quality function expression in metric space like for the length Formula (9). Then, metric variation may be taken into account. To this end, we propose to express all quality functions only in terms of dot products (and edge lengths) and volumes. In the following discussion various metric-based quality criteria are presented. These criteria or measures are by no means exhaustive and various mesh generators and solver pre-processors utilize differing ones.

Volume-length ratio. The first quality function commonly used in metric-based anisotropic mesh adaptation \( ^8,10,19,25 \) is the volume-length ratio which easily extend to metric space:

\[
Q_{\mathcal{M}}(K) = \frac{36}{3^\frac{2}{3}} \frac{|K|_{\mathcal{M}}^\frac{2}{3}}{\sum_{i=1}^{6} \ell_{\mathcal{M}}^i(\mathbf{e}_i)},
\]

where edge lengths and volume are computed according to Relations (9) and (10). Or, edge lengths and volume can be computed using Relations (2) and (4) assuming the metric is constant over element \( K \) and given by \( \mathcal{M}_K \).
MINIMUM EDGE WEIGHT FUNCTION. This quality function corresponds to the weight\(^6\) for an element edge shared by two faces:

\[
Q(K) = \min_{i=1,2} \frac{\|e_i\|_2}{\tan \theta_i} = \max_{i=1,2} \frac{\mathbf{n}_{F_{i,1}} \cdot \mathbf{n}_{F_{i,2}}}{6|K|},
\]

where \(\theta_i\) is the dihedral angle between two faces \(F_{i,1}\) and \(F_{i,2}\) sharing edge \(e_i\). It represents the geometric contribution to the value of the diagonal terms in a solution matrix for an elliptic equation. Consequently, it represents a quantity that can degrade performance. This is directly analogous to minimizing the maximum angle in 2D and similar to that in 3D. This quality function can be written only in function of dot products, edge lengths and volume as:

\[
\mathbf{n}_{F_{i,1}} \cdot \mathbf{n}_{F_{i,2}} = \mathbf{e}_i \cdot (\mathbf{e}_j \times (\mathbf{e}_k \times \mathbf{e}_i)) = \mathbf{e}_i \cdot ((\mathbf{e}_j \cdot \mathbf{e}_k)\mathbf{e}_i - (\mathbf{e}_j \cdot \mathbf{e}_i)\mathbf{e}_k).
\]

To compute this quality function in Riemannian metric space, we choose a first order approximation assuming the metric is constant over element \(K\) and given by \(\mathcal{M}_K\):

\[
Q_{\mathcal{M}}(K) = \max_{i=1,2} \frac{(\mathbf{e}_i, \mathcal{M}_K \mathbf{e}_j)(\mathbf{e}_i, \mathcal{M}_K \mathbf{e}_k) - \ell^2_{\mathcal{M}_K}(\mathbf{e}_j)(\mathbf{e}_j, \mathcal{M}_K \mathbf{e}_k)}{\sqrt{\det \mathcal{M}_K} |K|}.
\]

TETRAHEDRON DIHEDRAL ANGLE FUNCTION. The maximum value corresponds to the function for the edge with the minimum dihedral angle between the two faces that share the edge. The maximum value corresponds to the function for the edge with the maximum dihedral angle between the two faces that share the edge:

\[
Q(\mathbf{e}_i) = \cos \theta_i = \frac{\mathbf{n}_{F_{i,1}} \cdot \mathbf{n}_{F_{i,2}}^2}{\|\mathbf{n}_{F_{i,1}}\|^2 \|\mathbf{n}_{F_{i,2}}\|^2} = \frac{\mathbf{n}_{F_{i,1}} \cdot \mathbf{n}_{F_{i,2}}}{16|F_{i,1}|^2|F_{i,2}|^2},
\]

which can be rewritten

\[
Q(\mathbf{e}_i) = \frac{(\ell^2_{ij} \ell^2_{ik} - \ell^2_{ik} \ell^2_{jk})(\ell^2_{ij} \ell^2_{ik} - \ell^2_{ij} \ell^2_{jk})}{(\ell^2_{ij}^2 - \ell^2_{ij})^2 (\ell^2_{ik}^2 - \ell^2_{ik})^2}.
\]

The maximum element angle is useful as a basic measure of quality for a given element. Particularly at extremes. Large dihedral angles always correspond to low quality elements as defined by any quality criteria.
III.C. Metric-based anisotropic mesh transition

This section presents the new blended approach that smoothly blends the near-body pseudo-structured region into a generalized anisotropic field region. The unified approach taken here utilizes the best characteristics of both near body BL with pseudo-structured elements and field region unstructured mesh with metric-based anisotropic element.

First, the BL region mesh is generated using the advancing-normal BL mesh generation algorithm. Second, the unstructured anisotropic mesh of the outer field region is generated utilizing the advancing-front based/local-reconnection based approach. Here, the metric field does not come from an error estimate as in mesh adaptation but, instead, it is derived on the fly by interpolation when each new vertex is inserted into the volume mesh. The initial metric values are specified to be isotropic at all surface points except at the interface between the BL and outer tetrahedral region. That metric is determined from the normal and tangential spacing at the edge of the BL region as shown in Figure 5.

Two different methods have been tested. For the first one, called metric-based transition based on metric interpolation, each time a new vertex \( x \) is inserted in tetrahedron \( K \) its metric is simply interpolated from the metric of the vertices of \( K \):

\[
\mathcal{M}(x) = \frac{\exp \left( \sum_{p_i \in K} \omega_i \log (\mathcal{M}(p_i)) \right)}{\sum_{p_i \in K} \omega_i},
\]

where the weight are chosen to be \( \omega_i \) the barycentrics of point \( x \) with respect to \( K \), i.e., the classical FEM basis function coefficient.

For the second method, when a new vertex \( x \) is inserted in tetrahedron \( K \), its metric is computed from growth metric issued from vertices of \( K \). For each vertex \( p \) of tetrahedron \( K \) supplied with a metric \( \mathcal{M}(p) \), the growth metric issued from \( p \) at position \( x \), denoted \( \mathcal{M}_p(x) \), is computed. To this end, the metric growth law proposed in which is homogeneous in the physical space is chosen. This means that the eigenvalues are growing separately and differently, the shape of the metric is not more preserved. This law gradually makes the metric more and more isotropic as it gradually propagates in the domain. The growth factor associated independently with each eigenvalue of \( \mathcal{M} \) is given by:

\[
\eta^2_i(px) = \left( 1 + \sqrt{\lambda_i} \|px\|_2 \ln(\beta) \right)^{-2} \quad \text{for} \quad i = 1, \ldots, 3,
\]

where \( \beta \) is the specified growth rate factor. The growth metric at \( x \) is given by:

\[
\mathcal{M}_p(x) = ^tR \mathcal{N}(px) R \quad \text{where} \quad \mathcal{N}(px) = \begin{pmatrix} \eta^2_1(px) & 0 & 0 \\ 0 & \eta^2_2(px) & 0 \\ 0 & 0 & \eta^2_3(px) \end{pmatrix}.
\]

Finally, the metric at new vertex \( x \) is obtained by a weighted interpolation of all the vertices growth metrics of tetrahedron \( K \):

\[
\mathcal{M}(x) = \frac{\exp \left( \sum_{p_i \in K} \omega_i \log (\mathcal{M}_p(x)) \right)}{\sum_{p_i \in K} \omega_i},
\]
where \( \omega_i \) are again the barycentrics of point \( x \) with respect to \( K \). We will refer to this method as metric-based transition with growth.

### IV. Numerical comparison

Numerical results are presented here to demonstrate the basic characteristics of approach described in this work. These cases where chosen to best illustrate the properties in a graphical context. However, their resolution is relatively coarse in comparison to what is more typical in realistic aerospace configurations. In each case, an anisotropic pseudo-structured surface mesh is used.

#### IV.A. Wing-body

The first test case is a wing-body profile with relatively high-aspect ratio quad faces on the wing surface. Length scale transition between anisotropic and isotropic faces has been selected with relatively high growth to amplify the effect of the blending. Geometry and the initial surface mesh are depicted in Figure 6 for this configuration. The surface mesh is composed of 19,724 triangles and 1,170 quads.

Three distinctly different volume meshes were generated using the same BL region mesh characteristics. For each case, a volume mesh was generated using pentahedral BL elements to facilitate the graphic process. With hexahedra elements the number of BL elements are reduced by a factor of two.

One uses a pure isotropic approach with an immediate transition from the BL region to the outer tetrahedral region. The results are depicted in Figure 7. As shown, mesh quality suffers in the transition region. For this case, this is largely driven by the anisotropic surface mesh and there is simply no way to gracefully transition immediately from a high-aspect ratio face to an isotropic tet element.

The other two use metric-based anisotropic blending described in Section III.C. Of those one uses standard metric interpolation and the other uses metric-based transition with growth. For the case with metric interpolation the results are depicted in Figure 8. As shown the transition now is blended. And the mesh quality is improved from a length scale and element volume perspective. However, the transition is very slow and the mesh density maybe inappropriate for typical applications. Compared to the no blending case, the number of vertices has increased by a factor two and the number of tetrahedra in the outer region are nearly ten times more.

The results for the case using metric-based transition with growth are depicted in Figure 9. As with metric interpolation, the transition is blended and the mesh quality is again improved. However, growth has significantly improved the rate of transition and provided a more reasonable mesh density for the given configuration. Compared to the no blending case, the number of vertices has increased by a factor 1.25 and the number of tetrahedra in the outer region are 2.75 times more. By varying the parameter in the growth function, density can be increased. Here, it has been set to 1.01. Increasing it to 1.05 reduces the number of tetrahedra by a factor 2.75. And, reducing it to 1.0025 increases the number of tetrahedra by a factor two.

<table>
<thead>
<tr>
<th>Case</th>
<th># vertices</th>
<th># tets</th>
<th># prisms</th>
<th># pyramids</th>
</tr>
</thead>
<tbody>
<tr>
<td>No blending</td>
<td>377,563</td>
<td>313,766</td>
<td>625,859</td>
<td>389</td>
</tr>
<tr>
<td>Interpolation</td>
<td>838,477</td>
<td>3,049,515</td>
<td>625,859</td>
<td>389</td>
</tr>
<tr>
<td>Growth</td>
<td>469,390</td>
<td>859,133</td>
<td>625,859</td>
<td>389</td>
</tr>
</tbody>
</table>

Table 1. Wing-body mesh size statistics.
Figure 6. Wing-body hybrid surface mesh with high-aspect ratio quad faces on the wing.

Figure 7. Wing-body meshes obtained with no anisotropic blending.
Figure 8. Wing-body meshes obtained with metric-based transition based on metric interpolation.

Figure 9. Wing-body meshes obtained with metric-based transition with growth.
IV.B. Nacelle

The second test case is a nacelle configuration with a predominantly structured quad faces on the surface. Geometry and the initial surface mesh are depicted in Figure 10 for this configuration. The surface mesh is composed of 14,192 triangles and 24,640 quads.

Three distinctly different volume meshes were again generated using similar BL region mesh characteristics. However, the BL region was chosen to terminate at a smaller normal spacing to illustrate transition in that direction as well as tangential. As with the wing-body case, a volume mesh was generated using pentahedral BL elements to facilitate the graphic process. With hexahedra elements the number of BL elements are reduced by a factor of two.

The results for the pure isotropic approach with an immediate transition from the BL region to the outer tetrahedral region are presented in Figure 11. As before, mesh quality suffers in the transition region. For this case, this is driven by both the anisotropic surface mesh at the leading edge and the normal spacing at the edge of the BL region. Graceful and immediate transition from BL elements with anisotropy in both normal and tangential directions to the isotropic outer elements is not possible.

For the case with metric interpolation the results are depicted in Figure 12. As shown the transition now is blended. And the mesh quality is improved from a length scale and element volume perspective. However, the transition is again very slow and the mesh density maybe inappropriate for typical applications. Compared to the no blending case, the number of vertices has increased by a factor 1.6 and the number of tetrahedra in the outer region are nearly eight times more.

The results for the case using metric-based transition with growth are depicted in Figure 13. As with metric interpolation, the transition is blended and the mesh quality is again improved. Also, the metric-based transition is accounting for both the normal and tangential anisotropy. However, growth has once again significantly improved the rate of transition and provided a more reasonable mesh density for the given configuration. Compared to the no blending case, the number of vertices has increased by a nominal amount and the number of tetrahedra by a factor two.

<table>
<thead>
<tr>
<th>Case</th>
<th># vertices</th>
<th># tets</th>
<th># prisms</th>
<th># pyramids</th>
</tr>
</thead>
<tbody>
<tr>
<td>No blending</td>
<td>1,591,712</td>
<td>875,740</td>
<td>2,855,180</td>
<td>4,356</td>
</tr>
<tr>
<td>Interpolation</td>
<td>2,569,867</td>
<td>6,715,561</td>
<td>2,855,180</td>
<td>4,356</td>
</tr>
<tr>
<td>Growth</td>
<td>1,729,584</td>
<td>1,702,926</td>
<td>2,855,180</td>
<td>4,356</td>
</tr>
</tbody>
</table>

Table 2. Nacelle mesh size statistics.

V. Conclusion

A metric-based approach for smooth transition between BL region and tetrahedral outer region has been proposed. This unified method utilizes the best characteristics of both near body BL and generalized metric-based approaches. Metric-based formulations for quality functions and other geometric quantities require for mesh generation have been presented. Results point out that the metric-based transition can be used to improve mesh quality and density for configurations with anisotropic surface meshes and BL regions that do not reach outer region length scale.

Further work is required to demonstrate this approach combined with an adaptive approach to resolve off-body or field features.
Figure 10. Nacelle hybrid surface mesh with high-aspect ratio quad element.

Figure 11. Nacelle meshes obtained with no anisotropic blending.
Figure 12. Nacelle meshes obtained with metric-based transition based on metric interpolation.

Figure 13. Nacelle meshes obtained with metric-based transition with growth.
References