Commun. Comput. Phys.
doi: 10.4208/cicp.OA-2015-0021

Vol. **21**, No. 2, pp. 526-558
February 2017

Local Error Analysis and Comparison of the Swept- and Intersection-Based Remapping Methods

Matej Klima[1], Milan Kucharik[1,*] and Mikhail Shashkov[2]

[1] *Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University in Prague, Brehova 7, Praha 1, 115 19, Czech Republic.*
[2] *XCP-4 Group, MS-F644, Los Alamos National Laboratory, Los Alamos, NM 87545, USA.*

Received 12 December 2015; Accepted (in revised version) 17 July 2016

Abstract. In this paper, the numerical error of two widely used methods for remapping of discrete quantities from one computational mesh to another is investigated. We compare the intuitive, but resource intensive method utilizing intersections of computational cells with the faster and simpler swept-region-based method. Both algorithms are formally second order accurate, however, they are known to produce slightly different quantity profiles in practical applications. The second-order estimate of the error formula is constructed algebraically for both algorithms so that their local accuracy can be evaluated. This general estimate is then used to assess the dependence of the performance of both methods on parameters such as the second derivatives of the remapped distribution, mesh geometry or mesh movement. Due to the complexity of such analysis, it is performed on a set of simplified elementary mesh patterns such as cell corner expansion, rotation or shear. On selected numerical tests it is demonstrated that the swept-based method can distort a symmetric quantity distribution more substantially than the intersection-based approach when the computational mesh moves in an unsuitable direction.

PACS: 47.11.-j, 02.60.Ed

Key words: Conservative interpolation, remapping, numerical error analysis, swept regions, polygon intersections.

1 Introduction

For numerical simulations of fluid dynamics, the computational methods are typically categorized into two classes – the Eulerian and Lagrangian methods. In the pioneering work [11], the authors developed a more general framework combining best properties

*Corresponding author. *Email addresses:* klimamat@fjfi.cvut.cz (M. Klima), kucharik@newton.fjfi.cvut.cz (M. Kucharik), shashkov@lanl.gov (M. Shashkov)

 ©2017 Global-Science Press

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

of both Lagrangian and Eulerian approaches. This framework has been termed Arbitrary Lagrangian-Eulerian or ALE and since that, many authors have contributed to the investigation of its robustness, accuracy, or efficiency, see for example [1,3,10,15,24,25,29].

The ALE algorithm is usually separated in three distinct stages: (1) a Lagrangian stage, in which the fluid quantities and the computational mesh are advanced in time; (2) a rezoning stage, in which nodes of the (potentially distorted) computational mesh are moved to more optimal positions with respect to their geometrical quality; and (3) a remapping stage, in which all fluid quantities are conservatively transferred from the Lagrangian mesh to the rezoned one. The ALE algorithm preserves the advantages of the Lagrangian methods (such as low dissipation at the discontinuities or the computational mesh intrinsically following the fluid), while its Eulerian part (rezoning and remapping) prevents the computational mesh from degeneration often appearing in purely Lagrangian simulations. In this paper, we focus on the last part of the ALE algorithm – the remapping stage in single-material simulations.

In case of close computational meshes with the same topology, the remapping process can be formulated in a flux form, using fluxes of the involved quantities between cells which share the face. The fluxes are constructed by integrating the particular fluid quantity over certain transport volumes. Here, we discuss two methods of constructing such volumes [19,21,26,27]. The first method is more intuitive and employs the exact intersections of the computational cells with their neighbors in the new mesh. It is known to be more demanding in terms of computational resources, but it can, under certain circumstances, perform better in terms of solution symmetry (especially in case of discontinuous solutions and corner coupling), such as observed for example in [5,14]. The second widely used method approximates the calculated inter-cell fluxes using regions swept by the cell edges during the transformation from one mesh to another. It is robust and computationally less demanding, however, the approximation used raises concerns about its accuracy. The question is, whether it is possible to determine which method is better suited for a specific application or problem.

As far as we know, there exist two papers addressing the theoretical error analysis of the remapping methods. An error analysis based on the Fourier decomposition of the numerical error was performed in [23], showing that the faster swept-region-based method can under certain circumstances provide better results than using the intersections. Another analysis was presented earlier in [27], confirming the second order of accuracy and other properties of both methods. In the current paper, we theoretically analyze the numerical error of both methods locally, while treating the remapping process as an interpolation method rather than as a fluid flow through the computational cell edges.

However, to be able to determine which method is more suitable for specific data, an analysis of the overall accuracy for the entire computational domain is not sufficient. We need to perform an analysis of the distribution of the remapping error caused by each of the two methods. Either the fluid quantities, their derivatives (including the second derivatives), or the geometrical characteristics (such as cell volume or nodal movement

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

during the rezoning phase) can serve as parameters for the analysis.

This work extends and complements the preceding research of the hybrid remapping methods [4, 18, 20], where combination of both remapping approaches was employed depending on the presence of the material interfaces. This concept can be extended in the form of a "pseudo-hybrid" method, where the same combination of remapping approaches is used based on the local error estimate, trying to use the more accurate method in each particular computational cell. The local error analysis presented in this paper provides a theoretical justification for such combination, identifying particular terms in the error formula which are responsible for accuracy or symmetry violation of both methods.

The rest of the paper is organized as follows. In Section 2, the intersection-based and swept-region-based remapping algorithms in a flux form are overviewed and the sources of the numerical error for the swept-based algorithm are identified. In Section 3, the main part of the paper is presented. A particular remapping algorithm is analyzed in case of a general smooth function and general mesh motion. In the following subsections, several typical simplified mesh motions are analyzed in detail and the general error formula is adapted, showing which method is more suitable in such cases. In Section 4, the theoretical analysis is supported by selected numerical tests, including both static remapping and a full hydrodynamic simulation. The whole paper is concluded in Section 5.

2 Overview of swept- and intersection-based remapping algorithms

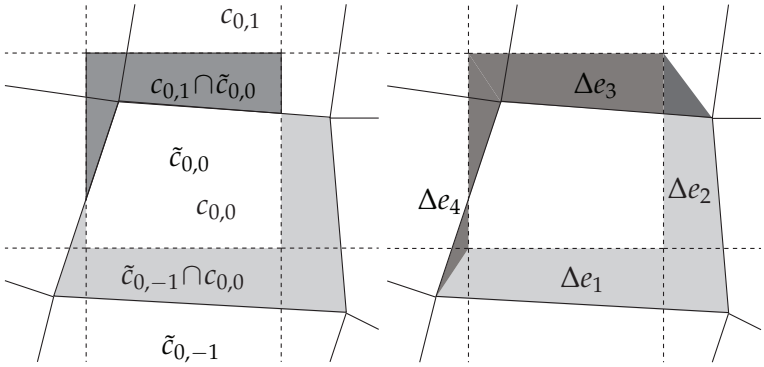
The remapping process represents essentially a conservative interpolation of a discrete quantity from one computational mesh to another. We assume that both meshes share the same topology and are close to each other so that no cell intersects any cell of the other mesh that is not in its immediate neighborhood. We will describe this process on a general polygonal mesh (although the numerical examples in this paper are shown on logically rectangular meshes). The discrete scalar quantity (such as density) is defined by the mean value in the computational cell c and will be referred to as f_c.

The remapping algorithm can be formulated in a flux form [9, 19, 21]:

$$\tilde{f}_c \tilde{V}_c = f_c V_c + \sum_{c' \in C(c)} F_{c' \to c}, \tag{2.1}$$

where $\tilde{f}_c$ is the remapped quantity and $\tilde{V}_c$ represents the new cell volume which is calculated from the known new mesh geometry. $C(c)$ is a set of all cells surrounding cell c (sharing at least one vertex with c). $F_{c' \to c}$ represents the quantity flux into cell c from its neighbor c'. The construction of the fluxes depends on the particular remapping method used. Remap defined by this formula is always conservative due to its flux nature, no matter how the fluxes are computed. In case of remapping fluid density ($f_c = \rho_c$), the formula represents remapping of mass and the fluxes $F_{c' \to c}$ represent mass fluxes between the cells.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press



(a) Exact intersections of cells (b) Regions swept by cell edges

Figure 1: Comparison of flux construction in the remapping methods, dashed line – new mesh, solid line – old mesh, ■ positive flux, ■ negative flux.

The intersection-based method can be used for remapping of fluid quantities between two arbitrary meshes. However, we only consider the case of close meshes with the same topology, allowing to express the remapped quantity in the flux form (2.1). The fluxes are calculated by integrating the reconstructed function over the intersections of the new cell polygon with each its neighbor in the old position and vice versa [19, 21], as seen in Fig. 1(a). The particular flux can be split in two terms, each representing its opposite directions (pointing inside and outside of cell c):

$$F_{c'\to c} = F_{c'\cap\tilde{c}} - F_{c\cap\tilde{c}'}, \tag{2.2}$$

where $\tilde{c}$ represents the region enclosed by the cell c in its new position. Each of these terms can be approximated by integrating a function reconstructed from the discrete values of f_c over the intersecting polygon,

$$F_{c'\cap\tilde{c}} = \int_{c'\cap\tilde{c}} f^{\text{rec}}_{c'}(x,y)\,dV. \tag{2.3}$$

In the second-order method presented here, a piecewise linear reconstruction is computed in the intersecting region:

$$f^{\text{rec}}_{c'}(x,y) = f_{c'} + \left(\frac{\partial f}{\partial x}\right)_{c'}(x-x_{c'}) + \left(\frac{\partial f}{\partial y}\right)_{c'}(y-y_{c'}), \tag{2.4}$$

where $x_{c'}$, $y_{c'}$ are the coordinates of the geometrical centroid of the respective cell,

$$x_{c'} = \frac{\int_{c'} x\,dV}{\int_{c'} 1\,dV}, \quad y_{c'} = \frac{\int_{c'} y\,dV}{\int_{c'} 1\,dV}. \tag{2.5}$$

The function derivatives are estimated numerically using a least squares minimization method with the Barth-Jespersen limiter [2, 17, 28]. Calculation of the fluxes then represents calculation of a first degree polynomial over a polygonal region, that can be evaluated analytically.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

The difficult part is to calculate the intersecting polygon coordinates accurately. In general, the design of a numerical polygon intersection algorithm is a complex issue out of the scope of this paper. Most often in case of mesh smoothing, the differences between the original and the rezoned computational meshes are very small. Therefore, the algorithm must be robust enough to handle regions with almost degenerate shapes and close-to-parallel edges [21].

In the swept-based algorithm, the fluxes are not calculated by the exact integration as in (2.3) in each direction separately (2.2), instead they are approximated by only one flux per edge:

$$\tilde{f}_c \tilde{V}_c = f_c V_c + \sum_{e \in E(c)} \int_{\Delta e} f^{\text{rec}}_{c(\Delta e)}(x,y)\, dV, \tag{2.6}$$

where $E(c)$ is a set of all edges of cell c and the integration is performed over the regions swept by each edge Δe (shown in Fig. 1(b)) during the transition from the old to the new computational mesh. The corner fluxes are not explicitly treated in this approach, they are approximated together with the adjacent cell fluxes [27]. $c(\Delta e)$ represents the cell from which the reconstruction is taken – either c or the neighboring cell sharing the edge e, depending on the sign of the oriented volume integral over c, which also determines the direction of the particular flux. The swept region is defined by the old and new positions of the cell vertices so an intersection algorithm is not needed. This makes the method very robust and fast. However, concerns about its accuracy exist. First, this method does not account for the corner fluxes – their contributions are approximated by the reconstruction from the adjacent edge-neighboring cells. This can be seen for example in Fig. 1(b) in the lower right corner of the cell. And second, if the cell edge of the new mesh intersects the original edge, there are two fluxes in the intersection-based approach with opposite directions, each with its own reconstructed function. In this situation, the swept-region method uses only one of the reconstructions which cannot be chosen properly, since it will not be accurate in the other part of the self-intersecting region (the "hourglass" case in Fig. 1(b), as shown at the left edge of the depicted cell). This effect is most significant when the swept region volume approaches zero. Let us note that this effect can be improved by decomposing the swept region to two triangles in this case (requiring only finding an intersections of two straight lines), such as described in [12]. However, the first effect of inaccurate reconstruction in the corner regions remains as full intersection of the swept region with the old mesh [20] is needed for correct reconstruction, which is just a different formulation of the intersection-based approach.

3 Estimating the local remapping error

In most cases, the accuracy of the remapping method utilizing swept regions integration is comparable to integrating the exact cell intersections [4, 20, 22, 27]. The swept-based method is, however, much simpler, robust and computationally more efficient; therefore it is often the first choice for single-material quantity remapping. Albeit in some

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

cases where symmetry preservation is crucial (eg. when remapping a radially symmetric quantity distribution), the exact method is known to yield better results. To design some criteria for selecting either intersections or swept regions, we need to know how the method error is influenced by the quantity distribution and the computational mesh.

In our analysis, we do not treat the remapping process as an advection scheme, but rather as an interpolation method. That way we can focus on the local remapping error on the level of individual cells. Often, the swept-based method distorts the solution only in a small portion of the cells in the computational mesh, so knowing where the exact intersection method is locally more accurate can be beneficial.

We assume that the discrete quantity f_c represents the mean value of a certain analytical function f in the cell c. This underlying function and its properties are not known in general, but for the purpose of this analysis we presume that this function is continuous and differentiable. Although this approximation may be less accurate for initially discontinuous distributions, it needs to be emphasized that we are making an independent local approximation for each computational cell that is valid only within the cell itself. As such, it will converge with increasing mesh resolution. We use the second-order Taylor expansion in the cell centroid for the purpose of local error estimation:

$$f_c = \frac{1}{V_c}\int_c \left(f(\mathbf{r_c}) + \nabla f(\mathbf{r_c})\cdot(\mathbf{r}-\mathbf{r_c}) + \frac{1}{2}(\mathbf{r}-\mathbf{r_c})^{\mathrm{T}}\mathbf{H}(\mathbf{r}-\mathbf{r_c}) + \mathcal{O}(\mathbf{r}^3) \right) dV, \tag{3.1}$$

where $\mathbf{r}$ is the general position vector, $\mathbf{r_c}$ is the location of the cell centroid and $\mathbf{H}$ is the Hessian matrix of the function f (matrix of its second derivatives). As we assume a second-order approximation, $\mathbf{H}$ is treated as constant in the selected cell and its neighbors – we do not need to evaluate it separately at a certain point. The linear part of this function is reduced to the function value evaluated at the cell centroid after integration:

$$f_c = f(\mathbf{r_c}) + \frac{1}{2V_c}\int_c (\mathbf{r}-\mathbf{r_c})^{\mathrm{T}}\mathbf{H}(\mathbf{r}-\mathbf{r_c}) + \mathcal{O}(\mathbf{r}^3)\, dV. \tag{3.2}$$

The integral in this formula can be analytically evaluated for a polygonal region, yielding a rather complicated result. We will further refer to this integral as:

$$I_c^{\mathbf{H}}(\mathbf{r_c}) = \int_c (\mathbf{r}-\mathbf{r_c})^{\mathrm{T}}\mathbf{H}(\mathbf{r}-\mathbf{r_c})\, dV. \tag{3.3}$$

This integral can be evaluated by expanding the quadratic form:

$$I_c^{\mathbf{H}}(\mathbf{r_c}) = \int_c \mathbf{r}^{\mathrm{T}}\mathbf{H}\mathbf{r} - 2\mathbf{r}^{\mathrm{T}}\mathbf{H}\mathbf{r_c} + \mathbf{r_c}^{\mathrm{T}}\mathbf{H}\mathbf{r_c}\, dV, \tag{3.4}$$

due to the fact that the Hessian is symmetric. The Hessian can be diagonalized in the following way:

$$I_c^{\mathbf{H}}(\mathbf{r_c}) = \int_c \mathbf{r}^{\mathrm{T}}\mathbf{P}\mathbf{\Lambda}\mathbf{P}^{\mathrm{T}}\mathbf{r} - 2\mathbf{r}^{\mathrm{T}}\mathbf{H}\mathbf{r_c} + \mathbf{r_c}^{\mathrm{T}}\mathbf{H}\mathbf{r_c}\, dV, \tag{3.5}$$

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

where the transformation matrix $\mathbf{P}$ is composed of the principal directions (ie. the eigenvectors of $\mathbf{H}$) as columns. In the first term, we can rotate the coordinates using the transformation $\mathbf{u}=\mathbf{P}^{\mathrm{T}}\mathbf{r}$ to simplify the integration. $\mathbf{P}$ can be considered a rotation matrix without loss of generality. The second integrand is a linear function and the third one a constant. The formula can be expressed as:

$$I_c^{\mathbf{H}}(\mathbf{r_c})=\int_c \mathbf{u}^{\mathrm{T}}\mathbf{\Lambda u}dV+\sum_{i=0}^{n}\begin{vmatrix}x_i & x_{i+1}\\ y_i & y_{i+1}\end{vmatrix}\left(\frac{\mathbf{r_c}}{2}-\frac{\mathbf{r}_i+\mathbf{r}_{i+1}}{3}\right)^{\mathrm{T}}\mathbf{H}\mathbf{r_c}, \tag{3.6}$$

where i is the index of the closed polygon vertices and $\mathbf{r}_i$ is the vertex position ($\mathbf{r}_0=\mathbf{r}_n$). The integral now consists of quadratic terms only and can be evaluated on a polygon in the following way:

$$\int_c \mathbf{u}^{\mathrm{T}}\mathbf{\Lambda u}dV=\frac{1}{12}\sum_{i=0}^{n}\begin{vmatrix}u_{x,i} & u_{x,i+1}\\ u_{y,i} & u_{y,i+1}\end{vmatrix}\begin{pmatrix}u_{x,i}^2+u_{x,i}u_{x,i+1}+u_{x,i+1}^2\\ u_{y,i}^2+u_{y,i}u_{y,i+1}+u_{y,i+1}^2\end{pmatrix}\cdot\begin{pmatrix}k_1\\ k_2\end{pmatrix}, \tag{3.7}$$

where $u_{x,i}$ is the x coordinate of the i-th vertex in the transformed coordinate system. k_1,k_2 are the principal curvatures – the eigenvalues of $\mathbf{H}$. We can now substitute back for $\mathbf{u}$:

$$\int_c \mathbf{r}^{\mathrm{T}}\mathbf{H}\mathbf{r}dV=\frac{1}{12}\sum_{i=0}^{n}\begin{vmatrix}x_i & x_{i+1}\\ y_i & y_{i+1}\end{vmatrix}\left(\mathbf{r}_i^{\mathrm{T}}\mathbf{H}\mathbf{r}_i+\mathbf{r}_i^{\mathrm{T}}\mathbf{H}\mathbf{r}_{i+1}+\mathbf{r}_{i+1}^{\mathrm{T}}\mathbf{H}\mathbf{r}_{i+1}\right) \tag{3.8}$$

and use this result in (3.6). After simplification, we obtain:

$$\begin{aligned}I_c^{\mathbf{H}}(\mathbf{r_c})=\frac{1}{12}\sum_{i=0}^{n}\|\mathbf{r_i}\times\mathbf{r_{i+1}}\|^2\Big((\mathbf{r_i}-\mathbf{r_c})^{\mathrm{T}}\mathbf{H}(\mathbf{r_i}-\mathbf{r_c})\\ +(\mathbf{r_i}-2\mathbf{r_c})^{\mathrm{T}}\mathbf{H}(\mathbf{r_{i+1}}-2\mathbf{r_c})+(\mathbf{r_{i+1}}-\mathbf{r_c})^{\mathrm{T}}\mathbf{H}(\mathbf{r_{i+1}}-\mathbf{r_c})\Big).\end{aligned} \tag{3.9}$$

This formula has been verified numerically for selected functions and mesh shapes and holds exactly.

3.1 Remapping error of an arbitrary single flux

The developed formula (3.9) cannot be analyzed for a general mesh motion and general function. Here, we show how to construct the error formula in the case when only one non-zero flux is present due to a special mesh movement, and in the next sections, we construct the particular error formulas for the swept-based and intersection-based methods in cases of certain elementary motion patterns.

As we can now evaluate the integral of the quadratic form, we can express the remapping error in case of a single flux by comparing the remapped value with the mean value of the analytical function in the new cell:

$$\epsilon_{\text{remap}}=\tilde{f}_c-\frac{1}{\tilde{V}_c}\int_{\tilde{c}}f(\mathbf{r})dV=\frac{1}{\tilde{V}_c}\left(f_cV_c+\int_F f_{c'}^{\text{rec}}(\mathbf{r})dV-\int_{\tilde{c}}f(\mathbf{r})dV\right), \tag{3.10}$$

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

where F is the region of the flux (in this simplified case, the swept- and intersection-based methods produce the same results), c and $\tilde{c}$ represent the remapped cell in its old and new positions, respectively. c' is the neighboring cell from which the reconstruction is obtained. The integration region of the new cell can be split into two parts, namely $\tilde{c}=c\cup F$, so the cell term cancels out:

$$\epsilon_{\text{remap}}=\frac{1}{\tilde{V}_c}\left(\int_F f_{c'}^{\text{rec}}(\mathbf{r})dV-\int_F f(\mathbf{r})dV\right). \tag{3.11}$$

We can now substitute for $f_{c'}^{\text{rec}}$ from (2.4) with an assumption that the numerical first derivative is exact (the methods used to calculate it may vary and comparing them is out of the scope of this paper). If we replace $f(\mathbf{r})$ with its second-order Taylor expansion in $\mathbf{r}_{c'}$, we obtain:

$$\epsilon_{\text{remap}}=\frac{1}{\tilde{V}_c}\left(\int_F f_{c'}-f(\mathbf{r}_{c'})-\frac{1}{2}(\mathbf{r}-\mathbf{r}_{c'})^{\mathrm{T}}\mathbf{H}(\mathbf{r}-\mathbf{r}_{c'})dV\right). \tag{3.12}$$

Note that the mean value $f_{c'}$ is not equal to the function value in the centroid. For $f_{c'}$, we can substitute from (3.2) and simplify the integrals as:

$$\epsilon_{\text{remap}}=\frac{V_F}{2\tilde{V}_c}\left(\frac{I_{c'}^{\mathbf{H}}(\mathbf{r}_{c'})}{V_{c'}}-\frac{I_F^{\mathbf{H}}(\mathbf{r}_{c'})}{V_F}\right), \tag{3.13}$$

where V_F is the volume of the flux region. Both terms here represent the mean value of the second-order approximation of the remapping error. They differ only in the integration region, the first one being the consequence of the approximate mean value of f in c' used in the reconstruction. The second one represents the error of the reconstruction in the flux region.

We would like to know also the sign of the remapping error. Generally, comparing two mean values of a certain function over a non-specific region is difficult. There are, however, some conditions that are sufficient to determine the sign:

1. $|\mathbf{H}|\geq 0;\ \mathbf{H}_{xx}+\mathbf{H}_{yy}>0;\ \frac{I_{c'}^{\mathbf{H}}(\mathbf{r}_{c'})}{V_{c'}}\leq(\mathbf{r_F}-\mathbf{r_{c'}})^T\mathbf{H}(\mathbf{r_F}-\mathbf{r_{c'}})\ \Rightarrow\ \epsilon_{\text{remap}}\leq 0,$
2. $|\mathbf{H}|\geq 0;\ \mathbf{H}_{xx}+\mathbf{H}_{yy}<0;\ \frac{I_{c'}^{\mathbf{H}}(\mathbf{r}_{c'})}{V_{c'}}\geq(\mathbf{r_F}-\mathbf{r_{c'}})^T\mathbf{H}(\mathbf{r_F}-\mathbf{r_{c'}})\ \Rightarrow\ \epsilon_{\text{remap}}\geq 0,$
3. $|\mathbf{H}|<0\ \cdots$ inconclusive.

Here, $\mathbf{r_F}$ represents the geometrical centroid of the flux region. Its value is known, because its calculation is performed in both remapping methods. The first and second conditions correspond to the case of a positive or negative semi-definite Hessian matrix, respectively. In the first case, we can make a lower estimate of $I_F^{\mathbf{H}}(\mathbf{r}_{c'})/V_F$ over the region of the flux by integrating the tangential plane of the error function (an elliptical paraboloid) instead. As we construct the plane in $\mathbf{r_F}$, this integral is equal to the value of

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

the error function in the centroid. This can be applied in a similar manner in the case of a negative semi-definite Hessian matrix. Note that the opposite statement is not generally valid, but on the other hand, it can be used as an approximation of the error even for an indefinite Hessian matrix ($|\mathbf{H}|<0$):

$$\epsilon_{\text{remap}} \sim \frac{V_F}{2\tilde{V}_c}\left(\frac{I^{\mathbf{H}}_{c'}(\mathbf{r}_{c'})}{V_{c'}} - (\mathbf{r_F}-\mathbf{r_{c'}})^T\mathbf{H}(\mathbf{r_F}-\mathbf{r_{c'}})\right). \tag{3.14}$$

In this approximation we assume that the dimensions of the flux region are small when compared to the size of the cell c'. Replacing the integral over the flux region with the value in the centroid can be useful in applications where it might be inconvenient to calculate the integrals for every flux.

In the next sections, we focus on several elementary patterns present during the rezoning process. For an overview of the mesh motion patters, see [7]. More complex error terms are derived using the single flux error expression (3.13). Then, we analyze the numerical error of the intersection and swept-based approaches in such simplified cases.

3.2 Remapping error of a corner movement

The first case in which the swept- and intersection-based remapping methods differ is the movement of the corner of one computational cell. In this section we describe a general case where the movement of the upper-right vertex is arbitrary. The other two neighboring vertices can move arbitrarily along the edges so that the intersection-based algorithm finds non-zero intersections only with cells $c_{0,1}$,$c_{1,1}$ and $c_{1,0}$. The swept-based algorithm treats this case as only two swept regions and uses the function reconstruction from cells $c_{0,1}$ and $c_{1,0}$, respectively. Using (3.13) for each flux, we can write the error of the swept-based method as follows:

$$\begin{aligned}\epsilon_{\text{sw}} = \frac{1}{2\tilde{V}_{c_{0,0}}}\Bigg(&\frac{V_{F_{\text{N}}+F_{\text{NE1}}}}{V_{c_{0,1}}} I^{\mathbf{H}}_{c_{0,1}}(\mathbf{r}_{c_{0,1}}) - I^{\mathbf{H}}_{F_{\text{N}}+F_{\text{NE1}}}(\mathbf{r}_{c_{0,1}}) \\ &+ \frac{V_{F_{\text{E}}+F_{\text{NE2}}}}{V_{c_{1,0}}} I^{\mathbf{H}}_{c_{1,0}}(\mathbf{r}_{c_{1,0}}) - I^{\mathbf{H}}_{F_{\text{E}}+F_{\text{NE2}}}(\mathbf{r}_{c_{1,0}})\Bigg),\end{aligned} \tag{3.15}$$

where F_{N}, F_{E}, F_{NE1}, and F_{NE2} represent the flux regions and $c_{i,j}$ are the corresponding cells (see Fig. 2). The error of the intersection-based method can be described in a similar way:

$$\begin{aligned}\epsilon_{\text{int}} = \frac{1}{2\tilde{V}_{c_{0,0}}}\Bigg(&\frac{V_{F_{\text{N}}}}{V_{c_{0,1}}} I^{\mathbf{H}}_{c_{0,1}}(\mathbf{r}_{c_{0,1}}) - I^{\mathbf{H}}_{F_{\text{N}}}(\mathbf{r}_{c_{0,1}}) + \frac{V_{F_{\text{NE1}}+F_{\text{NE2}}}}{V_{c_{1,1}}} I^{\mathbf{H}}_{c_{1,1}}(\mathbf{r}_{c_{1,1}}) \\ &- I^{\mathbf{H}}_{F_{\text{NE1}}+F_{\text{NE2}}}(\mathbf{r}_{c_{1,1}}) + \frac{V_{F_{\text{E}}}}{V_{c_{1,0}}} I^{\mathbf{H}}_{c_{1,0}}(\mathbf{r}_{c_{1,0}}) - I^{\mathbf{H}}_{F_{\text{E}}}(\mathbf{r}_{c_{1,0}})\Bigg).\end{aligned} \tag{3.16}$$

Now, we would like to express the difference between the two errors to analyze, which method is more accurate:

$$\Delta\epsilon = |\epsilon_{\text{int}}| - |\epsilon_{\text{sw}}|, \quad |\Delta\epsilon| \le |\epsilon_{\text{int}} - \epsilon_{\text{sw}}|. \tag{3.17}$$

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

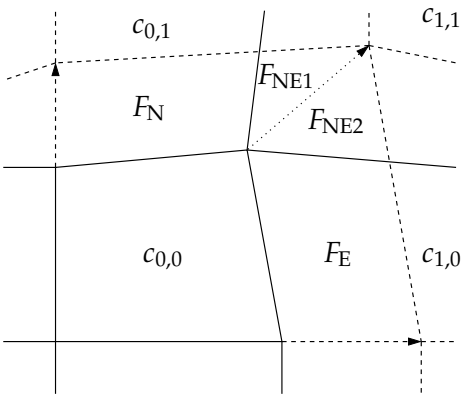


Figure 2: Upper right corner movement, dashed line – new mesh, solid line – old mesh.

Unfortunately, the signs of both errors cannot be determined easily for a general Hessian matrix and cell geometry. We can either directly evaluate both terms or use some approximation of the involved integrals (eg. (3.14)). In this case (distorted initial mesh), the complete analysis is impossible and we only make an upper estimate of the error magnitude. For certain simplified mesh patterns (such as the one presented in the next Section), it is possible to evaluate the error difference exactly and study the influence of the function and movement parameters on the error difference. Then, it is possible to explicitly identify, which method has a lower numerical error for given mesh and function parameters. In our case described in Fig. 2, we can at least make an upper estimate of the error difference magnitude:

$$|\Delta e| \leq \frac{1}{2\tilde{V}_{c_{0,0}}} \left| \frac{V_{F_{\mathrm{NE1}}+F_{\mathrm{NE2}}}}{V_{c_{1,1}}} I^{\mathbf{H}}_{c_{1,1}}(\mathbf{r}_{c_{1,1}}) - \frac{V_{F_{\mathrm{NE1}}}}{V_{c_{0,1}}} I^{\mathbf{H}}_{c_{0,1}}(\mathbf{r}_{c_{0,1}}) - \frac{V_{F_{\mathrm{NE2}}}}{V_{c_{1,0}}} I^{\mathbf{H}}_{c_{1,0}}(\mathbf{r}_{c_{1,0}}) \right.$$
$$\left. - I^{\mathbf{H}}_{F_{\mathrm{NE1}}+F_{\mathrm{NE2}}}(\mathbf{r}_{c_{1,1}}) + I^{\mathbf{H}}_{F_{\mathrm{NE1}}}(\mathbf{r}_{c_{0,1}}) + I^{\mathbf{H}}_{F_{\mathrm{NE2}}}(\mathbf{r}_{c_{1,0}}) \right|. \tag{3.18}$$

If the cells of the old computational mesh have similar geometry, the integrals over cells cancel out. The remaining terms can be approximated using a similar approach as in (3.14). This way, we obtain a simple formula for estimating the error difference magnitude which can show us whether the remapping method choice is important or not.

It is to be noted that in the opposite case of cell corner compression, ie. all its edges moving inwards, both methods are equivalent because all reconstructions are made using values from the cell itself. Therefore, it is not important whether we use a swept- or intersection-based representation of the fluxes in such case.

In a more general situation of a polygonal mesh, there can be a different number of cells sharing the corner vertex. This will result in similar error difference terms containing $N_C - 2$ corner triangles, where N_C is the number of such cells. In a similar manner, we can describe the situation where there are two swept regions with a triangular overlapping area, only the signs of the fluxes will be different.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

3.3 Simplified orthogonality-preserving diagonal corner movement

To see the effects of mesh movement on the remapping error in more details, we reduce the above-presented situation so that the affected horizontal and vertical grid-lines move equally by the distance d in the directions of the axes (see Fig. 3). This simplified mesh movement corresponds to the tensor product mesh motion often used for testing of remapping methods [27]. The old mesh is equidistant and has square cells with the edge length l.

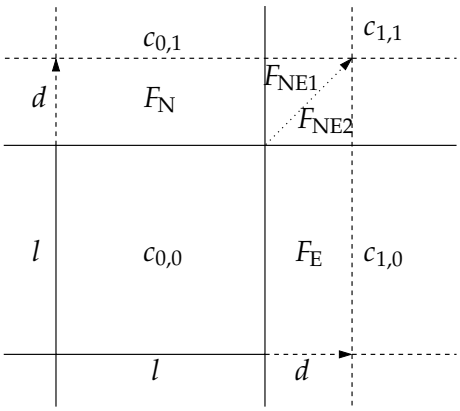


Figure 3: Diagonal movement of orthogonal corner, dashed line – new mesh, solid line – old mesh.

The difference between the methods is dependent on the area of the corner flux regions $V_{F_{\mathrm{NE1}}+F_{\mathrm{NE2}}}$ but also on other factors. However, here we have the possibility to investigate the dependence of the remapping error on a limited set of parameters, namely d, l, and the elements of $\mathbf{H}$. After substituting into (3.15), (3.16) and integrating, we obtain the following remapping error terms:

$$\epsilon_{\mathrm{sw}} = -\frac{d}{12\tilde{V}_{c_{0,0}}}\left[(2d^3+ld^2-2dl^2+l^3)(\rho_{xx}+\rho_{yy})+(3d^3+2ld^2-3dl^2)\rho_{xy}\right], \tag{3.19}$$

$$\epsilon_{\mathrm{int}} = -\frac{d}{12\tilde{V}_{c_{0,0}}}\left[(2d^3-ld^2-2dl^2+l^3)(\rho_{xx}+\rho_{yy})+(3d^3-6ld^2+3dl^2)\rho_{xy}\right], \tag{3.20}$$

where ρ_{xx}, ρ_{xy}, ρ_{yy} are the respective second derivatives, the elements of $\mathbf{H}$. These formulas can be simplified by rotating the coordinate system by $\pi/4$, which corresponds to the axis of symmetry for this problem:

$$x' = \frac{1}{\sqrt{2}}(x+y), \quad y' = \frac{1}{\sqrt{2}}(y-x). \tag{3.21}$$

We can transform the second derivatives similarly by multiplying them by the Jacobian matrix of the transformation,

$$\rho_{x'x'} = \frac{1}{2}(\rho_{xx}+\rho_{yy})+\rho_{xy}, \quad \rho_{x'y'} = \frac{1}{2}(\rho_{xx}-\rho_{yy}), \quad \rho_{y'y'} = \frac{1}{2}(\rho_{xx}+\rho_{yy})-\rho_{xy}. \tag{3.22}$$

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

After substituting the inverse transformation into (3.19), (3.20), we can see that the mixed derivative term cancels out and the matrix of second derivatives **H** has only two non-zero components. One in the direction of the movement and the other in the perpendicular direction. The relation between these two components plays the key role in determining which method produces larger remapping error. The error terms can now be expressed as

$$\epsilon_{\text{sw}} = -\frac{d}{24\tilde{V}_{c_{0,0}}}\left[(7d^3+4ld^2-7dl^2+2l^3)\rho_{x'x'}+(d^3-dl^2+2l^3)\rho_{y'y'}\right], \tag{3.23}$$

$$\epsilon_{\text{int}} = -\frac{d}{24\tilde{V}_{c_{0,0}}}\left[(7d^3-8ld^2-dl^2+2l^3)\rho_{x'x'}+(d^3+4ld^2-7dl^2+2l^3)\rho_{y'y'}\right]. \tag{3.24}$$

It is to be noted that this rotation of coordinates in the direction of movement is not useful for an arbitrary mesh movement. The mixed derivative is canceled out due to the symmetric initial conditions, in a general case there are many more factors that will affect the remapping error structure.

We can simplify both terms by dividing them by $K_1 = -dl^3\rho_{x'x'}/(24\tilde{V}_{c_{0,0}})$ and substituting $g = \rho_{y'y'}/\rho_{x'x'}$ as the ratio of the second derivative components and $t = d/l$ as the movement relative to the cell size. Due to the remapping stability condition [27] in this case the constraints of t are $0 < t < 0.5$. The derivative ratio g can be seen as a measure of eccentricity of the local second-order error. To determine which method performs better we can evaluate the error difference (3.17):

$$\begin{aligned}\Delta\epsilon = |K_1|\big(&|g(t^3-t+2)+7t^3+4t^2-7t+2| \\ &-|g(t^3+4t^2-7t+2)+7t^3-8t^2-t+2|\big).\end{aligned} \tag{3.25}$$

The sign of $\Delta\epsilon$ indicates which method performs better – this is shown in Fig. 4. We can see that for $|\rho_{y'y'}/\rho_{x'x'}| > 1$, the error difference $\Delta\epsilon < 0$ and thus the intersection method is more accurate than the swept region method in this area.

Concerning the magnitude of the error, we can approximate the error difference using the triangle inequality for the intersection error term and cancel the common terms out:

$$|\Delta\epsilon| \le \frac{d^2 l}{12\tilde{V}_{c_{0,0}}}\left|(3l-6d)\rho_{x'x'}+(2d-3l)\rho_{y'y'}\right|. \tag{3.26}$$

This can be also expressed in terms of total corner flux volume $V_{F_{\text{NE}}} = d^2$ and old cell volume $V_{c_{0,0}} = l^2$. After substituting $2ld = \tilde{V}_{c_{0,0}} - V_{c_{0,0}} - V_{F_{\text{NE}}}$, we obtain

$$|\Delta\epsilon| \le \frac{V_{F_{\text{NE}}}}{12V_{\tilde{C}}}\left|3(V_{F_{\text{NE}}}+2V_{c_{0,0}}-\tilde{V}_{c_{0,0}})\rho_{x'x'}-(\tilde{V}_{c_{0,0}}-4V_{c_{0,0}}-V_{F_{\text{NE}}})\rho_{y'y'}\right|. \tag{3.27}$$

This form of the error approximation by volumes only is useful because a corresponding formula can be applied also to non-rectangular cells, although the validity of this estimate for the general case cannot be easily verified. The dimension is that of mass. To obtain a dimensionless relative error difference estimate, this formula has to be divided by the cell mass.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

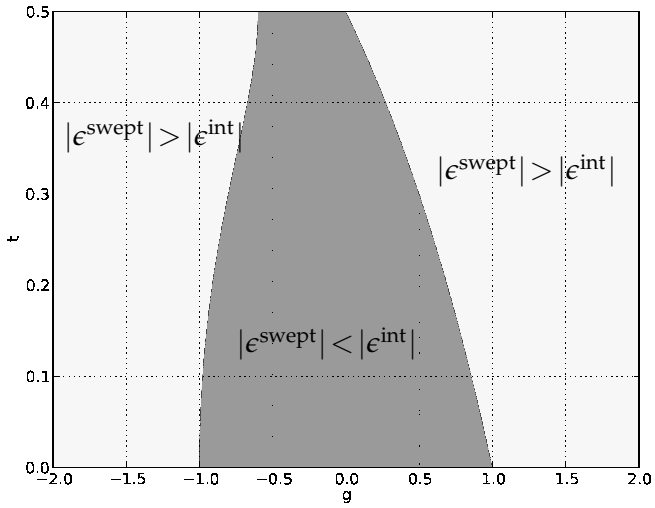


Figure 4: Two grid-lines motion – method preference. Different colors show regions where each method is more accurate.

3.4 Simplified diagonal corner movement - one vertex only

This type of movement concerns of a single node of a computational cell – the upper right vertex of the middle cell is moved in the diagonal direction, as shown in Fig. 5. This corresponds to the correction of the position of a single node, which is often encountered in mesh rezoning algorithms. It is expected to be more difficult to analyze, because not all edges remain orthogonal after the mesh rezoning and the resulting formulas are more complicated. Again, we evaluate the remapping error for the mesh configuration described here. Surprisingly, both errors can be simplified by using the diagonalization

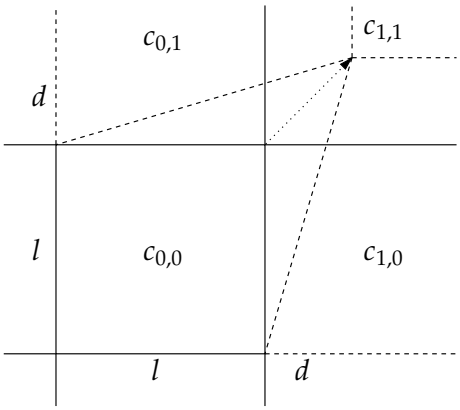


Figure 5: Diagonal one point corner movement, dashed line – new mesh, solid line – old mesh.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

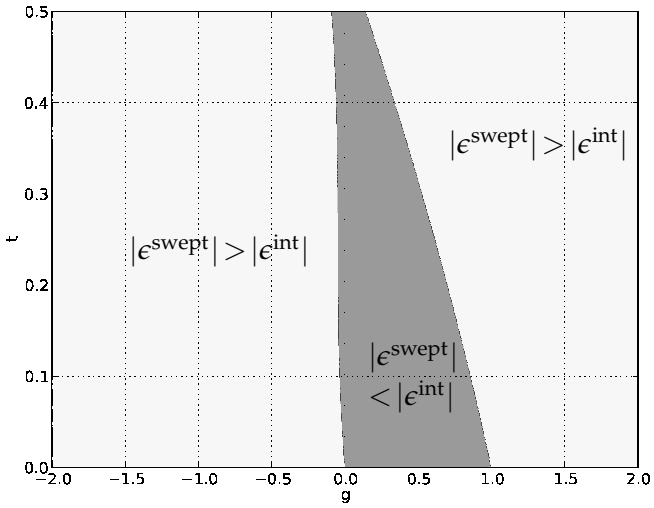


Figure 6: Single vertex movement – method preference. Different colors show regions where each method is more accurate.

process as described in (3.22):

$$\epsilon_{\text{sw}} = -\frac{d}{12(l+d)}\left[(2d^2-ld)\rho_{x'x'}+l^2\rho_{y'y'}\right], \tag{3.28}$$

$$\epsilon_{\text{int}} = -\frac{d}{12(l+d)^3}\left[(2d^4-d^3l-3d^2l^2+2dl^3)\rho_{x'x'}+(l^4-dl^3)\rho_{y'y'}\right]. \tag{3.29}$$

We can simplify them further by dividing by $K_2 = -d\rho_{x'x'}/(12(l+d)^3)$ and substituting t and g as in (3.25). The resulting error difference can be expressed as follows:

$$\Delta\epsilon = |K_2|\left[(1+t)^2|(2t^2-t+g)|-|g(t-1)-2t^4+t^3+3t^2-2t|\right]. \tag{3.30}$$

The sign of this error difference is shown in Fig. 6. As we can see, for this type of the mesh movement the swept-based method is more accurate approximately in the region defined as $0<\rho_{y'y'}/\rho_{x'x'}<1$, so the area is even smaller than in the previous example.

After approximating the absolute value of the error difference by the triangle inequality, we obtain the result in the following form:

$$|\Delta\epsilon| \le \frac{d^2l}{12(l+d)^3}\left|(3l^2-3ld-4d^2)\rho_{x'x'}-(3l^2+ld)\rho_{y'y'}\right|, \tag{3.31}$$

and after substituting for volumes where possible, we can write

$$|\Delta\epsilon| \le \frac{1}{24}\left(\frac{V_{F_{\text{NE}}}}{V_{\tilde{C}}}\right)^{\frac{3}{2}}\left|(9V_{c_{0,0}}-3\tilde{V}_{c_{0,0}}-5V_{F_{\text{NE}}})\rho_{x'x'}-(\tilde{V}_{c_{0,0}}+5V_{c_{0,0}}-V_{F_{\text{NE}}})\rho_{y'y'}\right|. \tag{3.32}$$

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

3.5 Hourglass edge movement

In the previous sections, we have analyzed movements where the swept region method produces higher numerical error due to the absence of the corner fluxes. Here, we examine the "hourglass" movement – rotating an edge around its center by moving its vertices in the opposite direction (see Fig. 7). This unwanted mesh patters is known to produce higher numerical error also when using the swept fluxes (due to the error resulting from the function reconstruction taken from a "wrong" cell in one part of the flux). Let us note that the analysis is performed for a smooth density function, as we did before. However, the largest error of the swept-based method would be obtained in case of a discontinuous function with the discontinuity oriented along the rotated edge because taking the reconstruction from just one cell in the entire swept region basically means that the discontinuity is ignored during flux construction.

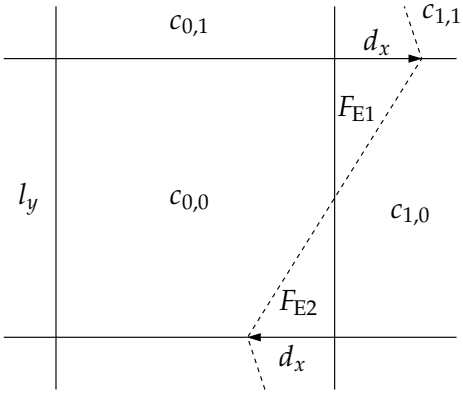


Figure 7: Hourglass movement, dashed line – new mesh, solid line – old mesh.

The situation depicted in the figure is again simplified, in a real-world remapping problem, the hourglass movement will be combined with corner and other movements and also the hourglass region does not have to be perfectly symmetric. The general expression for the remapping error terms is as follows:

$$\epsilon_{\mathrm{sw}} = \frac{1}{2\tilde{V}_{c_{0,0}}} \left(\frac{V_{F_{\mathrm{E1}}-F_{\mathrm{E2}}}}{V_{c_{1,0}}} I^{\mathbf{H}}_{c_{1,0}}(\mathbf{r}_{c_{1,0}}) - I^{\mathbf{H}}_{F_{\mathrm{E1}}-F_{\mathrm{E2}}}(\mathbf{r}_{c_{1,0}}) \right). \tag{3.33}$$

The swept method is constructed so that it perceives the self-intersecting polygon $F_{\mathrm{E1}} - F_{\mathrm{E2}}$ as having volume close to zero. However, based on the sign of its volume it decides which reconstruction will be used for the particular flux. In this case, we presume the region F_{E1} is slightly larger and so values from cell $c_{1,0}$ are used. The intersection method is not prone to such vulnerability because both regions are treated separately:

$$\epsilon_{\mathrm{int}} = \frac{1}{2\tilde{V}_{c_{0,0}}} \left(\frac{V_{F_{\mathrm{E1}}}}{V_{c_{1,0}}} I^{\mathbf{H}}_{c_{1,0}}(\mathbf{r}_{c_{1,0}}) - I^{\mathbf{H}}_{F_{\mathrm{E1}}}(\mathbf{r}_{c_{1,0}}) - \frac{V_{F_{\mathrm{E2}}}}{V_{c_{0,0}}} I^{\mathbf{H}}_{c_{0,0}}(\mathbf{r}_{c_{0,0}}) + I^{\mathbf{H}}_{F_{\mathrm{E2}}}(\mathbf{r}_{c_{0,0}}) \right). \tag{3.34}$$

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

We can also express the error difference magnitude, similarly as for the corner movement:

$$|\Delta\epsilon| \leq \frac{V_{F_{E2}}}{2\tilde{V}_{c_{0,0}}} \left| \frac{I^{\mathbf{H}}_{c_{1,0}}(\mathbf{r}_{c_{1,0}})}{V_{c_{1,0}}} - \frac{I^{\mathbf{H}}_{F_{E2}}(\mathbf{r}_{c_{1,0}})}{V_{F_{E2}}} - \frac{I^{\mathbf{H}}_{c_{0,0}}(\mathbf{r}_{c_{0,0}})}{V_{c_{0,0}}} + \frac{I^{\mathbf{H}}_{F_{E2}}(\mathbf{r}_{c_{0,0}})}{V_{F_{E2}}} \right|. \tag{3.35}$$

We can see that any dependence on the region F_{E1} was eliminated, the formula only concerns the region for which an incorrect reconstruction is used. If the swept method would originally use the values from cell $c_{0,0}$ instead, the formula would contain only F_{E1}-related terms.

In the idealized situation depicted in Fig. 7, both neighboring cells have the same geometry, thus we can claim that $I^{\mathbf{H}}_{c_{1,0}}(\mathbf{r}_{c_{1,0}}) \approx I^{\mathbf{H}}_{c_{0,0}}(\mathbf{r}_{c_{0,0}})$. Also, both halves of the hourglass polygon are symmetrically positioned relative to the centroid of the cell they are located in. The second-order error function $\mathbf{r}^T\mathbf{H}\mathbf{r}$ is centrally symmetric, therefore $I^{\mathbf{H}}_{F_{E1}}(\mathbf{r}_{c_{1,0}}) = I^{\mathbf{H}}_{F_{E2}}(\mathbf{r}_{c_{0,0}})$. If we apply both identities to (3.34), we can see that the intersection-based method is third-order accurate in this case. On the other hand, the swept-based method produces some error in this case, namely:

$$\epsilon_{\text{sw}} = \frac{d_x}{12}\left(d_x\rho_{xx} + l_y\rho_{xy}\right), \quad \epsilon_{\text{int}} = 0, \tag{3.36}$$

where d_x and l_y are the horizontal node movement and the vertical cell dimension, respectively (see Fig. 7). This formula shows that there is one term linear in mesh movement and dependent on the mixed derivative, and an additional quadratic term. This expression is simple to evaluate and does not require further analysis. In this special case, the swept region remapping is always worse than the intersection-based method.

3.6 Shear movement

This movement is an extension of the hourglass mesh motion described in the previous section. This pattern can be encountered in real hydrodynamic simulations (for example, at the material interfaces), so we present this situation here briefly. The bottom mesh nodes both move to the left by the distance d_x while the upper nodes move in the opposite direction, as shown in Fig. 8.

In this case, the intersection-based method produces no second-order error again. The swept-based method can be also as accurate, however, this depends on the selection of the cell from which the reconstruction is used. If the same side is selected on both moving edges (eg. cell $c_{1,0}$ for the right edge and $c_{0,0}$ for the left edge), this method produces numerical error as follows:

$$\epsilon_{\text{sw}} = \frac{d_x}{6}\left(d_x\rho_{xx} + l_y\rho_{xy}\right). \tag{3.37}$$

However, if the opposite neighbor is used for function reconstruction (e.g. cell $c_{1,0}$ for the right edge and $c_{-1,0}$ for the left edge or $c_{0,0}$ for both), the error terms for both sides cancel out. This is caused by the symmetric positioning of both swept regions relative to the cell from which the reconstruction is obtained.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

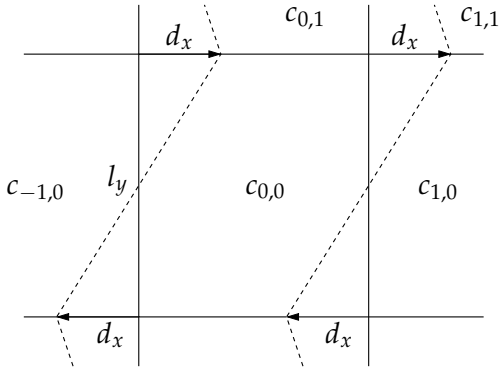


Figure 8: Symmetric cell shear, dashed line – new mesh, solid line – old mesh.

The shear movement consisting of only one edge (upper or lower) moving sideways is not detailed here, because in such case, there are only simple fluxes present and both remapping methods are equivalent.

3.7 Cell rotation

The last simplified movement presented in this paper is the cell rotation around its geometrical centroid shown in Fig. 9. The movement looks complicated to analyze as there are many intersections of the neighboring cells (or self-intersecting regions in the case of the swept-based method). However, the resulting formulas can be simplified due to the symmetric nature of this movement. The total volume of the cell does not change during this movement.

We will describe the results for an equidistant mesh with rectangular cells with dimensions l_x, l_y, as can be seen in Fig. 9. After using (3.13) for every flux and simplifying the results, we obtain the following terms for intersection- and swept-based method error

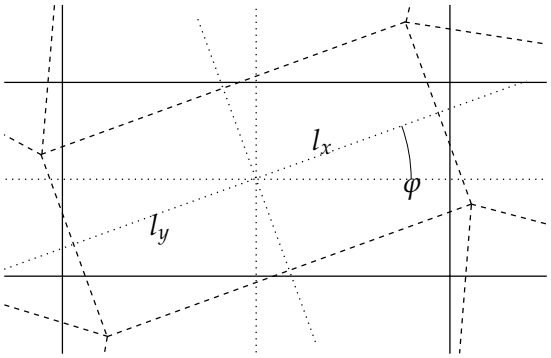


Figure 9: Cell rotation around its centroid by angle φ, dashed line – new mesh, solid line – old mesh.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

respectively:

$$\epsilon_{\text{int}} = \frac{1}{24}\left(l_x^2 - l_y^2\right)\sin\varphi\Big(\left(\rho_{xx}-\rho_{yy}\right)\sin\varphi - \rho_{xy}\cos\varphi\Big), \tag{3.38}$$

$$\epsilon_{\text{sw}} = \frac{1}{12}\left(1-\cos\varphi\right)\Big(\left(\rho_{xx}l_x^2 + \rho_{yy}l_y^2\right)\cos\varphi + \rho_{xy}\left(l_x^2 - l_y^2\right)\sin\varphi\Big). \tag{3.39}$$

We can see that the error of the intersection-based method is dependent on the difference of the squares of the cell dimensions and therefore it will approach zero when the cell dimensions are close to equal, independently on the values of the remapped distribution. The swept-based method does not have such properties.

To see the effect of the second derivatives on the remapping error, we can express the terms in the second-order gauge coordinates of the Hessian (eg. the coordinates in which **H** is diagonal). We presume that **H** is a symmetric matrix, so this transformation involves rotation of the coordinate system by the angle ψ. This angle is computed by solving the following equation:

$$\left(\rho_{xx}-\rho_{yy}\right)\sin 2\psi = 2\rho_{xy}\cos 2\psi, \quad \psi \in \left\langle -\frac{\pi}{4}, \frac{\pi}{4}\right\rangle. \tag{3.40}$$

Note that this transformation can change the order of the eigenvalues. The derivatives in the principal directions ρ_{dd}, ρ_{pp} (the eigenvalues of **H**) can be expressed as follows:

$$\rho_{dd} = \rho_{xx}\cos^2\psi + 2\rho_{xy}\sin\psi\cos\psi + \rho_{yy}\sin^2\psi, \tag{3.41}$$

$$\rho_{pp} = \rho_{xx}\sin^2\psi - 2\rho_{xy}\sin\psi\cos\psi + \rho_{yy}\cos^2\psi. \tag{3.42}$$

By substituting these transformations in (3.38), (3.39), we yield the following error terms:

$$\epsilon_{\text{int}} = \frac{1}{24}\left(l_x^2 - l_y^2\right)\left(\rho_{dd}-\rho_{pp}\right)\sin\varphi\sin\left(\varphi - 2\psi\right), \tag{3.43}$$

and

$$\epsilon_{\text{sw}} = \frac{1}{24}\left(1-\cos\varphi\right)\Big(\left(\rho_{dd}+\rho_{pp}\right)\left(l_x^2 + l_y^2\right)\cos\varphi + \left(\rho_{dd}-\rho_{pp}\right)\left(l_x^2 - l_y^2\right)\cos\left(\varphi - 2\psi\right)\Big). \tag{3.44}$$

Here, we can see another beneficial properties of the intersection-based method. If $\rho_{dd} = \rho_{pp}$, ie. the second derivative is radially symmetric, this method produces no second-order error terms again, regardless of the cell dimensions. It is also accurate when the principal directions are aligned with the axis of symmetry of the movement (half the rotation angle, so $\varphi = 2\psi$).

As we have said before, for a square cell the intersection-based method is third-order accurate. In this case the error term of the swept-based method can be simplified as:

$$\epsilon_{\text{sw},\square} = \frac{1}{12}l^2\left(\rho_{dd}+\rho_{pp}\right)\left(1-\cos\varphi\right)\cos\varphi, \tag{3.45}$$

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

where $l^2 = l_x^2 = l_y^2$. It is interesting that due to the involved symmetry, even the swept-based method has its error independent on the principal directions orientation. It is proportional to the mean curvature, cell area, and the rotation angle.

4 Numerical examples

In this section, we compare the performance of both remapping methods by applying them on a known distribution of discrete values. Analytical initial functions are selected so that they can be integrated over any cell geometry for the error assessment. Similarly, the accuracy of our second-order error estimation can also be evaluated and verified, if it equals to the error produced by the remapping of a quadratic function. In the first example, we can directly compare the numerical results with the analytic formulas. The error estimate is, however, not expected to be exact for later examples, because the function profiles are distorted after the first remapping step and contain higher-order terms. Moreover, when the entire mesh moves during rezoning, more complicated mesh motions than analyzed are present. In the last example, we apply both methods in the context of a full ALE fluid dynamics simulation and compare their results.

The L_1 error is used to measure the total remapping error over the whole domain and it is calculated as follows:

$$L_1 = \frac{\sum_{c\in C} |f_c - \bar{f}_c|}{\sum_{c\in C} \bar{f}_c}, \quad \bar{f}_c = \frac{\int_c f(\mathbf{r})\, dV}{V_c}, \tag{4.1}$$

where C is the set of all computational cells, f_c represents the remapped value in the cell and $\bar{f}_c$ is the analytical mean value in the cell.

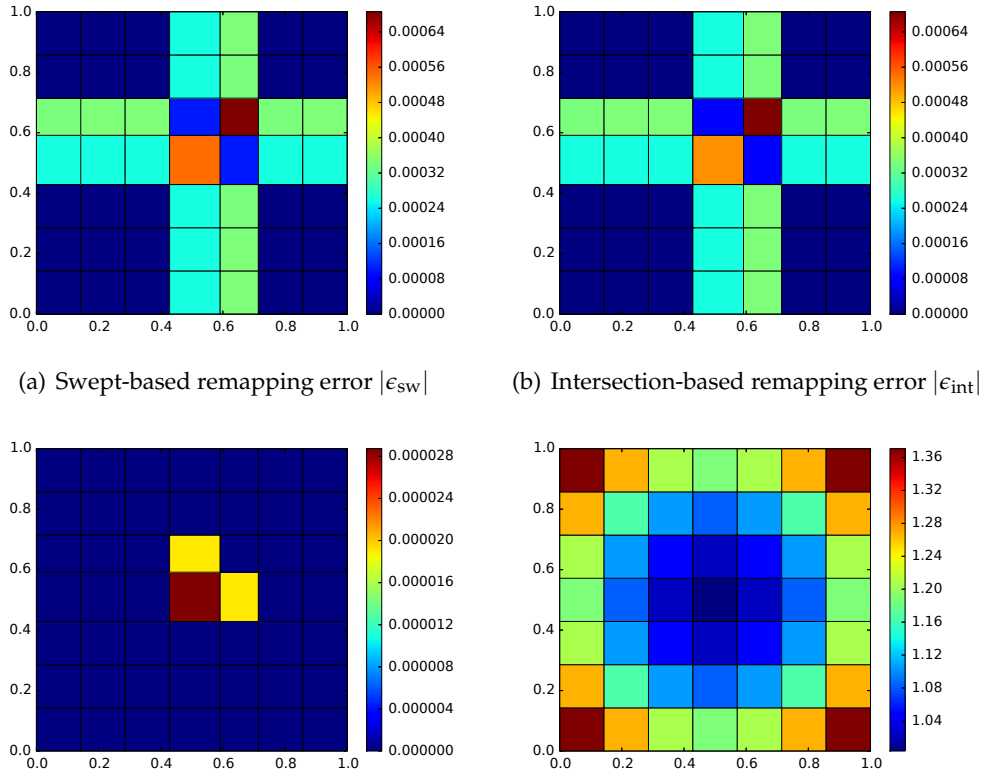
4.1 One remapping step on a small mesh

The first example involves the smallest computational mesh where there is at least one cell unaffected by any boundary effects during a single remapping step. The least squares minimization of the error functional [17] used for obtaining the function derivative approximation employs the values from the neighboring cells, so for the calculation of the second derivative used here, two such layers are needed. Therefore the smallest mesh with at least one cell unaffected has 7×7 cells.

We have selected $f=(x-1/2)^2+(y-1/2)^2+1$ as the initial distribution function. Analytical integration of this function is performed over the computational cells to get the initial values. Then, the upper right corner of the middle cell moves by $\mathbf{r}=(0.02,0.02)$. Other points on the same index-lines are moved in the horizontal or vertical direction only to keep all edges orthogonal (similar movement as in Fig. 2). The distribution is then remapped on this new computational mesh.

The remapping errors for both methods are shown in Fig. 10. Both methods are equivalent and produce the same error in most of the cells except the four cells in the center

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press



(a) Swept-based remapping error $|\epsilon_{\text{sw}}|$ (b) Intersection-based remapping error $|\epsilon_{\text{int}}|$

(c) Method error difference $\Delta\epsilon = |\epsilon_{\text{sw}}| - |\epsilon_{\text{int}}|$ (d) Remapped quantity – initial state

Figure 10: Numerical error of both remapping methods compared. Initial distribution function $(x-1/2)^2+(y-1/2)^2+1$, 7×7 mesh, one step of a diagonal movement.

Table 1: Numerical error of both remapping methods in the central and surrounding cells. Initial distribution function $(x-1/2)^2+(y-1/2)^2+1$, 7×7 mesh, one step of a diagonal movement.

	ϵ_{sw}	ϵ_{int}
$c_{4,4}$	$-5.460\cdot10^{-4}$	$-5.173\cdot10^{-4}$
$c_{5,4}$	$1.033\cdot10^{-4}$	$8.421\cdot10^{-5}$
$c_{4,5}$	$1.033\cdot10^{-4}$	$8.421\cdot10^{-5}$
$c_{5,5}$	$6.857\cdot10^{-4}$	$6.857\cdot10^{-4}$

of the domain. In the error difference plot, we can see that the swept-based remapping method is less accurate due to the missing corner flux. For exact values of the error terms in four central cells, see Table 1. As we can see, the swept-based method produces higher error in each of these cells. In this particular case, there is only second-order error present so the derived error estimate (3.13) equals the numerical remapping error exactly.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

4.2 Remapping of 1D symmetric distribution on a diagonally moving mesh

In the next test, we are remapping a distribution that does not depend on y, namely $f=(x+1)^2$. It undergoes a repeated diagonal mesh movement, a so-called tensor product rezoning [26]. Here, both horizontal and vertical grid-lines move in the perpendicular directions at different velocities. This produces a diagonal mesh deformation, although the cells always stay rectangular. The positions of vertices at step n are defined as follows:

$$\begin{aligned} &x_i^n = x_i^0(1-d^n)+\left(x_i^0\right)^3 d^n, \quad y_i^n = y_i^0(1-d^n)+\left(y_i^0\right)^2 d^n, \\ &d^n = \tfrac{1}{2}\sin(2\pi t^n), \quad t^n = n/n_{\max}, \quad n_{\max}=100, \end{aligned} \tag{4.2}$$

where (x_i^n, y_i^n) is the actual position of node i and (x_i^0, y_i^0) is the initial position. In general, the superscript 0 denotes the initial grid, n the actual grid, and $n_{\max}$ is the total number of the remapping steps required for a completion of the full movement period. The boundary vertices slide along the borderline in the same manner as the inner vertices.

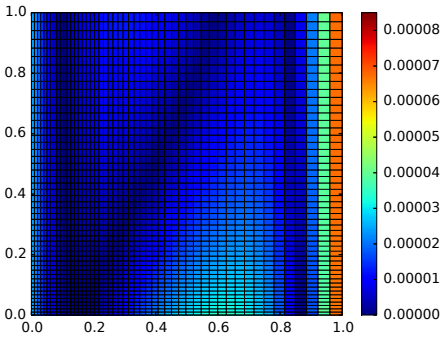
A visible difference between the error distributions for each method after 25 remapping steps (the mesh is most deformed at this moment) is shown in Fig. 11. We can compare the L_1 error of both methods. The numerical error of the swept-based method is $5.786\cdot10^{-6}$, which is slightly worse than the numerical error of the intersection-based method $4.920\cdot10^{-6}$. The swept-based method remapping error varies in the vertical direction due to the asymmetric mesh movement, while the numerical error distribution of the intersection-based method preserves the 1D character of the initial distribution. It is to be noted that in this calculation, the analytical Neumann boundary conditions are imposed on the first derivative:

$$\frac{\partial f}{\partial x}(0)=\frac{\partial f}{\partial x}(1)=2x+2, \quad \frac{\partial f}{\partial y}(0)=\frac{\partial f}{\partial y}(1)=0. \tag{4.3}$$

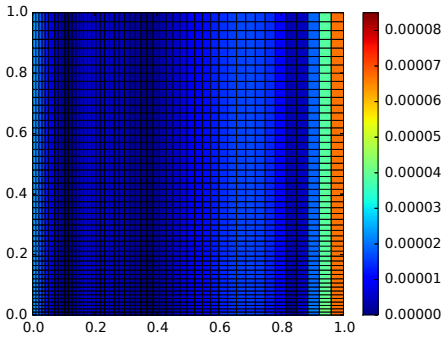
They are used to reduce the inaccuracy in the least squares approximation of the first derivative which arises at the boundary of the domain.

If the remapping steps are repeated it is no longer possible to maintain the exactness of our error estimate. In each successive step, the distribution being remapped is already slightly distorted from the previous one and does not exactly correspond to the second-order polynomial function used in the error analysis. The numerical remapping error presented in Fig. 11 is calculated from the integral mean values of the known analytical function in the computational cells. However, our second-order estimate uses the values from the previous remapping step as a reference and therefore it differs from the exact remapping error. In Fig. 12, a cumulative second-order error estimate (sum of error estimates (3.13) over time steps in each computational cell) for this remapping calculation is presented. The pattern is similar as in the analytical error calculation (compare Fig. 11), although here is a clearly visible shift in the direction of the mesh movement. Also, note the vertical variation of the error estimate values at the right boundary.

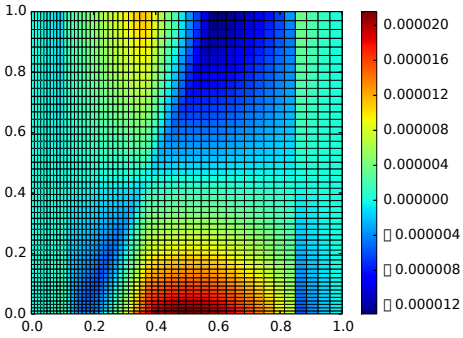
https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press



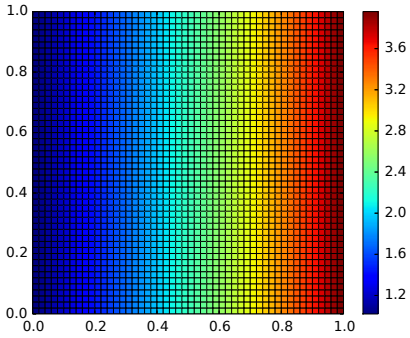
(a) Swept-based remapping error $|\epsilon_{\text{sw}}|$



(b) Intersection-based remapping error $|\epsilon_{\text{int}}|$

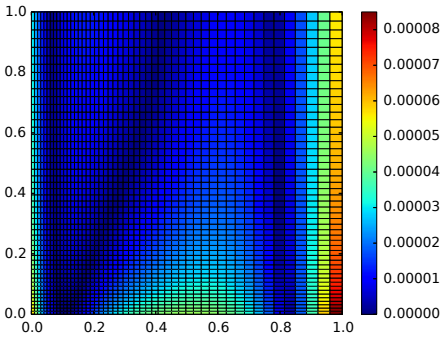


(c) Method error difference $\Delta\epsilon = |\epsilon_{\text{sw}}| - |\epsilon_{\text{int}}|$

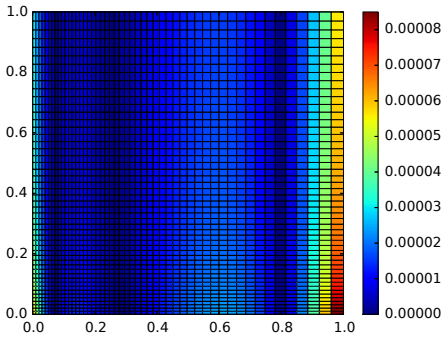


(d) Remapped quantity – initial state

Figure 11: Numerical remapping error of both methods compared. Initial distribution function $(x+1)^2$, 50×50 mesh, step $n=25$ of the tensor product movement.



(a) Swept-based remapping



(b) Intersection-based remapping

Figure 12: Estimate of the second-order remapping error. Initial distribution function $(x+1)^2$, 50×50 mesh, step $n=25$ of the tensor product movement.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

4.3 Cyclic remapping of a smooth, radially symmetric distribution

In the previous example, we have investigated the effect of the diagonal mesh movement on a 1D distribution and we have seen that the swept-based method tends to distort the symmetry of the resulting error. Now we will demonstrate the effect of a diagonal movement on a radially symmetric distribution:

$$f = 10\Big(\mathrm{sinc}(24r-8)+2\Big) = 10\left(\frac{\sin(24r-8)}{24r-8}+2\right), \tag{4.4}$$

$$r = \sqrt{(x-0.5)^2+(y-0.5)^2}. \tag{4.5}$$

This function is smooth and has a circular-shaped peak around the central area. The same mesh movement as in the previous example is used, with the only difference that here all 100 steps are performed. This means that the whole period of the movement will be accomplished and the final mesh will be the same as the initial one (hence the term "cyclic remapping").

The results for a 50×50 mesh are shown in Fig. 13. We can see that the intersection method produces a more symmetric distribution of the remapping error again, although the overall remapping error for the whole domain is actually slightly higher (the L_1 error in this case is $9.532\cdot10^{-3}$ for the swept-based method and $9.586\cdot10^{-3}$ for the intersection method).

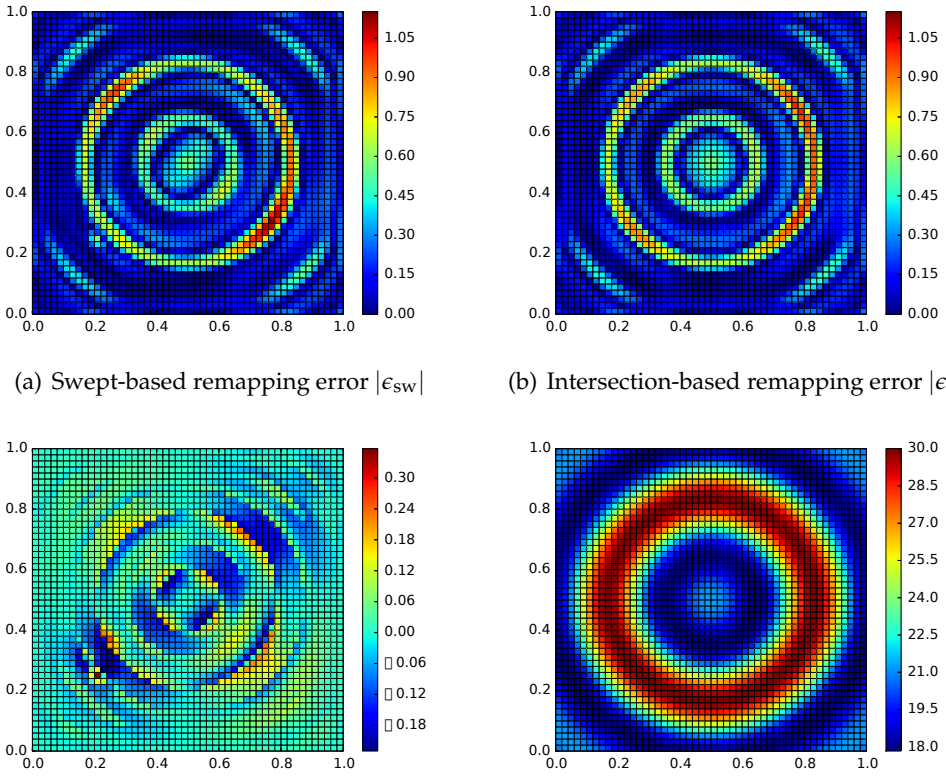
The amount of asymmetry introduced by the remapping of the distribution can be quantified by calculating the L_1 error in the 4 quadrants of the computational mesh separately. This is shown in Table 2. As we can see, the difference between the maximal and minimal quadrant L_1 error is significantly higher for the swept-based method, which confirms the asymmetry seen in Fig. 13.

Table 2: L_1 remapping error of both remapping approaches $m=$sw/int for quadrants 1-4 of the domain (with origin at $(0.5,0.5)$, numbering starts from upper right quadrant and going counter-clockwise). Initial distribution function $10\big(\mathrm{sinc}(24r-8)+2\big)$, 50×50 mesh, step $n=100$ of the tensor product movement.

m	sw	int
$L^1_{1,m}$	$9.351\cdot10^{-3}$	$1.045\cdot10^{-2}$
$L^2_{1,m}$	$9.305\cdot10^{-3}$	$8.676\cdot10^{-3}$
$L^3_{1,m}$	$8.056\cdot10^{-3}$	$8.892\cdot10^{-3}$
$L^4_{1,m}$	$1.142\cdot10^{-2}$	$1.033\cdot10^{-2}$
$\lvert \max_{Q=1\ldots4}(L^Q_{1,m}) - \min_{Q=1\ldots4}(L^Q_{1,m})\rvert$	$3.364\cdot10^{-3}$	$1.774\cdot10^{-3}$

In Table 3, we can see the convergence study for both remapping approaches. As we can see, both approaches exhibit second order convergence in the L_1 norm for the smooth function and their total numerical L_1 errors are very close. We can also see the standard

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press



(a) Swept-based remapping error $|\epsilon_{\text{sw}}|$ (b) Intersection-based remapping error $|\epsilon_{\text{int}}|$

(c) Method error difference $\Delta\epsilon = |\epsilon_{\text{sw}}| - |\epsilon_{\text{int}}|$ (d) Remapped quantity – initial state

Figure 13: Remapping error of both methods compared. Initial distribution function $10\big(\text{sinc}(24r-8)+2\big)$, 50×50 mesh, step $n=100$ of the tensor product movement.

Table 3: L_1 remapping error and its σ_{L_1} standard deviation (in %) for smooth function $10\big(\text{sinc}(24r-8)+2\big)$ and tensor product movement.

mesh size	$n_{\max}$	$L_{1,\text{sw}}$	$L_{1,\text{int}}$	$\sigma_{L_1}^{\text{sw}}$	$\sigma_{L_1}^{\text{int}}$
50×50	100	$9.532\cdot 10^{-3}$	$9.586\cdot 10^{-3}$	12.7	8.43
100×100	200	$2.337\cdot 10^{-3}$	$2.392\cdot 10^{-3}$	9.2	6.45
200×200	400	$5.571\cdot 10^{-4}$	$5.561\cdot 10^{-4}$	7.4	6.45
400×400	800	$1.266\cdot 10^{-4}$	$1.195\cdot 10^{-4}$	5.8	6.54

relative error deviation over the domain quadrants,

$$\sigma_{L_1}^{m} = \frac{1}{L_{1,m}}\sqrt{\frac{1}{4}\sum_{Q=1}^{4}\left(L_{1,m}^{Q}-L_{1,m}\right)^2}, \tag{4.6}$$

where $L_{1,m}^{Q}$ represents the numerical error of the remapping method m in one particular

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

Table 4: Time of simulation (in seconds) of both remapping approaches for smooth function $10\left(\text{sinc}\left(24r-8\right)+2\right)$ and tensor product movement for different mesh resolutions.

mesh size	$t_{sw}[s]$	$t_{int}[s]$
50×50	0.69	2.11
100×100	5.05	16.54
200×200	41.10	134.81
400×400	322.63	1069.46

quadrant Q. As we can see, the error deviation remains approximately constant for the intersection-based approach, the swept-based approach produces (typically) higher error deviation and is improving with increasing mesh resolution. For the highest mesh resolution, the deviations of both approaches are very close.

In Table 4, we can see the comparison of the computational times for different mesh resolutions. The tests were performed in the context of a simple research remapping code written in C++ on a standard Intel Core 3.4 GHz machine. As we can see, in our particular implementation, the intersection-based method is approximately three times slower than the swept-based approach. The main reason is the computation or more fluxes – we need to compute the 16 fluxes per cell (positive and negative flux for each neighbor, including the corner neighbors), while only 4 fluxes per cell (for edge neighbors only) are needed for the swept-based method. Moreover, polygon intersection has to be done for each flux, which is not needed for the swept remap. The rest of the process (density reconstruction, update of geometrical quantities, etc.) remains the same in both approaches.

4.4 Cyclic remapping of a discontinuous, radially symmetric distribution

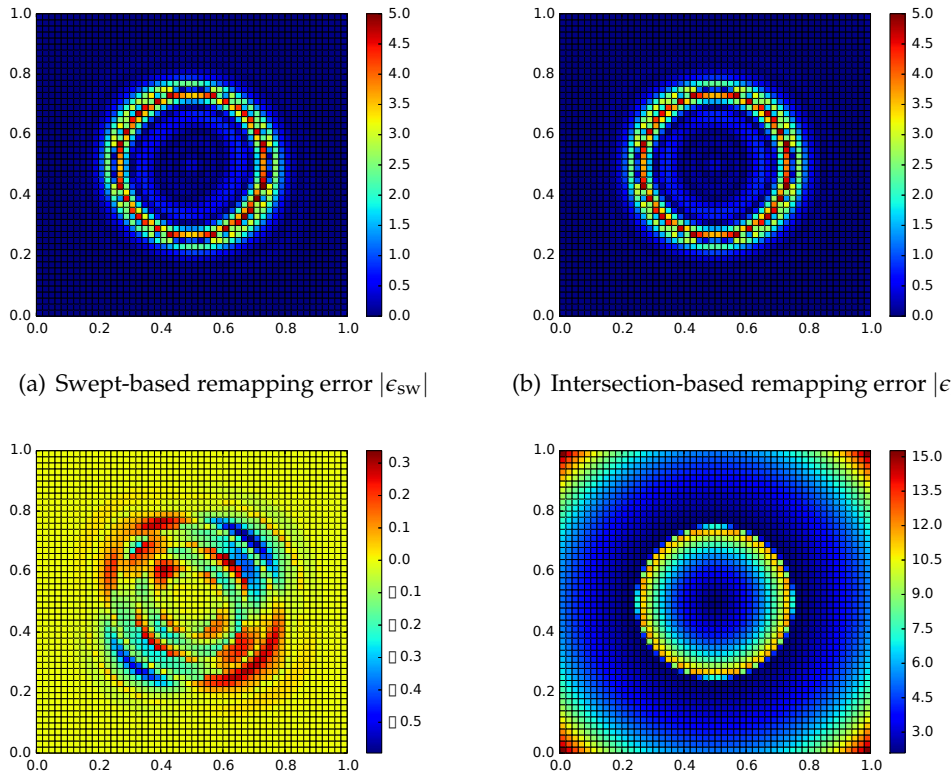
In the previous example, we have seen the convergence test in case of continuous radially-symmetric density function. It has been demonstrated that both methods have second order of convergence and their total numerical errors are comparable. The symmetry of the solution have been studied via the deviation of the error in all quadrants, this value was decreasing with the mesh resolution. In this section, we present similar test, in which the radially-symmetric double-exponential function contains a discontinuity as follows,

$$f(r)=\begin{cases} 1+e^{10r}, & \text{for } x\leq 1/4, \\ 1+e^{6r-1/4}, & \text{otherwise,} \end{cases} \tag{4.7}$$

where the radius r is defined as in (4.5). This test is taken from [4]. The results are shown in Fig. 14. In the error profiles, we can observe stronger asymmetry for the swept-based approach, especially in the diagonal direction corresponding to the mesh motion. This is even better visible in the plot of the error difference.

In Table 5, we can see the convergence study for both remapping approaches. As we can see, both approaches converge with just the first order for the discontinuous function,

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press



(a) Swept-based remapping error $|\epsilon_{\text{sw}}|$ (b) Intersection-based remapping error $|\epsilon_{\text{int}}|$

(c) Method error difference $\Delta\epsilon = |\epsilon_{\text{sw}}| - |\epsilon_{\text{int}}|$ (d) Remapped quantity – initial state

Figure 14: Remapping error of both methods compared. Initial distribution function – double exponential, 50×50 mesh, step $n=100$ of the tensor product movement.

Table 5: L_1 remapping error and its σ_{L_1} standard deviation (in %) for discontinuous double-exponential function and tensor product movement.

mesh size	$n_{\max}$	$L_{1,\text{sw}}$	$L_{1,\text{int}}$	$\sigma_{L_1}^{\text{sw}}$	$\sigma_{L_1}^{\text{int}}$
50×50	100	$7.702\cdot10^{-2}$	$7.873\cdot10^{-2}$	9.4	2.7
100×100	200	$4.859\cdot10^{-2}$	$4.940\cdot10^{-2}$	10.7	2.9
200×200	400	$2.930\cdot10^{-2}$	$2.982\cdot10^{-2}$	10.9	2.9
400×400	800	$1.750\cdot10^{-2}$	$1.782\cdot10^{-2}$	10.9	2.8

their total numerical errors are still very close. We can also see that the error deviation remains approximately constant for both methods, the value is significantly higher for the swept-based approach. This confirms our suspicion that the swept-based approach can violate symmetry stronger. This result was not observed so strongly for the previous smooth function, as the piecewise linear density reconstruction approximates the smooth function reasonably well and the error of the swept-based approach resulting from wrong

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

function approximation in the diagonal direction decreases. However, in the case of discontinuous function, this error is always present (due to erroneous density reconstruction close to the discontinuity) and leads to the symmetry violation.

Regarding the cost of the simulations, the results are almost identical with the results presented in Table 4. The whole remapping process is same as for the smooth function, the only differences are the produced density slopes, which do not affect the simulation time.

4.5 Application in an ALE simulation – Sedov test

In the previous static-remapping tests, we have seen the violation of symmetry significantly higher for the swept-based remapping approach. In this section, we demonstrate that the same behavior can be seen in case of a full hydrodynamic simulation. Due to its symmetric nature, we have chosen the standard Sedov explosion problem [32], which is often used for assessing the performance of hydrocodes. We are aware that the error analysis was performed for smooth functions while the analytic solution of the Sedov problem contains a discontinuity. However, as the numerical solution is discrete we can still see that the behavior of both methods differs in a similar manner as before.

We will compare the remapping methods in the context of a full ALE fluid dynamics simulation consisting of a Lagrangian stage (where the fluid dynamics PDEs are numerically solved) and a mesh smoothing and remapping stage. Note that the produced error is affected mainly by the Lagrangian part of the ALE algorithm. However, some differences resulting from the used remapping method can be also observed.

The underlying equations are the two-dimensional compressible Euler equations:

$$\frac{1}{\rho}\rho_t+\nabla\cdot\mathbf{u}=0, \tag{4.8}$$

$$\rho(\mathbf{u}_x)_t+\nabla p=0, \quad \rho(\mathbf{u}_y)_t+\nabla p=0, \tag{4.9}$$

$$\rho\varepsilon_t+p\nabla\cdot\mathbf{u}=0, \tag{4.10}$$

where ρ represents fluid density, p pressure, $\mathbf{u}$ velocity vector, and ε specific internal energy. The t subscript denotes the partial time derivative, whereas $\mathbf{u}_x$ and $\mathbf{u}_y$ are the velocity vector components in the reference spatial Cartesian coordinates. This system is complemented with the ideal gas equation of state,

$$p=\rho\varepsilon(\gamma-1). \tag{4.11}$$

The Lagrangian scheme employs a staggered predictor-corrector scheme [6] with bulk artificial viscosity [8] to solve these equations. The Winslow rezoning method [16] is employed to keep the computational mesh smooth. For quantity remapping, the flux based approach described in [21] is used. In this approach, the (intersection-based or swept-based) fluxes of mass are constructed by the integration of the limited density reconstruction. Internal energy is remapped in a similar flux form, its fluxes are constructed

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

by attaching the limited internal energy reconstruction values to the mass fluxes. The nodal mass is remapped, where the inter-nodal mass fluxes are interpolated from the inter-cell ones as in [30]. The nodal velocity (momentum) is remapped in a similar flux form, the momentum fluxes are constructed by attaching the reconstructed velocity to the inter-nodal mass fluxes. To guarantee total energy conservation, the standard energy fix [3] is performed.

The particular initial data are taken from [31]. In this test, the overall energy is released from the single computational cell closest to the origin of the coordinate system while the initial values of other quantities are constant in the whole domain, namely:

$$\rho = 1.0, \quad p = 0.4 \cdot 10^{-14}, \quad \mathbf{u} = \mathbf{0}, \quad \gamma = 1.4, \quad \varepsilon_{0,0} = 409.7, \quad t = 1.0,$$

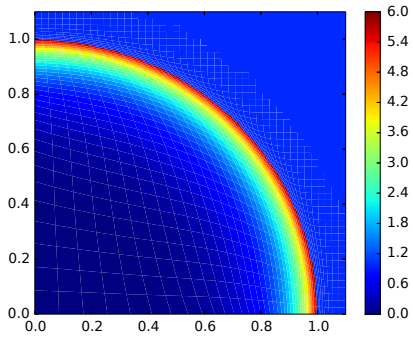
where $\epsilon_{0,0}$ is the specific internal energy in the bottom left computational cell, and t simulation time. This corresponds to the total blast energy $E_{\text{blast}} = 0.2448$. The initial computational mesh contains 45^2 uniformly-distributed cells in the $\langle 0,1.1 \rangle^2$ domain. Any numerical problems with negative internal energy are avoided by timestep reduction in the Lagrangian phase, and by limiting the internal energy reconstruction and subcycling in the remapping stage.

The result is a radially symmetric shock wave shown in Fig. 15. The advantage of this problem is that the analytical solution is known [13] and it is shown sampled on the initial mesh for comparison. The mesh rezoning algorithm used here is the Winslow smoothing [16], applied after every 10 Lagrangian steps (ALE10 regime). The solution is robust, but diffusive. The density shown in Fig. 15 is remapped using the intersection-based algorithm – the difference between both methods is very small and cannot be seen on such plot. Also the L_1 error is similar, $2.061 \cdot 10^{-1}$ for the swept-based method and $2.099 \cdot 10^{-1}$ for the intersection-based method.

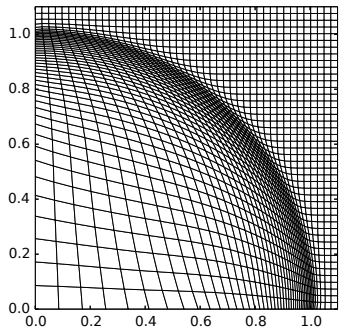
The differences between the methods are better visible, when we plot the radial distribution of the density error in the computational cells. This is shown in Fig. 16. We can see that although the overall error average is similar, the swept-based method shows higher variance on the trailing edge of the shock wave. This is caused by the inward rezoning movement counteracting the Lagrangian expansion. For some of the cells this happens in the diagonal direction, which is one of the weak points of the swept-based method.

In Fig. 17, we can see the angular distribution of the density error for both methods. To emphasize the differences, only cells between $r = 0.9$ and $r = 0.99$ (distance of the centroid from the origin of the coordinate system) are shown. As we can see, the overall behavior is similar for both methods, the error is slightly lower in the diagonal direction than in the direction of the axes. Let us note that this variance does not result from the boundary conditions, which was confirmed by an additional simulation of the Sedov problem performed on a full $\langle 0, 2\pi \rangle$ domain, but results from the variance of the gradients of fluid quantities and mesh edges directions over the domain. However, it confirms our conclusions from the previous tests – the swept-based approach can produce larger variance of the numerical error while the error of the intersection-based algorithm is distributed more uniformly.

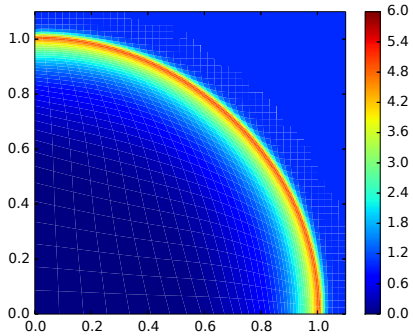
https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press



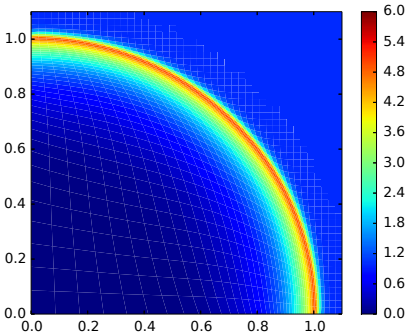
(a) Sampled analytical solution



(b) Computational mesh



(c) Swept, Winslow rezoning



(d) Intersections, Winslow rezoning

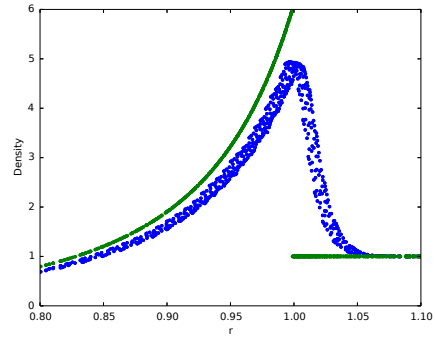
Figure 15: Sedov problem, 2D density profile (analytical and numerical), 45×45 mesh, $t=1.0$.

5 Conclusion

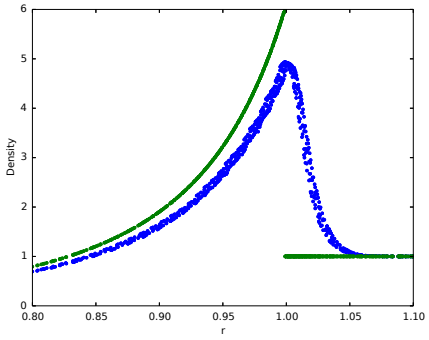
In this paper, we have analyzed the local numerical remapping error for the two often used flux-based remapping algorithms. We focused strictly on the situation of same-connectivity meshes and small mesh displacements, for which the swept-based remap is available. A general analytical formula representing the second-order error magnitude was derived. We have investigated several typical mesh motions and the error formula was adapted for such situations. It was shown that in certain special cases, such as corner movement, "hourglass" movement, or cell rotation, the remapping error produced by the swept-based method can have more nonuniform spatial distribution when compared to the error of the intersection-based method. The error of each method often depends, among other terms, on the orientation of the principal curvatures of the remapped distribution relative to the mesh movement.

If the initial quantity distribution is symmetric, then although the total remapping error is similar, its variance and spatial distribution is apparently less favorable in the case

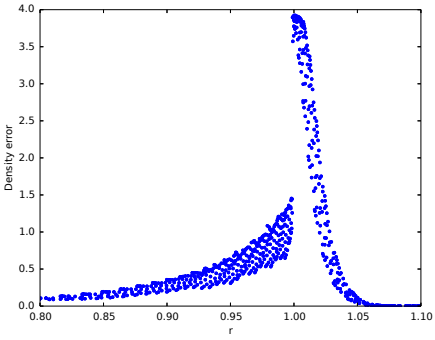
https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press



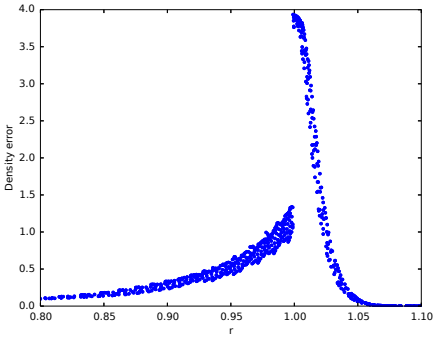
(a) Density, swept-based remapping



(b) Density, intersection-based remapping

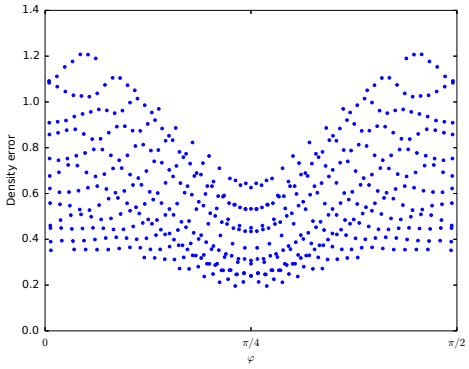


(c) Error, swept-based remapping

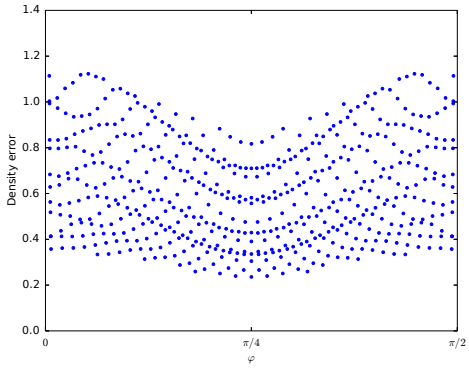


(d) Error, intersection-based remapping

Figure 16: Sedov problem, radial density (numerical and analytic) and density error $\rho_{\text{err}} = |\rho_{i,j} - \rho(\mathbf{r}_{i,j})|$, 45×45 mesh, $t = 1.0$.



(a) Swept-based remapping



(b) Intersection-based remapping

Figure 17: Angular distribution of density error $\rho_{\text{err}} = |\rho_{i,j} - \rho(\mathbf{r}_{i,j})|$ for Sedov test, 45×45 mesh, $t = 1.0$, both remapping approaches are compared. Only values in cells of radius between 0.9 and 0.99 (maximum difference between the methods) are shown.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

of the swept-based method. This was validated on several numerical examples where the static remapping procedure was performed on a selected test function and the resulting error was analyzed. Finally, we have demonstrated on the well-known Sedov blast wave problem that in a hydrodynamical application, the difference between the two remapping methods is visible in the form of a small, but still visible error variance in an otherwise symmetric problem.

Future work involves the development of an approach for approximate and fast estimation of the remapping methods error difference. We are developing a pseudo-hybrid remapping method, using this estimate as a switch between the swept- and intersection-based algorithms in different regions of the computational domain in order to minimize the numerical error and the symmetry distortions.

Acknowledgments

This work was performed under the auspices of the National Nuclear Security Administration of the US Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396 and supported by the DOE Advanced Simulation and Computing (ASC) program. The authors acknowledge the partial support of the DOE Office of Science ASCR Program. This work was partially supported by the Czech Technical University grant SGS16/247/OHK4/3T/14, the Czech Science Foundation project 14-21318S, and by the Czech Ministry of Education project RVO 68407700.

References

[1] R. W. Anderson, N. S. Elliott, and R. B. Pember. An arbitrary Lagrangian-Eulerian method with adaptive mesh refinement for the solution of the Euler equations. *Journal of Computational Physics*, 199(2):598–617, 2004.

[2] T. J. Barth. Numerical methods for gasdynamic systems on unstructured meshes. In D. Kroner, M. Ohlberger, and C. Rohde, editors, *An introduction to Recent Developments in Theory and Numerics for Conservation Laws, Proceedings of the International School on Theory and Numerics for Conservation Laws*, Berlin, 1997. Lecture Notes in Computational Science and Engineering, Springer. ISBN 3-540-65081-4.

[3] D. J. Benson. Computational methods in Lagrangian and Eulerian hydrocodes. *Computer Methods in Applied Mechanics and Engineering*, 99(2-3):235–394, 1992.

[4] M. Berndt, J. Breil, S. Galera, M. Kucharik, P.-H. Maire, and M. Shashkov. Two step hybrid remapping (conservative interpolation) for multimaterial arbitrary Lagrangian-Eulerian methods. *Journal of Computational Physics*, 230(17):6664–6687, 2011.

[5] D. E. Burton, M. A. Kenamond, N. R. Morgan, T. C. Carney, and M. J. Shashkov. An intersection based ALE scheme (xALE) for cell centered hydrodynamics (CCH). Talk at Multimat 2013, International Conference on Numerical Methods for Multi-Material Fluid Flows, San Francisco, September 2-6, 2013. LA-UR-13-26756.2.

[6] E. J. Caramana, D. E. Burton, M. J. Shashkov, and P. P. Whalen. The construction of compatible hydrodynamics algorithms utilizing conservation of total energy. *Journal of Computational Physics*, 146(1):227–262, 1998.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

[7] E. J. Caramana and M. J. Shashkov. Elimination of artificial grid distortion and hourglass-type motions by means of Lagrangian subzonal masses and pressures. *Journal of Computational Physics*, 142(2):521–561, 1998.
[8] E.J. Caramana, M.J. Shashkov, and P.P. Whalen. Formulations of artificial viscosity for multi-dimensional shock wave computations. *Journal of Computational Physics*, 144(1):70–97, 1998.
[9] J. K. Dukowicz and J. R. Baumgardner. Incremental remapping as a transport/advection algorithm. *Journal of Computational Physics*, 160(1):318–335, 2000.
[10] S. Galera, P.-H. Maire, and J. Breil. A two-dimensional unstructured cell-centered multi-material ALE scheme using VOF interface reconstruction. *Journal of Computational Physics*, 229(16):5755–5787, 2010.
[11] C. W. Hirt, A. A. Amsden, and J. L. Cook. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *Journal of Computational Physics*, 14(3):227–253, 1974.
[12] Ph. Hoch. An arbitrary Lagrangian-Eulerian strategy to solve compressible fluid flows. Technical report, CEA, 2009. HAL: hal-00366858. Available at `http://hal.archives-ouvertes.fr/docs/00/36/68/58/PDF/ale2d.pdf`.
[13] J.R. Kamm. Evaluation of the Sedov-von Neumann-Taylor blast wave solution. Technical Report LA-UR-00-6055, Los Alamos National Laboratory, 2000.
[14] M. A. Kenamond and D. E. Burton. Exact intersection remapping of multi-material domain-decomposed polygonal meshes. Talk at Multimat 2013, International Conference on Numerical Methods for Multi-Material Fluid Flows, San Francisco, September 2-6, 2013. LA-UR-13-26794.
[15] P. Kjellgren and J. Hyvarinen. An arbitrary Lagrangian-Eulerian finite element method. *Computational Mechanics*, 21(1):81–90, 1998.
[16] Patrick M. Knupp. Winslow smoothing on two-dimensional unstructured meshes. In *Proceedings of the Seventh International Meshing Roundtable, Park City, UT*, pages 449–457, 1998.
[17] M. Kucharik. *Arbitrary Lagrangian-Eulerian (ALE) Methods in Plasma Physics*. PhD thesis, Czech Technical University in Prague, 2006.
[18] M. Kucharik, J. Breil, S. Galera, P.-H. Maire, M. Berndt, and M. Shashkov. Hybrid remap for multi-material ALE. *Computers & Fluids*, 46(1):293–297, 2011.
[19] M. Kucharik and M. Shashkov. Flux-based approach for conservative remap of multi-material quantities in 2D arbitrary Lagrangian-Eulerian simulations. In Jaroslav Fořt, Jiří Fürst, Jan Halama, Raphaèl Herbin, and Florence Hubert, editors, *Finite Volumes for Complex Applications VI Problems & Perspectives*, volume 1 of *Springer Proceedings in Mathematics*, pages 623–631. Springer, 2011.
[20] M. Kucharik and M. Shashkov. One-step hybrid remapping algorithm for multi-material arbitrary Lagrangian-Eulerian methods. *Journal of Computational Physics*, 231(7):2851–2864, 2012.
[21] M. Kucharik and M. Shashkov. Conservative multi-material remap for staggered multi-material Arbitrary Lagrangian–Eulerian methods. *Journal of Computational Physics*, 258:268–304, 2014.
[22] M. Kucharik, M. Shashkov, and B. Wendroff. An efficient linearity-and-bound-preserving remapping method. *Journal of Computational Physics*, 188(2):462–471, 2003.
[23] P.H. Lauritzen, Ch. Erath, and R. Mittal. On simplifying 'incremental remap'-based transport schemes. *Journal of Computational Physics*, 230(22):7957–7963, 2011.
[24] R. Loubere, P.-H. Maire, M. Shashkov, J. Breil, and S. Galera. ReALE: A reconnection-based arbitrary-LagrangianEulerian method. *Journal of Computational Physics*, 229(12):4724–4761, 2010.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press

[25] L. G. Margolin. Introduction to "An arbitrary Lagrangian-Eulerian computing method for all flow speeds". *Journal of Computational Physics*, 135(2):198–202, 1997.
[26] L. G. Margolin and M. Shashkov. Second-order sign-preserving remapping on general grids. Technical Report LA-UR-02-525, Los Alamos National Laboratory, 2002.
[27] L. G. Margolin and M. Shashkov. Second-order sign-preserving conservative interpolation (remapping) on general grids. *Journal of Computational Physics*, 184(1):266–298, 2003.
[28] D. J. Mavriplis. Revisiting the least-squares procedure for gradient reconstruction on unstructured meshes. In *AIAA 2003-3986*, 2003. 16th AIAA Computational Fluid Dynamics Conference, June 23-26, Orlando, Florida.
[29] J. S. Peery and D. E. Carroll. Multi-material ALE methods in unstructured grids. *Computer Methods in Applied Mechanics and Engineering*, 187(3-4):591–619, 2000.
[30] R. B. Pember and R. W. Anderson. A comparison of staggered-mesh Lagrange plus remap and cell-centered direct Eulerian Godunov schemes for Eulerian shock hydrodynamics. Technical report, LLNL, 2000. UCRL-JC-139820.
[31] G. Scovazzi, E. Love, and M. Shashkov. Multi-scale Lagrangian shock hydrodynamics on Q1/P0 finite elements: Theoretical framework and two-dimensional computations. *Computer Methods in Applied Mechanics and Engineering*, 197(9-12):1056–1079, 2008.
[32] L.I. Sedov. *Similarity and Dimensional Methods in Mechanics, Tenth Edition*. Taylor & Francis, 1993.

https://doi.org/10.4208/cicp.OA-2015-0021 Published online by Cambridge University Press