Article:
Deng, S., Wang, Y. and Qin, N. (2023) Cross-field structured adaptive mesh using medial axis flow feature extraction. AIAA Journal. ISSN: 0001-1452

https://doi.org/10.2514/1.j063346

Cross-field structured adaptive mesh using medial axis flow feature extraction

Siqiang Deng* and Yibin Wang†
Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

and

Ning Qin‡
University of Sheffield, Sheffield, England S1 3JD, United Kingdom

A method to generate feature-aligned anisotropic structured meshes automatically is presented for high resolution of two-dimensional complicated flow features such as normal and oblique shock waves, shock reflection, shock-shock interaction, and shock-boundary layer interaction. The method starts from extraction of dominant flow features to be treated alongside geometric entities. This is followed by automatic generation of multi-block structured meshes of the flow domain using a cross-field based method so that the flow feature entities are embedded along with the geometries. The cross-field method allows for the automatic blocking so that anisotropic quadrilateral meshes generation and adaptation to resolve flow features can be carried out. The resulting mesh possesses high element quality for orthogonality and high resolution of the flow features, suitable for the shock capturing methods and higher-order reconstruction schemes. The method is demonstrated through a number of aerodynamic flow test cases with discussion on its efficiency, accuracy, and robustness.

Nomenclature

a, b, c, d, e = parameters of the bump

E = error estimate along grid edges

* PhD student, College of Aerospace Engineering.
† Associate Professor, College of Aerospace Engineering.
‡ Professor of Aerodynamics, Department of Mechanical Engineering. Associate Fellow AIAA. Email addresses: n.qin@sheffield.ac.uk (Corresponding author)

$\mathbf{F}_T$ = flux

$\mathbf{H}$ = Hessian matrix

k = index of the contained singularity in a triangular

k_{ij} = linear spring-stiffness

l = edge length

l_c = chord length

$\mathbf{M}$ = symmetric and positive-definite matrix obtained from $\mathbf{H}$

M = Mach number

$\mathbf{n}$ = face normal vector

n_c = integer associated with crosses

$\mathbf{Q}$ = conserved variables

Re = Reynolds number

$\mathbf{S}$ = source terms

t = time

$\boldsymbol{v}$ = velocity vector

x_i = node coordinates of index i

α = angle of attack

Φ_c = degree of the bisector of two boundary edges

θ = degree of a cross

θ_c = degree of two boundary edges

ω = relaxation factor

$\Delta\theta_{ij}$ = minimum angle difference between any two crosses

I. Introduction

IN solving flow problems in aerodynamics and fluid mechanics, both the distribution of mesh density and orthogonality are important for accurate numerical solutions. The balance of mesh resolution and computational cost needs to be struck for computational efficiency. Qin and Liu [1] proposed an efficient strategy for flow solutions based on the flow features to be resolved. The feature aligned mesh adaptation allows for the flow features, such as shock

waves and shear layers, to be resolved accurately with efficient meshes since the refinement only needs to be done in the direction normal to the features. Furthermore, the feature aligned meshes with good orthogonality can reduce the interpolation errors in the discretization and provide better numerical accuracy and faster convergence. As a result, a feature aligned grid demonstrated both high accuracy and efficiency for the approximate Riemann solvers and higher-order reconstruction schemes has to be generated after resolving the flow field.

Feature aligned adaptive mesh is the rationale in handling the complex flow features, which is a branch of edge-based mesh movement methods that nodes are moved along the edge direction according to the computational results in an iterative manner [2]. Mesh movement methods generally produce anisotropic elements with high aspect ratios, keeping the grid topology and the total number of nodes unchanged. The anisotropic adaptation can capture the flow features more efficiently due to more reasonable grid distribution. Ait-Ali-Yahia et al. [3] applied the anisotropic grid adaptation method based on the Hessian-based error estimate to a wide range of external compressible flows. In these cases, shocks are well resolved for transonic and supersonic benchmarks. They also pointed out the limitation and inflexibility of the adaptation. For example, the constant connectivity of the structured grid can limit the level of equi-distribution of the error. Nevertheless, the method is still the most efficient for accurate shock wave and shear layer resolution for high Reynolds number flows. Dompierre et al. [4] proposed a method including grid refinement, coarsening, and movement, based on an interpolation-based error estimate for unstructured meshes. The method shows the efficiency of the property of anisotropy compared to the isotropic adaptation, when the same resolution is obtained without the increase of grid nodes.

According to the context, anisotropic adaptation without flow feature alignment generates highly skewed cells unavoidably. Hence, flow feature aligned mesh adaptation draw more attention recently and the geometric representation of flow features is another concern. Shock waves, boundary layers and wakes are extracted as entities such as curves or surfaces embedded in the flow solution domain. Marcum and Gaither [5] used flow property gradient to extract shock waves and regenerated new feature aligned meshes based on the extraction. This showed a good resolution for more complicated shock cases, but was applied for unstructured triangular meshes. Qin and Liu [1] proposed a method to extract flow features based on the Hessian matrix and high-quality structured quadrilateral mesh blocks are placed around them. Employing the Hessian matrix of flow variables, the elements' size, stretching and orientation is determined. The remainder of the domain is then filled with triangular elements, and the mesh was then adapted using an anisotropic nodal movement algorithm. The Rankine-Hugoniot relation across the shock wave is

satisfied, if the cell interfaces are fully aligned with shock waves, which can reduce the misalignment error near shock. Nevertheless, the method poses a challenge that the extraction of features is hard to achieve when the flow structures become complex. Therefore, to increase flexibility and capability of handling more intricate cases, Harris and Qin [6, 7] followed along the line of feature aligned mesh adaption and made use of medial axis to represent complex flow structures as geometric entities after the flow field solution. The medial curves are embedded in the domain to guide the generation of an unstructured high-quality anisotropic quadrilateral mesh, which is locally aligned with flow features.

At present, feature aligned adaptive meshes are utilized to build hybrid grids, in which anisotropic structured cells are used within flow feature regions and unstructured cells are used elsewhere. However, some researchers might want fully structured feature aligned grids for the regular data structure and the anisotropy of cells. Feature aligned structured meshes also have some advantages over unstructured meshes in numerical simulation since they yield sparse banded system matrices and are perfect for multigrid acceleration. The main hurdle restricting the widespread application of feature aligned structured grid is that it is often necessary to divide the structured blocks manually for complex geometries and/or flow features, which requires considerable experience and skills, and requires a lot of time and labor cost. The cross-field based methods have the potential to solve this problem, which can automatically and efficiently generate multi-block structured grids with good orthogonality and smoothness.

Cross-field based methods offer better control of the orthogonality and cell size by distributing singularities. A singularity is defined as a node whose adjacent edges are not four in a quadrilateral grid. Bunin [8, 9] connected the cross-field on a two-dimensional manifold with the potential field on a three-dimensional manifold by using conformal transformation. Singularity distribution is obtained by solving inverse Poisson equation, and then a cross-field is obtained. Kowalski et al. [10] transformed the cross-field into a vector field and calculated the position of the singularities by solving the diffusion equation of the vector field. After that, the blocks are derived from the streamlines departing from singularities and corners. On this basis, Xiao et al. [11] applied boundary element method to solve the diffusion equation of the vector field. Marcon et al. [12] proposed the continuous and discontinuous Galerkin methods to obtain the vector field. Fogg et al. [13], on the basis of Bunin's continuous theory of cross-fields, derived the formula of angle change of crosses in triangles by using the principle of minimum potential energy, and then solved the cross-fields by the paving method. They also used streamlines to generate blocks. Then, they combined medial axis based method and cross-field based method to obtain a relatively reasonable block structure with high quality

[14]. The frame-field [10, 15, 16] and octahedral field [17] are also used to generate structured hexahedral meshes, but they have only been applied for simple geometries, and their aerodynamic applications remain to be explored. According to the literature, many problems need to be solved to improve the stability and robustness of the methods.

The current approach in this paper attempts to generate feature-aligned structured mesh automatically using a cross-field based method for two-dimensional flow problems. The initial structured meshes are produced and implemented to solve the flow problem studied as initial solution, and then regenerated with flow features being extracted and embedded as geometric entities using a medial axis method. The resulting grids are aligned with flow features and allow for high resolution of shock waves, boundary layers, and wakes in high gradient directions by utilizing high aspect ratio and efficient cells.

This paper is arranged as follows. Section 2 briefly describes the structured finite volume solver used in this work. The methodology of the feature aligned structured adaptive mesh generation is discussed in Section 3, covering the medial axis based flow feature extraction method, the cross-field based structured mesh generation method and the improved edge-based anisotropic mesh adaptation method. In Section 4, a number of test cases are presented using the proposed method, including inviscid and viscous flow cases. Conclusions are drawn from the work at the end of this paper.

II. Flow solver

The compressible Reynolds averaged Navier-Stokes equations can be written in an integral conservation form as follows, neglecting external forces and heat sources:

$$\frac{\partial}{\partial t}\int_V \mathbf{Q}\mathrm{d}V + \int_S \mathbf{F}_\mathrm{T}\cdot\mathbf{n}\mathrm{d}S = \mathbf{S} \tag{1}$$

where $\mathbf{Q}$, $\mathbf{F}_\mathrm{T}$ and $\mathbf{S}$ represent the conserved variables, flux vector and source terms, respectively.

In the following test cases, the Spalart-Allmaras turbulence model [18] is used for closure of the system of equations. Simulations are performed using an in-house finite volume RANS CFD code. The AUSMPW [19] flux splitting scheme is used to capture both the shock waves and shear layers accurately.

III. Methodology

A. Overview of the method

The general process of the proposed method is summarized in the following steps:

1) Input the flow conditions and geometry.
2) Generate a multi-block structured grid using **the cross-field method**.
3) Use an r-adaptive mesh method to solve the flow field.
4) Utilize **the medial axis approach** to extract the flow features in order to produce the flow feature curves.
5) Check to see if the number of critical nodes on the feature curves change, indicating whether the flow features converge. If so, output the current solution for the flow field; otherwise, repeat steps 2 to 4 using the flow feature curves incorporated into the flow solution domain.

The input geometry for Step 1 consists of nodes on the wall and the far field. These nodes are used in Step 2 to build a background grid that can be employed to solve for the cross-field and blocking the flow solution domain. This section refers to the articles of Fogg et al. [13] and Marcon et al. [12]. However, there are certain discrepancies in grid reconstruction, which are reflected in the processing of flow feature curves. In section 3.3, which also introduces filling the blocks with anisotropic cells, this procedure is briefly explained.

The structured r-adaptive mesh approach utilized by Ait-Ali-Yahia et al. [3] is enhanced in Step 3. The iterative coupling between the adaptive mesh and flow field solution will be covered in Section 3.4. Nonetheless, the mesh adaption is not required for solving the original flow field. We described the geometric process of flow features in Step 4 in Section 3.2, proposed by Harris et al. [6, 7], which mainly consists of three steps: marking flow feature nodes, recognizing boundaries based on α-shape, and extracting flow feature curves based on medial axis. Grid adaptive criteria and flow feature nodes marking are carried out using the Hessian matrix, while flow feature curve extraction is handled using the medial axis approach.

For Step 5, the number of critical nodes on the flow feature curves, namely the terminal nodes and the crossing nodes, are measured to determine whether the flow features have converged. Fig. 1 depicts the comprehensive flow chart in summary form.

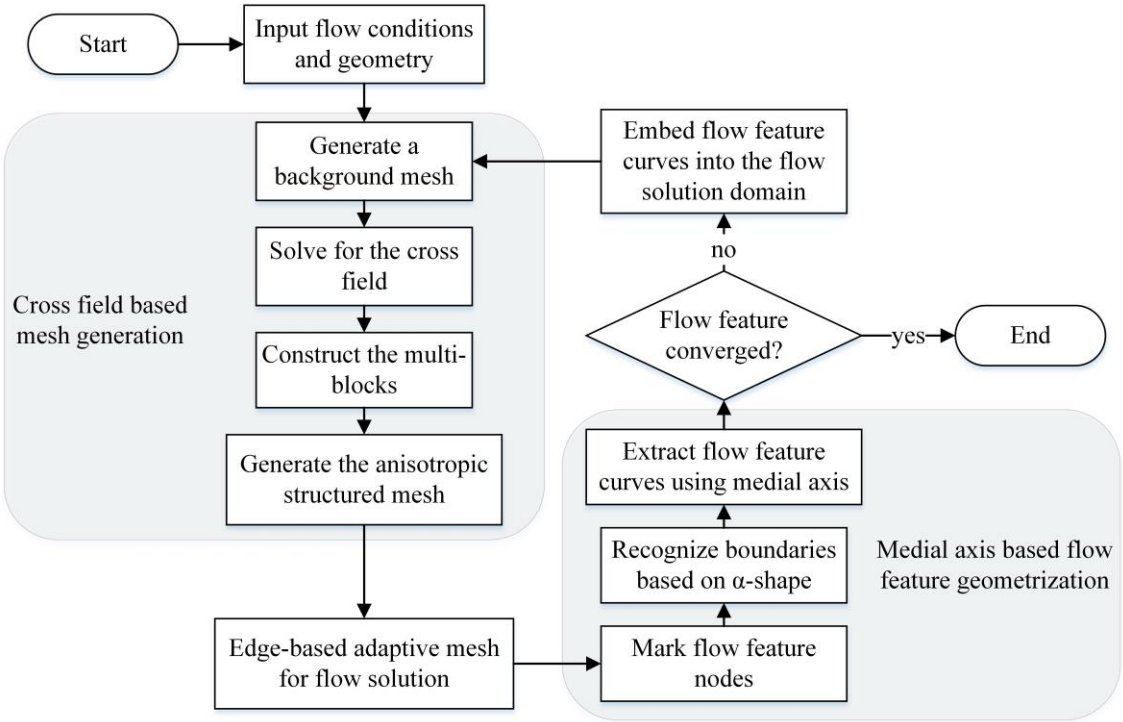


Fig. 1 Flow chart of the feature-aligned structured adaptive mesh generation.

B. Medial axis based flow feature extraction

As a demonstration, Fig. 2 shows the Mach number contour of a transonic flow field solution around the NACA0012 airfoil, where the free steam Mach number is 0.85 at a 1° angle of attack. Fig. 3 provides an illustration of the extraction procedure for the shock wave feature.

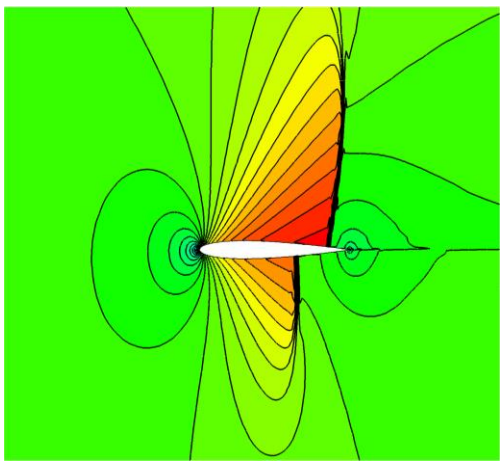
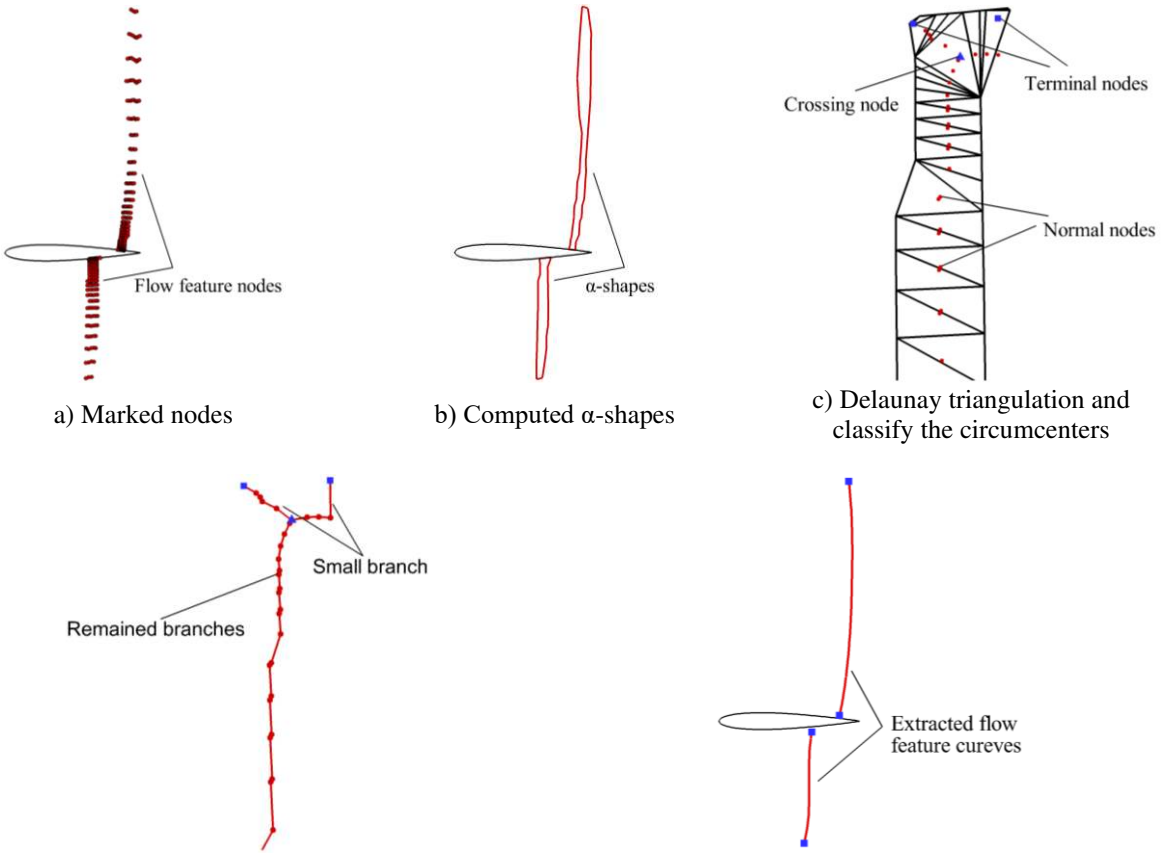


Fig. 2 The Mach number contour of a transonic flow field solution around the NACA0012 airfoil.



a) Marked nodes

b) Computed α-shapes

c) Delaunay triangulation and classify the circumcenters

d) Connected medial axis and split branches

e) Clean and smoothed curves corresponding to the shock waves

Fig. 3 Illustration of the flow feature curve extraction of an inviscid transonic flow solution.

The following equation, Eq.2, is used to calculate the error estimate along the grid edges before distributing it to the grid nodes [1, 2].

$$E = \int_0^1 \sqrt{(\boldsymbol{x}_2 - \boldsymbol{x}_1)^{\mathrm{T}} \mathbf{M}(t)(\boldsymbol{x}_2 - \boldsymbol{x}_1)}\,\mathrm{d}t \tag{2}$$

where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are two nodes of an edge, and $\mathbf{M}$ is a symmetric and positive-definite matrix obtained from the Hessian matrix $\mathbf{H}$. Flow feature nodes are marked when E exceeds a pre-specified threshold ([0.05, 0.15]). In the test cases, the threshold is set to 0.1. The threshold value is determined by the balance of the feature identification and the numerical noise. For this inviscid flow problem, the pressure is used as the physical quantity to compute $\mathbf{H}$ to capture the shock feature (Fig. 3a). For viscous flows, density or the Mach number is used as a quantity in $\mathbf{H}$ to capture both inviscid and viscous flow features.

The boundaries of the node set are determined using the α-shape technique [20], shown in Fig. 3b. This technique is utilized as a filter to get smooth boundaries, while zigzag lines would be produced if the borders were obtained

directly from the cell domains, as demonstrated in Fig. 4. A brief description is given here. If there is an empty circle of a radius α going through any two points in a node set, then those two points constitute of the set's α-shape. The value of α has an impact on the α-shape. When α is infinite, the convex hull of the set forms the α-shape. Concave shapes start to appear as α falls. At zero, the α-shape reverts to the original set. In our algorithm, however, α is not fixed when dealing with the set. For the two points being processed, the neighboring edges are searched in the dependent grid and α is set twice the maximum length of these edges. Obviously, looping nodes to obtain α-shapes are computationally intensive. In order to improve efficiency, these nodes are Delaunay triangulated first, and the edges are then identified as being of the α-shape by comparing their locations to those of the adjacent triangles. The specific algorithm can be seen in Alg.1. Shewchuk's two-dimensional quality mesh generator and Delaunay triangulator library [21], *Triangle*, is used as **triangulate (·)** here and below. The parameter, "·", is set to "points" to generate Delaunay triangulations, "points, edges" to generate constrained Delaunay triangulations, or "points, edges, area" to generate high-quality meshes that limits the maximum triangular area.

Fig. 4 The inherent boundary of the red node set (left), and the α-shape boundary (right).

The nodes on the α-shapes are then subjected to the constrained Delaunay triangulation, and the ~~outer centers~~ circumcenters of these triangles define the medial axis of the boundaries. In accordance with the quantity of neighbors of the related triangles, the nodes on the medial axis can be divided into crossing, terminal, and normal nodes (Fig. 3c). The terminal nodes are the extremities of the curves with just one neighbor, while the crossing nodes, which are connected to more than two curves, have three neighbors. The remaining nodes are normal nodes with two neighbors. The nodes are therefore named the critical nodes, the crossing nodes and the terminal nodes that segment the medial axis into branches, as presented in Fig. 3d. Small branches are defined as branches that have less than a pre-specified number of nodes (for example, five or six). For simplification of the medial axis, a small branch is handled in the

following two cases: (1) the branch is removed when at least one end is the terminal node; and (2) the branch degenerates to a crossing node when both ends are crossing nodes, by averaging the coordinates of nodes on the branch.

Following the simplification, the remaining branches are smoothened using the polynomial fitting techniques [6], and the node types on the fresh curves that were produced are altered. Fig. 3e shows the processed curves that represent the shock waves in the initial flow solution in Fig. 2. In the subsequent structured mesh regeneration, the flow feature curves are treated as geometric entities in the flow solution domain. Alg.1 is given conclusively.

Algorithm 1 Flow feature extraction

- ***Input:*** *gridData* ← the initial flow field including grid information
- ***Output:*** *featureCurves*

featureNodes ← [] *// A list to store flow feature nodes. Assume the nodes are in a connected region.*
featureCurves ← [] *// A line grid including nodes and their connectivity to represent the flow feature curves.*
// Step1 Mark flow feature nodes:
for every edge *i* in *gridData* **do**
  compute error estimate E_{edgei} in Eq.2
for every node *i* in *gridData* **do**
  interpolate E_{nodei} from E_{edgei}
  if $E_{nodei} > 0.1$ **then**
    mark node as *featureNodes*
// Step2 Implement the α-shape method:
triFeatureNodes ← triangulate(*featureNodes*) *// The Delaunay triangulation of the flow feature nodes.*
alphaEdges ← [] *// A list of edges to store the α-shape.*
for every edge *i* in *triFeatureNodes* **do**
  interpolate α_i from *gridData*
  p_1, p_2 ← two nodes of the edge *i*
  l_c, r_c ← the circumcenters of edge *i*'s neighbor triangle(s)
  if r_c == [] **then** *// This means the edge just has one neighbor.*
    line(l_c, r_c) ← A ray departing from l_c in the direction of (l_c, r_c)
    $a_{\min}$ ← the minimum distance form p_1 to *line*(l_c, r_c)
    if $a_{\min} \le \alpha_i$ **then**
      mark edge *i* as *alphaEdges*
  else *// The edge has two neighbors.*
    line(l_c, r_c) ← A line segment composed of l_c and r_c
    $a_{\min}$, $a_{\max}$ ← the minimum and maximum distances form p_1 to *line*(l_c, r_c)
    if $a_{\min} \le \alpha_i$ **then**
      mark edge *i* as *alphaEdges*
// Step3 Extract a flow feature curve:
alphaNodes ← the nodes in *alphaEdges*
triAlphaShape ← triangulate(*alphaNodes*, *alphaEdges*)
medialNodes ← the circumcenters of the triangles in *triAlphaShape*
numberofNeighbors ← the number of neighbors of the triangles in *triAlphaShape*
for every node *i* in *medialNodes* **do**
  if *numberofNeighbors*[i] == 1 **then**
    mark node *i* as *terminal*
  elseif *numberofNeighbors*[i] == 2 **then**
    mark node *i* as *normal*

else
mark node *i* as *crossing*
medialCurves ← make curves starting from nodes in *crossing* and *terminal* with nodes in *medialNodes*
for every curve *i* in *medialCurves* **do**
if number of nodes in curve $i \geq 6$ **then**
smooth curve *i*
mark curve *i* as *featureCurves*

C. Cross-field based structured mesh generation

1. *Preliminaries*

Basic definitions are described here to help a deeper understanding of the cross-field, and more details can be found in Bunin's continuum theory [8].

Definition 1 (Cross) *A cross is an element of* $\mathbb{R}^2/\sim$*, in which* $\sim$ *is defined as an equivalence of vectors in* $\mathbb{R}^2$*: for two vectors* $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^2$*,* $\mathbf{v}_1 \sim \mathbf{v}_2$ *if and only if* $\mathbf{v}_1, \mathbf{v}_2$ *are parallel or perpendicular.*

A given cross is hence a set of vectors, each pair of which is either perpendicular or parallel. A cross-field is thus the field of crosses. Let P represent a finite set of points in Ω and J be a finite set of points in $\partial\Omega$, where Ω and $\partial\Omega$ are, respectively, a surface and its boundary. P and J stand for singularities and corners, respectively, which constitute the cross-field's discontinuity.

Definition 2 (Cross-field) *A cross-field on a given manifold is a mapping* $V: \Omega \backslash (P \cup J) \to \mathbb{R}^2/\sim$ *such that:*

1) $\forall a, b \in \Omega \backslash (P \cup J)$*, the parallel transport of a vector* $\mathrm{V}(a)$ *at point* a *to point* b *is independent of the path, and is equal to* $\mathrm{V}(b)$*.*

2) $\forall a \in \partial\Omega \backslash (P \cup J)$*, the tangent belongs to the cross there.*

Definition 3 (Singularity) *A singularity is defined as the discontinuity that cannot be linear interpolated in a cross-field, or a node where the number of connected edges is not four in a quadrilateral mesh.*

Definition 4 (Corner) *A corner is defined as a point where a discontinuity in curvature exists on* $\partial\Omega$*.*

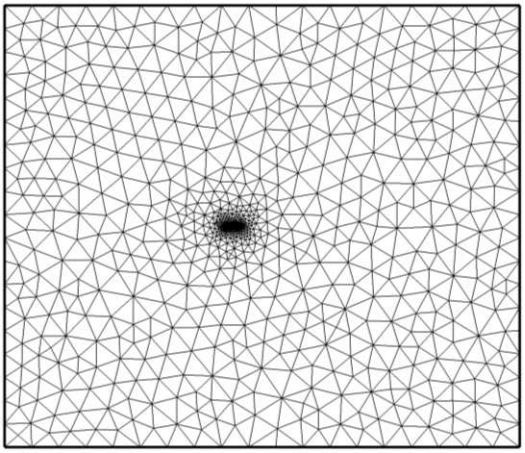
Singularities can theoretically appear anywhere on a surface Ω. However, there will be just corners and no singularities on $\partial\Omega$ in our research in order to guarantee the orthogonality of the boundary mesh. A cross field on Ω is then conformally mapped to a ϕ-field on $\partial\Omega$, in which Ω is a locally flat surface and singularities on Ω are transformed into corn points on it. The ϕ-field is governed by a Poisson equation to form a smooth cross field such that,

$$\Delta_{\Omega}\phi \;=\; K \;+\; \sum_{i=1}^{N} \frac{k_i \pi}{2} \delta_{\mathbf{p}i} \tag{3}$$

where Δ_{Ω} is the Laplace-Beltrami operator, K is the zero-valued Gaussian curvature of for flats, k_i is the integer associated with singularities and will be explained later, $\delta_{\mathbf{p}i}$ is the weighted Dirac delta function at cone points.

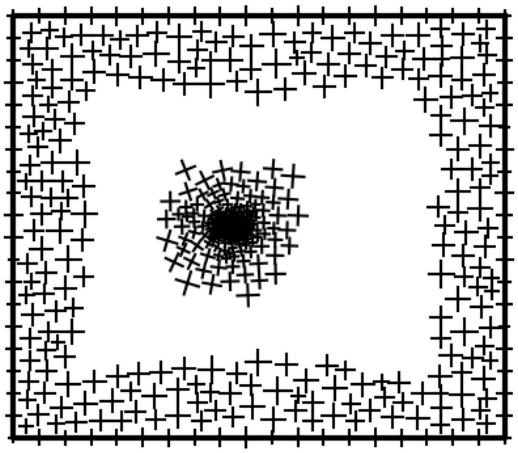
2. Blocking the flow solution domain

We follow the theory of cross-field elaborated in Bunin [8, 9]. The method is only briefly described here, and the pertinent algorithms are available in [12, 13, 16]. In our implementation, the cross-field based automatic decomposition of structured block topology technique can be divided into the steps in Fig. 5, in which the flow solution domain of the NACA0012 airfoil is used as an example.

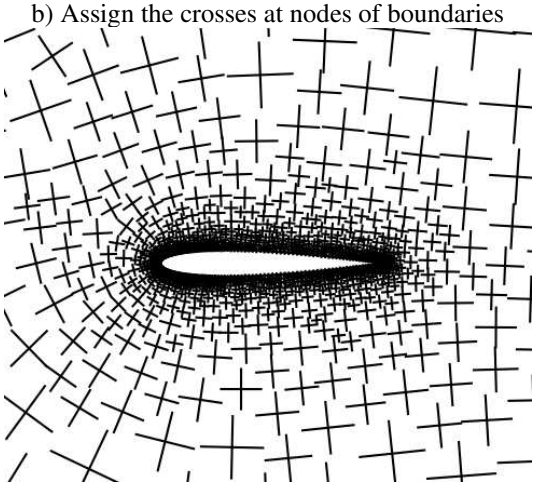


a) Generate the background mesh

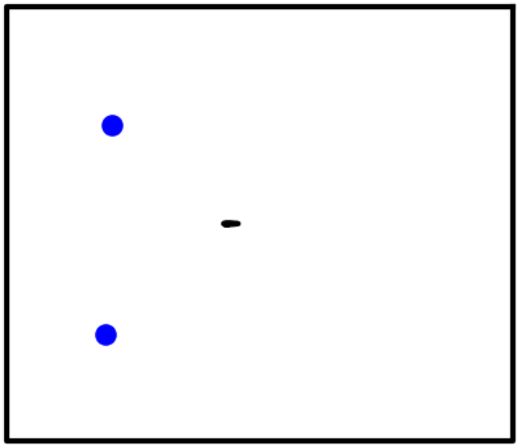
b) Assign the crosses at nodes of boundaries



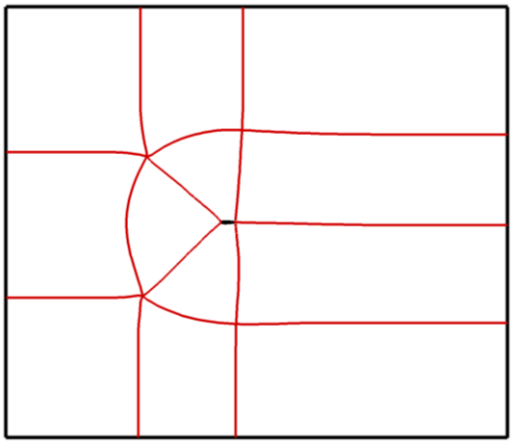
c) Propagate the crosses to the interior



d) Smooth the cross-field



e) Identify the singularities in the cross-field



f) Trace the streamlines to block the domain

Fig. 5 General steps of the cross field based structured block decomposition.

Each step in Fig. 5 is illustrated below.

a. Generate the background mesh

A background mesh as in Fig. 5a stores the cross-field of the flow solution domain after its initialization and smooth. The boundaries must be conformal. The library function **triangulate** (·) is used to generate the grid, and it limits the maximum area to $0.01l_c^2$, where l_c is the length of the airfoil chord.

b. Assign the crosses at nodes of boundaries

When assigning crosses on boundaries as shown in Fig. 5b, a technique is used to enhance the automation of the method. The two orthogonal directions associated with a cross on a boundary edge, according to the continuity theory, are perpendicular to and along the edge tangent. Assume that Φ_c and θ_c are the angles created by the bisector of the boundary and two adjacent edges of a discrete point, as shown in Fig. 6. Then an integer is defined to determine the cross at the point, that is,

$$n_c = \text{round}\left(\frac{\theta_c}{\frac{\pi}{2}}\right) \tag{4}$$

and the cross is represented by an angle,

$$\theta = \begin{cases} \Phi_c, & n_c \text{ is even} \\ \Phi_c + \frac{\pi}{4}, & n_c \text{ is odd} \end{cases} \tag{5}$$

where θ is adjusted in the domain of $\left[0, \frac{\pi}{2}\right)$. The point is classified as a corner when n_c is not equal to 2, which is a discontinuity of the cross-field.

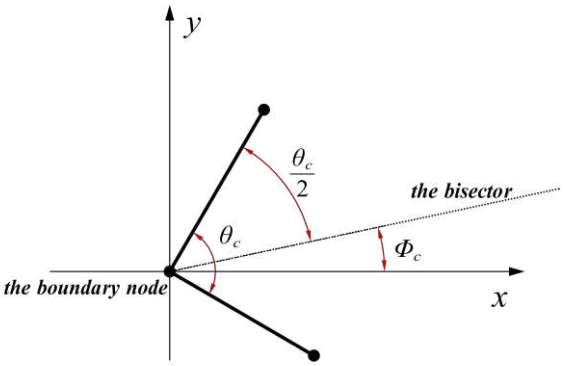


Fig. 6 A diagram of Φ_c and θ_c.

c. Propagate the crosses to the interior

Then, using the Fast Marching Method [22], crosses on boundaries are propagated to the interior on the background mesh following Fogg et al. [13], where a distance field is applied to direct the propagation direction with values of zero at the boundaries. The propagation of cross-field is illustrated in Fig. 5c.

d. Smooth the cross-field

Fogg et al. [13] locally optimized the cross-field because their approach would result in non-smooth regions along the medial axis of the flow solution domain. The smoothed cross-field is depicted in Fig. 5d.

e. Identify the singularities in the cross-field

The block topology is created by connecting the cross-field streamlines to the singularities and corners in the domain. The triangular elements of the background mesh that containing singularities in a smoothed cross-field is distinguished using Bunin's equation [8],

$$k = \text{round}\left(-\frac{2}{\pi}(\Delta\theta_{12}+\Delta\theta_{23}+\Delta_{31})\right) \tag{6}$$

In a triangle with crosses θ_1, θ_2, θ_3 at anticlockwise ordered nodes 1, 2, 3, $\Delta\theta_{ij} = \theta_j - \theta_i$ is the minimum angle difference between any two crosses within the range of $[-\pi/4, \pi/4]$. The singularity is to locate at the geometric center of the triangle to simplify computation (as shown in Fig. 5e).

f. Trace the streamlines to block the domain

Streamlines of the cross-field are employed to decompose the fluid domain into structured blocks. As demonstrated by Dorobantu [23], to trace streamlines through the cells, an explicit 2-stage Runge-Kutta method is implemented to numerically integrate the following equation of velocity vector field

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{v}(\boldsymbol{x}) \tag{7}$$

Streamlines may collide as they advance over the domain simultaneously from the points of origin, as outlined by Marcon [12]. At each advancing step, the front points of streamlines are compared in order to predict when they will meet. The streamlines are assumed to meet if the distance between their front points is less than both of the sizes of cells where the points locate and they are moving in opposite directions. After that, the two streamlines are combined into one by curve fitting. Fig. 5f shows the block topology with good orthogonality.

The flow feature curves are inserted into the initial geometry to regenerate the background mesh. The cross-field and blocks are generated with different approaches to flow feature curves. As seen in Fig. 7, the crosses on normal nodes and terminal nodes are in line with the directions of the tangent of curves where they are located. The positioning of corners is not considered because the curves are flattened. When n curves collide at a crossing node, the cross can be assigned to the node by viewing the node as n corners produced by two adjacent curves. When using the Fast Marching Method, a crossing node is divided into three corners and the location is divided into three areas signifying three routes of propagation, as shown in Fig. 8. Each corner has an impact on the cross propagation in the direction it faces under the given conditions. Fig. 9 depicts the reconstruction on the background grid and the recalculation of the cross field for the example case with the flow feature curves.

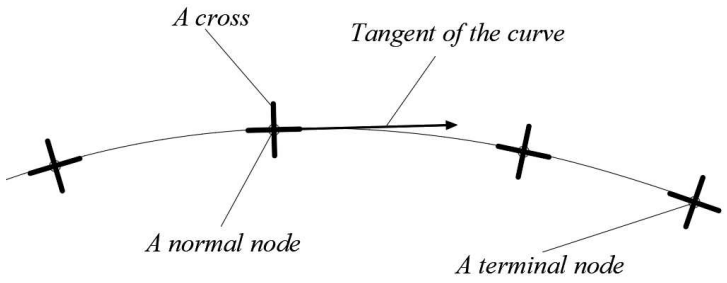


Fig. 7 Assignment of crosses on normal nodes and terminal nodes of feature curves.

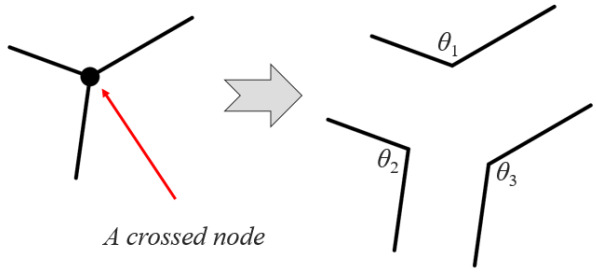


Fig. 8 A crossing node is divided into several corners.

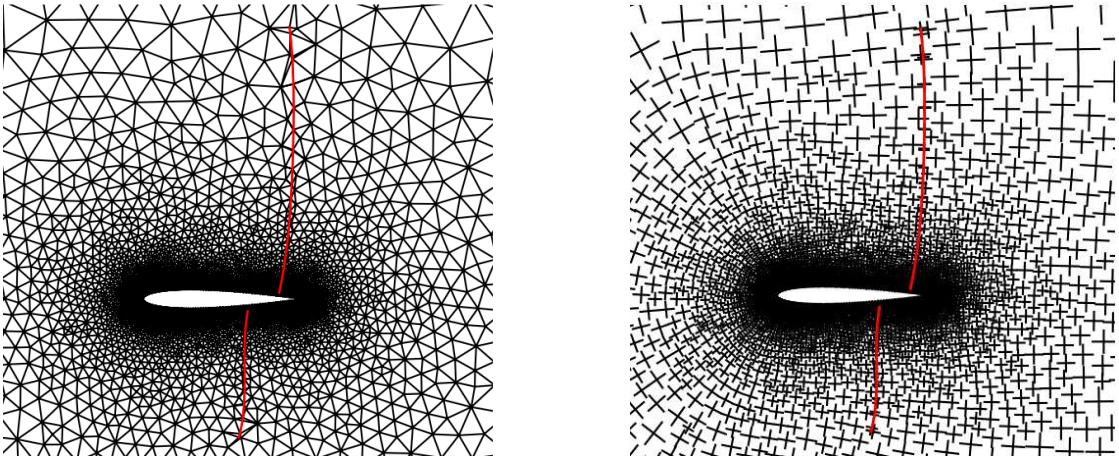


Fig. 9 Reconstruction on the background grid and the recalculation of the cross field.

The crossing nodes and terminal nodes on flow feature curves are also where home locations locate when tracing streamlines of the cross-field. The streamlines that emanate from the terminal nodes spread out in that direction. The integer n_c in Eq.3 is calculated for each detached corner at crossing nodes. As a result, within the range of each corner, there are $n_c - 1$ streamlines bisecting the angle if $n_c > 1$. In Fig. 10, the reconstructed block topology with flow feature curves embedded in the flow solution domain is presented.

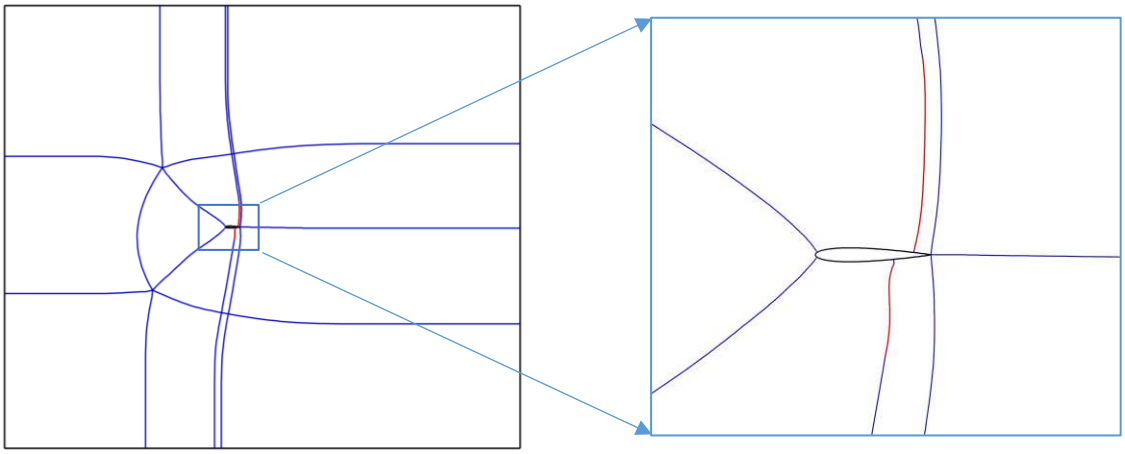


Fig. 10 Blocks with flow features (left) and its zoom view (right).

The pseudocode of this part together with multi-block construction is provided in Alg.2.

Algorithm 2 Multi-block construction on flow feature curves and initial flow field domain

- ***Input:*** *flowDomain* ← nodes and edges on boundaries, *featureCurves*
- ***Output:*** *blockData // The class to store nodes, curves and connectivity of the multi-block topology.*

area ← $0.01l_c^2$
backGrid ← triangulate(*flowDomain.nodes*+ *featureCurves.nodes*, *flowDomain.edges*+ *featureCurves.edges*, *area*)

Compute crosses on nodes in *flowDomain* and *featureCurves* and mark *corners* by Eq.4.
Initialize and propagate the cross-field following Fogg et al.'s method, see [13].
Identify *singularities* by Eq.6.
blockCurves ← Advance streamlines on *backGrid* from nods in *corners* + *singularities* + *crossing* + *terminal*
// using Marcon et al.'s method, see [12]
// *The entire curves are divided into blocks of curves and then joined into blocks.*
blockCurves ←*blockCurves* + *flowDomain* + *featureCurves*
Compute all crosspoints in *blockCurves* as *crosspoints*.
blockData.curves ← []
for every curve *i* in *blockCurves* **do**
  tempCurve ← []
  for every node *j* in *i* (except the first and last nodes) **do**
    if *j* in *crosspoints* **then**
      mark *j* as *tempCurve*
      mark *tempCurve* as *blockData.curves*
      tempCurve ← []
    else
      mark *j* as *tempCurve*
for every two curves *i, j* in *blockData.curves* **do**
  if *i* and *j* have endpoints that overlap **then**
    mark *i* and *j* are neighbors
blockData.blocks ← []
for every curve *i* in *blockData.curves* **do**
  for every two *i*'s neighbors *j*, *m* **do**
    if there exists *j.neighbor* == *m.neighbor* == *n* **then**
      blockData.blocks ← [*i*, *j*, *m*, *n*]

3. Filling the anisotropic cells in the blocks

Shock waves are treated as discontinuities in the mathematical modelling and modern shock capturing techniques can resolve the shock wave within one or two cell widths if the grid is normal to the shock waves. Therefore, there are fewer layers of refinement close to the shock waves than there are at the shear layers including boundary layers and wakes with continuous high gradients. Unless otherwise stated, the minimum spacing between refinement layers is set at 0.1% of the chord length l_c for shock waves in both inviscid and viscous flows. The transition rule from dense to sparse node distribution follows the hyperbolic tangent rate [24], such that

$$\frac{t_i}{h} = 1 + \frac{\tanh[B(i-1)]}{\tanh B} \tag{8}$$

where h is the current curve length, B a parameter described in the reference, t_i the distance from the start node to the ith node along the curve.

In blocks, an equal number of nodes on opposite boundaries are necessary, while both adjacent blocks have a common boundary. Thus, the block boundaries of a fluid domain need to be correlated so that all corresponding boundaries have the same number of nodes, by looping blocks and tracing the blocks of directions of its neighbors to search the opposite boundaries. As a result, the sets of the associated boundaries are obtained. Taking blocks in Fig. 11 as an example, the associated boundaries should be (1, 3, 6), (8, 9, 11), (2, 4, 10), (5, 7, 12).

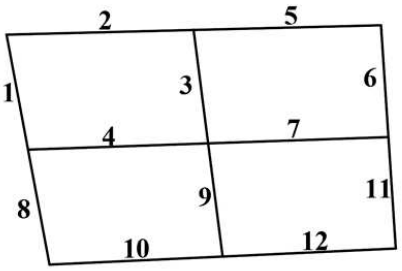


Fig. 11 A simple example of block boundaries.

According to node distribution on block boundaries, structured cells are generated separately in each block, by solving the Poisson equation [25]

$$\begin{cases} \xi_{xx} + \xi_{yy} = P(\xi, \eta) \\ \eta_{xx} + \eta_{yy} = Q(\xi, \eta) \end{cases} \tag{9}$$

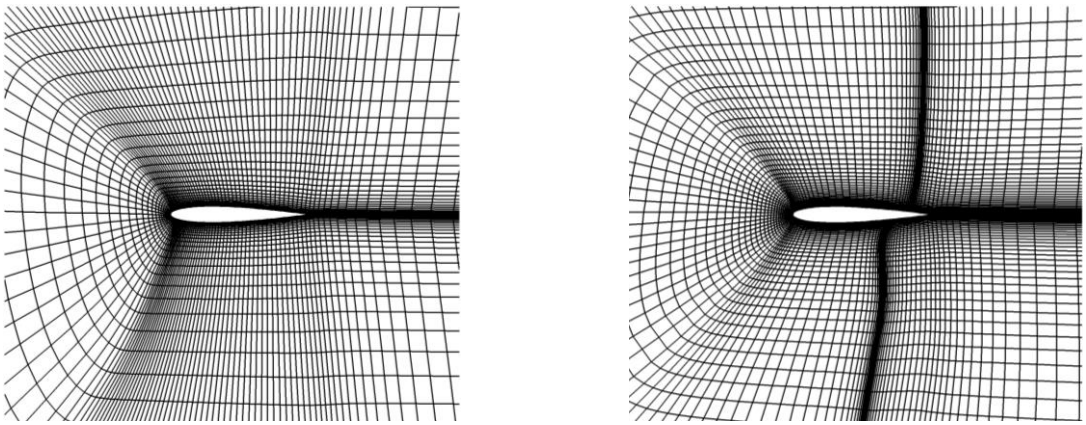
where (x, y) is the physical plane, (ξ, η) is the transformation space, P and Q are the source items to be solved.

The algorithm is presented in Alg.3. Fig. 12 illustrates the initial structured grid and the feature-aligned structured grid, in which the refined mesh along the shock waves can be observed for the latter.

Algorithm 3 Fill cells in blocks

- ***Input:*** *blockData*
- ***Output:*** *multiBlockGrid*

multiBlockGrid ← []
associate ← []
for every block *i* in *blockData.block* **do**
  for every curves *j* in *i* **do**
    if *j* has neighbor in featureCurves then
      set the spacing of this side as $0.1\%l_c$
    Compute node number on *j* using Eq.8.
  for every pair curves *m*, *n* in *i* **do**
    if exit *k* that *m* or *n* in *associate*[*k*] **then**
      mark the other as *associate*[*k*]
    else
      mark [*m*, *n*] as *associate*
for every *i* in *associate* **do**
  $node_i$ ← find maximum node number in *i*
  for every curve *j* in i **do**
    Set *j*'s node number as $node_i$.
    Recompute *j*'s node distribution based on Eq.8.
for every block *i* in *blockData.block* **do**
  grid ← solve Eq.9
  mark *grid* as *multiBlockGrid*



a) b)

Fig. 12 The (a) initial structured grid and (b) feature-aligned structured grid for the NACA0012 airfoil.

D. r-adaptive mesh with feature aligned mesh

After mesh regeneration, the existing solution is interpolated onto the newly created feature-aligned mesh to start the r-adaptive solution. Equi-distribution of the error is implemented by a linear spring-stiffness approach, in which each grid edge with nodes *i*, *j* is thought of as a spring with a stiffness set to

$$k_{ij} = \frac{e}{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|} \tag{10}$$

The feature aligned meshes have a high degree of anisotropy and desirable density normal to the feature. The function of mesh adaptation is to adapt the node position to achieve higher accuracy of solving flow features. In the feature-aligned adaptive mesh approach proposed here, the mesh can be adapted normal and along the captured flow features effectively. This flow feature based adaptive mesh method is defined as f-adaptive mesh in this paper. The node movement scheme is carried out on the feature-aligned mesh such that,

$$\boldsymbol{x}_i^{\text{new}} = \boldsymbol{x}_i^{\text{old}} + \omega \frac{\sum_j (\boldsymbol{x}_j^{\text{old}} - \boldsymbol{x}_i^{\text{old}}) k_{ij}}{\sum_j k_{ij}} \tag{11}$$

where ω is a relaxation factor and $\omega \in [0,1]$. The spring system created by the grid edges is brought to a state of close equilibrium by repeatedly moving the nodes. Notable, when resolving shock waves, a minimum edge length in r-adaptive mesh should be set. Otherwise, all the points will be attracted to the shock waves as the gradient at the shock is infinity. Fig. 13 depicts the shock waves of the adaptive solution applied to the feature-aligned mesh.

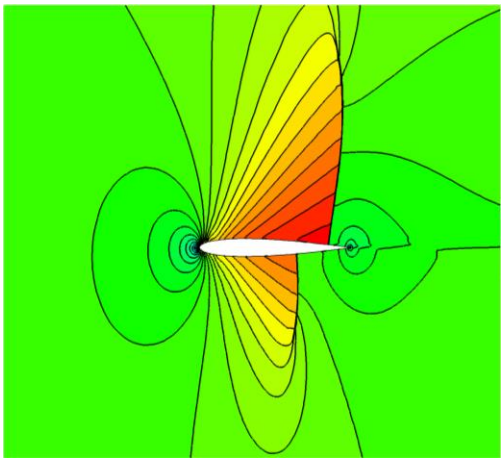


Fig. 13 The f-adaptive solution (Mach number contour) from Fig. 12b.

IV. Test cases

The method of automatic anisotropic structured quadrilateral mesh generation for capturing complicated flow features (or feature alignment) presented above is demonstrated for inviscid flow Euler solutions with shock waves, shock reflection and shock-shock interaction, high Reynolds number viscous flow RANS solutions including shock waves, boundary layers and their interaction. To investigate the efficiency of the method against some meshing

strategies, all the cases reported here were run on a single core (4.1GHz) on the same computer with Intel i5-9400F processors.

A. Test Case 1: Supersonic inviscid flow around NACA 0012 airfoil

The case, from the AGARD Working Group 07 [26], tests a low supersonic inviscid flow issue with a free stream Mach number of 1.2 and an angle of attack $\alpha = 7°$. The initial structured grid for Euler solution is the same as that in Fig. 12 (a). As demonstrated in the converged solution in Fig. 14, the shock wave structure in the flow field is complex as a result, causing a relatively strong bow shock wave detached from the leading edge and a rather weak oblique fishtail shock wave at the trailing edge.

Obviously, due to numerical dissipation, the initial flow solution is difficult to be used to extract a more complete geometry of shock wave features. A better way is to carry out several coupling steps of flow solution and a structured r-adaptive mesh before the extraction. Fig. 15 shows the r-adaptive solution and the corresponding adaptive grid. This flow solution shows a better shock capture accuracy and resolves the weak shock wave under the airfoil more clearly despite the high skewness of the mesh in relation to the shocks and in the flow field. The resolution of far-field shock waves, wakes and other features is very important for the application of multi-field coupling calculation, such as the calculation of sound field and electromagnetic field. The poor mesh quality and distorted elements of the grid limit the further adaptation for this case using the structured r-refinement. The flow feature extraction of the shock waves is similar to the demonstration test case in Section 3.2.

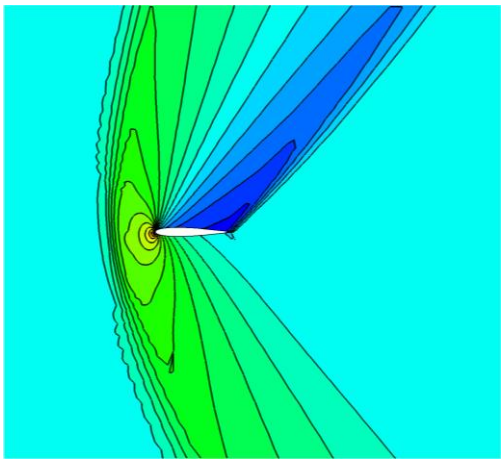


Fig. 14 The f-adaptive solution (Mach number contour).

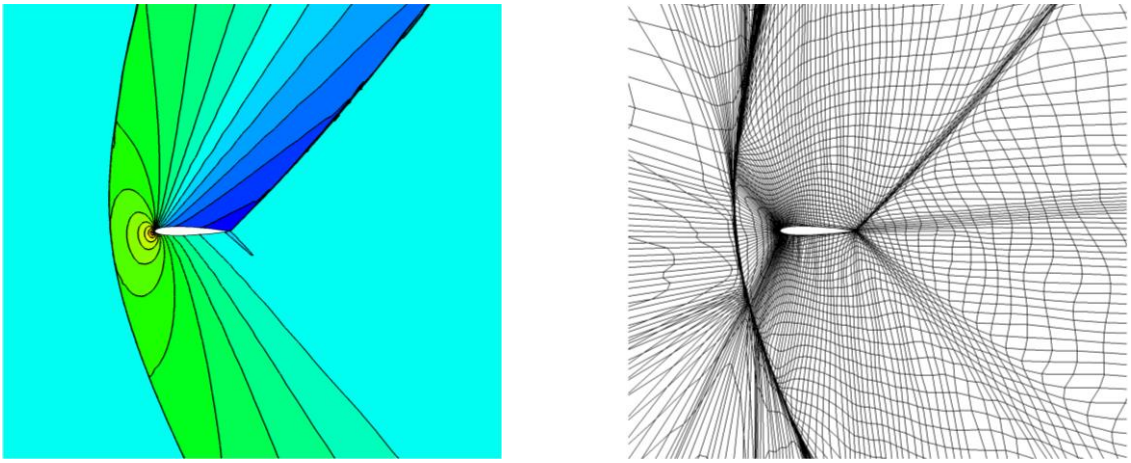


Fig. 15 The pressure contour of r-adaptive solution (left) and the r-adaptive grid (right).

Fig. 16 shows the flow feature lines extracted and Fig. 17 and Fig. 18 show the block structure along with the flow feature curves and corresponding feature-aligned grid, respectively. The grid has 18,810 nodes and 17,180 cells. The high quality orthogonality of the mesh along the flow features and in the flow field can be observed.

Fig. 16 Extracted flow feature lines (red lines) for shock wave capturing around the airfoil.

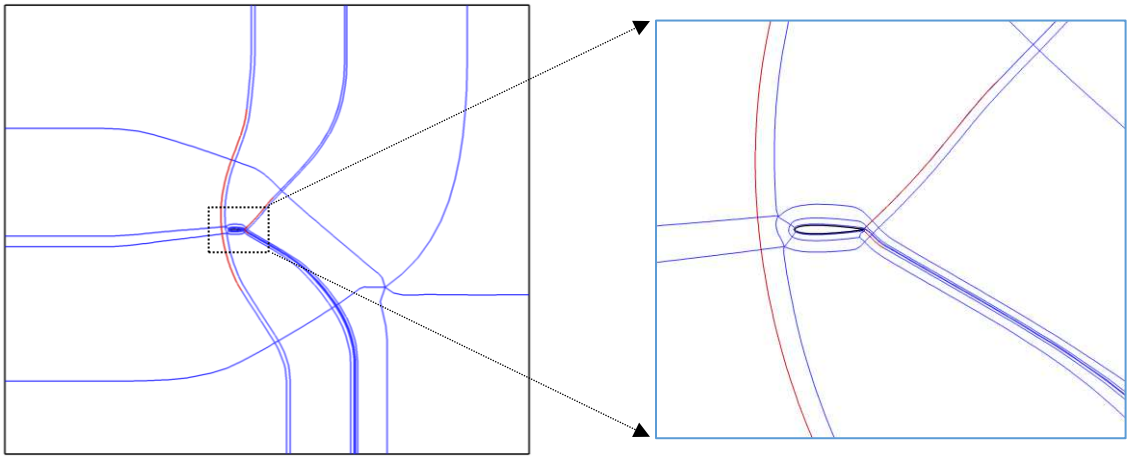


Fig. 17 Feature-aligned blocks for test case 1: farfield view (left) and near airfoil view (right).

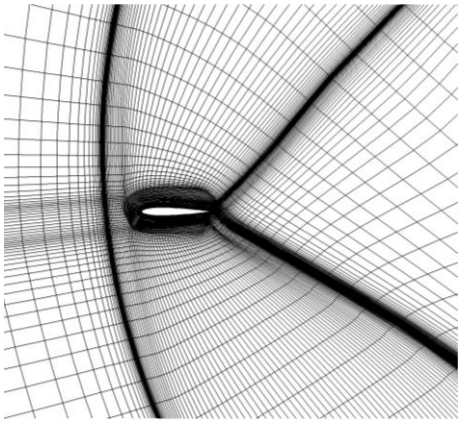


Fig. 18 Feature-aligned mesh for test case 1: near airfoil view.

The previous flow solution was interpolated into the feature-aligned grid to continue the calculation, and the f-adaptive solution were obtained, as shown in Fig. 19. The final pressure contour shows a clear shock structure, and a better resolution of weak shock wave under the trailing edge. Comparing the three results in Fig. 20, the length of the f-adaptive solution for weak shock wave is improved by 1747% and 270%, respectively, compared with the initial solution and the structured r-adaptive solution, which illustrates a significant improvement in the solution quality with a relative efficient mesh. In order to evaluate the quality of a structured mesh, a mesh quality metric [27] is introduce as q, which assesses the skewness and the size of the mesh. Briefly, the essential properties of the quadrilateral size-skew metric are:

(a) $q = 1$, the element has equal angles and the same size as the ideal element;

(b) $q = 0$, the element is degenerated.

Therefore, $q_{\min}$ denotes the worst quality cell in the grid and $q_{\max}$ the average quality of all cells in the grid. The f-adaptive mesh maintains good mesh quality, and the changes in mesh quality statistics are listed in Table 1. The minimum and average cell quality of the f-adapted grid are found to be 55.3 and 9.17 times, respectively, higher than those of the r-adaptive grid.

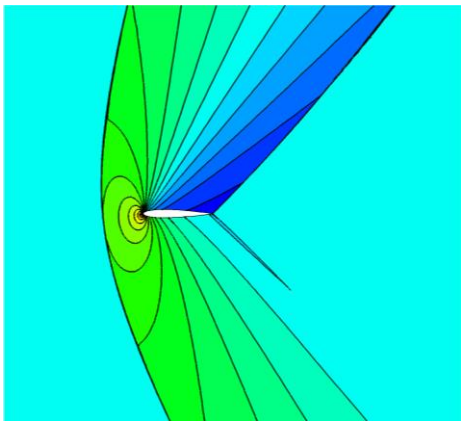
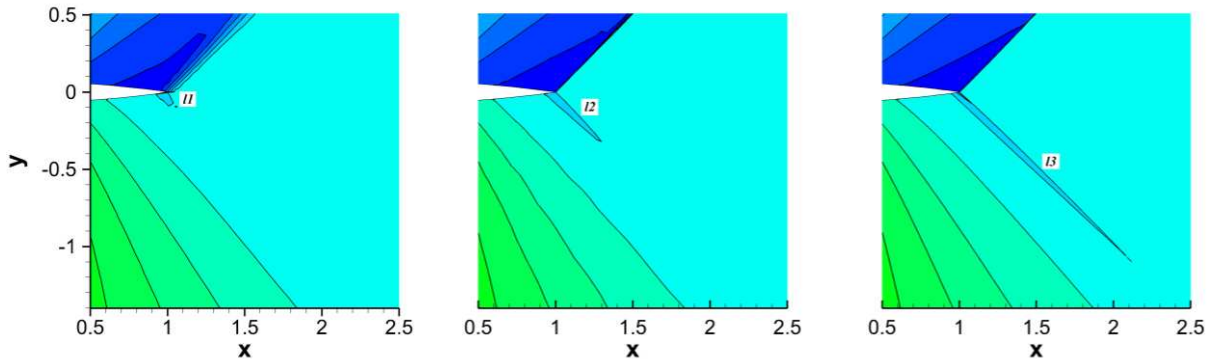


Fig. 19 The f-adaptive solution (Mach number contour).



(a) $l1$ = 0.085 (b) $l2$ = 0.424 (c) $l3$ = 1.570

Fig. 20 Comparison of the length of the weak shock wave under the trailing edge between: (a) initial solution, (b) r-adaptive solution and (c) f-adaptive solution.

Table 1 Comparison of mesh quality statistics for test case 1

Grid title	$q_{\min}$	$q_{\max}$
r-adaptive grid	0.003	0.105
feature-aligned grid	0.531	0.976
f-adaptive grid	0.166	0.963

B. Test Case 2: Transonic inviscid flow over a double wedge airfoil profile

A transonic inviscid flow problem containing shock wave reflection and interaction is tested for a double wedge airfoil profile at the center of a 8×2 region whose upper and lower boundary conditions are inviscid wall. The chord of the airfoil is 1 and the maximum vertical height is 0.1. A Mach 1 free stream at a zero angle of attack flows from the left boundary. The schematic diagram of the flow solution domain is shown in Fig. 21. A relative coarse structured grid with 5548 cells and 6000 nodes is generated. The converged initial solution, solved by the Euler equation, is shown in Fig. 22. After compression and expansion of the airfoil surface, two oblique shock waves are generated at the trailing edge, which are reflected and interacts. Because of numerical dissipation, the shock waves away from the airfoil are poorly resolved.

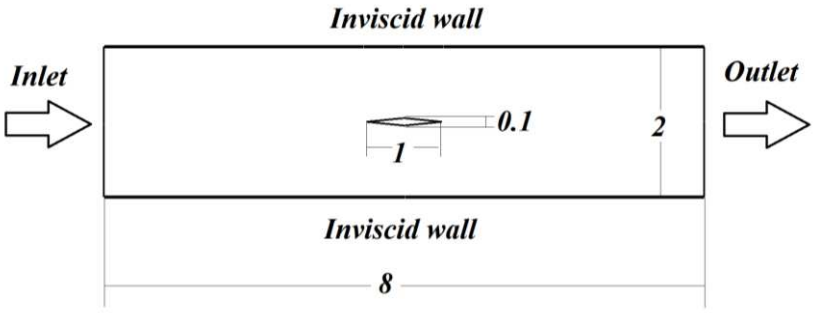


Fig. 21 A transonic inviscid flow over a double wedge airfoil profile.

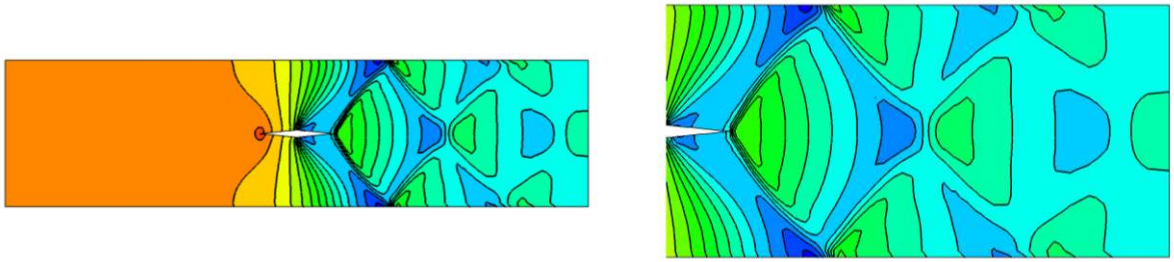


Fig. 22 Pressure contours solved on the initial mesh: farfield view (left) and close up of shock region (right).

The r-adaptive solution and grid are presented in Fig. 23. The extracted flow feature curves from the first and second iterations are shown in Fig. 24, and for the latter, the flow feature converges with a high resolution. Fig. 25 shows the f-adaptive mesh generation process in the second loop. Compared with the initial results, the f-adaptive result shows a significant improvement in the precision of shock wave capturing, including shock reflection and shock-shock interaction processes.

To compare the simulation efficiency, two different adaptive meshing methods were implemented and tested. One used the h-adaptive method on the initial structured grid, while the other applied the r-adaptive method on a refined structured grid with 100,000 nodes and 98,208 cells. The two methods were converged to the same degree to that using our proposed method, and the final mesh and convergence results (pressure contours) are shown in Fig. 26. Table 2 lists the comparison of grid information and the CPU time of the three methods for this case. Considering the same calculation conditions, the proposed method shows advantages in grid size, total calculation time and efficiency of each calculation step.

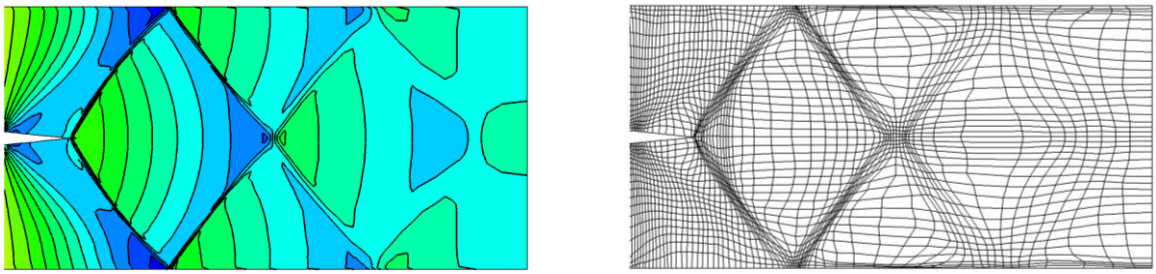


Fig. 23 The r-adaptive solution (pressure contour) and grid.

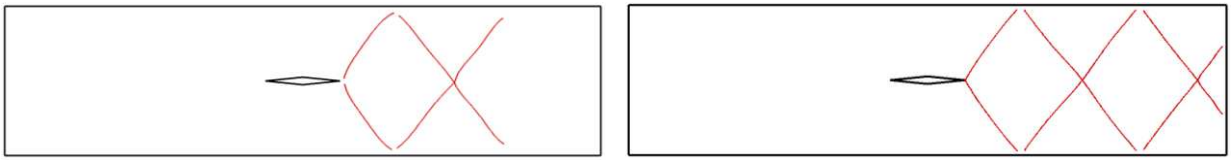
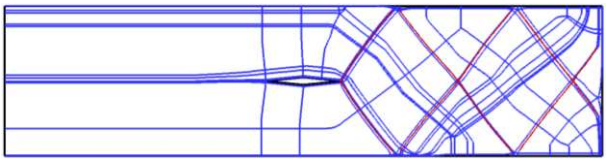
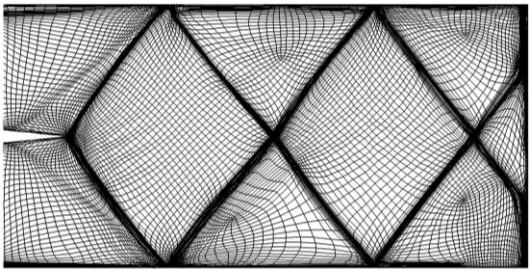


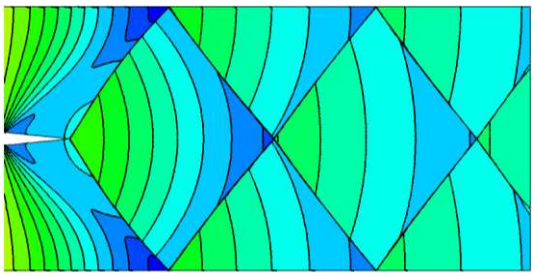
Fig. 24 Extracted flow feature lines: the first (left) and the second (right) iteration respectively.



a) Cross field based block regeneration

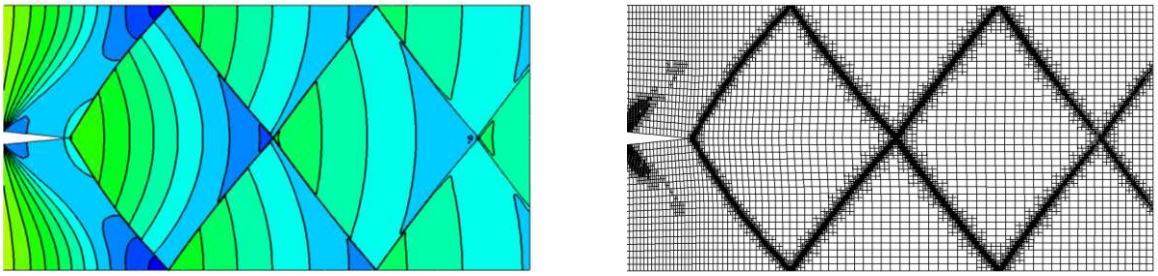


b) The f-adaptive mesh

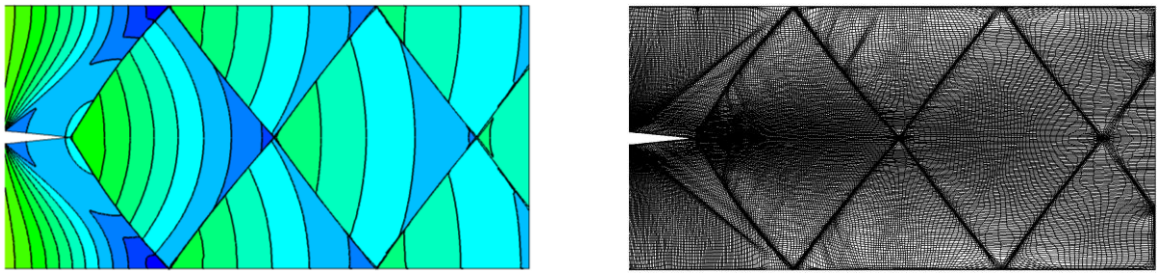


c) Pressure contour of the f-adaptive solution

Fig. 25 Description of the feature-aligned adaptive mesh generation for Test Case 2.



a) The h-adaptive method on the coarse initial grid



b) The r-adaptive method on the refined initial grid

Fig. 26 The adapted solution and grids using two other adaptive strategies.

Table 2 Comparison of mesh number and CPU time for test case 2

Adaptive strategy	Number of nodes		Number of cells		CPU time (second)	
	initial	final	initial	final	total	average each solution step
r-adaptive	10000	10000	98208	98208	11363	1.00
h-adaptive	6000	53235	5548	44977	4503	0.45
f-adaptive	6000	24208	5548	21202	2881	0.03

C. Test Case 3: Delery Channel Bump (Case C)

This is a classic case of shock wave and turbulent boundary layer interaction from Delery's transonic wind tunnel experiment [28]. The domain is taken from the test section of the wind tunnel, with a height of 100 mm and a length of 515 mm. The bump has a length of 286.37 mm and a crest height of 12 mm, and 75 mm away from the inlet. In

order to simulate the experimental condition, the inlet properties are set with free stream Mach number = 0.615, total temperature = 300 K, the total pressure = 96 kPa and Reynolds number = 1.1×10^7 [29]. The back pressure is set as 61.5 kPa. The initial solution in Fig. 28 is calculated on a 120×120 initial structured grid shown in Fig. 27. A lambda shock pattern is shown in the Mach number contour, the oblique leg of which induces boundary layer separation on the lower wall. The trailing normal shock that sits on top of the separated region merges with the oblique shock at the triple point with a strong normal shock. Near the upper wall, a much weaker lambda shock pattern is captured clearly from the f-adaptive solution.

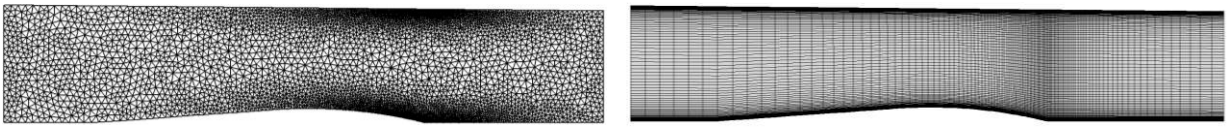


Fig. 27 The background (left) and structured (right) grid for simulation of the initial solution in test case 3.

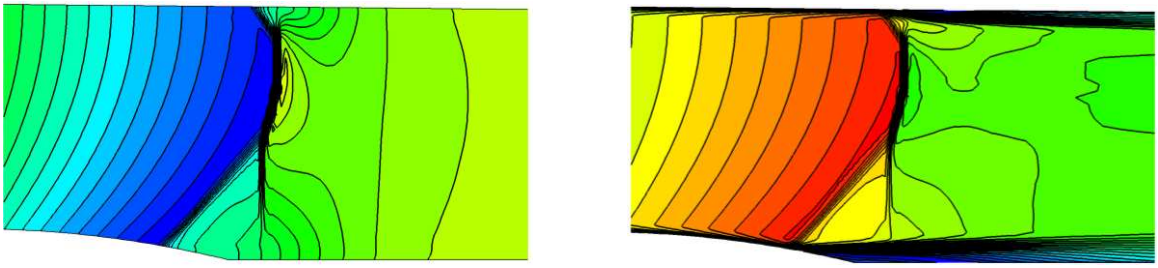


Fig. 28 The initial solution: Pressure contour (left) and Mach number contour (right).

For improvement of the viscous compressible flow feature resolution, the density was used as the flow variable in the calculation of the Hessian matrix. As seen in Fig. 29, the geometric entities representing the dominant flow features, such as lambda shock waves and shear layers, are retrieved and embedded in the solution domain. Fig. 30 shows the cross-field method generated structured block topology.

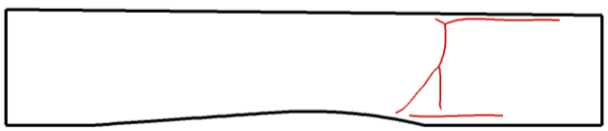


Fig. 29 The extracted flow feature curves (red lines) and the domain boundaries (black lines).

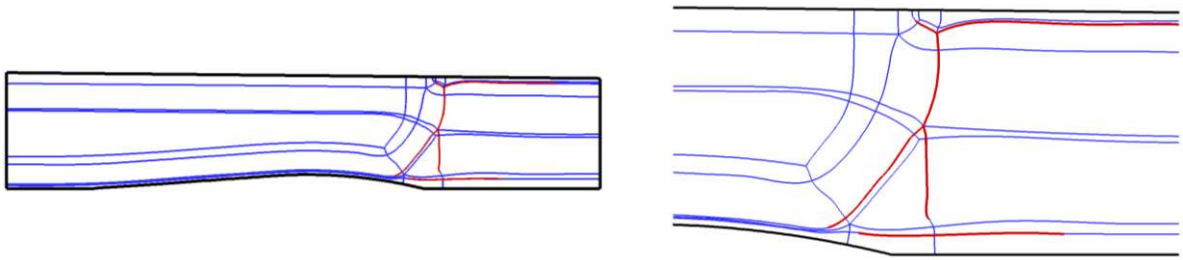


Fig. 30 Structured blocks (left) with flow feature curves being inserted and its close-up view (right) for Test Case 3.

Fig. 31 shows the corresponding feature-aligned grid generated (left) and the f-adaptive mesh (right). The f-adaptive mesh exhibits good feature alignment properties even if the original feature-aligned mesh did not exactly match the feature position. The main flow features are resolved explicitly in the f-adaptive grids (as shown in Fig. 32), namely, two lambda shock waves close to the lower and upper walls respectively, the shear layers generated by the separation of the boundary layer induced by the oblique shocks, and the shear layers from the two triple points. The Mach number contours with top value capped to show the shear layers from the triple points (middle and top) are shown in Fig. 33.

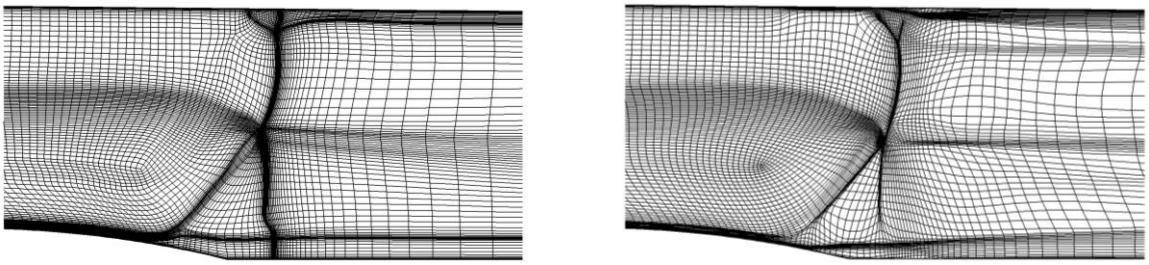


Fig. 31 Initial feature aligned mesh(left) and after f-adaptation (right).

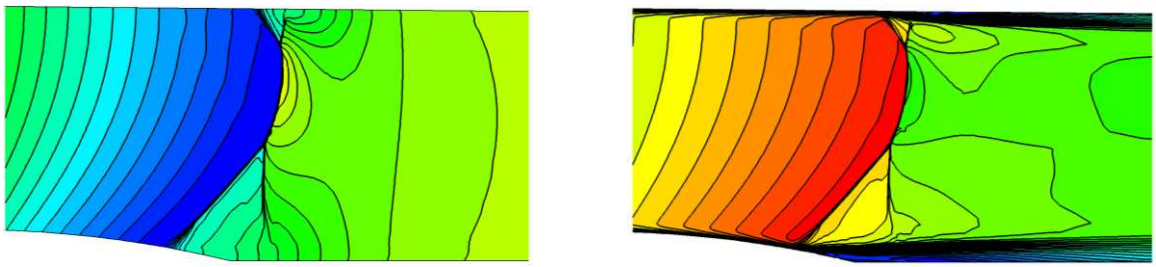


Fig. 32 The f-adaptive solution for Test Case 3: pressure contour (left) and Mach number contour (right).

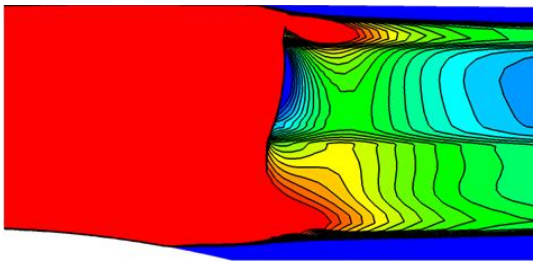


Fig. 33 The Mach number contours showing the boundary layers and the shear layers from triple points of shock interaction from the f-adaptive solution.

D. Test case 4: Transonic turbulent flow over RAE2822 airfoil with a bump

The wave drag for airfoils at a transonic speed can be effectively reduced using a shock control bump as presented by Qin et al.[30]. In this last test case, we demonstrate the application of the proposed f-adaptive mesh for the RAE2822 airfoil with a shock control bump. The flow conditions are $M = 0.785$ at $\alpha = 2°$ and $Re = 1.68 \times 10^7$. The bump design parameters are shown in Fig. 34 and Table 3. The bump is broken at the apex and is described by two cubic splines, $y = a_1x^3 + a_2x^2 + a_3x + a_4$. The bump curve is continuously differentiable at the apex and both ends.

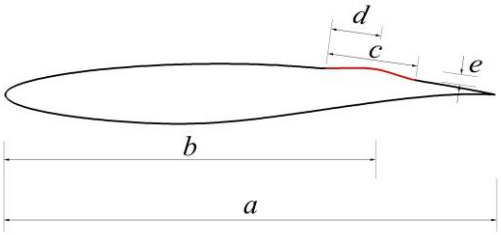


Fig. 34 The RAE2822 airfoil with a bump.

Table 3 The parameters of the bump for Test Case 4

Parameters	Values	Description
a	1	The chord length of the airfoil
b	0.75	The acme position of the bump along the chord
c	0.215	The length of the bump
d	0.55	The acme position along the bump
e	0.011	The height of the bump

An initial structured grid with 14607 cells (Fig. 35) is generated using the cross-field approach for the initial solution. Fig. 36 presents the pressure contour and the Mach number contour. Flow field shows a lambda shock wave, shock-induced flow separation at the trailing edge of the bump, and a wake from the trailing edge of the airfoil. The first cell y^+ distribution of the wall surface is in O(1) to resolve the viscous layer.

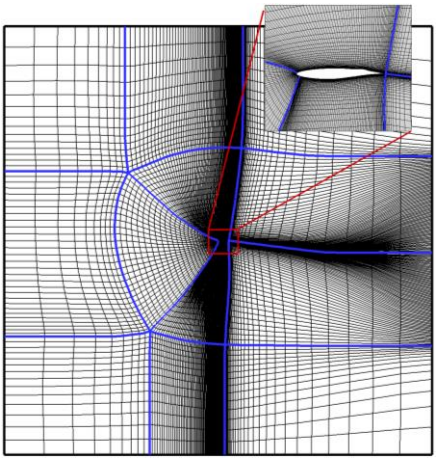


Fig. 35 The structured grid and blocks before flow feature extraction for Test Case 4.

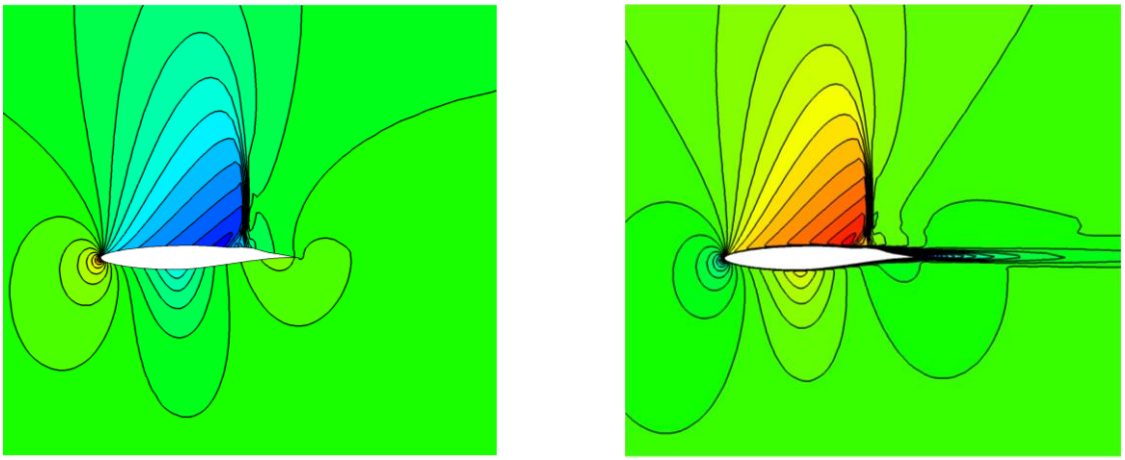


Fig. 36 The initial solution for Test Case 4: Pressure contour (left) and Mach number contour (right).

The above flow features are extracted as curves and the feature-aligned grid is then generated, as shown in Fig. 37. One of the advantages for the mesh is that the interaction between shock waves and boundary layers is also marked at the foot of the shock wave, and flow feature curves guide different directions of mesh generation, which allows for anisotropic mesh adaptation. The result in Fig. 38 show the improved flow feature resolution against the initial solution.

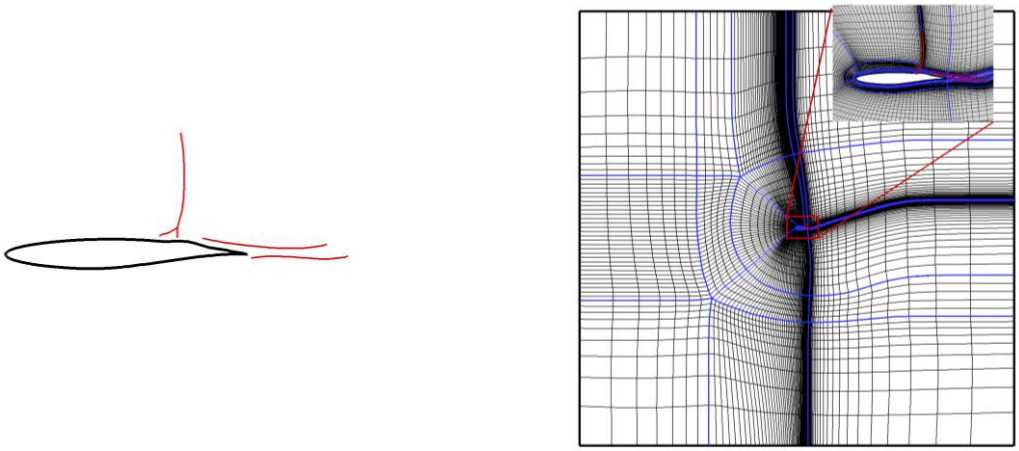
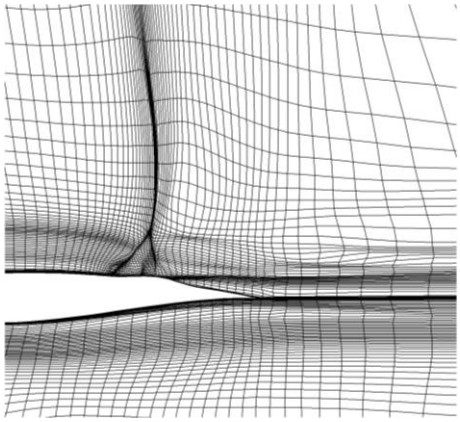
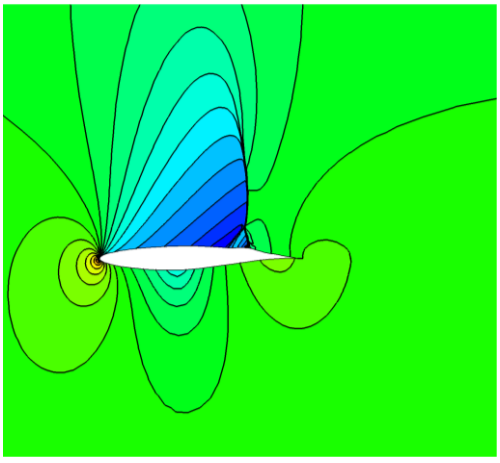


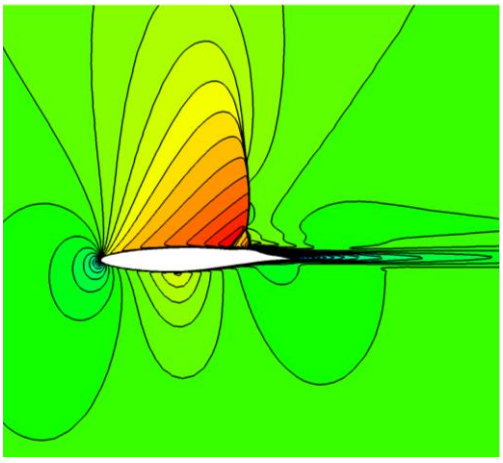
Fig. 37 The extracted flow feature curves and corresponding feature-aligned mesh.



a) The f-adaptive grid



b) Pressure contour



c) Mach number contour

Fig. 38 The f-adaptive solution of Test Case 4.

Quantitatively, Fig. 39 shows a drag comparison of a mesh sensitivity study and the f-adaptive mesh. In the former, the mesh is continuously refined until the drag value becomes stable or converged. For the mesh sensitivity of the f-

adaptive grid, we also doubled the grid number by refinement. As shown in Eq.7, the results remain almost the same. It can be seen from the total drag coefficient comparison, Fig. 39 (top), that f-adaptive result reaches similar drag value with about 4% of the cell count of the final refined mesh, which demonstrate that a significant computational efficiency can be achieved with the f-adaptive mesh. It is also interesting to point out that, for the shock boundary layer interaction problem, the pressure drag, Fig. 39 (middle), is more sensitive to the mesh resolution as compared with the skin friction drag, Fig. 39 (bottom). This is believed to be due to the sensitivity of the pressure drag to the lambda shock structure resolution by the mesh, which is better resolved using the f-adaptive approach.

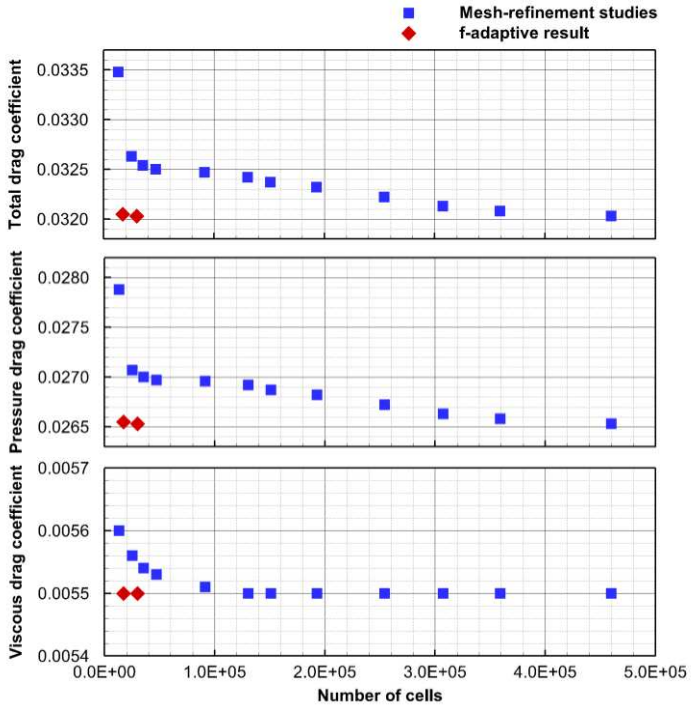


Fig. 39 Comparison of the drag coefficient for Test Case 4.

V. Conclusion

A method for generating feature-aligned anisotropic and multi-block structured adaptive meshes has been proposed in this paper for solving two-dimensional flow problems. By applying the cross-filed based method, the feature-aligned multi-block structured mesh can be automatically generated with high efficiency and high mesh quality. The flow feature curves are extracted using the medial axis method and embedded in the physical domain to guide the mesh alignment for the multi-block structured mesh regeneration and thus provide smooth and orthogonal structured meshes for capturing flow features such as shock waves, boundary layers and their interactions. The method

has been demonstrated for both inviscid and viscous test cases. High resolution was accomplished by utilizing a f-adaptive mesh, for some weak flow features, such as weak shock waves, slip lines and weak shock boundary layer interaction, which are not captured well by other adaptive solutions tested. Comparing with the traditional r-adaptive and h-adaptive method, the method has shown better efficiency for the adaptive mesh to resolve flow features due to feature alignment. In particular, the mesh convergence for pressure drag is significantly improved for the airfoil shock boundary layer interaction case with a shock control bump due to flow feature aligned adaptation.

The idea presented in this paper is applicable for three-dimensional problems. For extension to 3D, we anticipate two key steps. The first is the extraction of three-dimensional flow features. The flow features can be identified as 3D surfaces, in which the concept of mid-surface proposed by Ramanathan and Gurumoorthy [31] may be exploited. The second necessary extension is replacing the cross-field method for 2D to the frame-field method in 3D, as proposed by Kowalski et al. [16], for automatic generation of 3D multi-block structured grids.

Acknowledgments

This work is partially funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions of China and national natural science foundation to the 2nd author (No.12032011).

References

[1] Qin, N. and Liu, X., "Flow feature aligned grid adaptation," *International Journal for Numerical Methods in Engineering,* vol. 67, no. 6, 2006, pp. 787-814.
doi: 10.1002/nme.1648

[2] Habashi, W. G., Dompierre, J., Bourgault, Y., Ait-Ali-Yahia, D., Fortin, M., and Vallet, M.-G., "Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: general principles," *International Journal for Numerical Methods in Fluids,* vol. 32, no. 6, 2000, pp. 725-744.
doi: 10.1002/(SICI)1097-0363(20000330)32:6<725::AID-FLD935>3.0.CO;2-4

[3] Ait-Ali-Yahia, D., Baruzzi, G., Habashi, W. G., Fortin, M., Dompierre, J., and Vallet, M.-G., "Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part II. Structured grids," *International Journal for Numerical Methods in Fluids,* vol. 39, no. 8, 2002, pp. 657-673.
doi: 10.1002/fld.356

[4] Dompierre, J., Vallet, M.-G., Bourgault, Y., Fortin, M., and Habashi, W. G., "Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part III. Unstructured meshes," *International journal for numerical methods in fluids,* vol. 39, no. 8, 2002, pp. 675-702.
doi: 10.1002/fld.357

[5] Marcum, D. and Gaither, K., "Solution adaptive unstructured grid generation using pseudo-pattern recognition techniques," in *13th Computational Fluid Dynamics Conference*, 1997, pp. 1860.

[6] Harris, M. J. and Qin, N., "Using the medial axis to represent flow features for feature-aligned unstructured quad-dominant mesh generation," *Computers & Fluids,* vol. 102, pp. 1-14, 2014.
doi: 10.1016/j.compfluid.2014.06.017

[7] Harris, M. J. and Qin, N., "Geometric representation of flow features using the medial axis for mesh generation," *AIAA Journal,* vol. 53, no. 1, 2015, pp. 246-259.
doi: 10.2514/1.J053088

[8] Bunin, G., "A continuum theory for unstructured mesh generation in two dimensions," *Computer Aided Geometric Design,* vol. 25, no. 1, 2008, pp. 14-40.
doi: 10.1016/j.cagd.2007.05.002

[9] Bunin, G., "Towards unstructured mesh generation using the inverse poisson problem," *eprint arXiv:0802.2399,* 2008.
doi: 10.48550/arXiv.0802.2399

[10] Kowalski, N., Ledoux, F., and Frey, P., "Automatic domain partitioning for quadrilateral meshing with line constraints," *Engineering with Computers,* vol. 31, no. 3, 2015, pp. 405-421.
doi: 10.1007/s00366-014-0387-5

[11] Xiao, Z., He, S., Xu, G., Chen, J., and Wu, Q., "A boundary element-based automatic domain partitioning approach for semi-structured quad mesh generation," *Engineering Analysis with Boundary Elements,* vol. 113, 2020, pp. 133-144.
doi: 10.1016/j.enganabound.2020.01.003

[12] Marcon, J., Kopriva, D. A., Sherwin, S. J., and Peiró, J., "A high resolution PDE approach to quadrilateral mesh generation," *Journal of Computational Physics,* vol. 399, 2019, p. 108918.
doi: 10.1016/j.jcp.2019.108918

[13] Fogg, H. J., Armstrong, C. G., and Robinson, T. T., "Automatic generation of multiblock decompositions of surfaces," *International Journal for Numerical Methods in Engineering,* vol. 101, no. 13, 2015, pp. 965-991.
doi: 10.1002/nme.4825

[14] Fogg, H. J., Armstrong, C. G., and Robinson, T. T., "Enhanced medial-axis-based block-structured meshing in 2-D," *CAD Computer Aided Design,* vol. 72, 2016, pp. 87-101.

doi: 10.1016/j.cad.2015.07.001

[15] Li, Y., Liu, Y., Xu, W., Wang, W., and Guo, B., "All-hex meshing using singularity-restricted field," *ACM Transactions on Graphics,* vol. 31, no. 6, 2012, pp. 1-11.
doi: 10.1145/2366145.2366196

[16] Kowalski, N., Ledoux, F., and Frey, P., "Smoothness driven frame field generation for hexahedral meshing," *CAD Computer Aided Design,* vol. 72, 2016, pp. 65-77.
doi: 10.1016/j.cad.2015.06.009

[17] Liu, H., Zhang, P., Chien, E., Solomon, J., and Bommes, D., "Singularity-constrained octahedral fields for hexahedral meshing," *ACM Transactions on Graphics,* vol. 37, no. 4, 2018, p. Article 93.
doi: 10.1145/3197517.3201344

[18] Spalart, P. and Allmaras, S., "A one-equation turbulence model for aerodynamic flows," in *30th aerospace sciences meeting and exhibit*, 1992, pp. 439.

[19] Kim, K. H., Lee, J. H., Rho, O. H. J. C., and fluids, "An improvement of AUSM schemes by introducing the pressure-based weight functions," *Computers and Fluids,* vol. 27, no. 3, 1998, pp. 311-346.
doi: 10.1016/S0045-7930(97)00069-8

[20] Edelsbrunner, H., Kirkpatrick, D., and Seidel, R., "On the shape of a set of points in the plane," *IEEE Transactions on Information Theory,* vol. 29, no. 4, 1983, pp. 551-559.
doi: 10.1109/TIT.1983.1056714

[21] Shewchuk, J. R., Triangle, Software Package, Ver. 1.6, 2005. URL: https://www.cs.cmu.edu/~quake/triangle.html

[22] Elias, R. N., Martins, M. A. D., and Coutinho, A., "Simple finite element-based computation of distance functions in unstructured grids," *International Journal for Numerical Methods in Engineering,* Article vol. 72, no. 9, 2007, pp. 1095-1110.
doi: 10.1002/nme.2079

[23] Dorobantu, M., *Efficient streamline computations on unstructured grids*. Kungliga Tekniska Högskolan. Institut för Numerisk Analys och Datalogi, 1997.

[24] Vinokur, M., "On one-dimensional stretching functions for finite-difference calculations," *Journal of Computational Physics,* vol. 50, no. 2, 1983, pp. 215-234.
doi: 10.1016/0021-9991(83)90065-7

[25] Thompson, J. F., Warsi, Z. U., and Mastin, C. W., *Numerical grid generation: foundations and applications*. Elsevier North-Holland, Inc., 1985.

[26] Pulliam, T. and Barton, J., "Euler computations of AGARD working group 07 airfoil test cases," in *23rd Aerospace Sciences Meeting*, 1985, pp. 18.

[27] Knupp, P. M., "Algebraic mesh quality metrics for unstructured initial meshes," *Finite Elements in Analysis and Design,* vol. 39, no. 3, 2003 pp. 217-241.
doi: 10.1016/S0168-874X(02)00070-7

[28] Delery, J. M., "Experimental investigation of turbulence properties in transonic shock/boundary-layer interactions," *AIAA journal,* vol. 21, no. 2, pp. 180-185, 1983.
doi: 10.2514/3.8052.

[29] Qin, N. and Zhu, Y., "Grid adaption for shock/turbulent boundary-layer interaction," *AIAA journal,* vol. 37, no. 9, 1999, pp. 1129-1131.
doi: 10.2514/2.825

[30] Qin, N., Wong, W., and Le Moigne, A., "Three-dimensional contour bumps for transonic wing drag reduction," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering,* vol. 222, no. 5, 2008, pp. 619-629.
doi: 10.1243/09544100JAERO333

[31] Ramanathan, M. and Gurumoorthy, B., "Generating the Mid-Surface of a Solid using 2D MAT of its Faces," *Computer-Aided Design and Applications*, vol. 1, no. 1-4, 2004, pp. 665-674.
doi: 10.1080/16864360.2004.10738312