Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **8**, No. 5, pp. 847-870

DOI: 10.4208/aamm.2015.m1087
October 2016

# Solution Reconstruction on Unstructured Tetrahedral Meshes Using $P^1$-Conservative Interpolation

Biao Peng[1], Chunhua Zhou[1,*] and Junqiang Ai[2]

[1] *Department of Aerodynamics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*
[2] *The First Aircraft Institute, Aviation Industry Corporation of China, Xi'an 710089, China*

Received 4 May 2015; Accepted (in revised version) 6 September 2015

**Abstract.** This paper extends an algorithm of $P^1$-conservative interpolation on triangular meshes to tetrahedral meshes and thus constructs an approach of solution reconstruction for three-dimensional problems. The conservation property is achieved by local mesh intersection and the mass of a tetrahedron of the current mesh is calculated by the integral on its intersection with the background mesh. For each current tetrahedron, the overlapped background tetrahedrons are detected efficiently. A mesh intersection algorithm is proposed to construct the intersection of a current tetrahedron with the overlapped background tetrahedron and mesh the intersection region by tetrahedrons. A localization algorithm is employed to search the host units in background mesh for each vertex of the current mesh. In order to enforce the maximum principle and avoid the loss of monotonicity, correction of nodal interpolated solution on tetrahedral meshes is given. The performance of the present solution reconstruction method is verified by numerical experiments on several analytic functions and the solution of the flow around a sphere.

**AMS subject classifications**: 41A05, 65D05

**Key words**: Solution reconstruction, solution transfer, conservative interpolation, tetrahedral mesh, solution interpolation, mesh intersection.

## 1 Introduction

Solution reconstruction via interpolation from one mesh to another is absolutely vital in many numerical simulations, such as multi-physics simulations [1], fluid-structure interaction problems, numerical weather prediction and climate simulation [2–4] and mesh adaptation for time-dependent problems [5]. As repeated interpolation is needed for

*Corresponding author.
*Email:* pb_nuaa@126.com (B. Peng), chzhou@nuaa.edu.cn (C. H. Zhou), avic603ajq@163.com (J. Q. Ai)

 ©2016 Global Science Press

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

these problems, the accuracy and conservation of the interpolation are very important. Maintaining the conservation of the interpolation can avoid the excessive accumulation of errors [5, 6]. Therefore, some interpolation methods preserving the conservation are developed.

The remapping process of arbitrary Lagrangian-Eulerian (ALE) method is one of the important applications of conservative interpolation. In Lagrangian step, points and cells move together with the flow until certain cells become extremely distorted, then a rezoning procedure is performed. Therefore, a remapping procedure to transfer data from one mesh to another is needed. The related literatures on remapping can be found in [7–9]. Meshes used in ALE method are usually quadrilateral or hexahedral meshes with a fixed topology. A conservative remapping scheme for grids in spherical coordinates is proposed by Jones [10]. Some of these methods have been extended to the application for two unrelated meshes [11, 12]. Integral (or global) remapping permits that the current mesh and the background mesh are unrelated. Therefore, it has complete flexibility for the two meshes at the cost of calculating the integral volume of the intersection of background and current cells. A conservative interpolation method that converts the volume integral to a surface integral by using the divergence theorem is proposed by Dukowicz [13]. This method can simplify the problem of volume intersection computing, but it suffers from excessive diffusion due to its first order nature. Some work has been done to improve the accuracy by Dukowicz and Kodis in [7]. Geuzaine et al. proposed a Galerkin projection method that can be used to mesh-to-mesh interpolation [14]. A common-refinement based data transfer method applicable to surface meshes is presented by Jiao and Heath [1]. By using a similar concept with common-refinement, a conservative interpolation method based on Galerkin projection is proposed by Farrell et al. in [15, 16]. This method can be used for two unrelated volume meshes, but it involves complex matrix operations and the construction of the so-called supermesh.

In [5], Alauzet and Mehrenberger proposed an effective two-dimensional $P^1$-conservative interpolation algorithm. The mass conservation property of the interpolation operator is achieved by local triangle-triangle intersection. For a given solution on the background mesh, the mass and gradient of the interpolated solution on each triangle of the current mesh is first computed from those on all triangles of background mesh it overlaps. Then, the solution on each node of the current triangle can be reconstructed by using the mass and gradient transferred from the background mesh. In order to avoid the loss of monotonicity, the maximum principle is enforced by a specific correction of the interpolated solution. This algorithm is $P^1$ exact (order 2) and permits that the current mesh and the background mesh are unrelated while the interpolation algorithms used in the ALE methods are usually for quadrilateral and hexahedral meshes with a fixed topology (unchanged connectivity) [7–9] or only of the first order accuracy [13]. Compared to the algorithm of Farrell et al. [15, 16] in which resolution of a linear system and construction of the super-mesh are required, it is a matrix-free approach based on local mesh intersections and appropriate local reconstructions. The locality is primordial for efficiency and robustness. The conservation property of this algorithm has been checked in [5]. Howev-

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

er, it is still a challenge to extend this algorithm to three-dimensional tetrahedral meshes because of the complexity of mesh intersection and reconstruction.

Zhou and Ai applied the algorithm of $P^1$-conservative interpolation on triangular meshes proposed by Alauzet and Mehrenberger to the mesh adaptation for the simulation of two-dimensional unsteady flows with moving boundaries [17]. Solution interpolation is a key stage in the mesh adaptation for unsteady flows because it transfers the previous flow solution on the old mesh to the new adapted mesh and allows restarting the simulation of unsteady flows on the new mesh from the previous state. Preserving the mass conservation in this recursive stage is crucial in the dynamic mesh adaption for time-dependent problems as the errors induced by solution transferring may accumulate in the long-time scale simulations. The impact of the errors due to solution transferring by standard interpolation has been discussed in [5, 17]. For time-dependent problems, dynamic mesh adaptation is more important since the physical phenomena may evolve arbitrarily in the computational domain. Without mesh adaptation, a uniform fine mesh is needed to preserve the accuracy of numerical solution. Using such a mesh is computationally prohibitive, especially for three-dimensional problems.

In this paper, we aim to extend the algorithm of Alauzet and Mehrenberger to three-dimensional tetrahedral meshes. As discussed above, such an extension is a must for dynamic unstructured mesh adaption in the simulation of complex three-dimensional time-dependent problems. The mass conservation for a tetrahedron of the current mesh is preserved by calculating the integral on its intersection of the background mesh. Therefore, we proposed a detection algorithm to find all the background tetrahedrons overlapped by a specified current tetrahedron and a mesh intersection algorithm to construct the intersection of a current tetrahedron with an overlapped background tetrahedron and mesh the intersection region by tetrahedrons. The maximum principle is satisfied by the correction to the nodal interpolated solution on tetrahedral meshes.

The outline of the paper is as follows. Section 2 introduces the solution reconstruction based on $P^1$-conservative interpolation. Section 3 describes the method of overlapped tetrahedron detection. Section 4 presents the localization algorithm, which is the prerequisite for seeking overlapping units. Details of the mesh intersection algorithm are given in Section 5. Several numerical examples are presented in section 6 to validate the proposed scheme. The paper closes with some conclusions.

## 2 $P^1$-conservative solution reconstruction on tetrahedral meshes

Consider a bounded domain $\Omega \in R^3$ and two different tetrahedral meshes of this domain $H^1$ and $H^2$. Denote the project operator from the background mesh $H^1$ to the current mesh $H^2$ by $\Pi$, the $P^1$ discrete solution defined at the vertices of $H^1$ by $u_1$, and the projected solution $\Pi u_1$ on $H^2$ by $u_2$. If $\int_{H^1} u_1 d\Omega = \int_{H^2} u_2 d\Omega$, the operator $\Pi$ is said to be conservative.

At first, we introduce the signed volume of a tetrahedron and the barycentric coor-

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

dinate of a given point. Denote the four vertices of a tetrahedron $K$ by $P_0,P_1,P_2,P_3$. The numeration of the four faces is in the same order as their viewing vertices. The signed volume $V_K$ is defined as

$$V_K=\frac{1}{6}\begin{vmatrix} x_0 & y_0 & z_0 & 1 \\ x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \end{vmatrix}. \tag{2.1}$$

The value of the volume defined in this way can be positive or negative. Let $P$ be a given point. When the vertex $P_i$ of the tetrahedron $K(P_0,P_1,P_2,P_3)$ is replaced by $P$, a virtual tetrahedron is constituted. We denote this virtual tetrahedron by $K_i$. For example, when the vertex $P_2$ of $K$ is substituted by $P$, the four points $P_0$, $P_1$, $P$, $P_3$ constitute the virtual tetrahedron $K_2$. The four associated barycentric coordinates of $P$ are given by

$$\beta_i=\frac{V_{K_i}}{V_K},\quad i=0,1,2,3. \tag{2.2}$$

If $P$ is inside $K$, each $\beta_i$ is greater or equal to zero.

Provided that $P$ is a vertex of the current mesh $H^2$ that locates inside the tetrahedron $K^1\in H^1$, the classical $P^1$ interpolation scheme can be written as

$$u_2(P)=\sum_{i=0}^{3}\beta_i(P)u_1(P_i). \tag{2.3}$$

This scheme is $P^1$ exact, monotone and satisfies the maximum principle but it does not conserve the mass.

In this paper, we extend the $P^1$-conservative interpolation algorithm proposed by Alauzet and Mehrenberger in [5] to three dimensions and construct a solution reconstruction method for unstructured tetrahedral meshes.

We consider the solution on the background mesh $H^1$, which is piecewise linear by elements. For each background tetrahedron $K^1\in H^1$, we have the constant gradient $(\nabla u_1)|_{K^1}$ and the mass

$$m_{K^1}=\int_{K^1}u_1 d\Omega=|K^1|u_1(G), \tag{2.4}$$

where $G$ is the barycenter of the tetrahedron and the absolute value means the volume of the tetrahedron. For each tetrahedron $K^2\in H^2$, all tetrahedrons of the background mesh $H^1$ it overlaps can be computed as $\{K_j^1\}$. Denote the intersection region between $K^2$ and $K_j^1$ by $\Omega_j=K^2\bigcap K_j^1$, which can be discretized by a simple tetrahedral mesh $T_j$. Then, for

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

each tetrahedron $K^2$, the mass and the gradient is given as

$$m_{K^2} = \int_{K^2} u_2 d\Omega = \sum_j \int_{\Omega_j} u_1 d\Omega, \tag{2.5a}$$

$$(\nabla u_2)|_{K^2} = \frac{\sum_j \int_{\Omega_j} \nabla u_1 d\Omega}{|K^2|}. \tag{2.5b}$$

Since the solution $u_1$ is piecewise linear by elements, the integrals in (2.5a) and (2.5b) can be calculated as

$$\int_{\Omega_j} u_1 d\Omega = V_{\Omega_j} u_1(G_{\Omega_j}), \tag{2.6a}$$

$$\int_{\Omega_j} \nabla u_1 d\Omega = V_{\Omega_j} (\nabla u_1)|_{K^1}. \tag{2.6b}$$

In the above equations, $G_{\Omega_j}$ and $V_{\Omega_j}$ are the barycenter and volume of the intersection region, respectively. Since $\Omega_j$ is discretized by $T_j$, $G_{\Omega_j}$ and $V_{\Omega_j}$ can be calculated easily.

For the sake of simplicity, in the following discussions, the interpolated solution $u_2 = \Pi u_1$ on the current mesh $H^2$ is replaced by $u$ and a tetrahedron $K^2 \in H^2$ by $K$. Denote by $u_K$ the interpolated solution on the tetrahedron $K$. The value of $u_K$ at the barycenter of $K$ and the gradient on $K$ are given by

$$u_K(G_K) = \frac{m_K}{|K|}, \tag{2.7a}$$

$$\nabla u_K = (\nabla u)|_K. \tag{2.7b}$$

The value of interpolated solution at each vertex $P_i$ of the element $K$ can be obtained by Taylor expansion

$$u_K(P_i) = u_K(G_K) + \nabla u_K \cdot \overrightarrow{G_K P_i}. \tag{2.8}$$

This solution reconstruction is conservative and $P^1$ exact. Solution gradient reconstruction brings higher-order accuracy. However, it can lead to the loss of monotonicity. The maximum principle can be enforced by correcting the interpolated solution [5]. Here, we extend the correction for the interpolated nodal solution on triangular meshes [5] to tetrahedral meshes.

Let $L$ be the set of the vertices of the tetrahedrons of $H^1$, which the tetrahedron $K \in H^2$ overlaps. For each vertex $P_i$ of $K$, the nodal values satisfy the maximum principle if

$$u_{\min} = \min_{Q \in L} u_1(Q) \le u_K(P_i) \le \max_{Q \in L} u_1(Q) = u_{\max}. \tag{2.9}$$

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

The indices are reordered firstly such that $u_K(P_0)\leq u_K(P_1)\leq u_K(P_2)\leq u_K(P_3)$. Then, we set

$$u_K^M(P_3)=\min(u_K(P_3),u_{\max}), \tag{2.10a}$$

$$u_K^M(P_2)=\min\left(u_K(P_2)+\frac{1}{3}\max(0,u_K(P_3)-u_{\max}),u_{\max}\right), \tag{2.10b}$$

$$u_K^M(P_1)=\min\left(u_K(P_1)+\frac{1}{3}\max(0,u_K(P_3)-u_{\max})+\frac{1}{2}\max\left(0,u_K(P_2)+\frac{1}{3}(u_K(P_3)-u_{\max})-u_{\max}\right),u_{\max}\right), \tag{2.10c}$$

$$u_K^M(P_0)=4u_K(G)-u_K^M(P_1)-u_K^M(P_2)-u_K^M(P_3). \tag{2.10d}$$

The new corrected nodal values $\tilde{u}_K(P_i)$ can be obtained as follows

$$\tilde{u}_K(P_0)=\max(u_K^M(P_0),u_{\min}), \tag{2.11a}$$

$$\tilde{u}_K(P_1)=\max\left(u_K^M(P_1)-\frac{1}{3}\max(0,u_{\min}-u_K^M(P_0)),u_{\min}\right), \tag{2.11b}$$

$$\tilde{u}_K(P_2)=\max\left(u_K^M(P_2)-\frac{1}{3}\max(0,u_{\min}-u_K^M(P_0))-\frac{1}{2}\max\left(0,u_{\min}-\left(u_K^M(P_1)-\frac{1}{3}\max(0,u_{\min}-u_K^M(P_0))\right)\right),u_{\min}\right), \tag{2.11c}$$

$$\tilde{u}_K(P_3)=4u_K(G)-\tilde{u}_K(P_0)-\tilde{u}_K(P_1)-\tilde{u}_K(P_2). \tag{2.11d}$$

The corrected reconstruction $\tilde{u}_K$ satisfies the maximum principle, and is $P^1$ exact and conservative. Its $P^1$ exactness and conservation are obvious. The proof of its satisfaction of the maximum principle is given in Appendix.

This interpolated solution on the current mesh $H^2$ is conservative and piecewise linear by elements, but discontinuous. One more stage is needed to retrieve a solution at the vertices of $H^2$. The corrected solution (2.11a)-(2.11d) can be redistributed to each vertex $P$ of the current mesh $H^2$ by averaging

$$\tilde{u}(P)=\frac{\sum|K|\tilde{u}_K(P)}{\sum|K|}, \tag{2.12}$$

where the summations are over all tetrahedrons that share vertex $P$. The final interpolated solution (2.12) is continuous piecewise linear and conservative, and still satisfies the maximum principle.

From the above discussions, it can be seen that detecting the overlapped tetrahedrons $\{K_j^1\}$ for each tetrahedron $K^2\in H^2$, searching the host tetrahedron in the background mesh $H^1$ for each vertex of the current mesh $H^2$, and constructing and meshing the intersection region $\Omega_j=K^2\bigcap K_j^1$ are key steps for the proposed solution reconstruction methods. The mentioned techniques will be presented in detail in the following sections.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

## 3 Detection of overlapped tetrahedrons

Inspired by the technique to detect overlapped triangles proposed in [16], we develop a method for the detection of overlapped tetrahedrons. Using this method, for each tetrahedron $K^2$ of the current mesh $H^2$ we can find all the overlapped tetrahedrons in the background mesh $H^1$. The method is summarized as below.

Step 1 For each tetrahedron $K^2 \in H^2$, create an array to store the tetrahedrons of the background mesh $H^1$ it overlaps. The initial elements of this array are the host tetrahedrons of the four vertices of $K^2$. Obviously, $K^2$ overlaps these tetrahedrons.

Step 2 Compute the intersection points between $K^2$ and the unchecked neighbors of each tetrahedron which are already stored in the array. The computation of intersection points is included in Section5. When the number of the intersection points is greater than 3 and the intersection points are not coplanar, this neighbor is overlapped by $K^2$. And then, all overlapped neighbors are added into the array. Here, the neighbors of a tetrahedron mean the other tetrahedrons that share a face with it.

Step 3 Perform the intersection check in Step 2 successively for the neighbors of the tetrahedrons in the array until no new overlapped tetrahedron can be found.

## 4 Localization algorithm

Localization is to search the host tetrahedron in the background mesh $H^1$ for each vertex of the current mesh $H^2$. The determination of these tetrahedrons is also the key of the detection of the overlapped tetrahedrons described in the previous section. In order to avoid the naive quadratic scheme in $\mathcal{O}(N_v^2 \times N_t^1)$ ($N_v^2$ is the number of vertices of $H^2$ and $N_t^1$ the number of tetrahedrons of $H^1$), we introduce the localization algorithm presented in [18,19], which utilizes the barycentric coordinates and the topology of the background mesh.

Denote the four faces of a tetrahedron $K$ by $S_0$, $S_1$, $S_2$, $S_3$. From the definition of the signed volume, we know that the barycentric coordinates $\beta_i$ of a given point $P$ can be positive or negative. When $\beta_i$ is positive, the point $P$ and the vertex $P_i$ of $K$ locate on the same side of the plane of $S_i$, and vice versa. When its four barycentric coordinates are all non-negative, $P$ must be inside the tetrahedron $K$. If $\beta_i$ has a negative value, $P$ locates outside the tetrahedron and just at the outer side of the face $S_i$. An example for $\beta_3 < 0$ is shown in Fig. 1. Localization algorithm takes advantage of this property to identify the host tetrahedron of each vertex of the current mesh $H^2$.

Precisely for two tetrahedral meshes, let $P$ be a vertex of the current mesh $H^2$, $K^1$ a tetrahedron of the background mesh $H^1$. According to the signs of the four barycentric coordinates of $P$ associated with $K^1$, several possible cases may arise.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

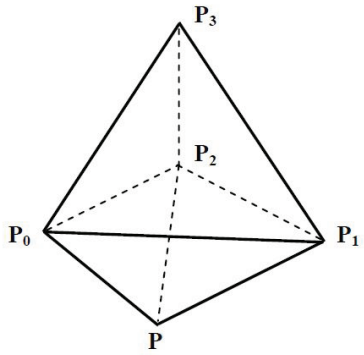



Figure 1: Illustration of the case $\beta_3<0$: $P$ at the outer side of the face $S_3(P_0P_1P_2)$.

1. If all barycentric coordinates are non-negative, $P$ is located inside $K^1$, which is the host tetrahedron of $P$.
2. If there is only one negative $\beta_i$, we will move to the neighboring element which shares face $S_i$ with $K^1$. In this case, we say that $P$ is viewed by the face $S_i$.
3. Else if there are more than one negative $\beta_i$s, more than one neighboring tetrahedrons are possible for the next move. A random choice is adopted.

We choose randomly one of the background tetrahedrons as the initial candidate of the host tetrahedron of $P$ to start the above searching process. According to the signs of the barycentric coordinates, we move to the corresponding neighbor and find the next candidate. Repeating this process until the four barycentric coordinates are all non-negative, we can find the visited tetrahedron that contains the point $P$, i.e., the host tetrahedron of $P$. The complexity of this search algorithm is $\mathcal{O}(N_v^2\times n)$, where $n$ is the averaged number of visited tetrahedrons for each localization.

The above barycentric coordinate-based algorithm presented in [18, 19] only utilizes the topology of the background mesh $H^1$. It is very time-consuming if the number of the visited elements in $H^1$ is large. To reduce the number of visited tetrahedrons and improve the algorithm, we utilize the topologies of both the background mesh $H^1$ and the current mesh $H^2$ as proposed in [5] for triangular meshes. The idea is that once a vertex $P$ has been located in a background tetrahedron $K_p^1$, we set $K_p^1$ as the initial candidate of the host tetrahedrons for the set of vertices that are connected to $P$ by a cell edge. Consequently, the number of visited tetrahedrons is drastically reduced as the initial guess of the searching process is close to the target. In practice, the number of visited tetrahedrons is almost equal to that of the background tetrahedrons which are overlapped by a current tetrahedron.

Usually, there are some objects in the mesh. The above searching process will fail when the object surface is encountered. Our solution is to follow a path on the surface of the object in order to bypass the obstacle. However, it is not always effective. In the

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

case where the obstacle cannot be bypassed, for each unlocated vertex of $H^2$, we can find its nearest tetrahedron of the background mesh $H^1$ adjacent to the object surface and set this tetrahedron as the initial candidate of its host tetrahedron. Then we can perform the mentioned searching process and complete finally the localization of all vertices of $H^2$.

It is worth mentioning that the vertex $P$ of a tetrahedron $K^2$ of the current mesh may be at the face, edge or vertex of a tetrahedron of the background mesh. In this case, all the tetrahedrons which contain this face, edge or vertex are recorded as the host tetrahedrons of $P$. These hosts are likely to overlap $K^2$.

## 5 Mesh intersection algorithm

Mesh intersection algorithm is to construct the intersection of a tetrahedron $K^1$ of the background mesh $H^1$ with a tetrahedron $K^2$ of the current mesh $H^2$ and mesh the intersection region by tetrahedrons. If the two tetrahedrons overlap each other, the intersection region must be a convex polyhedron. The key to construct the intersection region is to find out all the vertices of the convex polyhedron. The vertices of the convex polyhedron can be classified as two groups: edge-face intersection points and vertices of a tetrahedron locating inside another. The number of vertices of this polyhedron is between 4 and 12 [20,21].

Using the values of its barycentric coordinates about a given tetrahedron $K^2 \in H^2$, one can easily determine whether a vertex of $K^1 \in H^1$ is inside $K^2$ or not. If the four barycentric coordinates of each vertex of $K^1$ are all non-negative, $K^1$ is contained by $K^2$ and the intersection is just $K^1$ itself. Conversely, using the values of the barycentric coordinates of each vertex of $K^2$ about $K^1$, we can also determine whether $K^2$ is contained by $K^1$ or not. If $K^2$ is contained by $K^1$, the intersection is just $K^2$. For these two cases, the number of vertices of the intersection polyhedron is 4.

Except the above two cases, the intersection between an edge of $K^1$ (resp. $K^2$) and a triangular face of $K^2$ (resp. $K^1$) must be calculated. The calculation will be described in Subsection 5.1. When the number of vertices of the intersection polyhedron is greater than 4, the polyhedron must be partitioned into a number of non-overlapping tetrahedrons. The generation of a simple tetrahedral mesh of the intersection region will be presented in Subsection 5.2.

### 5.1 Calculation of edge-face intersection

Calculation of intersection between an edge of one tetrahedron $K^1$ and a triangular face of another tetrahedron $K^2$ is divided into three steps. The first is to determine whether the edge intersects the infinite plane $S$ which contains the triangle, the second is to determine whether the edge and the triangle intersect with each other, and the third is to calculate the intersection points.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

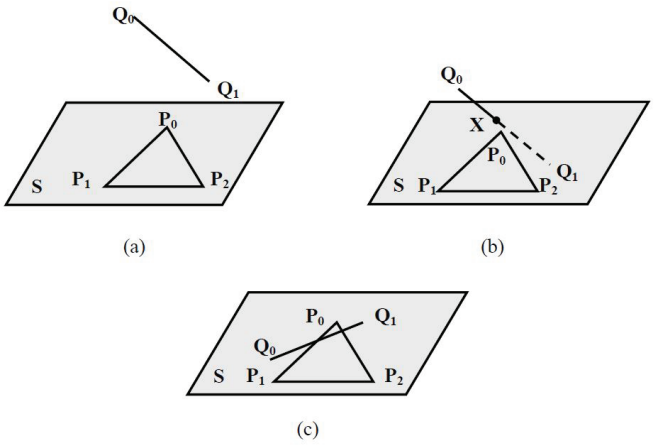



Figure 2: Illustration of the edge-plane intersection.

The question of whether an edge intersects an infinite plane can be answered by determining whether the two endpoints of the edge are on the different side of the plane. As shown in Fig. 2, there are three cases for the position of an edge $Q_0Q_1$ of the tetrahedron $K^1(Q_0Q_1Q_2Q_3)$ relative to the plane $S$ containing the triangular face $P_0P_1P_2$ of the tetrahedron $K^2(P_0P_1P_2P_3)$. The position of edge $Q_0Q_1$ relative to the plane $S$ is determined by the signed volumes of the tetrahedrons $Q_0P_0P_1P_2$ and $Q_1P_0P_1P_2$ as below.

1. If $V_{Q_0P_0P_1P_2}\times V_{Q_1P_0P_1P_2}>0$, then $Q_0$ and $Q_1$ locate on the same side of $S$ as shown in Fig. 2(a), and therefore the edge does not intersect $S$.
2. If $|V_{Q_0P_0P_1P_2}|+|V_{Q_1P_0P_1P_2}|\neq 0$ and $V_{Q_0P_0P_1P_2}\times V_{Q_1P_0P_1P_2}\leq 0$, then $Q_0$ and $Q_1$ locate on the different sides of the plane as shown in Fig. 2(b), and the line segment intersects $S$. The intersection point $X$ can be calculated easily in terms of the signed volumes of $Q_0P_0P_1P_2$ and $Q_1P_0P_1P_2$

$$X=Q_0+\frac{V_{Q_0P_0P_1P_2}}{V_{Q_0P_0P_1P_2}-V_{Q_1P_0P_1P_2}}\cdot\overrightarrow{Q_0Q_1}. \tag{5.1}$$

3. If $V_{Q_0P_0P_1P_2}=V_{Q_1P_0P_1P_2}=0$, then $Q_0$ and $Q_1$ locate within the plane as shown in Fig. 2(c) and the number of the intersection points between $Q_0Q_1$ and the edges of triangle $P_0P_1P_2$ may be 0, 1 or 2. Actually, the number of the edge-edge intersection points still needs further determination. Note that edge-edge intersection points in this case are the vertices of the convex polyhedron formed by the tetrahedron-tetrahedron intersection.

For the case shown in Fig. 2(b) where the edge intersects the infinite plane $S$ containing the triangular surface $P_0P_1P_2$, We still need to know whether the intersection point $X$ locates inside the triangle. This problem can be solved by using three barycentric coordinates of $X$ about the tetrahedron $P_0P_1P_2P_3$. As shown in Fig. 3(a), when the three

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

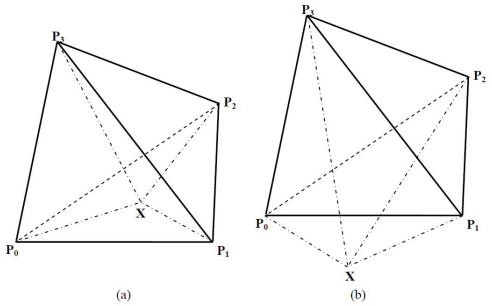



Figure 3: Determination of whether the intersection point $X$ locates inside the triangle $P_0P_1P_2$.

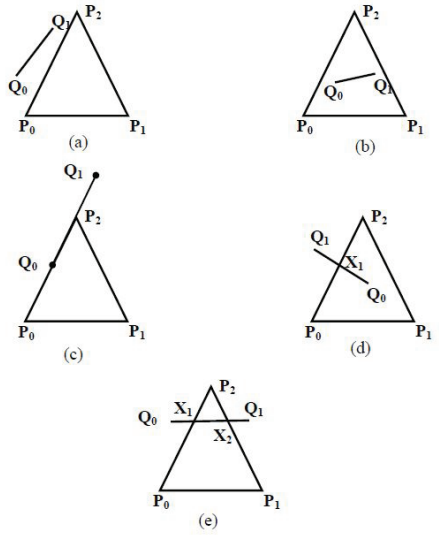



Figure 4: Illustration of edge-triangle intersection when the edge $Q_0Q_1$ locates in the infinite plane which contains the triangle.

barycentric coordinates $\beta_0\beta_1\beta_2$ of $X$ about the tetrahedron $P_0P_1P_2P_3$ are all non-negative, the point locates inside the triangle. If there is any negative barycentric coordinate, the point $X$ locates outside the triangle (an example for $\beta_2<0$ is shown in Fig. 3(b)).

For the case shown in Fig. 2(c) where the edge $Q_0Q_1$ locates in the infinite plane $S$, the number of edge-edge intersection points may be 0, 1 or 2. There are five cases of the edge-edge intersection as shown in Fig. 4.

Fig. 4(a) shows the case where the segment $Q_0Q_1$ does not intersect any triangle edge. Any end point of $Q_0Q_1$ which locates inside the triangle $P_0P_1P_2$, such as $Q_0$ and $Q_1$ in Fig. 4(b), and $Q_0$ in Fig. 4(c) and Fig. 4(d), is a vertex of the intersection polyhedron. Therefore, they are recorded as edge-face intersection points. The previous checking whether each vertex of $K^1$ is inside $K^2$ has determined whether $Q_0$ and $Q_1$ locate inside the triangle surface or not. The points of intersection $X_1$ in Fig. 4(d), and $X_1$, $X_2$ in Fig. 4(e) belong to the same class: the intersection of a segment and triangle edges.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

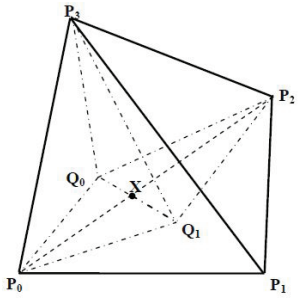



Figure 5: Illustration of the edge-edge intersection.

When the edge $Q_0Q_1$ locates in the infinite plane which contains the triangle, we can also determine whether $Q_0Q_1$ intersects an edge of the triangle by the concept of signed volume. For an example shown in Fig. 5, if $V_{Q_0P_0P_2P_3}\times V_{Q_1P_0P_2P_3}<0$ and $V_{P_0Q_0Q_1P_3}\times V_{P_2Q_0Q_1P_3}<0$, $Q_0Q_1$ intersects the edge $P_0P_2$ at point $X$. The intersection point can be calculated as

$$X=Q_0+\frac{V_{Q_0P_0P_2P_3}}{V_{Q_0P_0P_2P_3}-V_{Q_1P_0P_2P_3}}\cdot\overrightarrow{Q_0Q_1}. \tag{5.2}$$

If the values of the mentioned volumes are all equal to zero, the line segment $Q_0Q_1$ is collinear with the edge of the triangle as shown in Fig. 4(c). Any of the endpoints of $Q_0Q_1$ which lies on the edge of the triangular face is the vertex of the intersection polyhedron. Therefore, it is recorded as edge-face intersection. The previous checking whether each vertex of $K^1$ is inside $K^2$ has determined whether $Q_0$ or $Q_1$ locates on the edge of a current tetrahedron.

## 5.2 Meshing of a convex polyhedron

After all the vertices of the convex polyhedron formed by the intersection of two overlapping tetrahedrons are obtained, we mesh the polyhedron by tetrahedrons for the evaluation of the integrals in (2.5a) and (2.5b). The procedure of mesh generation is summarized as below.

Step 1 Take four non-coplanar points from the list of the vertices of the convex polyhedron to construct a tetrahedron which is the first element of the tetrahedron set $T_j$, and delete the four points from the vertex list and update the list.

Step 2 Take one point $P$ from the current vertex list and compute its barycentric coordinates $\beta_i$ $(i=1,2,3,4)$ about each tetrahedron $K\in T_j$. If $\beta_i$ is negative and the face $S_i$ of $K$ is a boundary face of $T_j$, connect $P$ with the three vertices of $S_i$ and build a new tetrahedron. The number of the new built tetrahedrons may be greater than one. Then, add the built tetrahedrons into $T_j$ and delete $P$ from the vertex list.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

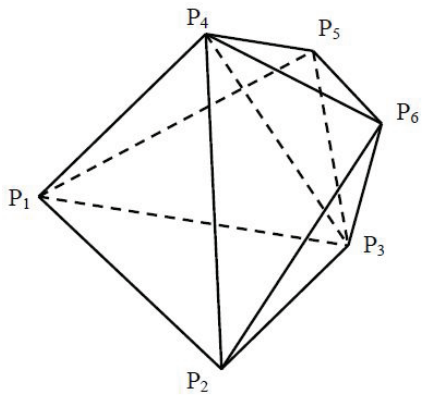



Figure 6: Illustration of meshing of a convex polyhedron with 6 vertices.

Step 3 Repeat Step 2 till the vertex list is empty. The intersection region of two overlapping tetrahedrons is finally discretized by $T_j$.

An example for meshing of a convex polyhedron with 6 vertices is shown in Fig. 6. $P$'s subscript represents the order in which the vertices are deleted from the list.

## 6 Numerical experiments

In the Subsections 6.1, 6.2 and 6.3, through the analysis on three analytical functions defined on the domain $[0,1]\times[0,1]\times[0,1]$, we firstly verify the behaviors of solution transfer/reconstruction by the $P^1$ conservative interpolation, such as the mass conservation, the accuracy and the convergence order. To investigate the convergence order of solution transfer, two series of meshes denoted by $H_i^1|_{i=1,3}$ and $H_i^2|_{i=1,3}$ are generated by the uniform and successive refinement of the initial meshes $H_1^1$ and $H_1^2$, respectively. The tetrahedral cells of $H_1^1$ and $H_1^2$ are obtained by partitioning the hexahedral cells of the corresponding Cartesian meshes. The series of meshes are summarized in Table 1. For each test case, the analytical function is applied on $H_i^1$ and $H_i^2$ to provide the solution fields $u_{1,i}$ and $u_{2,i}$, respectively. If we transfer $u_{1,i}$ from $H_i^1$ to $H_i^2$ and get $\Pi u_{2,i}$, this solution transfer is called transfer $H_i^1 \to H_i^2$. The mass variation of the solution transfer is computed as

$$mv_i = \left| \int_{H_i^2} \Pi u_{2,i} - \int_{H_i^1} u_{1,i} \right|. \tag{6.1}$$

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

Table 1: Mesh sizes of the series of mesh.

| Mesh level $i$ | Vertex number of $H_i^1$ | Vertex number of $H_i^2$ |
|---|---|---|
| 1 | 9261 | 4096 |
| 2 | 68921 | 29791 |
| 3 | 531441 | 226981 |

The error of solution transfer $H_i^1 \to H_i^2$ is computed in $L^1$ -norm and $L^2$ -norm

$$\varepsilon_i^{1\to 2}|_{L^1} = \int_{H_i^2} |u_{2,i} - \Pi u_{2,i}|, \tag{6.2a}$$

$$\varepsilon_i^{1\to 2}|_{L^2} = \sqrt{\int_{H_i^2} |u_{2,i} - \Pi u_{2,i}|^2}. \tag{6.2b}$$

We can also analyze the error when the solution is re-interpolated back to $H_i^1$. That is to say, $u_{1,i}$ is first transferred from $H_i^1$ to $H_i^2$ to produce $\Pi u_{2,i}$, and then $\Pi u_{2,i}$ is transferred from $H_i^2$ to $H_i^1$ to produce $\Pi u_{1,i}$. This double solution transfer is denoted by $H_i^1 \to H_i^2 \to H_i^1$. The error $\varepsilon_i^{1\to 2\to 1}$ is obtained by computing the gap in $L^1$-norm and $L^2$-norm between $u_{1,i}$ and $\Pi u_{1,i}$ on $H_i^1$.

In Subsection 6.4, solution transfers between meshes of quite different size are performed, and in Subsection 6.5 we reconstruct the numerical solution of a practical flow filed.

## 6.1 A polynomial function

The first analytical function is a polynomial function defined as

$$u_1(x,y,z) = 4z^3 + 3y^2 + 7x + z + 5. \tag{6.3}$$

As shown in Fig. 7(a), the mass variation of the $P^1$ conservative interpolation is of the order of the machine error ($\approx 10^{-14}$). The mass variation of the classic $P^1$ interpolation is about $10^{-3}$ and it decreases as the mesh interval decreases.

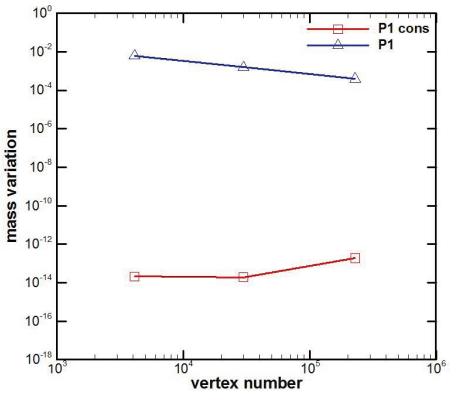
The accuracy and convergence orders of solution transfer by $P^1$ conservative interpolation and classic linear interpolation are presented in Figs. 7(b)-(e). Both $P^1$ interpolation schemes converge at the second order for the transfers $H_i^1 \to H_i^2$ and $H_i^1 \to H_i^2 \to H_i^1$. As regards the accuracy, the $P^1$ conservative interpolation is more accurate than the classic one, especially for the double transfer case.

## 6.2 A Gaussian function

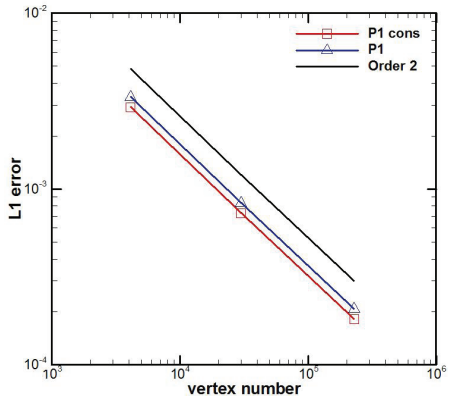
The second analytical function is a smooth Gaussian function given by

$$u_2(x,y,z) = \exp(-30(x^2 + y^2 + z^2)). \tag{6.4}$$

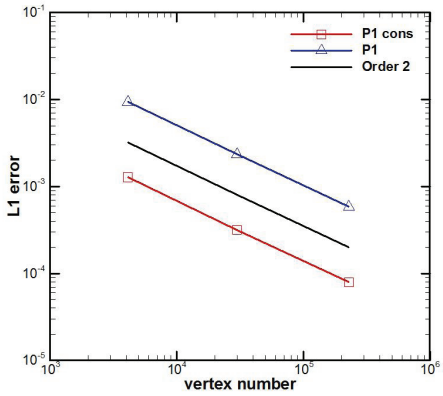
https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press



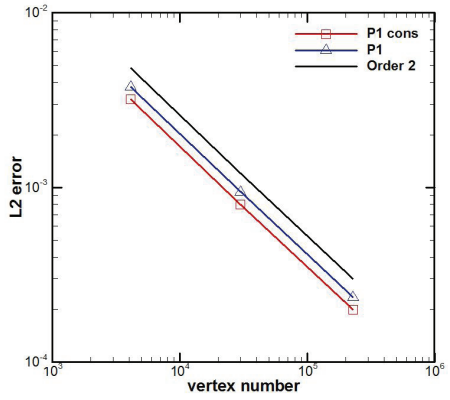

(a) Mass variation for transfer $H_i^1 \to H_i^2$



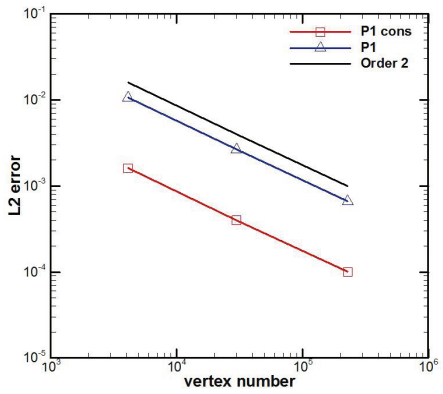

(b) $L^1$ error for transfer $H_i^1 \to H_i^2$




(c) $L^1$ error for transfer $H_i^1 \to H_i^2 \to H_i^1$



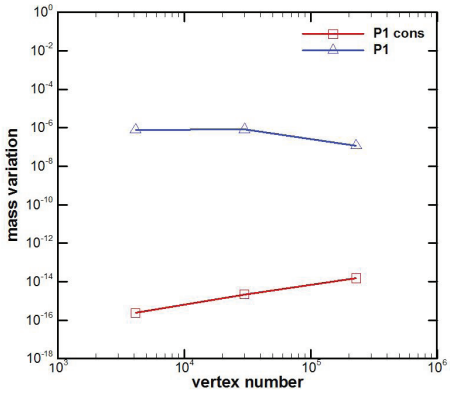

(d) $L^2$ error for transfer $H_i^1 \to H_i^2$



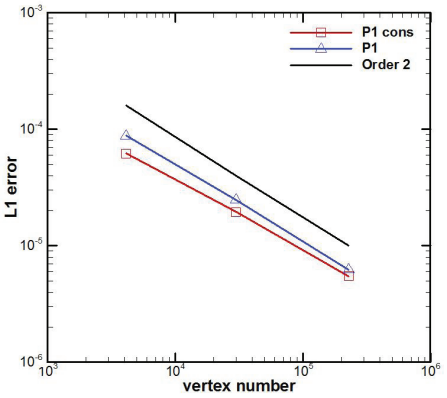

(e) $L^2$ error for transfer $H_i^1 \to H_i^2 \to H_i^1$

Figure 7: Analysis on a Polynomial function.

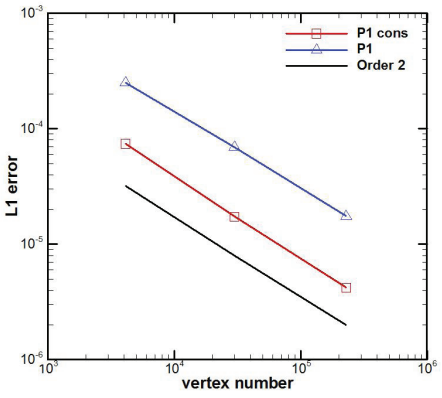
https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press



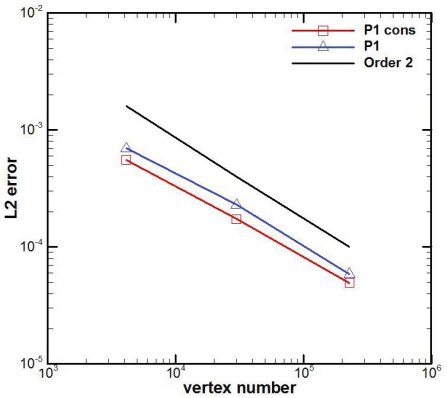

(a) Mass variation for transfer $H_i^1 \to H_i^2$



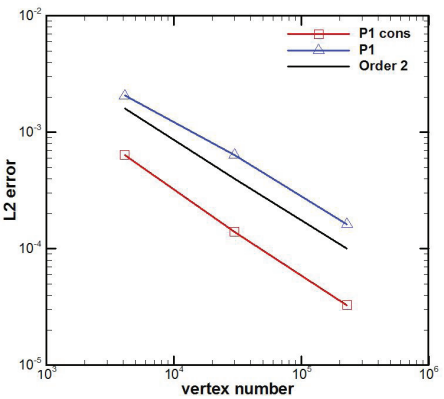

(b) $L^1$ error for transfer $H_i^1 \to H_i^2$




(c) $L^1$ error for transfer $H_i^1 \to H_i^2 \to H_i^1$




(d) $L^2$ error for transfer $H_i^1 \to H_i^2$




(e) $L^2$ error for transfer $H_i^1 \to H_i^2 \to H_i^1$

Figure 8: Analysis on a Gaussian function.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

As shown in Fig. 8(a), the mass variation of the solution transfer by classic linear interpolation is about $10^{-6}$ while the variation of the transfer by $P^1$-conservative interpolation is of the order of the machine error $(\approx 10^{-15})$.

It can be seen from Figs. 8(b)-(e) that both the conservative $P^1$ interpolation scheme and the classic $P^1$ interpolation scheme converge at the second order for the transfers $H_i^1 \to H_i^2$ and $H_i^1 \to H_i^2 \to H_i^1$ and the $P^1$ conservative interpolation is more accurate than the classic one in both cases, especially for the double transfer case.

### 6.3 A multi-scale smooth function

The third test is for a multi-scale smooth function, which presents sinusoidal variations with different scales. There exist two orders of magnitudes of small- and large-scale variations. This function is described as

$$u_3(x,y,z) = \begin{cases} 0.01\sin(50xyz), & \text{if } \left(xyz \le \dfrac{-\pi}{50}\right), \\ \sin(50xyz), & \text{if } \left(\dfrac{-\pi}{50} < xyz \le \dfrac{2\pi}{50}\right), \\ 0.01\sin(50xyz), & \text{if } \left(\dfrac{2\pi}{50} < xyz\right). \end{cases} \tag{6.5}$$

As shown in Fig. 9(a), the mass variation of the classic $P^1$ interpolation is between $10^{-2}$ and $10^{-3}$ and it decreases as mesh interval decreases. The mass variation of the $P^1$ conservative interpolation is still of the order of the machine error $(\approx 10^{-15})$ for all interpolations on different mesh levels.

The results of the accuracy and the convergence order are presented in Figs. 9(b)-(e). It can be observed that the conservative $P^1$ interpolation scheme behaves in a similar way as it does for the other two functions (6.3) and (6.4).

### 6.4 Interpolation between meshes of quite different sizes

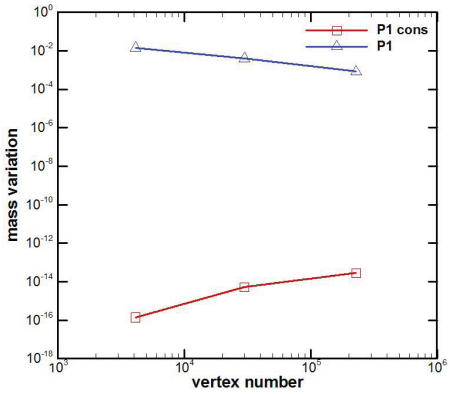
In the above numerical experiments, two sets of meshes $H_i^1$ and $H_i^2$ have the relatively close sizes for the same level. To validate the solution reconstruction method for meshes of quite different sizes, we perform the solution transfers $H_1^2 \to H_3^1$ and $H_1^2 \to H_3^1 \to H_1^2$. The mass variation and the errors computed in $L^1$-norm and $L^2$-norm of the solution transfer are given in Table 2 and Table 3.

We can see that the mass variation of the $P^1$ conservative interpolation is still of the order of the machine error. For both solution transfers $H_1^2 \to H_3^1$ and $H_1^2 \to H_3^1 \to H_1^2$, the error of the $P^1$ conservative interpolation is smaller than the classic $P^1$ interpolation.

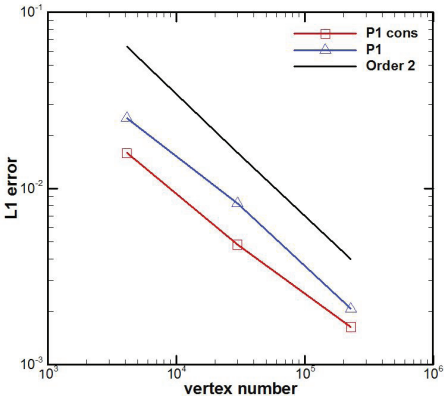
### 6.5 Flow field over a sphere

In order to verify the present method for practical problems, the flow over a stationary sphere at Reynolds number of 250 is considered. This flow is steady but asymmetric.

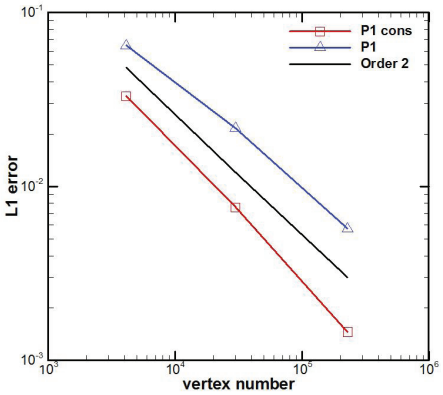
https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press



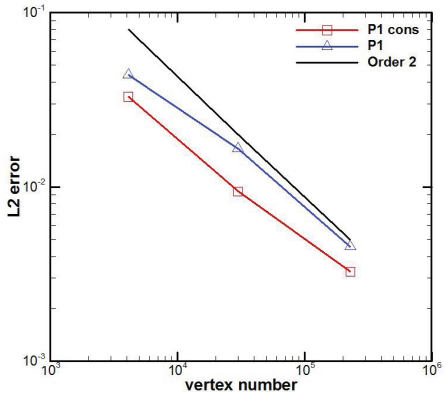

(a) Mass variation for transfer $H_i^1 \to H_i^2$



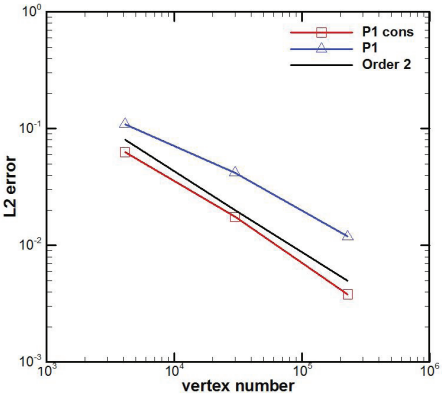

(b) $L^1$ error for transfer $H_i^1 \to H_i^2$




(c) $L^1$ error for transfer $H_i^1 \to H_i^2 \to H_i^1$




(d) $L^2$ error for transfer $H_i^1 \to H_i^2$




(e) $L^2$ error for transfer $H_i^1 \to H_i^2 \to H_i^1$

Figure 9: Analysis on a multi-scale function.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

Table 2: Analysis on a Gaussian function.

| Interpolation | $mv$ | $H_1^2 \to H_3^1$ | | $H_1^2 \to H_3^1 \to H_1^2$ | |
|---|---|---|---|---|---|
| | | $L^1$ error | $L^2$ error | $L^1$ error | $L^2$ error |
| $P^1$ | $1.58\times10^{-7}$ | $1.92\times10^{-4}$ | $1.77\times10^{-3}$ | $5.62\times10^{-5}$ | $4.41\times10^{-4}$ |
| $P^1$ cons | $4.08\times10^{-15}$ | $1.90\times10^{-4}$ | $1.76\times10^{-3}$ | $1.25\times10^{-5}$ | $1.02\times10^{-4}$ |

Table 3: Analysis on a multi-scale function.

| Interpolation | $mv$ | $H_1^2 \to H_3^1$ | | $H_1^2 \to H_3^1 \to H_1^2$ | |
|---|---|---|---|---|---|
| | | $L^1$ error | $L^2$ error | $L^1$ error | $L^2$ error |
| $P^1$ | $4.58\times10^{-4}$ | $5.77\times10^{-2}$ | $1.12\times10^{-1}$ | $1.29\times10^{-2}$ | $2.08\times10^{-2}$ |
| $P^1$ cons | $1.44\times10^{-15}$ | $5.65\times10^{-2}$ | $1.11\times10^{-1}$ | $4.53\times10^{-3}$ | $7.91\times10^{-3}$ |

The computational domain is $15D\times10D\times10D$, where $D$ is the diameter of the sphere. The unstructured tetrahedral meshes are symmetric about the $(x-y)$ plane and $(x-z)$ plane. The meshes $H^1$ and $H^2$ contain 333082 and 262152 nodes, respectively. Fig. 10 illustrates the meshes on the $(x-y)$ plane.

The streamlines of the numerical solution on $H^1$ and those of the solution reconstructed by $P^1$ conservative interpolation on $H^2$ are compared in Fig. 11. The streamlines are constructed from in-plane velocity vectors in the $(x-y)$ and $(x-z)$ planes. It can be seen that the numerical solution and the reconstructed solution are very close to each other. Fig. 12 illustrates the differences between the results of $P^1$ conservative interpolation and those of the classic $P^1$ interpolation by contour lines of the pressure and the velocity component in $x$ direction. The differences are observable although they are very slight. The mass variations of the two interpolations are given in Table 4. From this table, we can see that the mass variation of the $P^1$ conservative interpolation is much smaller than that of the classic $P^1$ interpolation.

## 7 Summary and conclusions

In this paper, we extend the algorithm of $P^1$-conservative interpolation on triangular meshes proposed by Alauzet and Mehrenberger in [5] to tetrahedral meshes for three-dimensional solution reconstruction. The algorithm is fully conservative, matrix-free, $P^1$ exact (order 2) and satisfies the maximum principle. The mass of a tetrahedron of the current mesh is calculated by the integral on its intersection with the background mesh. So it preserves the mass naturally. In an efficient way, all the background tetrahedrons overlapped by a current tetrahedron are detected. A mesh intersection algorithm is proposed to construct the intersection of a current tetrahedron with the overlapped background tetrahedron and mesh the intersection region by tetrahedrons. To avoid the loss of monotonicity caused by the solution gradient reconstruction, an approach to correct the nodal interpolated solution on tetrahedral meshes is developed for the satisfaction of the maximum principle.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

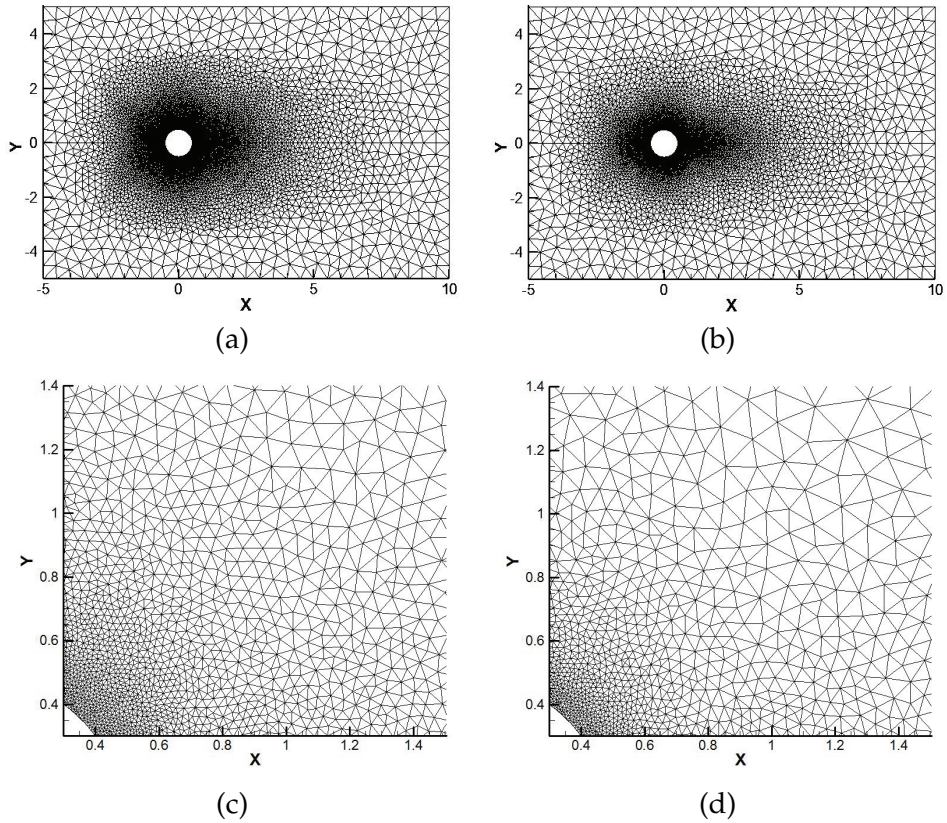



Figure 10: Illustration of meshes $H^1$ and $H^2$ on the (x-y) plane. (a): global view of $H^1$; (b): global view of $H^2$; (c): partial view of $H^1$; (d): partial view of $H^2$.

We validate the present algorithm by a series of numerical experiments. From the results of the numerical experiments on three analytical functions, we confirm that the present solution reconstruction method is fully conservative and achieves a second order of convergence. It can also be seen that the present $P^1$ conservative interpolation is a little more accurate than the classic $P^1$ interpolation. The applicability of the present solution reconstruction method for practical problems has been verified by solution transfers between meshes of quite different sizes and the reconstruction of numerical solution of the flow around a sphere.

Table 4: The mass variation of the $P^1$ conservative interpolation and the classic $P^1$ interpolation.

| Transfer | Interpolation | $mv_p$ | $mv_u$ | $mv_v$ | $mv_w$ |
|---|---|---|---|---|---|
| $H^1 \to H^2$ | $P^1$ | $5.74\times10^{-4}$ | $6.20\times10^{-2}$ | $3.56\times10^{-3}$ | $1.21\times10^{-8}$ |
| | $P^1$ cons | $6.26\times10^{-10}$ | $5.25\times10^{-8}$ | $2.23\times10^{-9}$ | $1.52\times10^{-12}$ |
| $H^2 \to H^1$ | $P^1$ | $8.46\times10^{-4}$ | $4.96\times10^{-2}$ | $2.53\times10^{-3}$ | $6.48\times10^{-7}$ |
| | $P^1$ cons | $2.98\times10^{-10}$ | $2.93\times10^{-7}$ | $4.72\times10^{-8}$ | $1.22\times10^{-10}$ |

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

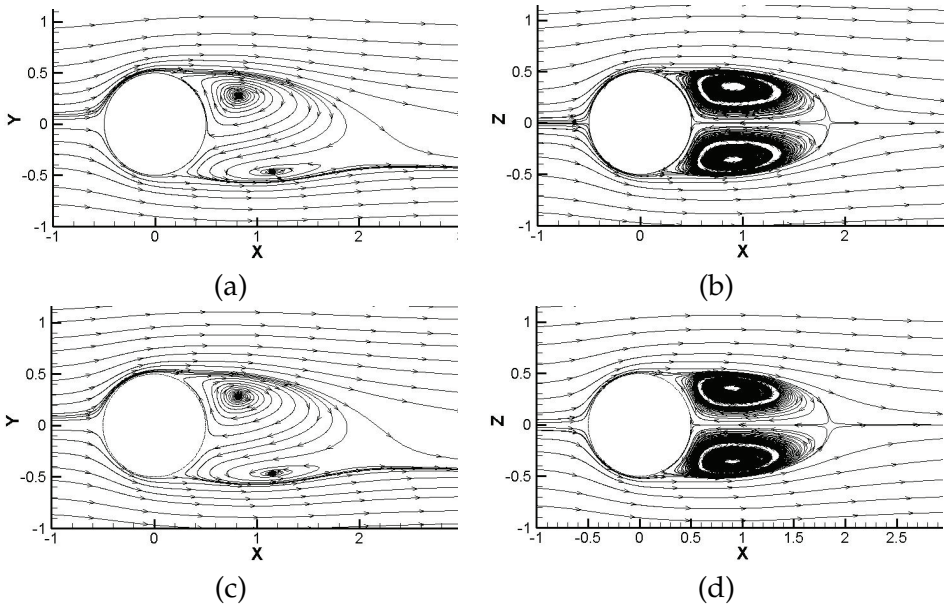



Figure 11: Streamlines of projected velocity vectors at $Re=250$. (a) and (b): numerical solution; (c) and (d): solution reconstructed by the $P^1$ conservative interpolation.

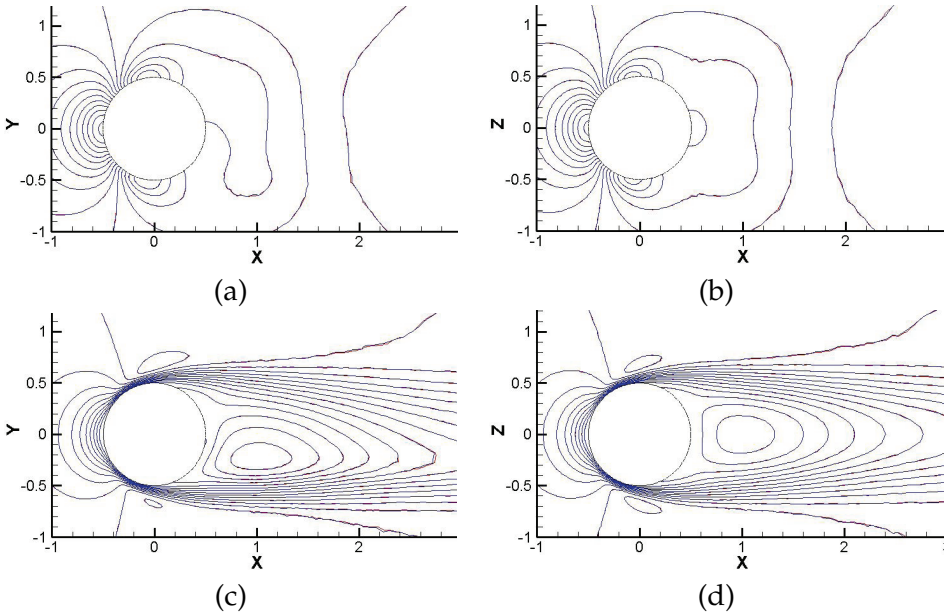



Figure 12: Contour lines of reconstructed solution. (a) and (b): pressure; (c) and (d): $x$-component of velocity; red lines: $P^1$ conservative interpolation; blue lines: classic $P^1$ interpolation.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

# Appendix

The corrected reconstruction $\tilde{u}_K$ not only satisfies maximum principle, but also

$$u_K(P_0)\le u_K(P_1)\le u_K(P_2)\le u_K(P_3)\Rightarrow \tilde{u}_K(P_0)\le \tilde{u}_K(P_1)\le \tilde{u}_K(P_2)\le \tilde{u}_K(P_3)$$

and

$$u_{\min}\le u_K(P_i)\le u_{\max}\Rightarrow \tilde{u}_K(P_i)=u_K(P_i)\quad \text{for } i=0,1,2,3.$$

The proof is given as below. Firstly, we prove that

$$u_K^M(P_0)\le u_K^M(P_1)\le u_K^M(P_2)\le u_K^M(P_3)\le u_{\max}. \tag{A.1}$$

If $u_K(P_3)\le u_{\max}$, then $u_K^M(P_i)=u_K(P_i)$ for $i=0,1,2,3$, and the inequality (A.1) holds.

Otherwise, if $u_K(P_3)>u_{\max}$, we have $u_K^M(P_3)=u_{\max}$ and then

$$u_K^M(P_2)=\min(u_K(P_2)+(u_K(P_3)-u_{\max})/3,u_{\max}).$$

Therefore, if $3u_K(P_2)+u_K(P_3)\le 4u_{\max}$, we have

$$u_K^M(P_i)=u_K(P_i)+(u_K(P_3)-u_{\max})/3\quad \text{for } i=0,1,2,$$

and (A.1) is obtained. If $3u_K(P_2)+u_K(P_3)>4u_{\max}$, we have $u_K^M(P_2)=u_{\max}$ and

$$u_K^M(P_1)=\min(u_K(P_1)+(u_K(P_2)+u_K(P_3))/2-u_{\max},u_{\max}).$$

If $2u_K(P_1)+u_K(P_2)+u_K(P_3)\le 4u_{\max}$, we thus have

$$u_K^M(P_i)=u_K(P_i)+(u_K(P_2)+u_K(P_3))/2-u_{\max}\quad \text{for } i=0,1,$$

and we get (A.1). Finally, if $2u_K(P_1)+u_K(P_2)+u_K(P_3)>4u_{\max}$, we have $u_K^M(P_1)=u_{\max}$ and $u_K^M(P_0)=4u_K(G)-3u_{\max}\le u_{\max}$. Then, the inequality (A.1) is obtained.

In a similar way, we can prove

$$u_{\min}\le \tilde{u}_K(P_0)\le \tilde{u}_K(P_1)\le \tilde{u}_K(P_2)\le \tilde{u}_K(P_3).$$

The inequality $\tilde{u}_K(P_3)\le u_{\max}$ is remained to be proved.

It is already known that $u_K^M(P_3)\le u_{\max}$. If $u_K^M(P_0)\ge u_{\min}$, we have directly $\tilde{u}_K(P_3)=u_K^M(P_3)\le u_{\max}$.

We continue the proof for the case $u_K^M(P_0)<u_{\min}$. If $3u_K^M(P_1)+u_K^M(P_0)\ge 4u_{\min}$, we have

$$\tilde{u}_K(P_3)=u_K^M(P_3)-(u_{\min}-u_K^M(P_0))/3\le u_K^M(P_3)\le u_{\max}.$$

If $3u_K^M(P_1)+u_K^M(P_0)<4u_{\min}$, we get $\tilde{u}_K(P_1)=u_{\min}$ and

$$\tilde{u}_K(P_2)=\max(u_K^M(P_2)+(u_K^M(P_0)+u_K^M(P_1))/2-u_{\min},u_{\min}).$$

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

If $2u_K^M(P_2)+u_K^M(P_1)+u_K^M(P_0)\geq 4u_{\min}$, we have

$$\tilde{u}_K(P_i)=u_K^M(P_i)+(u_K^M(P_0)+u_K^M(P_1))/2-u_{\min} \quad \text{for} \quad i=2,3.$$

Then, from $u_K^M(P_0)<u_{\min}$ and $3u_K^M(P_1)+u_K^M(P_0)<4u_{\min}$, the following inequality is true

$$(u_K^M(P_0)+u_K^M(P_1))/2<u_{\min}.$$

We thus have

$$\tilde{u}_K(P_3)=u_K^M(P_3)\leq u_{\max}.$$

Finally, if $2u_K^M(P_2)+u_K^M(P_1)+u_K^M(P_0)<4u_{\min}$, we get $\tilde{u}_K(P_2)=u_{\min}$ and $\tilde{u}_K(P_3)=4u_K(G)-3u_{\min}=u_K^M(P_0)+u_K^M(P_1)+u_K^M(P_2)+u_K^M(P_3)-3u_{\min}$. Then, from $u_K^M(P_0)<u_{\min}$ and $3u_K^M(P_1)+u_K^M(P_0)<4u_{\min}$, we obtain

$$u_K^M(P_0)+u_K^M(P_1)+u_K^M(P_2)<3u_{\min}.$$

We thus obtain

$$\tilde{u}_K(P_3)<u_K^M(P_3)\leq u_{\max}.$$

The proof of the maximum principle is finished. The last property can be easily checked from formulae (2.10a)-(2.11d).

## Acknowledgments

This work has been supported by Natural Science Foundation of China under Grant No. 11472134.

## References

[1] X. JIAO AND M. HEATH, *Common-refinement-based data transfer between non-matching meshes in multiphysics simulations*, Int. J. Numer. Methods Eng., 61(14) (2004), pp. 2402–2427.
[2] T. RINGLER AND D. RANDALL, *A potential entropy and energy conserving numerical scheme for solution of the shallow-water equations on a geodesic grid*, Monthly Weather Review, 130(5) (2002), pp. 1397–1410.
[3] J. THUBURN, *Some conservation issues for the dynamical cores of NWP and climate models*, J. Comput. Phys., 227(7) (2007), pp. 3715–3730.
[4] M. CULLEN, *Modeling atmospheric flows,* Acta Numerica, 16 (2007), pp. 67–154.
[5] F. ALAUZET AND M. MEHRENBERGER, *$P^1$-conservative solution interpolation on unstructured triangular meshes*, Int. J. Numer. Methods Eng., 84 (2010), pp. 1552–1588.
[6] F. ALAUZET, P. FREY, P. GEORGE AND B. MOHAMMADI, *3D transient fixed point mesh adaptation for time-dependent problems: application to CFD simulations*, J. Comput. Phys., 222 (2007), pp. 592–623.
[7] J. DUKOWICZ AND J. KODIS, *Accurate conservative remapping (rezoning) for arbitrary Lagrangian-Eulerian computations*, SIAM J. Sci. Statist. Comput., 8(3) (1987), pp. 305–321.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press

[8] R. GARIMELLA, M. KUCHARIK AND M. SHASHKOV, *An efficient linearity and bound preserving conservative interpolation (remapping) on polyhedral meshes*, Comput. Fluids, 36(2) (2007), pp. 224–237.
[9] L. MARGOLIN AND M. SHASHKOV, *Second-order sign-preserving conservative interpolation (remapping) on general grids*, J. Comput. Phys., 184(1) (2003), pp. 266–298.
[10] P. W. JONES, *First- and second-order conservative remapping schemes for grids in spherical coordinates*, Monthly Weather Review, 127(9) (1999), pp. 2204–2210.
[11] J. CHENG AND C. W. SHU, *A high order accurate conservative remapping method on staggered meshes*, Appl. Numer. Math., 58(7) (2008), pp. 1042–1060.
[12] J. GRANDY, *Conservative remapping and regions overlays by intersecting polyhedral*, J. Comput. Phys., 148(2) (1999), pp. 433–466.
[13] J. K. DUKOWICZ, *Conservative rezoning (remapping) for general quadrilateral meshes,* J. Comput. Phys., 54 (1984), pp. 411–424.
[14] C. GEUZAINE, B. MEYS, F. HENROTTE, P. DULAR AND W. LEGROS, *A Galerkin projection method for mixed finite elements*, IEEE Transactions on Magnetics, 35(3) (1999), pp. 1438–1441.
[15] P. FARRELL, M. PIGGOTT, C. PAIN AND G. GORMAN, *Conservative interpolation between unstructured meshes via supermesh construction*, Comput. Methods Appl. Mech. Eng., 198 (2009), pp. 2632–2642.
[16] P. E. FARRELL AND J. R. MADDISON, *Conservative interpolation between volume meshes by local Galerkin projection,* Comput. Methods Appl. Mech. Eng., 200 (2011), pp. 89–100.
[17] C. H. ZHOU AND J. Q. AI, *Mesh adaptation for simulation of unsteady flow with moving immersed boundaries*, Int. J. Numer. Methods Fluids, 72 (2013), pp. 453–477.
[18] P. FREY AND P. GEORGE, Mesh Generation, Application to Finite Elements (2nd edn), ISTE Ltd and Wiley: New York, 2008.
[19] R. LÖHNER, Applied CFD Techniques, An Introduction Based on Finite Element Methods, Wiley: New York, 2001.
[20] F. P. PREPARATA AND M. I. SHAMOS, Computational Geometry: An Introduction, Second ed., Springer, 1985.
[21] R. SEIDEL, Convex Hull Computations, in: J. E. Goodman and J. óRourke (Eds.), Handbook of Discrete and Computational Geometry, Chapman & Hall/CRC, 2004, pp. 495–512.

https://doi.org/10.4208/aamm.2015.m1087 Published online by Cambridge University Press