Commun. Comput. Phys.
doi: 10.4208/cicp.OA-2021-0143

Vol. **32**, No. 2, pp. 490-523
August 2022

A NURBS-Enhanced Finite Volume Method for Steady Euler Equations with Goal-Oriented h-Adaptivity

Xucheng Meng[1,2] and Guanghui Hu[3,4,5,*]

[1] *Research Center for Mathematics, Beijing Normal University, Zhuhai 519087, China.*
[2] *BNU-HKBU United International College, Zhuhai 519087, China.*
[3] *Department of Mathematics, University of Macau, Macao SAR, China.*
[4] *Zhuhai UM Science and Technology Research Institute, Zhuhai, Guangdong Province, China.*
[5] *Guangdong-Hong Kong-Macao Joint Laboratory for Data-Driven Fluid Dynamics and Engineering Applications, University of Macau, Macao SAR, China.*

Received 2 July 2021; Accepted (in revised version) 28 May 2022

Abstract. In [*A NURBS-enhanced finite volume solver for steady Euler equations, X. C. Meng, G. H. Hu, J. Comput. Phys., Vol. 359, pp. 77–92*], a NURBS-enhanced finite volume method was developed to solve the steady Euler equations, in which the desired high order numerical accuracy was obtained for the equations imposed in the domain with a curved boundary. In this paper, the method is significantly improved in the following ways: (i) a simple and efficient point inversion technique is designed to compute the parameter values of points lying on a NURBS curve, (ii) with this new point inversion technique, the h-adaptive NURBS-enhanced finite volume method is introduced for the steady Euler equations in a complex domain, and (iii) a goal-oriented *a posteriori* error indicator is designed to further improve the efficiency of the algorithm towards accurately calculating a given quantity of interest. Numerical results obtained from a variety of numerical experiments with different flow configurations successfully show the effectiveness and robustness of the proposed method.

AMS subject classifications: 76M12, 65N50, 35Q31, 65D07

Key words: Steady Euler equations, NURBS-enhanced finite volume method, goal-oriented *a posteriori* error estimation, non-oscillatory k-exact reconstruction, point inversion.

1 Introduction

Aerodynamic shape optimal design [17,30,43] plays an increasingly important role in the design of vehicle and aircraft. The main components of the aerodynamic shape optimal

*Corresponding author. *Email addresses:* xcmeng@bnu.edu.cn (X. Meng), garyhu@um.edu.mo (G. Hu)

design include a flow solver, shape parameterization, optimization algorithms, and mesh deformation methods, we refer to [12, 43, 46] and the references therein for the details. To efficiently implement the aerodynamic shape optimization, one needs to develop an efficient flow solver and an appropriate shape parameterization technique.

The high order numerical methods using the goal-oriented mesh adaptation technique have been commonly used to efficiently compute the quantity of interest [6, 7, 16, 18, 21, 29, 42, 48]. On the other hand, the B-splines and Non-Uniform Rational B-splines (NURBS) [36] have been widely utilized for the shape parameterization in both the structural shape optimization problem [11, 37, 45] and the aerodynamic shape optimization problem [3, 43, 46]. Although the goal-oriented mesh adaptation technique is popular in numerical analysis, and NURBS are prevalent in Computer Aided Design (CAD), it should be pointed out that their quality combination, which should be very attractive, is not a trivial task. Hence, the primary goal of this paper is to investigate those two techniques in a synthesize way to solve the steady Euler equations by following our previous works [23, 26, 27, 35]. It is worthwhile to note that the adaptive refinement is still primarily an academic endeavor rather than an industrial technology, and the reason for this phenomenon may be that the link for the communication between the mesh refinement and the CAD system is often unavailable [28].

In [35], a NURBS-enhanced high order finite volume scheme on unstructured grids was developed to solve the two-dimensional (2D) steady Euler equations in a curved domain. Although the numerical results presented there have shown that the proposed method possesses the high order behavior, the method is unsatisfactory since uniformly refined meshes were used for the simulations. The uniform mesh refinement is not an efficient way to reduce the error and to save computational cost. For example, in the case of an inviscid subsonic flow through a Gaussian bump [47], the high order finite volume solver using uniformly refined meshes is not an efficient way to reduce the entropy error since the error around the Gaussian bump accounts for the majority of total entropy error. The benefits of high order methods are reduced when singularities are present in the solutions and the uniformly refined meshes are used [4, 38, 42, 49].

The NURBS-enhanced finite volume method [35] uses the NURBS to represent the curved boundary of physical domain, and the mesh refinement procedure does not need to communicate with the CAD system. To introduce the *h*-adaptive mesh refinement method in the numerical solver developed in [35], two issues need to be resolved well. The first one is how to efficiently obtain the parameter values of points lying on a NURBS curve. The second one is how to design reliable error indicators. We need the former one to flexibly insert and/or remove the grid points lying on the NURBS curve, and to efficiently obtain the quadrature information on the curved edges locating on the NURBS curve, while the latter one is not only for the efficiency of the algorithm, but also for the application of the method to shape optimization problems.

The *p*-adaptivity is another way to save the computational cost, and it has been studied in [41] for the Stokes flows by using the NURBS-enhanced finite element method (NEFEM) in combination with the hybridisable discontinuous Galerkin (HDG) method.

To the best of our knowledge, the *h*-adaptivity has not been studied within the framework of NURBS-enhanced type numerical methods.

In this paper, based on our previous works [26, 27, 35], the NURBS-enhanced high order finite volume method with goal-oriented *h*-adaptivity is developed to solve the 2D steady Euler equations. The numerical framework is based on a Newton-Geometric Multigrid (Newton-GMG) finite volume method [24, 25, 34], and mainly consists of three steps. Firstly, the cell-centered finite volume method is used to discretize the non-linear governing equations. Secondly, the Newton method is adopted to linearize the discrete formulation, and finally, the GMG is utilized to solve the derived system of linear equations. The non-oscillatory *k*-exact reconstruction proposed in [27] is adopted in the method for the high order solution reconstruction. To achieve the high order accuracy for problems imposed in the domain with a curved boundary, we use the NURBS to obtain an exact or a high order representation of the curved boundary of domain [35, 39, 40].

As aforementioned, in the process of *h*-adaptive mesh refinement, it is necessary to locally insert and/or remove gird points on the boundary of cells. Since the NURBS curve is of parametric form and the parameter values for points lying on it are not available in traditional numerical solvers, an efficient approach to calculating those parameter values is necessary for the NURBS-enhanced type numerical methods. The point inversion technique introduced in [36] can be used to achieve this goal. However, the initial guess for the Newton method used in the point inversion technique developed in [36] is point-dependent, which means that the additional computational cost is required to obtain the initial guesses. Furthermore, the technique is not suitable for the NURBS curves of C^1-continuity since the second-order derivative of NURBS curve is needed [36]. In this paper, we design a simplified and efficient point inversion technique which is applicable to C^1-continuous NURBS curves, and the initial guesses for the Newton method are fixed. On obtaining the parameter values, the quadrature information and the mid-point of any curved edge can be easily achieved.

The efficiency and effectiveness of the *h*-adaptive mesh refinement method are guaranteed by two factors. The first one is the use of Hierarchy Geometry Tree (HGT) data structure to manage the mesh refinement. The HGT [33] is based on the four-fork tree data structure in the 2D case. With HGT, the local refinement and/or coarsening of the mesh can be implemented efficiently. In addition, the solution interpolation between two meshes can also be implemented efficiently. The second factor is a goal-oriented *a posteriori* error estimation based on the numerical solutions. Based on this error estimation, an error indicator can be designed to drive the mesh to be locally refined to accurately calculate the given quantity of interest. This feature makes the method very attractive in the shape optimization problems, see for example [16, 32, 46]. It is noted that in the Newton-MGM finite volume method, the Jacobian matrix of the numerical flux function is approximated by a finite difference scheme. The same difference scheme can be used to numerically compute the Jacobian matrices of both the discrete residual equations and the given quantity of interest.

The numerical method is realized on a C++ library called AFVM4CFD [26], and a

variety of numerical experiments with different configurations are tested in the paper. First of all, the robustness of the numerical method can be observed from the fact that the convergence to the steady state can be achieved for all numerical examples with one set of parameters. Secondly, with the help of the goal-oriented h-adaptive refinement method, the desired convergence of the error in quantity of interest is obtained successfully, which shows the potential application of our method to the shape optimization problems.

The rest of this paper is organized as follows. In Section 2, we briefly introduce the 2D steady Euler equations and the high order finite volume method developed in [27]. In Section 3, we discuss how to generate a high order unstructured mesh based on NURBS, and we present the simplified point inversion technique. In Section 4, we derive the goal-oriented *a posteriori* error estimation to achieve the error indicators. A variety of numerical experiments are presented in Section 5 to demonstrate the effectiveness and robustness of the proposed method. Conclusions and the future works are given in the last section.

2 A Newton-GMG high order finite volume method for steady Euler equations

In this section, we briefly introduce the 2D steady Euler equations and the related discretization method for them. The interested reader is referred to [24–27,34] and the references therein for the details.

2.1 The finite volume scheme for 2D steady Euler equations

The 2D steady Euler equations of gas dynamics read

$$\frac{\partial \boldsymbol{f}(\boldsymbol{U})}{\partial x}+\frac{\partial \boldsymbol{g}(\boldsymbol{U})}{\partial y}=\boldsymbol{0}, \tag{2.1}$$

where $\boldsymbol{U}$ is the vector of conserved variables, $\boldsymbol{f}(\boldsymbol{U})$ and $\boldsymbol{g}(\boldsymbol{U})$ are the flux functions in the x- and y- directions, respectively. Specifically, they are defined by

$$\boldsymbol{U}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ E\end{bmatrix},\quad \boldsymbol{f}(\boldsymbol{U})=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\end{bmatrix},\quad \boldsymbol{g}(\boldsymbol{U})=\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\end{bmatrix}, \tag{2.2}$$

where $\boldsymbol{u}=(u,v)^T$ is the velocity field, ρ is the density, p is the pressure, and E is the total energy per unit volume.

Let $\mathbf{F}(\mathbf{U})=(\mathbf{f}(\mathbf{U}),\mathbf{g}(\mathbf{U}))$, then the 2D steady Euler equations (2.1) can be written more compactly as

$$\boldsymbol{\mathcal{R}}(\boldsymbol{U}):=\nabla\cdot\boldsymbol{F}(\boldsymbol{U})=\boldsymbol{0}. \tag{2.3}$$

An equation of state is needed to complete the system of equations (2.1). For a polytropic ideal gas, it reads

$$p=(\gamma-1)\Big(E-\frac{1}{2}\rho(u^2+v^2)\Big), \tag{2.4}$$

where γ is the ratio of the specific heats, and is set to 1.4 for the numerical examples.

To discretize the non-linear governing equations (2.1), the cell-centered finite volume scheme is used in this paper. Let $\mathcal{T}_h=\{T_i\}_{i=1}^{N_h}$ be an unstructured triangulation of the physical domain $\Omega\subset\mathbb{R}^2$, where N_h is the number of cells of the triangulation. We first consider the integral form of the governing equations (2.3) over $T_i\in\mathcal{T}_h$, where $1\le i\le N_h$, and then use the divergence theorem, we obtain

$$\int_{T_i}\nabla\cdot\boldsymbol{F}(\boldsymbol{U})dxdy=\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\boldsymbol{F}(\boldsymbol{U})\cdot\boldsymbol{n}_{ij}ds=\boldsymbol{0},\quad\text{for}\quad i=1,2,\cdots,N_h, \tag{2.5}$$

where $e_{ij}=\partial T_i\cap\partial T_j$, here T_j denotes the von Neumann neighbor of T_i, and $\boldsymbol{n}_{ij}$ is the unit outward normal to the edge e_{ij} with respect to T_i.

We replace the flux $\boldsymbol{F}(\boldsymbol{U})$ appearing in Eq. (2.5) by the numerical flux $\bar{\boldsymbol{F}}(\boldsymbol{U}_i,\boldsymbol{U}_j)$, then we achieve the following discrete system

$$\boldsymbol{r}_i:=\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\bar{\boldsymbol{F}}(\boldsymbol{U}_i,\boldsymbol{U}_j)\cdot\boldsymbol{n}_{ij}ds=\boldsymbol{0},\quad\text{for}\quad i=1,2,\cdots,N_h, \tag{2.6}$$

where $\boldsymbol{r}_i$ is referred to as the cell residual associated with the cell T_i [12]. The Harten-Lax-van Leer-Contact (HLLC) [44] numerical flux is used for all numerical examples. Let $\boldsymbol{\mathcal{R}}_{\boldsymbol{h}}=(\boldsymbol{r}_1,\boldsymbol{r}_2,\cdots,\boldsymbol{r}_{N_h})^T$ be the residual vector, then the discrete system (2.6) can be written as

$$\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h)=\boldsymbol{0}. \tag{2.7}$$

2.2 The Newton iteration method and geometric multigrid method

Following [25, 27, 34], the Newton iteration method is adopted to linearize the nonlinear discrete system (2.7). By setting $\boldsymbol{U}_i^{(n+1)}=\boldsymbol{U}_i^{(n)}+\delta\boldsymbol{U}_i^{(n)}$ and keeping the linear terms in the Taylor's expansion, where $\delta\boldsymbol{U}_i^{(n)}$ is the increment of the conserved quantities in the cell T_i, we obtain the following linearized system

$$\begin{aligned}\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})\cdot\boldsymbol{n}_{ij}ds+\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\delta\boldsymbol{U}_i^{(n)}\frac{\partial\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})}{\partial\boldsymbol{U}_i^{(n)}}\cdot\boldsymbol{n}_{ij}ds\\+\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\delta\boldsymbol{U}_j^{(n)}\frac{\partial\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})}{\partial\boldsymbol{U}_j^{(n)}}\cdot\boldsymbol{n}_{ij}ds=\boldsymbol{0},\end{aligned} \tag{2.8}$$

where $\partial\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})/\partial\boldsymbol{U}_i^{(n)}$ and $\partial\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})/\partial\boldsymbol{U}_j^{(n)}$ denote the Jacobian matrices of the numerical flux. The Jacobian matrices can be approximated by the finite difference scheme, and we refer to [25, 27] and the references therein for the details. Actually, the linearized system (2.8) is singular, and in [34], the l^1-norm of cell residual is employed to regularize the system, that is, the regularized system reads

$$\begin{aligned}\beta\Big\|\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})\cdot\boldsymbol{n}_{ij}ds\Big\|_{l^1}\delta\boldsymbol{U}_i^{(n)}+\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\delta\boldsymbol{U}_i^{(n)}\frac{\partial\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})}{\partial\boldsymbol{U}_i^{(n)}}\cdot\boldsymbol{n}_{ij}ds\\ +\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\delta\boldsymbol{U}_j^{(n)}\frac{\partial\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})}{\partial\boldsymbol{U}_j^{(n)}}\cdot\boldsymbol{n}_{ij}ds=-\sum_{\forall e_{ij}\in\partial T_i}\int_{e_{ij}}\bar{\boldsymbol{F}}(\boldsymbol{U}_i^{(n)},\boldsymbol{U}_j^{(n)})\cdot\boldsymbol{n}_{ij}ds,\end{aligned}\tag{2.9}$$

where β is a positive parameter, and is set to 2 for all numerical examples.

To efficiently solve the regularized system (2.9), the geometric multigrid method (GMG) developed in [34] is adopted. The main components of the GMG are briefly summarized as follows. Following the idea of mesh aggregation algorithm presented in the Section 9.4 of [10], the coarse grid is generated by fusing the cells of the finer grid with their neighbors. The block lower-upper symmetric Gauss-Seidel (LU-SGS) iteration introduced in [14] is used as the smoother. For the multigrid method used in this work, the V-cycle type iteration is employed, and the number of steps of multigrid iteration is set to 2 in the computations. We refer the interested reader to [10, 23, 34] for the details.

2.3 The high order solution reconstruction

To prevent spurious oscillations around discontinuities and to achieve high order accuracy for flows with smooth solutions, the non-oscillatory k-exact reconstruction developed in [27] is used. In this paper, the case $k=2$ is considered, hence the third order numerical accuracy can be achieved for the flow problem with a smooth analytical solution. The basic idea of the method is that the k-exact reconstruction [8] is performed to obtain the candidate polynomials, and then the weighted essentially non-oscillatory (WENO) [22] reconstruction is implemented to prevent the nonphysical oscillations, we refer the interested reader to [27] for the details.

In the simulations, the initial mesh is an unstructured mesh consisting of triangles, and the h-adaptive mesh refinement is adopted to improve the computational efficiency. The hanging nodes would appear on the edges of some cells of h-adaptive meshes. In this paper, the mesh refinement strategy proposed in [33] is adopted to locally refine the meshes, and fortunately, it can guarantee that a cell would contain at most one hanging node, we refer to Fig. 1 for a locally refined mesh.

The finite volume method can be implemented on control volumes of any shape, and the k-exact reconstruction on a target cell can be performed as long as the cell averages distributed on its reconstruction patch are available. Therefore, the existence of hanging

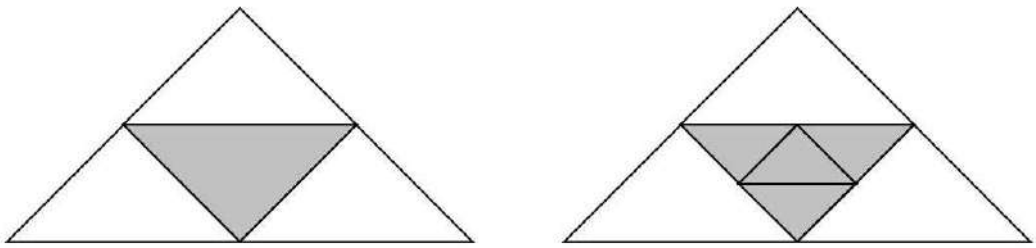

Figure 1: Left: a mesh consisting of 4 triangles, and the shaded triangle is to be locally refined. Right: the shaded triangle has been divided into four smaller triangles.

nodes has no effect on the solution reconstruction. For the cell containing a hanging node, we treat it as a triangle with four vertexes [33]. When the multiple reconstruction stencils for each cell are constructed, see [26] for the details on the construction of reconstruction stencils, the non-oscillatory k-exact reconstruction can be performed in the usual way, and we refer to [13, 23] and the references therein for more details on the polynomial reconstruction over h-adaptive meshes.

The reconstruction stencils designed in [26] are adopted to perform the non-oscillatory k-exact reconstruction. To be specific, for a given cell T_i, its Moore neighbors [35] and T_i itself are used to construct the reconstruction stencil $\mathcal{P}_i$. Moreover, when T_i is a cell interesting the boundary of domain, some additional cells along the inner normal direction to the edge located on the boundary of physical domain are selected as elements of $\mathcal{P}_i$. The numerical results presented in [26, 35] show that the convergence to steady state can be significantly improved by using such a revised reconstruction stencil for the boundary cell. Furthermore, for the two- and three-dimensional linear advection equation, it has been theoretically proved in [20] that the use of a larger reconstruction stencil would result in a more stable and robust scheme when the least squares slope reconstruction method is adopted.

3 NURBS curve, high order unstructured mesh, and point inversion algorithm

In this section, we first briefly recall the basic concepts of NURBS curve and curve fitting technique. Then we show how high order unstructured meshes can be generated by using EasyMesh [1] and NURBS [36]. Finally, we describe the new point inversion technique in detail. We refer the interested reader to [28, 36] and the references therein for more details on NURBS.

3.1 NURBS curve

A 2D p-th degree NURBS curve is a parametric curve, and is defined by

$$\mathbf{C}(\xi)=\big(x(\xi),y(\xi)\big)^T=\sum_{i=1}^{n}\mathbf{P}_i R_{i,p}(\xi),\quad 0\le\xi\le 1, \tag{3.1}$$

where n is the number of NURBS basis functions, $\mathbf{P}_i=(x_i,y_i)^T\in\mathbb{R}^2$ is the i-th control point, with $i=1,2,\cdots,n$. Moreover, $R_{i,p}(\xi)$, $i=1,2,\cdots,n$, are the NURBS basis functions, and are defined by

$$R_{i,p}(\xi)=\frac{w_i N_{i,p}(\xi)}{\sum_{j=1}^{n} w_j N_{j,p}(\xi)},\quad \text{for}\quad i=1,2,\cdots,n, \tag{3.2}$$

where $\{N_{i,p}(\xi)\}_{i=1}^{n}$ are the B-spline basis functions of degree p, and $\{w_i\}_{i=1}^{n}$ are the related weights. The i-th B-spline basis function, $N_{i,p}(\xi)$, is defined over a non-decreasing knot vector

$$\Xi=\{\xi_1=0\le\xi_2\le\cdots\le\xi_{n+p+1}=1\}. \tag{3.3}$$

Specifically, $N_{i,p}(\xi)$, where $1\le i\le n$, is recursively defined by

$$N_{i,k}(\xi)=\frac{\xi-\xi_i}{\xi_{i+k}-\xi_i}N_{i,k-1}(\xi)+\frac{\xi_{i+k+1}-\xi}{\xi_{i+k+1}-\xi_{i+1}}N_{i+1,k-1}(\xi), \tag{3.4}$$

for $k=1,2,\cdots,p$, where

$$N_{i,0}(\xi)=\begin{cases}1, & \text{if } \xi_i\le\xi<\xi_{i+1},\\ 0, & \text{otherwise.}\end{cases} \tag{3.5}$$

Note that the quotient $0/0$ is assumed to be 0 in the computations. The derivative of $N_{i,p}(\xi)$ is given by

$$N'_{i,p}(\xi)=\frac{p}{\xi_{i+p}-\xi_i}N_{i,p-1}(\xi)-\frac{p}{\xi_{i+p+1}-\xi_{i+1}}N_{i+1,p-1}(\xi). \tag{3.6}$$

The detailed implementation and the properties of NURBS can be found in [36].

Some commonly used geometries, including conic sections, disks, spheres and cylinders, can be exactly described by NURBS [28,36]. Due to this fact and the excellent properties of NURBS basis functions [28,36], NURBS have been widely used in the commercial CAD software as well as in the numerical analysis [28].

However, many geometries of practical applications can not be exactly represented by the NURBS, and sometimes what we know about the geometry are the points lying on the boundary of domain. In this case, the curve fitting technique is required to construct a NURBS curve/surface.

3.2 NURBS curve fitting

Generally speaking, there are two types of curve fitting technique to construct a NURBS curve when the points lying on the curved boundary of domain are available, i.e., the interpolation and approximation, and both methods can be utilized in either a global or a local way. In this work, we only consider the global interpolation as it is very easy to implement. We refer the interested reader to [36, 46] for other curve fitting methods.

Let $\{\mathbf{Q}_i\}_{i=1}^n$ be a set of points lying on the curved boundary of domain. The global interpolation is the process of determining a NURBS curve of degree p and a set of parameter values $\{\bar{\xi}_i\}_{i=1}^n$ such that

$$\mathbf{Q}_k=\mathbf{C}(\bar{\xi}_k)=\sum_{i=1}^{n}\mathbf{P}_i R_{i,p}(\bar{\xi}_k),\quad \text{for}\quad k=1,2,\cdots,n,\quad \text{and}\quad \bar{\xi}_k\in[0,1], \tag{3.7}$$

where, for simplicity, we set $w_1=w_2\cdots=w_n=1$ as in [36].

Following [36], the parameter values $\{\bar{\xi}_i\}_{i=1}^n$ are defined by

$$\bar{\xi}_1=0,\quad \bar{\xi}_k=\bar{\xi}_{k-1}+\frac{||\mathbf{Q}_{k+1}-\mathbf{Q}_k||_2}{d},\quad \text{for}\quad k=2,3,\cdots,n-1,\quad \text{and}\quad \bar{\xi}_n=1, \tag{3.8}$$

where $d=\sum_{k=1}^{n-1}\|\mathbf{Q}_{k+1}-\mathbf{Q}_k\|_2$ is the chord length of the points $\{\mathbf{Q}_i\}_{i=1}^n$, and $\|\cdot\|_2$ denotes the Euclid norm of a vector in $\mathbb{R}^2$.

Furthermore, as in [36], the knot vector Ξ is obtained by setting

$$\xi_{p+j}=\frac{1}{p}\sum_{i=j}^{j+p-1}\bar{\xi}_i,\quad \text{for}\quad j=2,\cdots,n-p, \tag{3.9}$$

with

$$\xi_1=\xi_2=\cdots=\xi_{p+1}=0,\quad \text{and}\quad \xi_{n+1}=\cdots=\xi_{n+p+1}=1.$$

Finally, the control points $\{\mathbf{P}_i\}_{i=1}^n$ can be determined by solving the non-singular system of linear equations (3.7), and we refer to [19,35,36] for the details. Now the NURBS curve of degree p is available to provide a high order description of the curved boundary of domain.

3.3 The generation of high order unstructured mesh

In this work, the 2D unstructured polygonal mesh is generated by EasyMesh [1]. Given a sequence of points lying on the boundary of physical domain, the EasyMesh would generate an unstructured mesh with straight edges, see Fig. 2 (the middle one) for a typical unstructured mesh generated by the EasyMesh.

It is well known in computational fluid dynamics (CFD) that the geometrical error introduced by using a polygon to approximate the curved boundary of domain not only destroys the high order behavior of high order methods, but also may produce spurious

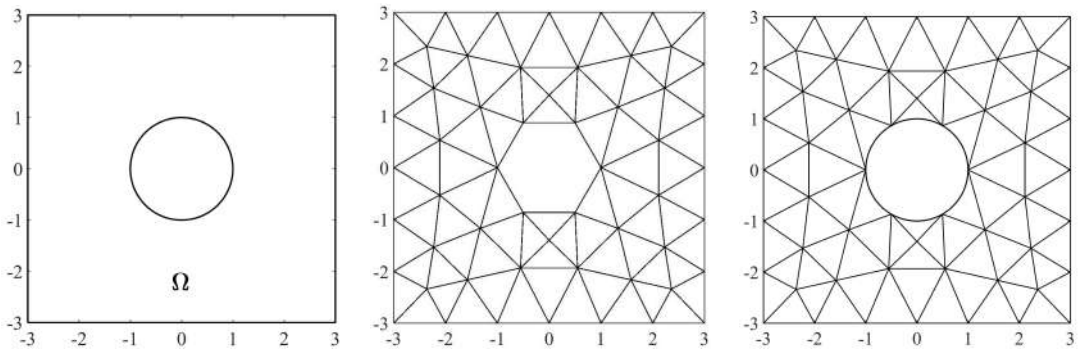

Figure 2: Left: the physical domain $\Omega=\{(x,y)\in\mathbb{R}^2|x^2+y^2\geq 1,-3\leq x,y\leq 3\}$. Middle: an unstructured mesh generated by EasyMesh. Right: the high order unstructured mesh obtained by using NURBS to describe the curved boundary.

numerical solutions [9,31]. Therefore, it is necessary to construct a high order description of the curved boundary in the development of high order methods, see [9,39,47] and the references therein for the details. In this paper, following [35,39], we utilize the NURBS curve to obtain an exact or a high order representation of the curved boundary. The numerical methods using NURBS to describe the curved boundary of domain are referred to as the NURBS-enhanced type methods [35,39,40].

The 2D high order unstructured mesh used in the NURBS-enhanced finite volume method [35] can be achieved by accomplishing the following steps:

(1) we first generate an unstructured mesh by using the EasyMesh and the given points lying on the boundary of domain;

(2) when the curved boundary of domain can not be exactly represented by any NURBS curve, by using the points lying on the curved boundary of domain and the curve fitting technique presented in Subsection 3.2, we can construct a NURBS curve which is a high order approximation to the curved boundary of domain; if the curved boundary of domain can be exactly represented by a NURBS curve, then we go to the next step;

(3) finally, we replace the straight edges locating on the curved boundary of domain by the related curved edges locating on the NURBS curve.

A set of curved triangles will be obtained when the above three steps have been done. As in [40], a curved triangle has one curved edge and two straight edges. In Fig. 2 (the right one), based on the unstructured mesh generated by the EasyMesh, we show the high order unstructured mesh obtained by using a quadratic NURBS curve to exactly represent the unit circle.

To uniformly/locally refine a high order unstructured mesh, the mid-points of all edges of this mesh are needed. For a curved edge locating on a NURBS curve, the technique presented in Remark 3.1 is used to compute its mid-point.

3.4 Numerical integration over curved geometries

For the cell-centered finite volume method, it is necessary to find the numerical quadrature information over edges and cells. The quadrature information on the boundary of a control volume is needed to compute the inviscid flux, and it includes the quadrature points, the quadrature weights, and the associated unit outward normal vectors on the boundary of the control volume. Moreover, the numerical quadrature information over a control volume T_i is needed to compute the area of T_i and the smoothness indicators used in the WENO reconstruction over T_i. The techniques presented in [35, 39, 40] are used to obtain the information of numerical quadrature over the curved edges/triangles. Here we present a brief description of these techniques for the sake of completeness.

Let us assume that the boundary $\partial\Omega$, or a part of it, has been approximately or exactly represented by NURBS curve(s), and suppose that T_e is a curved triangle containing a curved edge Γ_e, where Γ_e is on the NURBS curve $\mathbf{C}(\xi)=(x(\xi),y(\xi))^T$ with $\xi\in[\bar{\xi}_1^e,\bar{\xi}_2^e]$, that is, $\Gamma_e=\mathbf{C}([\bar{\xi}_1^e,\bar{\xi}_2^e])$. Moreover, let $\mathbf{A}_1^e=\mathbf{C}(\bar{\xi}_1^e)$ and $\mathbf{A}_2^e=\mathbf{C}(\bar{\xi}_2^e)$ be the two end points of Γ_e.

We first consider the line integral $\int_{\Gamma_e} f(x,y)ds$. Since

$$\int_{\Gamma_e} f(x,y)ds=\int_{\bar{\xi}_1^e}^{\bar{\xi}_2^e} f(x(\xi),y(\xi))\sqrt{x'(\xi)^2+y'(\xi)^2}d\xi, \tag{3.10}$$

the one-dimensional Gauss-Legendre quadrature formula can be used to approximate the line integral, i.e.,

$$\int_{\Gamma_e} f(x,y)ds\approx\sum_{i=1}^{n_{gl}}\omega_i f(\mathbf{C}(\xi_i^e))\sqrt{x'(\xi_i^e)^2+y'(\xi_i^e)^2}, \tag{3.11}$$

where n_{gl} is the number of Gauss quadrature points, ξ_i^e and ω_i are the corresponding quadrature points and weights in the interval $[\bar{\xi}_1^e,\bar{\xi}_2^e]$, respectively. Note that $n_{gl}=p+1$ is used in the computation, where p is the degree of NURBS basis functions.

We next consider the cell integral $\int_{T_e} f(x,y)dxdy$. A mapping [35, 39] which maps the rectangle $R=[\bar{\xi}_1^e,\bar{\xi}_2^e]\times[0,1]$ to the curved triangle T_e is used to compute this integral. The mapping is defined by

$$\boldsymbol{\varphi}:(\xi,\lambda)\in R\to(x,y)\in T_e,\quad \boldsymbol{\varphi}(\xi,\lambda)=(1-\lambda)\mathbf{C}(\xi)+\lambda\mathbf{A}_3^e, \tag{3.12}$$

where $\mathbf{A}_3^e=(x_3^e,y_3^e)^T\in T_e$ is the opposite vertex with respect to the curved edge Γ_e. Then the cell integral $\int_{T_e} f(x,y)dxdy$ can be approximated by

$$\int_{T_e} f(x,y)dxdy=\int_R f(\boldsymbol{\varphi}(\xi,\lambda))|J_{\boldsymbol{\varphi}}|d\xi d\lambda\approx\sum_{i=1}^{n_{gl}}\sum_{j=1}^{m_{gl}}\omega_i\tilde{\omega}_j f(\boldsymbol{\varphi}(\xi_i^e,\lambda_j))|J_{\boldsymbol{\varphi}}(\xi_i^e,\lambda_j)|, \tag{3.13}$$

where $J_{\boldsymbol{\varphi}}(\xi,\lambda)=\partial(x,y)/\partial(\xi,\lambda)=(1-\lambda)\left[x'(\xi)\left(y_e^e-y(\xi)\right)-y'(\xi)\left(x_3^e-x(\xi)\right)\right]$ is the Jacobian determinant of the transformation $\boldsymbol{\varphi}(\xi,\lambda)$, and n_{gl} and m_{gl} are the number of Gauss quadrature points in $[\bar{\xi}_1^e,\bar{\xi}_2^e]$ and $[0,1]$, respectively, and in the simulations, $n_{gl}=p+1$ and $m_{gl}=2$ are used. Moreover, ξ_i^e and ω_i are the quadrature points and weights in $[\bar{\xi}_1^e,\bar{\xi}_2^e]$, and λ_j and $\tilde{\omega}_j$ are the quadrature points and weights in $[0,1]$.

3.5 Point inversion algorithm

When a traditional finite volume solver is available, the main difficulty in the implementation of the NURBS-enhanced finite volume method is how to efficiently include the information of a NURBS curve into the traditional numerical solver. This fact has also been mentioned in [40], where the NURBS-enhanced finite element method (NEFEM) is proposed.

Specifically speaking, for a curved edge $\Gamma_e=\mathbf{C}([\bar{\xi}_1^e,\bar{\xi}_2^e])$ with $\mathbf{A}_1^e=\mathbf{C}(\bar{\xi}_1^e)$ and $\mathbf{A}_2^e=\mathbf{C}(\bar{\xi}_2^e)$, the coordinates of $\mathbf{A}_1^e$ and $\mathbf{A}_2^e$ are known in the traditional numerical solvers, but the parameter values $\bar{\xi}_1^e$ and $\bar{\xi}_2^e$ are not available in those solvers. Hence, in order to efficiently implement the NURBS-enhanced finite volume solver, especially on an h-adaptive mesh, we use a simplified point inversion technique to compute the parameter value ξ of a given point $\mathbf{Q}$, where $\mathbf{Q}=(x_0,y_0)^T$ is a point lying on the NURBS curve $\mathbf{C}(\xi)$.

The most straightforward point inversion algorithm is to numerically solve the non-linear equation

$$f_1(\xi):=\|\mathbf{C}(\xi)-\mathbf{Q}\|_2^2:=\big(x(\xi)-x_0\big)^2+\big(y(\xi)-y_0\big)^2=0, \tag{3.14}$$

by the Newton iteration method with an appropriate initial guess $\xi_0\in[0,1]$ and a prescribed tolerance ϵ, where $\epsilon=10^{-15}$ is used in this paper.

The classical point inversion technique proposed in [36] is to numerically solve the following non-linear equation

$$f_2(\xi):=\mathbf{C}'(\xi)\cdot\big(\mathbf{C}(\xi)-\mathbf{Q}\big)=0, \tag{3.15}$$

by the Newton iteration method. It can be seen from Eq. (3.15) that the Newton method using $f_2(\xi)$ requires the second-order derivative of the NURBS curve. Therefore, the Newton method based on $f_2(\xi)$ is not applicable to the C^1-continuous NURBS curves. On the other hand, since the Newton method is sensitive to the initial guess, additional work is required to find an appropriate initial guess, see [36].

In this paper, we design a simplified point inversion algorithm which is not sensitive to the initial guess. To be specific, we design a function that is simpler than $f_1(\xi)$ and $f_2(\xi)$, and can be applicable to the NURBS curves of C^1-continuity. The simplified version is described as follows.

We divide the points lying on the curved boundary of physical domain into several parts such that the x- or y-coordinates of points in each part are monotone, and in each part, we construct a NURBS curve with the related points. In this way, the $x(\xi)$ or $y(\xi)$ of each NURBS curve $\widetilde{\boldsymbol{C}}(\xi)=(x(\xi),y(\xi))^T$ would be a monotone function. If $x(\xi)$ is a monotone function, then the parameter value ξ can be obtained by solving

$$f_3(\xi):=x(\xi)-x_0=0, \tag{3.16}$$

with the Newton method. The similar treatment can be used when $y(\xi)$ is a monotone function. Compared to the functions $f_1(\xi)$ and $f_2(\xi)$, $f_3(\xi)$ is much simpler, and the associated Newton method does not require the second-order derivative of the NURBS curve,

which implies that it is applicable to C^1-continuous NURBS curves. It is worthwhile to note that the most important aspect of the simplified version is that the Newton iteration is not sensitive to the initial guess ξ_0, and it is unnecessary to find initial guesses for different points.

Remark 3.1. Let Γ_e be a curved edge locating on a NURBS curve $\mathbf{C}(\xi)=(x(\xi),y(\xi))^T$ of degree p, and assume that the curvature of Γ_e is not too large. Then the mid-point of Γ_e can be efficiently obtained by the Newton method when the parameter values of the two end points of Γ_e are available. Specifically, we first compute the length of the curved edge Γ_e by evaluating the line integral

$$l_e=\int_{\Gamma_e} ds=\int_{\bar{\xi}_1^e}^{\bar{\xi}_2^e}\sqrt{x'(\xi)^2+y'(\xi)^2}d\xi,$$

where $\bar{\xi}_1^e$ and $\bar{\xi}_2^e$ are the related parameter values of the two end points of Γ_e, and $\bar{\xi}_1^e<\bar{\xi}_2^e$. Note that the Gauss-Legendre numerical quadrature formula of $p+1$ points is used to compute the line integral. We next use the Newton method to find $\bar{\xi}_{mid}^e\in(\bar{\xi}_1^e,\bar{\xi}_2^e)$ such that

$$\int_{\bar{\xi}_1^e}^{\bar{\xi}_{mid}^e}\sqrt{x'(\xi)^2+y'(\xi)^2}d\xi=l_e/2,$$

then the mid-point of Γ_e is $\mathbf{P}_{mid}^e=\mathbf{C}(\bar{\xi}_{mid}^e)$. With the knowledge of mid-point and parameter values of the two end points of every curved edge, the NURBS-enhanced finite volume solver with uniform or h-adaptive mesh refinement can be easily implemented when the traditional finite volume solver is available. The proposed strategy can be easily extended to other integral form based high order methods using NURBS to describe the curved boundary, and the modification of the extension is slight.

Remark 3.2. If the curved boundary of domain is symmetric about the x- or y-axis, and the x- or y-coordinates of the points of each part are monotone, there is no need to divide the points lying on the curved boundary into two parts. The initial guess of the Newton iteration method is $\xi_0=0.25$ or $\xi_0=0.75$, and the specific initial guess depends on which part of the boundary the point $\mathbf{Q}$ belongs to. In such a situation, if we use one NURBS curve to represent the closed boundary, i.e., $\mathbf{C}(0)=\mathbf{C}(1)=\mathbf{P}_1=\mathbf{P}_n$, the parameter value for the point $\mathbf{Q}=\mathbf{P}_1=\mathbf{P}_n$ is $\xi=0$ or $\xi=1$, and its exact value depends on which the curved edge $\mathbf{Q}$ belongs to.

To demonstrate the effectiveness and advantage of the simplified point inversion algorithm, we take the upper surface of NACA0012 airfoil as an example, see Fig. 3 (the black solid curve). The mathematical expression of the upper surface of NACA0012 airfoil is

$$y=0.6(0.2969\sqrt{x}-0.1260x-0.3516x^2+0.2843x^3-0.1036x^4),$$

where $x\in[0,1]$.

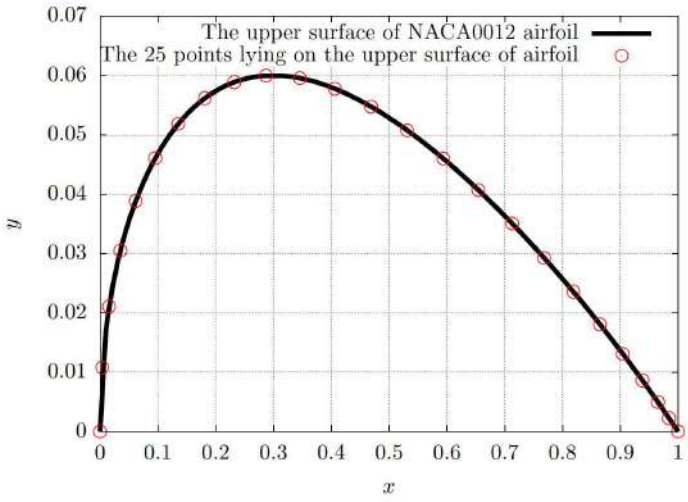

Figure 3: The upper surface of NACA0012 airfoil, and the distribution of points on it.

There are 25 points on the upper surface of NACA0012 airfoil, see Fig. 3 (the small red circles) for those points. Let $\{\mathbf{Q}_k := (x_k, y_k)^T\}_{k=0}^{24}$ be the set containing those points, where $0 = x_0 < x_1 < \cdots < x_{24} = 1$. Note that the x-coordinates of those points are monotonically increasing. We can construct a cubic NURBS curve by using those points and the curve fitting technique presented in Subsection 3.2.

Although the parameter values $\{\bar{\xi}_i\}_{i=0}^{24}$ have been achieved in the process of curve fitting, they are not available in a traditional finite volume solver. We use the point inversion technique to compute them. It follows from $\mathbf{C}(0) = \mathbf{Q}_0$ and $\mathbf{C}(1) = \mathbf{Q}_{24}$ that $\bar{\xi}_0 = 0$ and $\bar{\xi}_{24} = 1$. Therefore, the Newton method is needed to compute the parameter values for the points $\mathbf{Q}_k$ with $1 \le k \le 23$. The initial guess of the Newton method is $\xi_0 = 0.5$ for all points, and the software and hardware used for the simulation can be found in Section 5.

In Fig. 4 (left one), we show the number of Newton iteration steps needed to compute the parameter value for $\mathbf{Q}_k$ with $1 \le k \le 23$. As a comparison, we also present the results obtained by using the Newton method based on the functions $f_1(\xi)$ and $f_2(\xi)$ in the figure. It can be observed from the figure that the number of iterations required by the Newton method based on $f_3(\xi)$ is comparable with that needed by the Newton method with $f_2(\xi)$ for most of the points, and is much fewer than that needed by the Newton method with $f_1(\xi)$.

In Fig. 4 (middle one), we show the l^1-norm errors, i.e., $\|\mathbf{C}(\tilde{\xi}_k) - \mathbf{Q}_k\|_1$ with $1 \le k \le 23$, where $\tilde{\xi}_k$ denotes the parameter value obtained by using the point inversion technique for $\mathbf{Q}_k$. We can see from this figure that the error can be effectively reduced to around 10^{-16} with less than or equal to 6 Newton iteration steps by using the simplified function $f_3(\xi)$ with a fixed initial guess $\xi_0 = 0.5$.

The total CPU time needed to compute those parameter values using the Newton method with the functions $f_1(\xi)$, $f_2(\xi)$ and $f_3(\xi)$ are about 3.46×10^{-4} s, 3.80×10^{-5} s,

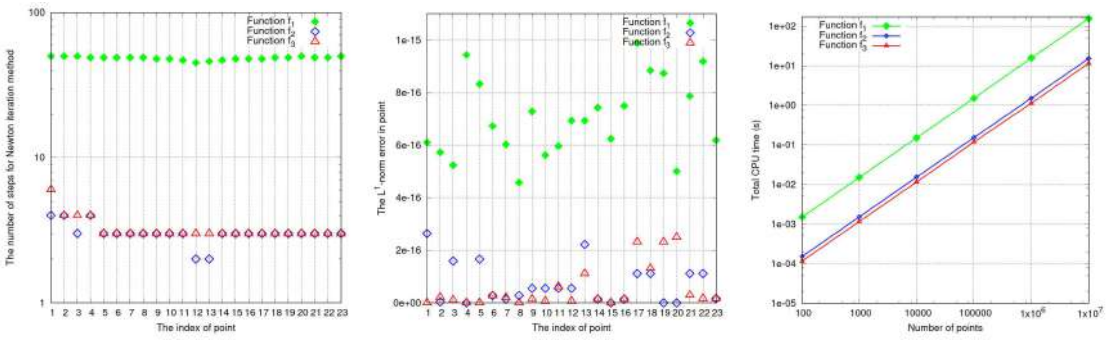

Figure 4: The results obtained by using three different functions to implement the point inversion algorithm. Left: the number of iteration steps of the Newton method for the 23 points lying on the upper surface of airfoil. Middle: the l^1-norm error between the point $\mathbf{Q}_k$ and $\mathbf{C}(\tilde{\xi}_k)$, where $1 \le k \le 23$. Right: the CPU time needed to implement the point inversion technique for the points on the NURBS curve.

and 2.82×10^{-5} s, respectively, which demonstrates that the simplified one is the most efficient one in terms of the CPU time.

Finally, to further confirm that the simplified point inversion algorithm is efficient, in Fig. 4 (right one), we show the total CPU time needed to implement the point inversion technique using the three functions to calculate the parameter values uniformly distributed in $[0.0001,0.9999]$. It can be seen from the figure that the total CPU time is increasing linearly as the increase of the number of points, and the simplified point inversion algorithm is the most efficient one. It is worthwhile to note that the total CPU time needed to perform the point inversion using the simplified function $f_3(\xi)$ for 10^5 points is about 0.1182 s, which shows that the CPU time needed by the simplified point inversion algorithm is negligible in practical simulations.

4 The goal-oriented *a posteriori* error estimation

Since the main concern in most engineering applications is to accurately compute some scalar functionals called quantities of interest or output functionals, e.g., the lift and drag coefficients in the external aerodynamics flow simulations, it is highly desirable to develop an efficient and accurate numerical method to evaluate the quantity of interest. In this paper, the goal-oriented *a posteriori* error estimation technique is used to drive the mesh adaptation to obtain an accurate approximation to the output functional. The goal-oriented error estimation used in this paper is based on the discrete adjoint method [12,17,42]. Here, we present a brief review of the goal-oriented error estimation, the interested reader is referred to [26] and the references therein for the details.

Let the quantity of interest be $\mathcal{J}(\boldsymbol{U})$, where $\boldsymbol{U}$ is the solution to the steady Euler equations. Let $\mathcal{T}_H$ denote a triangulation of the computational domain Ω, and let $\boldsymbol{U}_H$

denote the numerical solution to the steady Euler equations, which is obtained by solving

$$\boldsymbol{\mathcal{R}}_H(\boldsymbol{U}_H)=\boldsymbol{0}, \tag{4.1}$$

where the NURBS-enhanced high order finite volume method developed in [35] is used. Then the corresponding quantity of interest evaluated on $\mathcal{T}_H$ is $\mathcal{J}_H(\boldsymbol{U}_H)$. Let $\mathcal{T}_h$ be a mesh obtained by uniformly refining $\mathcal{T}_H$, and we assume that $\boldsymbol{U}_h$ is the numerical approximation to $\boldsymbol{U}$ on $\mathcal{T}_h$, i.e., $\boldsymbol{U}_h$ is the solution to $\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h)=\boldsymbol{0}$. Hence the related quantity of interest evaluated on $\mathcal{T}_h$ is $\mathcal{J}_h(\boldsymbol{U}_h)$.

Now let us consider an approximation to $\mathcal{J}_h(\boldsymbol{U}_h)$

$$\mathcal{J}_h(\boldsymbol{U}_h)\approx\mathcal{J}_h(\boldsymbol{U}_h^H)+\frac{\partial\mathcal{J}_h}{\partial\boldsymbol{U}_h}\Big|_{\boldsymbol{U}_h^H}(\boldsymbol{U}_h-\boldsymbol{U}_h^H), \tag{4.2}$$

where $\boldsymbol{U}_h^H=\boldsymbol{I}_h^H\boldsymbol{U}_H$ is the solution on $\mathcal{T}_h$, which is obtained through the interpolation of the solution $\boldsymbol{U}_H$ on the coarser grid $\mathcal{T}_H$ to the finer grid $\mathcal{T}_h$, and $\boldsymbol{I}_h^H$ is the related interpolation operator, see [26] for the construction of $\boldsymbol{U}_h^H$.

On the other hand, for the discrete residual equations $\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h)=\boldsymbol{0}$, we have

$$\boldsymbol{0}=\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h)\approx\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h^H)+\frac{\partial\boldsymbol{\mathcal{R}}_h}{\partial\boldsymbol{U}_h}\Big|_{\boldsymbol{U}_h^H}(\boldsymbol{U}_h-\boldsymbol{U}_h^H). \tag{4.3}$$

It follows from the above expression that

$$\boldsymbol{U}_h-\boldsymbol{U}_h^H\approx-\Big(\frac{\partial\boldsymbol{\mathcal{R}}_h}{\partial\boldsymbol{U}_h}\Big|_{\boldsymbol{U}_h^H}\Big)^{-1}\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h^H). \tag{4.4}$$

Combining Eqs. (4.2) and (4.4), we can obtain an approximation to the error between $\mathcal{J}_h(\boldsymbol{U}_h)$ and $\mathcal{J}(\boldsymbol{U}_h^H)$

$$\mathcal{J}_h(\boldsymbol{U}_h)-\mathcal{J}_h(\boldsymbol{U}_h^H)\approx\boldsymbol{\psi}_h^T\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h^H), \tag{4.5}$$

where $\boldsymbol{\psi}_h=-\big(\frac{\partial\mathcal{J}_h}{\partial\boldsymbol{U}_h}\big|_{\boldsymbol{U}_h^H}\big(\frac{\partial\boldsymbol{\mathcal{R}}_h}{\partial\boldsymbol{U}_h}\big|_{\boldsymbol{U}_h^H}\big)^{-1}\big)^T$ is the so-called adjoint solution, which is obtained by solving the discrete adjoint equations

$$\Big(\frac{\partial\boldsymbol{\mathcal{R}}_h}{\partial\boldsymbol{U}_h}\Big|_{\boldsymbol{U}_h^H}\Big)^T\boldsymbol{\psi}_h+\Big(\frac{\partial\mathcal{J}_h}{\partial\boldsymbol{U}_h}\Big|_{\boldsymbol{U}_h^H}\Big)^T=\boldsymbol{0}. \tag{4.6}$$

It can be observed from the above equation that the adjoint solution $\boldsymbol{\psi}_h$ is computed on the finer grid $\mathcal{T}_h$, which is prohibitively expensive. To resolve this problem, the adjoint solution $\boldsymbol{\psi}_h$ can be approximated by following [6,26] as

$$\boldsymbol{\psi}_h\approx\boldsymbol{\psi}_h^H=\boldsymbol{I}_h^H\boldsymbol{\psi}_H, \tag{4.7}$$

where $\boldsymbol{\psi}_h^H$ is the solution on the finer mesh $\mathcal{T}_h$ obtained through the interpolation of $\boldsymbol{\psi}_H$ on $\mathcal{T}_H$ to $\mathcal{T}_h$, and $\boldsymbol{\psi}_H$ is the adjoint solution on the coarser grid $\mathcal{T}_H$, and can be achieved by solving the discrete adjoint equation on $\mathcal{T}_H$

$$\Big(\frac{\partial\boldsymbol{\mathcal{R}}_H}{\partial\boldsymbol{U}_H}\Big|_{\boldsymbol{U}_H}\Big)^T\boldsymbol{\psi}_H+\Big(\frac{\partial\mathcal{J}_H}{\partial\boldsymbol{U}_H}\Big|_{\boldsymbol{U}_H}\Big)^T=\boldsymbol{0}. \tag{4.8}$$

Therefore, by (4.5) and (4.7), the error between $\mathcal{J}_h(\boldsymbol{U}_h)$ and $\mathcal{J}_h(\boldsymbol{U}_h^H)$ can be approximated by

$$\mathcal{J}_h(\boldsymbol{U}_h)-\mathcal{J}_h(\boldsymbol{U}_h^H)\approx(\boldsymbol{\psi}_h^H)^T\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h^H). \tag{4.9}$$

Now the error indicator for each cell of the coarser mesh $\mathcal{T}_H$ can be derived from the computable quantity given by the right hand side of (4.9). Since the finer mesh $\mathcal{T}_h$ is obtained by uniformly refining the coarser mesh $\mathcal{T}_H$, and the unstructured mesh consisting of triangles is used in this paper, the cell $T_i\in\mathcal{T}_H$ would contain four sub-cells belonging to $\mathcal{T}_h$, that is, $T_i=\bigcup_{j=1}^4 T_{i,j}$, where $T_{i,j}\in\mathcal{T}_h$, and $1\le j\le 4$. Therefore, the error indicator for the i-th cell $T_i\in\mathcal{T}_H$ is

$$\eta_i^H=\sum_{j=1}^{4}\left|(\boldsymbol{\psi}_h^H)^T|_{T_{i,j}}\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h^H)|_{T_{i,j}}\right|, \tag{4.10}$$

where $(\boldsymbol{\psi}_h^H)^T|_{T_{i,j}}$ and $\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h^H)|_{T_{i,j}}$ denote the elements of the vectors $(\boldsymbol{\psi}_h^H)^T$ and $\boldsymbol{\mathcal{R}}_h(\boldsymbol{U}_h^H)$ related to $T_{i,j}\in\mathcal{T}_h$, respectively.

For a cell $T_i\in\mathcal{T}_H$ and a given adaptation tolerance $TOL>0$, if the error indicator is much larger than the given TOL, then the cell T_i will be refined into four sub-cells. On the other hand, if the indicators in the four sub-cells are small enough, the four sub-cells will be coarsened into a larger cell, we refer to [33] for the details. The Hierarchy Geometry Tree (HGT) [33] data structure is then used to efficiently manage the locally refined meshes obtained by the adjoint-based h-adaptive method [6, 21], we refer the interested reader to [26] and the references therein for the details.

5 Numerical results

In this section, we present several numerical examples to illustrate the effectiveness and robustness of the proposed method. The density and velocity of the free stream are set as $\rho_\infty=1$ and $v_\infty=(u_\infty,v_\infty)^T=(\cos\alpha,\sin\alpha)^T$, respectively, where α is the angle of attack. The numerical examples presented in Subsections 5.1 and 5.2 are used to demonstrate the effectiveness and advantage of using the NURBS-enhanced high order finite volume method with goal-oriented h-adaptivity to compute the quantity of interest, and the test cases presented in Subsection 5.3 are used to further show the robustness of the proposed method.

In the following, the free-stream flow conditions are used to set the initial guess of the Newton iteration method on both the initial and the h-adaptive meshes, and the computations were performed until the l^1-norm of density residual reaches the stop tolerance 10^{-10}. All of the tests are ran sequentially on a C++ package called AFVM4CFD which is still under development. The hardware is a Dell Precision 5530 Mobile Workstation with Intel (R) Xeon (R) E-2176M CPU @ 2.70 GHz and 32 Gb memory.

5.1 Inviscid subsonic flow through a channel with a bump

We first consider an inviscid subsonic flow through a channel with a bump containing a semicircle at a free-stream Mach number $M_\infty = 0.3$ and an attack angle $\alpha = 0^\circ$. Since the semicircle can be exactly represented by a quadratic NURBS curve [36], the purpose of this example is not only to demonstrate the effectiveness of the goal-oriented mesh adaptation method but also to show the advantage of using NURBS to represent the curved wall boundary. Note that this numerical example is taken from [15, 39] and is a modified version of the test case presented in [27, 47].

The configuration of this test case is described as follows. The physical domain is $\Omega = \{(x,y)|-1 \le x \le 1, f(x) \le y \le 0.8\}$, where $f(x)$ represents a bump with a semicircle of radius $r = 0.25$, and is defined by

$$f(x) = \begin{cases} 0, & \text{if } -1 \le x < -r, \\ \sqrt{r^2 - x^2}, & \text{if } -r \le x \le r, \\ 0, & \text{if } r < x \le 1. \end{cases} \tag{5.1}$$

The subsonic inflow boundary and outflow boundary are imposed at the left and right boundaries, respectively, and the bottom and top boundaries are the wall boundary. The far field flow features are used to set the initial guess for the Newton method on both the initial and h-adaptive meshes, and the initial guess for the implementation of the point inversion technique is $\xi_0 = 0.5$. Fig. 6 (top left) shows a regular triangulation of Ω, and this mesh consists of 641 cells.

Since the flow is isentropic, the entropy s is a constant in the flow field, which means that the quantity $S = p/\rho^\gamma$ is also a constant. Hence, the $L^2(\Omega)$-norm of the entropy error evaluated at the steady state is used to evaluate the accuracy of the numerical scheme, and the entropy error is defined by

$$\epsilon_{ent} = \frac{S - S_\infty}{S_\infty}, \tag{5.2}$$

where $S_\infty = p_\infty/\rho_\infty^\gamma$, and p_∞ and ρ_∞ are the far field pressure and density, respectively.

Furthermore, in this test case, the quantity of interest is related to the entropy production

$$\mathcal{J}(U) = \frac{1}{|\Omega|}\int_\Omega \epsilon_{ent}^2 dxdy. \tag{5.3}$$

We first show the distribution of L^2-norm of entropy error in each cell of initial mesh in Fig. 5, and the error is obtained by the NURBS-enhanced third-order finite volume solver. It can be observed from the figure that the entropy errors distributed near the semicircle and the downstream of bottom boundary are larger than those in other regions.

The distributions of Mach number isolines obtained by the proposed NURBS-enhanced goal-oriented h-adaptive method on the initial mesh and the locally refined meshes are presented in Fig. 6 (right column), where the plotted values for isolines of

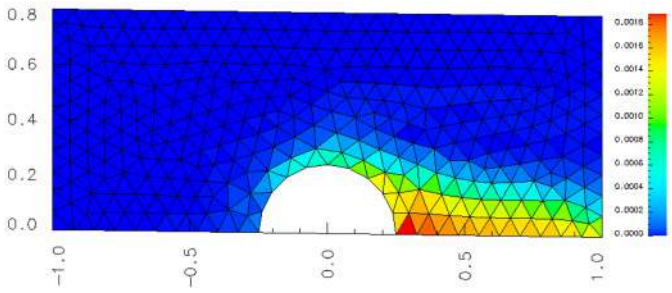

Figure 5: The distribution of L^2-norm of entropy error in each cell of the initial mesh evaluated at the steady state.

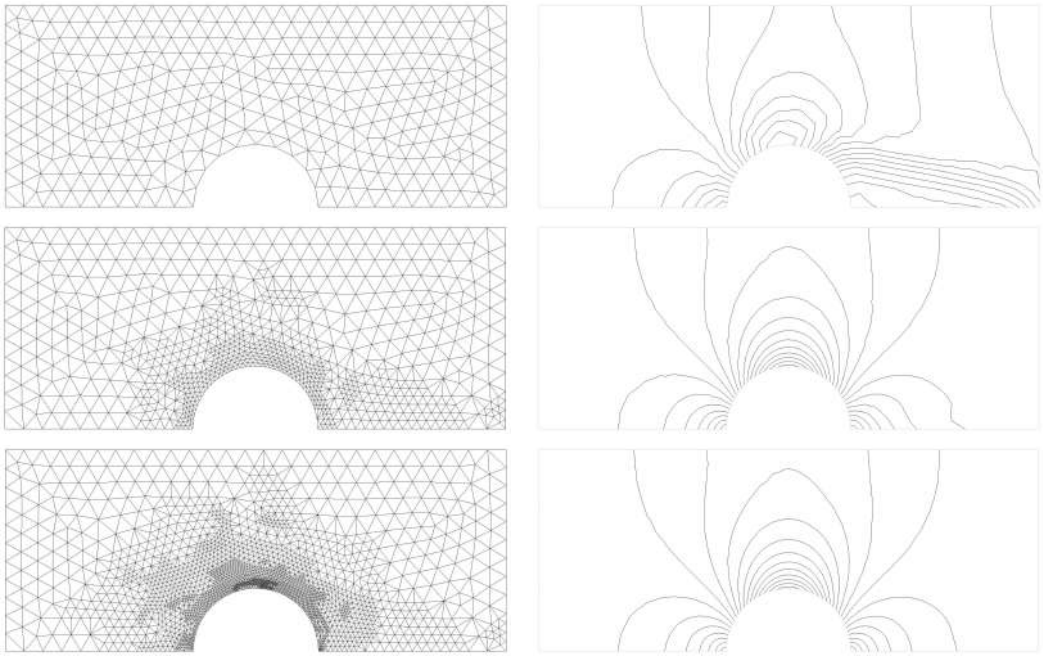

Figure 6: Left column: a sequence of adaptively refined meshes (from top to bottom) obtained by the NURBS-enhanced finite volume method with the goal-oriented h-adaptivity. Right column: the distribution of Mach number isolines on the related adapted meshes, where $M_i = i\Delta M$ with $\Delta M = 0.04$, $i = 0,1,2,\cdots$.

Mach number are given by $M_i = i\Delta M$, where $\Delta M = 0.04$, and $i = 0,1,\cdots$. Moreover, we show the locally refined meshes in Fig. 6 (left column). It can be observed from the figure that as the initial mesh is adaptively refined two times, the distribution of the Mach number isolines is smooth and symmetric about the y-axis, which implies the convergence of our scheme. On the other hand, it can be observed from the adaptively refined meshes that the regions near the semicircle and the downstream of bottom boundary exhibit the major part of the local refinement, which shows the effectiveness of the goal-oriented h-adaptive method.

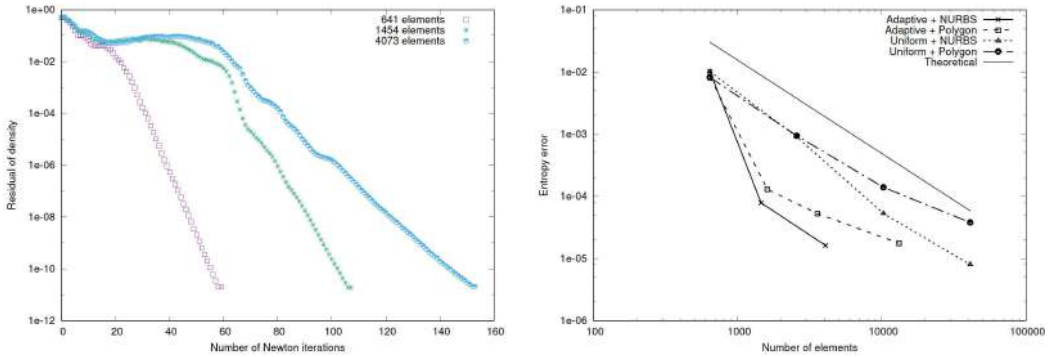

Figure 7: Left: The convergence history of l^1-norm of density residual against the steps of Newton iteration on three adaptively refined meshes. Right: The convergence histories of $L^2(\Omega)$-norm of entropy errors with respect to the number of cells.

The related convergence history of l^1-norm of the density residual against the Newton iteration steps on those adaptively refined meshes is shown in Fig. 7 (left one), and it can be observed from the figure that the residual can be reduced to the stop tolerance on all meshes.

Finally, we show the convergence history of the $L^2(\Omega)$-norm of entropy error in Fig. 7 (right one). As a comparison, we also present the related results computed on polygonal meshes. Note that the polygonal meshes are obtained by uniformly refining the initial mesh in a body-fitted manner. The following two facts can be observed from the figure: (1). the optimal convergence rate can not be achieved when the curved wall boundary is approximated by the polygon, even the goal-oriented mesh adaptation scheme has been used; on the contrary, the optimal convergence rate can be achieved when the NURBS are used; and (2). the goal-oriented mesh adaptation scheme needs much fewer cells than those needed by the global refinement counterpart to achieve the same error level. These numerical results show the great advantage of using the goal-oriented h-adaptive mesh refinement method in combination with NURBS to describe the curved boundary when the high order finite volume methods are used.

5.2 Compressible inviscid flow around the NACA0012 airfoil

In this subsection, we consider the compressible inviscid flow around a NACA0012 airfoil with the following three different free-stream conditions

- Subsonic flow with Mach number M_∞=0.5 and an attack angle $\alpha = 2^\circ$. The distinct feature of this flow is the existence of a geometrical singularity at the sharp trailing edge, and this test case is used to demonstrate the advantage of using goal-oriented mesh adaptation for a problem containing a geometrical singularity.
- Transonic flow with Mach number M_∞=0.8 and an attack angle $\alpha = 1.25^\circ$. In the flow field, there exist a strong shock on the upper surface of airfoil and a weak

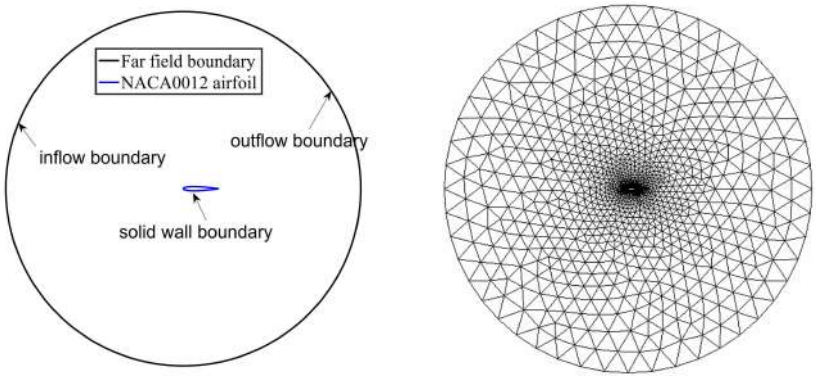

Figure 8: The physical domain (left), and the initial mesh (right) for the simulation of inviscid flow around the NACA0012 airfoil.

shock on the lower surface. The purpose of this test case is to demonstrate the advantage of using the goal-oriented mesh adaptation method with a combination of the NURBS-enhanced high order finite volume method for problems containing discontinuities.

- Supersonic flow with Mach number M_∞=1.5 and an attack angle $\alpha=0^\circ$. In the flow field, there is a strong detached bow shock in the front of airfoil [5].

The analytical expression for the NACA0012 airfoil [47] is

$$y=\pm 0.6c(0.2969\sqrt{x}-0.1260x-0.3516x^2+0.2843x^3-0.1036x^4), \tag{5.4}$$

where $x\in[0,1]$, and $c=1$ is the chord length of the airfoil.

Throughout this subsection, the circular far field boundary is adopted for all of the simulations, and the location of the far field boundary is $100c$ away from the airfoil, we refer to Fig. 8 (left) for the physical domain and the configuration of boundary conditions.

The initial mesh, see Fig. 8 (right), includes 3487 cells, 5283 edges, and 1796 vertexes. Furthermore, there are 48 mesh points on the surface of airfoil, see Fig. 9 (the red squares) for the distribution of those points. As the airfoil can not be exactly represented by a NURBS curve, the curve fitting technique discussed in Section 3 is needed to construct a cubic NURBS curve. The NACA0012 airfoil and the mesh points on it are symmetric about the x-axis, so is the NURBS curve. The initial guesses for the implementation of point inversion algorithm for the points lying on the lower and the upper parts of the NURBS curve are 0.25 and 0.75, respectively.

In the following, the distribution of pressure coefficient along the surface of airfoil is used to evaluate the performance of the proposed numerical method, and it is defined by

$$C_p:=\frac{p-p_\infty}{0.5\rho_\infty\|\boldsymbol{v}_\infty\|_2^2}, \tag{5.5}$$

where p_∞ and $\boldsymbol{v}_\infty$ are the far field pressure and velocity, respectively.

The quantity of interest for the simulation of flow around a airfoil is the drag coefficient along the airfoil, which is defined by

$$\mathcal{J}(U)=C_d:=\int_{\partial\Omega_a} p\boldsymbol{\beta}\cdot\boldsymbol{n}ds, \tag{5.6}$$

where $\partial\Omega_a$ is the surface of airfoil, $\boldsymbol{n}$ is the unit outward normal vector to $\partial\Omega_a$, and the vector $\boldsymbol{\beta}$ is given by

$$\boldsymbol{\beta}=(\cos\alpha,\sin\alpha)^T/C_\infty,$$

where $C_\infty=\frac{1}{2}\gamma p_\infty M_\infty^2 c$, M_∞ and c are the Mach number of the free stream and the chord length of airfoil, respectively, $\gamma=1.4$, and α is the angle of attack.

5.2.1 On geometrical error

As aforementioned, since the NACA0012 airfoil can not be exactly represented by a NURBS curve, there would exist the geometrical error when the airfoil is approximated by the cubic NURBS curve obtained by the curve fitting method using the 48 interpolation points. In this subsection, we investigate the effect of geometrical error on the quality of numerical solution.

Since the mathematical expressions of the airfoil and the cubic NURBS curve are available, we can compute the related geometrical error. We plot the point-wise geometrical error in Fig. 9 (right), and it can be observed from the figure that the geometrical error introduced by the cubic NURBS curve is much smaller than that introduced by the polygonal approximation when the same interpolation points are used. We can also observe from the figure that the geometrical errors around the leading edge are relatively larger than those in other regions. The reason for this phenomenon is that the curvature around the leading edge of NACA0012 airfoil is large due to the existence of $\sqrt{x}$ in the analytical expression of airfoil, see Eq. (5.4). To reduce the geometrical error, dense gird points should be placed in regions with high curvature, see [10]. The less accurate approximation to the regions around the leading edge would adversely affect the performance of numerical methods.

As the effect of geometrical error can be obviously observed from the numerical results on dense meshes, in Fig. 10 (left column) we show the distributions of pressure coefficient along the surface of airfoil computed on a mesh obtained by uniformly refining the initial mesh three times, where the mid-points of curved wall edges are located on the cubic NURBS curve with 48 interpolation points during each mesh refinement step. As a comparison, in Fig. 10 (middle column), we show the pressure coefficient computed on a polygonal mesh which is obtained by uniformly refining the initial mesh three times, where the mid-points of the polygonal edges lying on the airfoil are mapped to the true airfoil during each mesh refinement step, and now the number of mesh points lying on the surface of airfoil is 384. We can observe from the figure that (1) the results computed on the polygonal mesh with 384 points lying on the airfoil are more accurate than those obtained on the mesh using a NURBS curve with 48 interpolation points; in fact, the distributions of pressure coefficient are not smooth near the leading edge when the cubic

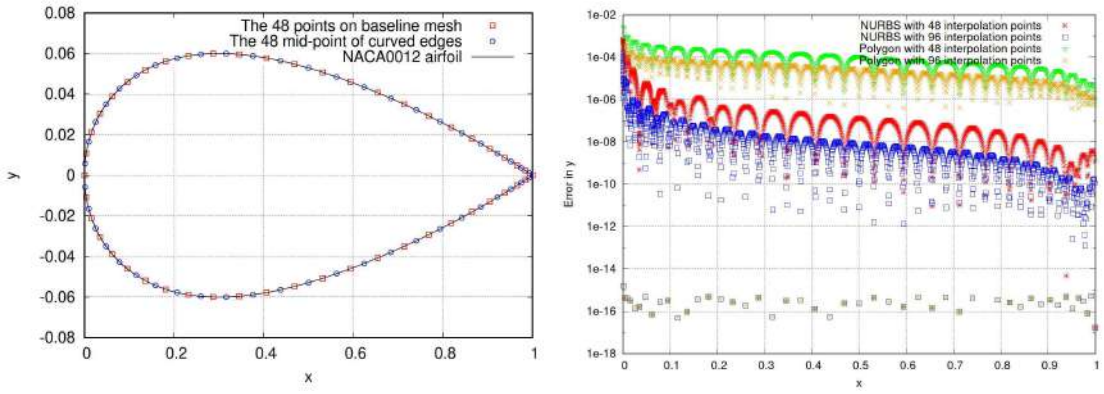

Figure 9: Left: the points lying on the NACA0012 airfoil. Right: the distribution of absolute errors in y-coordinates.

NURBS curve is used to approximate the airfoil, the reason for this phenomenon is that the geometrical error near the leading edge introduced by the polygonal approximation with 384 interpolation points is smaller than that introduced by the cubic NURBS curve with 48 interpolation points; and (2) for the transonic flow problem, no numerical oscillations is generated around the two shocks, which demonstrates that the non-oscillatory k-exact reconstruction method is capable of preventing numerical oscillations.

In order to achieve a more accurate approximation to the airfoil, we construct a new cubic NURBS curve using 96 interpolation points lying on the surface of airfoil, and those points are obtained by combining the mid-points (see Fig. 9 (left one, the blue circles)) of the original 48 curved edges locating on the NACA0012 airfoil with the original 48 interpolation points (see Fig. 9 (left one, the red squares)). The related geometrical errors introduced by the new NURBS curve are presented in Fig. 9 (right one). We present the related distributions of pressure coefficient for the three different flow configurations computed on a mesh obtained by uniformly refining the initial mesh three times in Fig. 10 (right column). It can be seen from the figure that the distributions of pressure coefficient are smooth around the leading edge for the three different free-stream configurations. Note that the results are indistinguishable from those obtained on the polygonal mesh with 384 points lying on the airfoil.

5.2.2 On efficiency of the proposed method

In this subsection, we numerically demonstrate that the NURBS-enhanced finite volume scheme with the goal-oriented h-adaptivity can efficiently reduce the error in quantity of interest. The mesh adaptation is driven to reduce the error in the drag coefficient along the airfoil.

We first show the results for the test case of subsonic flow. To compute the error in drag coefficient, a reference value for drag coefficient $C_{d,ref}$ =3.414415e-05 is used, and it is computed on a goal-oriented h-adaptive mesh with approximately 4.77×10^5 degrees of

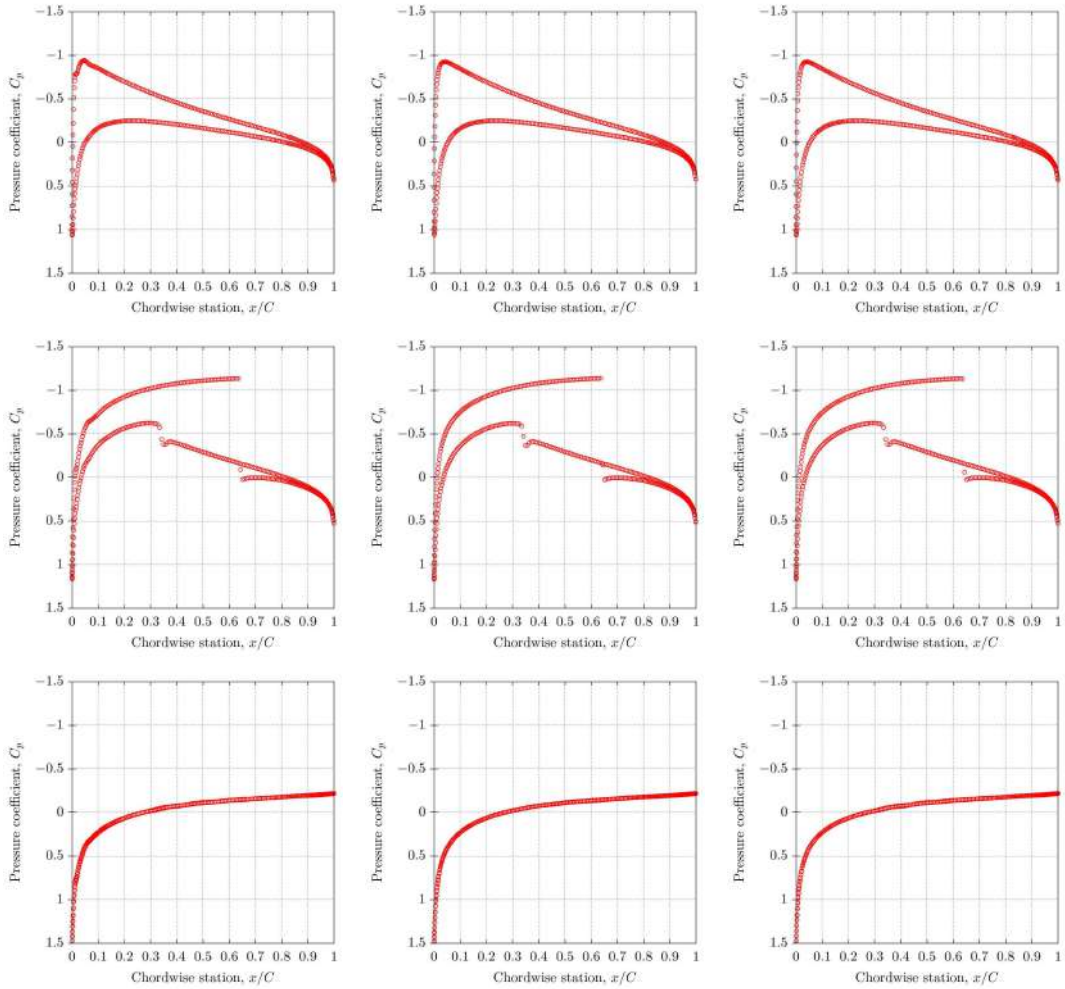

Figure 10: (NACA0012) The distribution of pressure coefficient along the surface of airfoil with different flow configurations computed on the mesh obtained by uniformly refined the initial mesh three times using three different approximations to the airfoil. Left column: NURBS curve with 48 interpolation points. Middle column: body-fitted polygonal approximation to airfoil, where 384 interpolation points are used. Right column: NURBS curve with 96 interpolation points. Top row: subsonic flow. Middle row: transonic flow. Bottom row: supersonic flow.

freedom, and the high order unstructured mesh is based on the cubic NURBS curve with 96 interpolation points lying on the airfoil. The convergence histories of the absolute error in drag coefficient are shown in Fig. 11 (top left). In the figure, both the *h*-adaptive meshes and uniformly refined meshes are considered. It can be observed from the figure that (1). due to the existence of a geometrical singularity at the sharp trailing edge, the optimal convergence rate can not be achieved when the uniformly refined meshes are used, which means that the benefit of high order numerical methods can not be fully utilized in such a situation [4, 42, 49]; and (2). compared with the uniform mesh refinement strategy, the

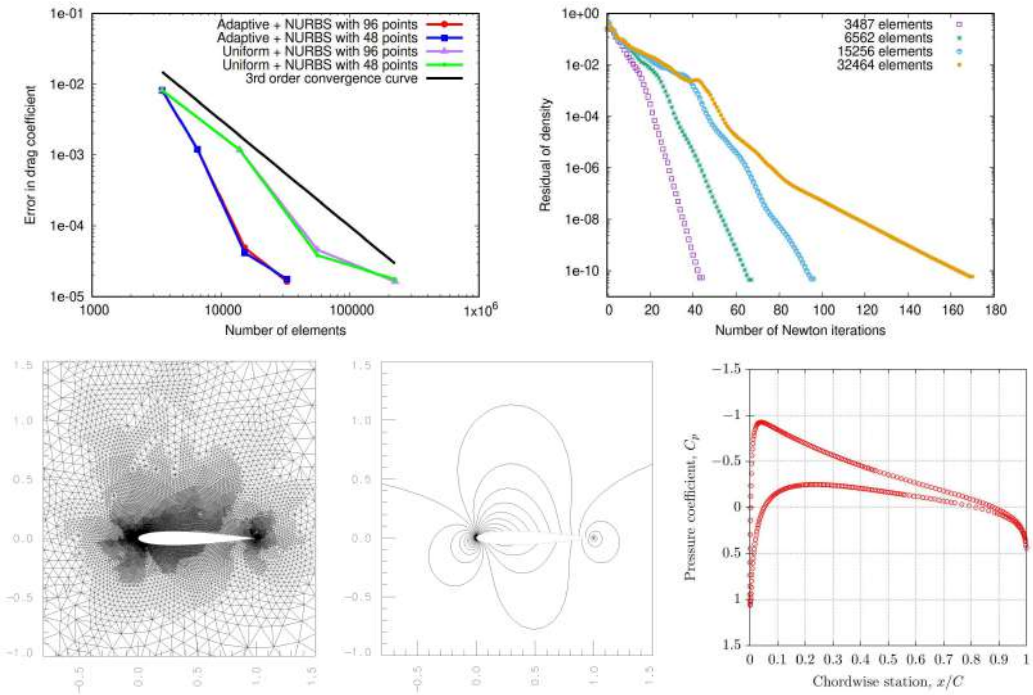

Figure 11: (NACA0012, M_∞=0.5, α=2°) Top left: convergence histories of the absolute error in drag coefficient, where two different approximations to the airfoil are considered. Top right: the convergence history of l^1-norm of density residual against the steps of Newton iteration. Bottom left: the final adapted mesh around the airfoil. Bottom middle: the distribution of Mach number isolines on the final adapted mesh. Bottom right: the distribution of pressure coefficient along the surface of airfoil.

goal-oriented mesh adaptive scheme needs much fewer cells to reach the same error level in drag coefficient, which shows that the goal-oriented mesh adaptation method is more efficient than the uniform mesh refinement scheme in computing the quantity of interest.

The numerical results presented in the last subsection show that the better results can be achieved when the approximation to the airfoil is more accurate, we therefore only present the related numerical results computed on high order unstructured meshes using the cubic NURBS curve with 96 interpolation points. We present the convergence history of l^1-norm of the density residual in Fig. 11 (top right), which shows that the convergence to steady state can be reached on all h-adaptive meshes. The final adapted mesh around the airfoil is shown in Fig. 11 (bottom left), the related distribution of Mach number isolines is presented in Fig. 11 (bottom middle), and the distribution of pressure coefficient along the surface of airfoil is displayed in Fig. 11 (bottom right). It can be observed from the figure of final adapted mesh that the regions around the trailing and leading edges and in the neighborhood of airfoil have been adaptively refined. On the other hand, the distributions of Mach number isolines and pressure coefficient are quite smooth, which demonstrate that the NURBS curve with 96 interpolation points is an accurate approximation to the airfoil.

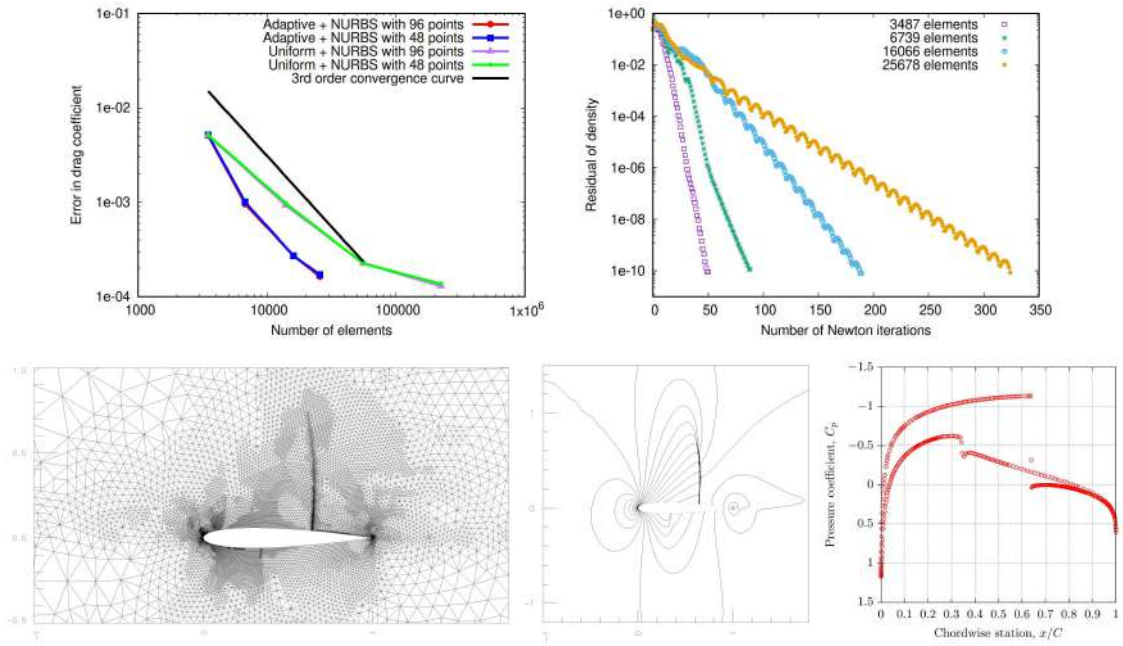

Figure 12: (NACA0012, $M_\infty = 0.8$, $\alpha = 1.25^\circ$) Top left: the convergence histories of the absolute error in drag coefficient computed on adaptively and uniformly refined meshes, where two different approximations to the airfoil are considered. Top right: the convergence history of l^1-norm of density residual computed on adaptively refined meshes. Bottom left: the final adapted mesh around the airfoil. Bottom middle: the distribution of Mach number isolines on the final adapted mesh. Bottom right: the distribution of pressure coefficient along the surface of airfoil.

We next show the numerical results for the test case of transonic flow. The reference drag coefficient for this test case is $C_{d,ref}$ =2.264595e-02, which is obtained from the h-adaptive mesh with 175020 degrees of freedom, where the NURBS curve with 96 interpolation points is used to construct the high order unstructured mesh. Note that the reference drag coefficient used in our simulation is between C_d=2.2628e-02 [49] and C_d=2.265319e-02 [5].

The convergence histories of the absolute error in drag coefficient are shown Fig. 12 (top left), where two different approximations to the airfoil are considered. It can be observed from the figure that (1). due to the existence of shock waves and the geometrical singularity, the optimal convergence rate can not be achieved when the uniform mesh refinement is used, similar numerical results can be observed in [4, 38, 49]; and (2). compared with the uniform mesh refinement, the goal-oriented mesh adaptation strategy needs much fewer cells to reach the same error level in drag coefficient, which demonstrates that the goal-oriented h-adaptive refinement is more efficient than the uniform mesh refinement in computing the drag coefficient.

The convergence history of the l^1-norm of density residual on a sequence of adaptively refined meshes is presented in Fig. 12 (top right), where the high order meshes are constructed from the cubic NURBS curve with 96 interpolation points. It can be ob-

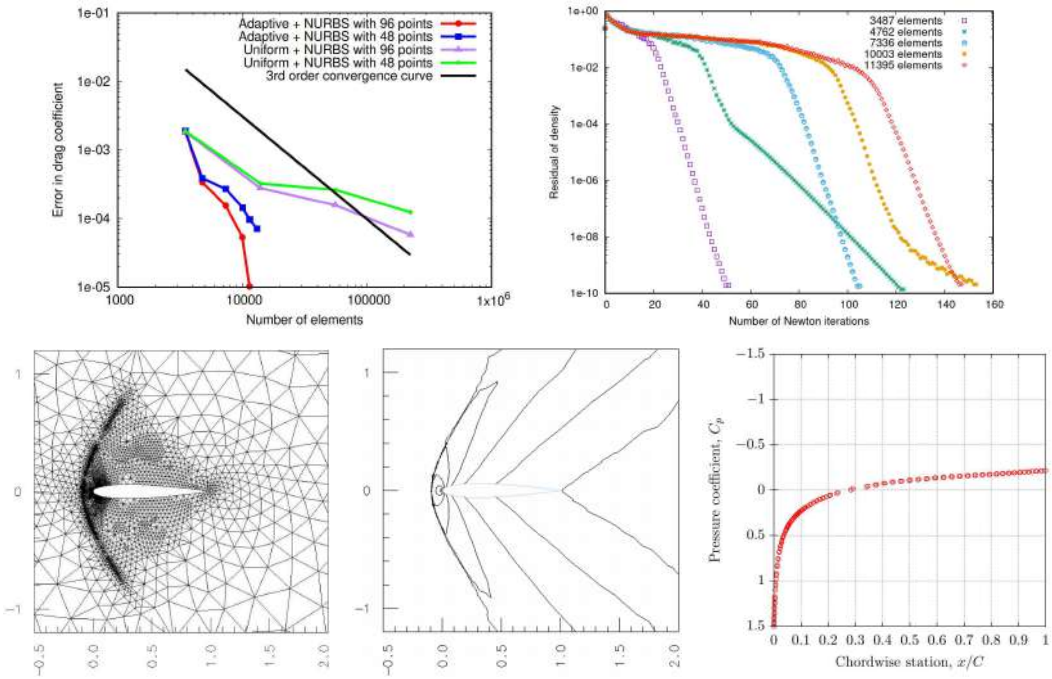

Figure 13: (NACA0012, $M_\infty=1.5$, $\alpha=0^\circ$) Top left: the convergence history of absolute error in drag coefficient. Top right: the convergence history of density residual against the steps of Newton iteration. Bottom left: the final adapted mesh around the airfoil. Bottom middle: the distribution of Mach number isolines on the final adapted mesh. Bottom right: the distribution of pressure coefficient along the surface of airfoil.

served from the figure that the residual can successfully reach the stop tolerance on all h-adaptive meshes.

In the following, we only show the related results based on the cubic NURBS curve using 96 interpolation points, and the related results are obtained on the adapted mesh used to compute the reference drag coefficient. The adapted mesh around the airfoil is presented in Fig. 12 (bottom left). It can be seen from the figure that the regions around the two shocks, near the leading and trailing edges, and in the neighborhood of airfoil have been adaptively refined, which demonstrates that the goal-oriented mesh adaptation method has effectively detected the regions that are critical to compute the drag coefficient. The distribution of Mach number isolines in $[-1.0,1.8]\times[-1.2,1.5]$ obtained on the final adapted mesh is shown in Fig. 12 (bottom middle), where the values for the Mach number of isolines are set as $M_i=i\Delta M$, $i=0,1,\cdots$, and $\Delta M=0.05$. The distribution of pressure coefficient along the surface of airfoil is shown in Fig. 12 (bottom right). It can be observed from those two figures that no numerical oscillations is generated around the shocks, which shows that the non-oscillatory 2-exact reconstruction method is capable of preventing numerical oscillations around the discontinuities.

Finally, we present the numerical results for the supersonic flow case. We first show the convergence histories of the absolute error in drag coefficient in Fig. 13 (top left),

where the reference drag coefficient is $C_{d,ref}$=9.629886e-02 [5]. From the figure, the same conclusions can be drawn as in the transonic flow case, i.e., due to the existence of the strong shock, the optimal convergence rate can not be achieved when the uniformly refined meshes are used. On the other hand, it can be observed from the figure that the goal-oriented h-adaptive mesh refinement scheme could significantly save the computational cost when comparing with the uniform mesh refinement.

The convergence history of the l^1-norm of density residual computed on several adaptively refined meshes is presented in Fig. 13 (top right), where the airfoil is approximated by a cubic NURBS curve with 96 interpolation points. It can be seen from the figure that the residual can successfully converge to the stop tolerance on those adapted meshes. The associated final adapted mesh around the airfoil is presented in Fig. 13 (bottom left), from which we can see that the regions near the strong shock, the leading edge and the surface of airfoil have been locally refined. The distribution of Mach number isolines in $[-0.5,2.0]\times[-1.2,1.2]$ obtained on the final adapted mesh is displayed in Fig. 13 (bottom middle), where the values for the Mach number of isolines are set as $M_i=i\Delta M$, $i=0,1,\cdots$, and $\Delta M=0.2$. Note that the Mach number contour agrees well with that reported in [5], and no numerical oscillations is generated around the strong shock. Furthermore, the distribution of pressure coefficient along the surface of airfoil is shown in Fig. 13 (bottom right). It can be observed from the figure that the distribution of pressure coefficient is smooth.

5.3 On robustness of the proposed method

In this subsection, two test cases with different geometrical configurations are simulated to further demonstrate the robustness of the proposed method. Specifically, the geometrical configurations based on the RAE 2822 airfoil and the two-body airfoils are considered.

We first consider the transonic inviscid flow with free-stream Mach number $M_\infty=0.75$ and an angle of attack $\alpha=1^\circ$ around the RAE 2822 airfoil. The location of far field boundary is $100c$ away from the airfoil, where $c=1$, and the initial mesh (see Fig. 14 (top left)) consists of 3618 cells, and there are 126 points on the airfoil, the interested reader is referred to [2] for the information of those points. In fact, 128 discrete points are given in [2], but the two points that are closest to the point $(1,0)^T$ are discarded to construct a high quality mesh by EasyMesh, and two cubic NURBS curves can be constructed by using those 126 points to approximate the lower and upper surfaces of the RAE2822 airfoil. The initial guess for the implementation of point inversion algorithm for each NURBS curve is $\xi_0=0.5$. As in the last subsection, the quantity of interest is the drag coefficient along the airfoil.

The final adapted mesh including 9285 cells, obtained by adaptively refining the initial mesh 5 times, is displayed in Fig. 14 (top middle). It is observed from the figure that the regions around the shock, near the leading and trailing edges, and in the neighborhood of the surface of airfoil have been locally refined, which demonstrates the effectiveness of the goal-oriented h-adaptive mesh refinement method. The Mach number

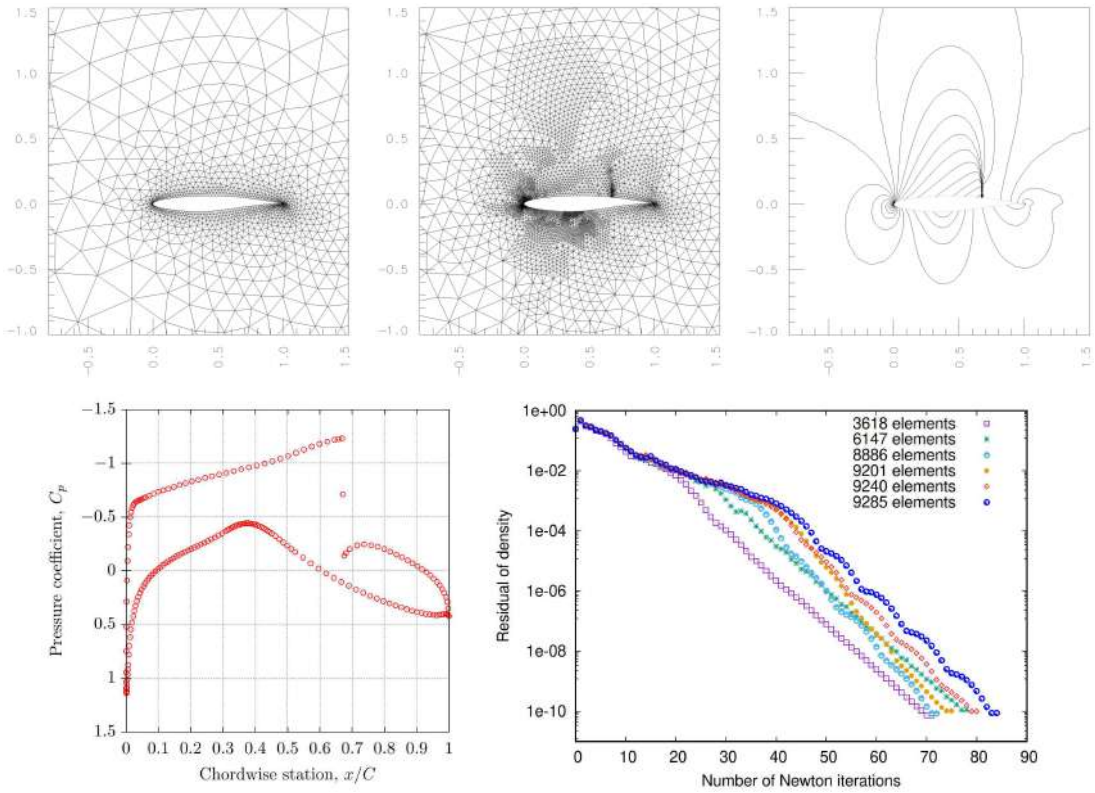

Figure 14: (RAE 2822, $M_\infty=0.75$, $\alpha=1^\circ$) Top left: the initial mesh around the RAE2822 airfoil. Top middle: the final adapted mesh around the airfoil. Top right: the distribution of Mach number isolines on the final adapted mesh, where the increment is $\Delta M=0.05$. Bottom left: the distribution of pressure coefficient along the surface of airfoil. Bottom right: the convergence history of the l^1-norm of density residual on several adaptively refined meshes.

contour obtained by using the NURBS-enhanced goal-oriented h-adaptive method on the final adapted mesh is shown in Fig. 14 (top right), and the distribution of pressure coefficient is presented in Fig. 14 (bottom left). It is observed from the figure that no numerical oscillations is generated around the discontinuity, which shows the effectiveness of the non-oscillatory k-exact reconstruction method. The convergence history of the l^1-norm of density residual is displayed in Fig. 14 (bottom right). One can see that the residual successfully converges to the stop tolerance on all adapted meshes.

Finally, we consider the transonic flow with free-stream Mach number $M_\infty=0.8$ and an angle of attack $\alpha=1.25^\circ$ around the two-body airfoils. In this test case, the two-body airfoils consist of two NACA0012 airfoils, and the physical domain is obtained by inserting another NACA0012 airfoil into the physical domain of the last subsection, where the newly inserted airfoil is obtained by moving the initial airfoil along the vector $\boldsymbol{a}=(1,0.5)^T$. For each NACA0012 airfoil, there are 96 grid points on the surface of airfoil, and those points are used to construct two cubic NURBS curves.

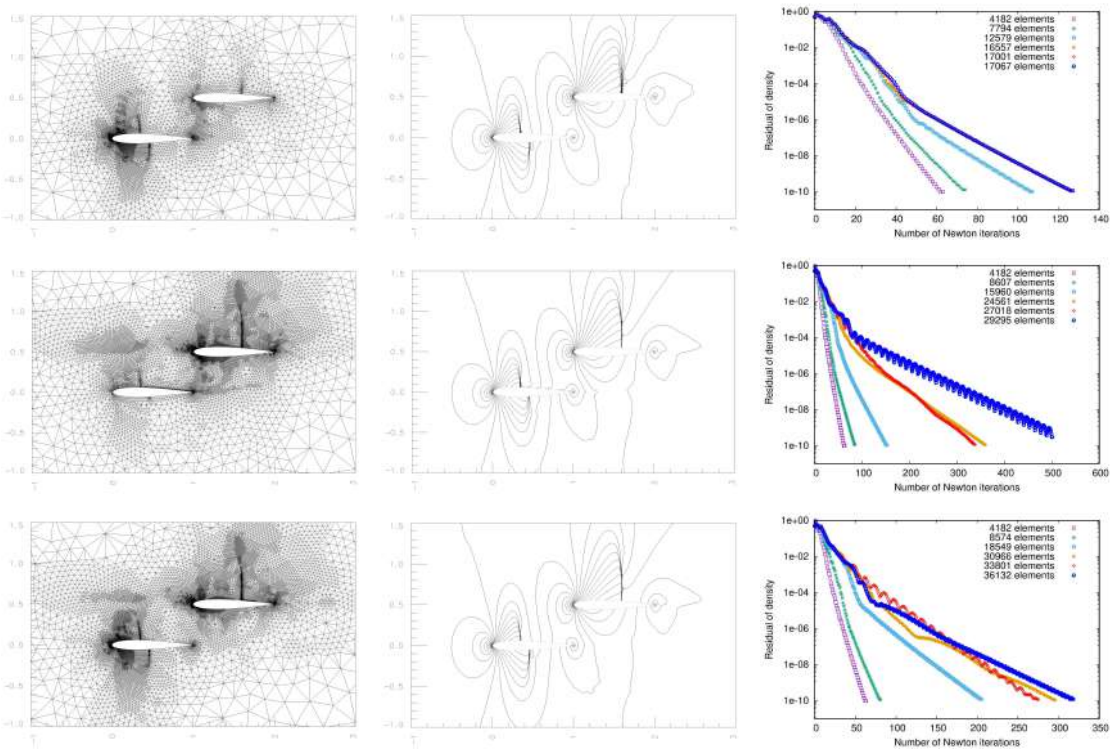

Figure 15: (Two-body airfoils, $M_\infty = 0.8$, $\alpha = 1.25°$) The final adapted meshes (left column), the associated distributions of Mach number isolines (middle column), and the convergence histories of l^1-norm of density residual (right column) computed by using three different quantities of interest. Top row: the drag coefficient based on the lower airfoil. Middle row: drag coefficient based on the upper one. Bottom row: drag coefficient based on both two airfoils.

In this test case, the drag coefficient along the surface of airfoil(s) is adopted as the quantity of interest. We first adopt the drag coefficient along the surface of lower airfoil as the quantity of interest, and we show the final adapted mesh, the related distribution of Mach number isolines, and the convergence history of l^1-norm of density residual on a sequence of adaptively refined meshes in Fig. 15 (top row). The related results obtained by using the drag coefficient along the surface of upper airfoil as the quantity of interest are presented in Fig. 15 (middle row), and the results obtained by using the drag coefficient along both the lower and upper airfoils as the quantity of interest are shown in Fig. 15 (bottom row). It can be observed from the adapted meshes (see Fig. 15 (left column)) that the goal-oriented h-adaptive refinement method can locally refine the mesh around the specified airfoil when the quantity of interest is about that airfoil, which shows the effectiveness of the proposed method. On the other hand, one can see from the figure that no numerical oscillations is generated around the shocks, which demonstrates the effectiveness of the non-oscillatory k-exact reconstruction. Furthermore, one can see that the residual successfully converges to the stop tolerance on all adapted meshes.

6 Conclusions

In this paper, a NURBS-enhanced finite volume method with the goal-oriented *h*-adaptivity is proposed to solve the two-dimensional steady Euler equations imposed in a domain with the curved boundary. The method is based on the finite volume discretization, and is built on a Newton-GMG framework for the discrete formulation of the steady Euler equations. To achieve the high order numerical accuracy, a non-oscillatory *k*-exact solution reconstruction and the NURBS representation of the curved boundary are employed in the method.

To improve the numerical accuracy and the efficiency of the algorithm, an *h*-adaptive mesh refinement method is introduced in the proposed numerical framework. To guarantee the performance of the *h*-adaptive method within the framework of NURBS-enhanced finite volume method, a simple and efficient point inversion technique is proposed to find the parameter values of mesh points lying on the NURBS curve, while a goal-oriented *a posteriori* error indicator is designed to efficiently calculate the given quantity of interest. The robustness of the proposed algorithm is demonstrated by a variety of numerical experiments, i.e., the convergence to the steady state of different flow configurations can be achieved with one set of parameters. The effectiveness of the goal-oriented *h*-adaptivity is demonstrated by the desired convergence of the error in a given quantity of interest in the simulations.

As future works, the aerodynamic shape optimization problem will be studied based on the proposed method. Furthermore, the extension of the method to the three-dimensional case will also be considered.

Acknowledgments

The research of Xucheng Meng is supported by the National Natural Science Foundation of China (Grant No. 12101057), the Scientific Research Fund of Beijing Normal University (Grant No. 28704-111032105), and the Start-up Research Fund from BNU-HKBU United International College (Grant No. R72021112). The research of Guanghui Hu is supported by FDCT of the Macao S. A. R. (0082/2020/A2), National Natural Science Foundation of China (Grant Nos. 11922120, 11871489), the Multi-Year Research Grant (MYRG2020-00265-FST) of University of Macau, and a grant from Department of Science and Technology of Guangdong Province (2020B1212030001).

References

[1] `https://web.mit.edu/easymesh_v1.4/www/easymesh.html`.
[2] `http://airfoiltools.com/airfoil/`.
[3] W.K. Anderson and V. Venkatakrishnan. Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation. *Computers & Fluids*, 28(4):443–480, 1999.

[4] J. Andren, H. Gao, M. Yano, D. Darmofal, C. Ollivier-Gooch, and Z. Wang. A comparison of higher-order methods on a set of canonical aerodynamics applications. In *20th AIAA Computational Fluid Dynamics Conference, AIAA paper 2011-3230*, 2011.
[5] A. Balan, M. Woopen, and G. May. *hp*-adaptivity on anisotropic meshes for hybridized discontinuous Galerkin scheme. In *22nd AIAA Computational Fluid Dynamics Conference, AIAA paper 2015-2006*, 2015.
[6] R. Balasubramanian and J.C. Newman. Comparison of adjoint-based and feature-based grid adaptation for functional outputs. *International Journal for Numerical Methods in Fluids*, 53(10):1541–1569, 2007.
[7] R. Balasubramanian and J.C. Newman. Adjoint-based error estimation and grid adaptation for functional outputs: Application to two-dimensional, inviscid, incompressible flows. *Computers & Fluids*, 38(2):320–332, 2009.
[8] T.J. Barth. Recent developments in high order *k*-exact reconstruction on unstructured meshes. *AIAA paper 93-0668*, 1993.
[9] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2D Euler equations. *Journal of Computational Physics*, 138(2):251–285, 1997.
[10] J. Blazek. *Computational Fluid Dynamics: Principles and Applications, Second edition*. Elsevier Science, 2005.
[11] V. Braibant and C. Fleury. Shape optimal design using B-splines. *Computer Methods in Applied Mechanics and Engineering*, 44(3):247–267, 1984.
[12] G. Carpentieri, B. Koren, and M.J.L. van Tooren. Adjoint-based aerodynamic shape optimization on unstructured meshes. *Journal of Computational Physics*, 224(1):267–287, 2007.
[13] L. Chen, G.H. Hu, and R. Li. Integrated Linear Reconstruction for Finite Volume Scheme on Arbitrary Unstructured Grids. *Communications in Computational Physics*, 24(2):454–480, 2018.
[14] R.F. Chen and Z.J. Wang. Fast, block lower-upper symmetric Gauss-Seidel scheme for arbitrary grids. *AIAA Journal*, 38(12):2238–2245, 2000.
[15] B. Cockburn, G.E. Karniadakis, and C.-W. Shu. The development of discontinuous Galerkin methods. In B. Cockburn, G.E. Karniadakis, and C.-W. Shu, editors, *Discontinuous Galerkin Methods*, volume 11 of *Lecture Notes in Computational Science and Engineering*, pages 3–50. Springer, Berlin, Heidelberg, 2000.
[16] K.J. Fidkowski and D.L. Darmofal. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA Journal*, 49(4):673–694, 2011.
[17] M.B. Giles and N.A. Pierce. An introduction to the adjoint approach to design. *Flow Turbulence and Combustion*, 65(3):393–415, 2000.
[18] M.B. Giles and E. Süli. Adjoint methods for PDEs: *A posteriori* error analysis and postprocessing by duality. *Acta Numerica*, 11:145–236, 2002.
[19] R. Goldenthal and M. Bercovier. Spline curve approximation and design by optimal control over the knots. In S. Hahmann, G. Brunnett, G. Farin, and R. Goldman, editors, *Geometric Modelling*, pages 53–64. Springer, 2004.
[20] F. Haider, J.-P. Croisille, and B. Courbet. Stability analysis of the cell centered finite-volume Muscl method on unstructured grids. *Numerische Mathematik*, 113(4):555–600, 2009.
[21] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *Journal of Computational Physics*, 183(2):508–532, 2002.
[22] C.Q. Hu and C.-W. Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 150:97–127, 1999.
[23] G.H. Hu. An adaptive finite volume method for 2D steady Euler equations with WENO reconstruction. *Journal of Computational Physics*, 252:591–605, 2013.

[24] G.H. Hu, R. Li, and T. Tang. A robust high-order residual distribution type scheme for steady Euler equations on unstructured grids. *Journal of Computational Physics*, 229(5):1681–1697, 2010.
[25] G.H. Hu, R. Li, and T. Tang. A robust WENO type finite volume solver for steady Euler equations on unstructured grids. *Communications in Computational Physics*, 9(3):627–648, 2011.
[26] G.H. Hu, X.C. Meng, and N.Y. Yi. Adjoint-based an adaptive finite volume method for steady Euler equations with non-oscillatory k-exact reconstruction. *Computers & Fluids*, 139:174–183, 2016.
[27] G.H. Hu and N.Y. Yi. An adaptive finite volume solver for steady Euler equations with non-oscillatory k-exact reconstruction. *Journal of Computational Physics*, 312:235–251, 2016.
[28] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39-41):4135–4195, 2005.
[29] A. Jalali and C. Ollivier-Gooch. An hp-adaptive unstructured finite volume solver for compressible flows. *International Journal for Numerical Methods in Fluids*, 85(10):563–582, 2017.
[30] A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3(3):233–260, 1988.
[31] L. Krivodonova and Berger M. High-order accurate implementation of solid wall boundary conditions in curved geometries. *Journal of Computational Physics*, 211(2):492–512, 2006.
[32] D. Li and R. Hartmann. Adjoint-based airfoil optimization with discretization error control. *International Journal for Numerical Methods in Fluids*, 77(1):1–17, 2015.
[33] R. Li. On multi-mesh H-adaptive methods. *Journal of Scientific Computing*, 24(3):321–341, 2005.
[34] R. Li, X. Wang, and W.B. Zhao. A multigrid block LU-SGS algorithm for Euler equations on unstructured grids. *Numerical Mathematics: Theory, Methods and Applications*, 1(1):92–112, 2008.
[35] X.C. Meng and G.H. Hu. A NURBS-enhanced finite volume solver for steady Euler equations. *Journal of Computational Physics*, 359:77–92, 2018.
[36] L. Piegl and W. Tiller. *The NURBS Book (Monographs in Visual Communication), Second edition*. Springer-Verlag, New York, 1997.
[37] X.P. Qian. Full analytical sensitivities in NURBS based isogeometric shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 199(29):2059–2071, 2010.
[38] C.J. Roy, C.L. Rumsey, and E.N. Tinoco. Summary data from the Sixth AIAA Computational Fluid Dynamics Drag Prediction Workshop: Code verification. *Journal of Aircraft*, 55(4):1338–1351, 2018.
[39] R. Sevilla, S. Fernández-Méndez, and A. Huerta. NURBS-enhanced finite element method for Euler equations. *International Journal for Numerical Methods in Fluids*, 57(9):1051–1069, 2008.
[40] R. Sevilla, S. Fernández-Méndez, and A. Huerta. NURBS-enhanced finite element method (NEFEM). *International Journal for Numerical Methods in Engineering*, 76(1):56–83, 2008.
[41] R. Sevilla and A. Huerta. HDG-NEFEM with degree adaptivity for Stokes flows. *Journal of Scientific Computing*, 77(3):1953–1980, 2018.
[42] L. Shi and Z.J. Wang. Adjoint-based error estimation and mesh adaptation for the correction procedure via reconstruction method. *Journal of Computational Physics*, 295:261–284, 2015.
[43] S.N. Skinner and H. Zare-Behtash. State-of-the-art in aerodynamic shape optimisation methods. *Applied Soft Computing*, 62:933–962, 2018.

[44] E.F. Toro, M. Spruce, and W. Speares. Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4(1):25–34, 1994.
[45] W.A. Wall, M.A. Frenzel, and C. Cyron. Isogeometric structural shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 197(33):2976–2988, 2008.
[46] K. Wang, S.J. Yu, Z. Wang, R.Z. Feng, and T.G. Liu. Adjoint-based airfoil optimization with adaptive isogeometric discontinuous Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 344:602–625, 2019.
[47] Z.J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H.T. Huynh, N. Kroll, G. May, P.-O. Persson, B. van Leer, and M. Visbal. High-order CFD methods: Current status and perspective. *International Journal for Numerical Methods in Fluids*, 72(8):811–845, 2013.
[48] M. Woopen, G. May, and J. Schütz. Adjoint-based error estimation and mesh adaptation for hybridized discontinuous Galerkin methods. *International Journal for Numerical Methods in Fluids*, 76(11):811–834, 2014.
[49] M. Yano and D. L. Darmofal. Case C1.3: Flow over the NACA 0012 airfoil: Subsonic inviscid, transonic inviscid, and subsonic laminar flows. In *First International Workshop on High-Order CFD Methods*, 2012.