Available online at www.sciencedirect.com

ScienceDirect

Journal of Computational Physics 226 (2007) 1152–1180

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A cell by cell anisotropic adaptive mesh ALE scheme for the numerical solution of the Euler equations

J.M. Morrell [a], P.K. Sweby [a,*], A. Barlow [b]

[a] Department of Mathematics, The University of Reading, P.O. Box 220, Reading RG6 6AX, UK
[b] AWE, Aldermaston, UK

Received 7 December 2006; received in revised form 14 May 2007; accepted 15 May 2007
Available online 20 July 2007

Abstract

In this paper a cell by cell anisotropic adaptive mesh technique is added to an existing staggered mesh Lagrange plus remap finite element ALE code for the solution of the Euler equations. The quadrilateral finite elements may be subdivided isotropically or anisotropically and a hierarchical data structure is employed. An efficient computational method is proposed, which only solves on the finest level of resolution that exists for each part of the domain with disjoint or hanging nodes being used at resolution transitions. The Lagrangian, equipotential mesh relaxation and advection (solution remapping) steps are generalised so that they may be applied on the dynamic mesh. It is shown that for a radial Sod problem and a two-dimensional Riemann problem the anisotropic adaptive mesh method runs over eight times faster.
Crown Copyright © 2007 Published by Elsevier Inc. All rights reserved.

PACS: 65M50; 76N15

Keywords: ALE; Mesh refinement

1. Introduction

Many physical features of interest, such as shocks and boundary layers, represent large physical variations over small length scales. Mesh refinement and mesh movement focus the resolution where the physics requires it, while requiring fewer elements. Lagrangian schemes allow the mesh to follow the movement of the material. These schemes are often combined with a remap step to reduce mesh tangling. The grid is relaxed and then the state variables are remapped or advected on to the new grid. Lagrangian plus remap schemes are a type of Arbitrary Lagrangian Eulerian, ALE, method [11,5,6]. The total number of elements remains fixed so that any increase in mesh resolution in one area will cause the mesh to become coarser in other areas of the domain.

* Corresponding author. Tel.: +44 118 378 8675; fax: +44 118 9313423.
E-mail address: p.k.sweby@reading.ac.uk (P.K. Sweby).

0021-9991/$ - see front matter Crown Copyright © 2007 Published by Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2007.05.040

An alternative approach to the problem of resolution is adaptive mesh refinement, AMR, which allows the total number of elements in a problem to be increased by introducing local regions of finer meshing. These techniques have traditionally been applied on Eulerian meshes. A hierarchical set of grids, representing different refinement levels, is automatically created as further resolution is required [9,10,18]. An ALE adaptive mesh refinement method has also been developed by Anderson et al. [2]. In this work refinement occurs in rectangular blocks and solutions must be obtained on all levels. In contrast the approach discussed here considers cell by cell refinement and we solve only on the finest resolution that exists for each part of the domain. The solution is obtained on the dynamic mesh containing both coarse and fine elements, while retaining the efficiency of a hierarchical array based data structure.

For many features of interest the large physical variations occur only in one dominant direction. Anisotropic refinement achieves the required resolution in the direction of interest without requiring unnecessary refinement in the other direction. Anisotropic refinement of quadrilaterals has been considered by van der Vegt and van der Ven [21], Aftosmis [1], Keats and Lien [13], and Apel [3]. Kallinderis and Baron [12] use a cell based connectivity array data structure similar to that used in this work and described in Section 3. In all of these papers the meshes remain fixed and do not move with the flow in contrast to the Lagrangian or ALE approach that is employed in this work.

The remainder of this paper is organised as follows. Section 2 gives an outline of the ALE method. Section 3 describes the cell by cell refinement strategy including isotropic and anisotropic refinement. Section 4 considers the changes required to the equipotential mesh relaxation method so that it can be applied on the dynamic mesh, which contains coarse, isotropic and anisotropic elements. In Section 5 the modifications to the advection or solution remapping step for the unstructured dynamic mesh are detailed. Results for a piston problem, a radial Sod problem and a two-dimensional Riemann problem are given in Section 6. It is shown that the anisotropic refinement preserves the accuracy of the uniform fine calculation but runs eight times faster. Finally, conclusions and areas for further work are highlighted in Section 7.

2. The original ALE scheme

This section summarises the main details of the staggered grid ALE method used here; [5] provides more information.

The Euler equations in a Lagrangian reference frame are

$$\frac{\mathrm{D}\rho}{\mathrm{D}t} = -\rho\nabla\cdot\mathbf{u}, \tag{1}$$

$$\rho\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\nabla p, \tag{2}$$

$$\rho\frac{\mathrm{D}\epsilon}{\mathrm{D}t} = -p\nabla\cdot\mathbf{u}, \tag{3}$$

where

$$\frac{\mathrm{D}}{\mathrm{D}t} = \frac{\partial}{\partial t} + \mathbf{u}\cdot\nabla \tag{4}$$

is the Lagrangian derivative, ρ is the density, $\mathbf{u}$ is the velocity vector, p is the pressure and ϵ is the specific internal energy. These equations represent respectively conservation of mass, momentum and energy. The system of equations is closed by including the ideal gas equation of state

$$p = (\gamma - 1)\rho\epsilon. \tag{5}$$

Density, energy and pressure are defined as cell centred variables, while position and velocity are nodal variables. A predictor–corrector time discretisation is used for the implicit pressure dependence in the Euler equations. The energy equation and the equation of state are used to obtain a half step pressure prediction. This predicted pressure is then used in the momentum equation to derive full step nodal velocities. All state variables are then recalculated during a full time step correction. The time step is limited by the CFL condition. The ∇p term in the momentum equation and the $\nabla\cdot\mathbf{u}$ term in the energy equation are evaluated using quadrilateral bilinear isoparametric finite elements.

A two-dimensional generalisation of Christensen's scalar monotonic artificial viscosity [5] is implemented by averaging the edge viscosities

$$q_{\text{edge}} = c_q\rho|\Delta u|^2(1-\Phi^2) + c_l\rho c_s|\Delta u|(1-\Phi), \tag{6}$$

where c_s is the element sound speed, c_q and c_l are the quadratic and linear Christensen artificial viscosity coefficients usually taken to be 0.75 and 0.5 respectively. The limiter Φ is similar to a van Leer limiter [22,23]. The artificial viscosity is treated as an additional pressure.

To reduce mesh tangling, without losing all of the Lagrangian nature of the mesh, the mesh can be relaxed. The Winslow–Crowley equipotential mesh relaxation equations are used to calculate new nodal positions [25,19]. A smooth mesh is obtained if the potential lines ϕ and ψ satisfy Laplace's equation in x and y coordinates. The inverse equations are

$$\frac{\alpha}{4}\frac{\partial^2 x}{\partial\phi^2} - \beta\frac{\partial^2 x}{\partial\phi\psi} + \frac{\gamma}{4}\frac{\partial^2 x}{\partial\psi^2} = 0, \tag{7}$$

$$\frac{\alpha}{4}\frac{\partial^2 y}{\partial\phi^2} - \beta\frac{\partial^2 y}{\partial\phi\psi} + \frac{\gamma}{4}\frac{\partial^2 y}{\partial\psi^2} = 0, \tag{8}$$

where

$$\alpha = 4\left(\left(\frac{\partial x}{\partial\psi}\right)^2 + \left(\frac{\partial y}{\partial\psi}\right)^2\right), \tag{9}$$

$$\beta = 2\left(\frac{\partial x}{\partial\phi}\frac{\partial x}{\partial\psi} + \frac{\partial y}{\partial\phi}\frac{\partial y}{\partial\psi}\right), \tag{10}$$

$$\gamma = 4\left(\left(\frac{\partial x}{\partial\phi}\right)^2 + \left(\frac{\partial y}{\partial\phi}\right)^2\right). \tag{11}$$

The derivatives can be discretised using central differences resulting in

$$\begin{aligned} x_{\phi,\psi} = &\frac{1}{2(\alpha_{\phi,\psi}+\gamma_{\phi,\psi})}[\alpha_{\phi,\psi}(x_{\phi+1,\psi}+x_{\phi-1,\psi}) + \gamma_{\phi,\psi}(x_{\phi,\psi+1}+x_{\phi,\psi-1}) \\ &+ \beta_{\phi,\psi}(x_{\phi+1,\psi-1} - x_{\phi+1,\psi+1} + x_{\phi-1,\psi+1} - x_{\phi-1,\psi-1})], \end{aligned} \tag{12}$$

$$\begin{aligned} y_{\phi,\psi} = &\frac{1}{2(\alpha_{\phi,\psi}+\gamma_{\phi,\psi})}[\alpha_{\phi,\psi}(y_{\phi+1,\psi}+y_{\phi-1,\psi}) + \gamma_{\phi,\psi}(y_{\phi,\psi+1}+y_{\phi,\psi-1}) \\ &+ \beta_{\phi,\psi}(y_{\phi+1,\psi-1} - y_{\phi+1,\psi+1} + y_{\phi-1,\psi+1} - y_{\phi-1,\psi-1})], \end{aligned} \tag{13}$$

where

$$\alpha_{\phi,\psi} = (x_{\phi,\psi+1} - x_{\phi,\psi-1})^2 + (y_{\phi,\psi+1} - y_{\phi,\psi-1})^2, \tag{14}$$

$$\beta_{\phi,\psi} = \frac{1}{2}((x_{\phi+1,\psi} - x_{\phi-1,\psi})(x_{\phi,\psi+1} - x_{\phi,\psi-1}) + (y_{\phi+1,\psi} - y_{\phi-1,\psi})(y_{\phi,\psi+1} - y_{\phi,\psi-1})), \tag{15}$$

$$\gamma_{\phi,\psi} = (x_{\phi+1,\psi} - x_{\phi-1,\psi})^2 + (y_{\phi+1,\psi} - y_{\phi-1,\psi})^2. \tag{16}$$

These formulae can then be applied iteratively to give new relaxed positions.

In the advection or remapping step the old solution, φ_α^-, must be transferred to the new relaxed mesh using a two-dimensional extension of van Leer's approach [22]. The technique for each cell centred variable is similar, the approach is outlined for density, for specific internal energy the volume terms are replaced with mass terms. As the mesh moves from the Lagrangian mesh to the relaxed mesh each element side sweeps out an overlap volume, for example $\Delta V_{\alpha i}$ for side i of element α. These overlap volumes are associated with fluxes through the element sides. The fluxes $\Delta\varphi_{\alpha i}$ are calculated using Benson's volume coordinate approach [7], where $\Delta V_{\alpha i}$ is multiplied by the cell centred value φ_α^- interpolated to the middle of the overlap volume

$$\begin{aligned}
\Delta\varphi_{\alpha 1} &= \Delta V_{\alpha 1}\left(\varphi_\alpha^- - \varphi'_{\alpha,\xi}\left(V_{\alpha 1} + V_{\alpha 2} + \frac{1}{2}\Delta V_{\alpha 1}\right)\right),\\
\Delta\varphi_{\alpha 2} &= \Delta V_{\alpha 2}\left(\varphi_\alpha^- + \varphi'_{\alpha,\eta}\left(V_{\alpha 2} + V_{\alpha 3} + \frac{1}{2}\Delta V_{\alpha 2}\right)\right),\\
\Delta\varphi_{\alpha 3} &= \Delta V_{\alpha 3}\left(\varphi_\alpha^- + \varphi'_{\alpha,\xi}\left(V_{\alpha 3} + V_{\alpha 4} + \frac{1}{2}\Delta V_{\alpha 3}\right)\right),\\
\Delta\varphi_{\alpha 4} &= \Delta V_{\alpha 4}\left(\varphi_\alpha^- - \varphi'_{\alpha,\eta}\left(V_{\alpha 4} + V_{\alpha 1} + \frac{1}{2}\Delta V_{\alpha 4}\right)\right),
\end{aligned} \tag{17}$$

where $\varphi'_{\alpha,\xi}$ and $\varphi'_{\alpha,\eta}$ are monotonic slopes in the η and ξ directions respectively derived in volume coordinates and the $V_{\alpha j}$ are partial volumes associated with element α as shown in Fig. 1.

Benson [7] calculates a parabolic slope by fitting a parabola through elements $\alpha-1$, α, $\alpha+1$ with centres $x_{\alpha-1}$, x_α, $x_{\alpha+1}$ as illustrated in Fig. 1

$$\frac{\partial\varphi_\alpha}{\partial x} = \frac{(\varphi_{\alpha+1}-\varphi_\alpha)\Delta x_\alpha^2 + (\varphi_\alpha - \varphi_{\alpha-1})\Delta x_{\alpha+1}^2}{\Delta x_\alpha \Delta x_{\alpha+1}(\Delta x_\alpha + \Delta x_{\alpha+1})}, \tag{18}$$

where the volume between $x_{\alpha-1}$ and x_α is $\Delta\, x_\alpha$ and between x_α and $x_{\alpha+1}$ is $\Delta\, x_{\alpha+1}$. These volumes are given by

$$\Delta x_\alpha = V^-_{\alpha-12} + V^-_{\alpha-13} + V^-_{\alpha 1} + V^-_{\alpha 4}, \tag{19}$$

$$\Delta x_{\alpha+1} = V^-_{\alpha 2} + V^-_{\alpha 3} + V^-_{\alpha+11} + V^-_{\alpha+14}, \tag{20}$$

where $^-$ denotes old mesh quantities.

A second order limited slope is obtained by requiring that the interpolated value at the edge of the element is not above or below that of the neighbour. The limited slope is

$$\varphi'_{\alpha,\eta} = \frac{1}{2}(\mathrm{sgn}(\Delta\varphi_\alpha) + \mathrm{sgn}(\Delta\varphi_{\alpha+1}))\min\left(\left|\frac{\partial\varphi_\alpha}{\partial x}\right|, |\Delta\varphi_\alpha|, |\Delta\varphi_{\alpha+1}|\right), \tag{21}$$

where the left and right limiting slopes are

$$\begin{aligned}
\Delta\varphi_\alpha &= \frac{\varphi_\alpha - \varphi_{\alpha-1}}{V^-_{\alpha 1} + V^-_{\alpha 4}},\\
\Delta\varphi_{\alpha+1} &= \frac{\varphi_{\alpha+1} - \varphi_\alpha}{V^-_{\alpha 2} + V^-_{\alpha 3}}.
\end{aligned} \tag{22}$$

The new values on the relaxed mesh are then obtained from

$$\varphi_\alpha^+ = \frac{1}{V_\alpha^+}\left(\varphi_\alpha^- V_\alpha^- + \Delta\varphi_{b3} + \Delta\varphi_{t1} + \Delta\varphi_{l2} + \Delta\varphi_{r4} - \sum_{i=1}^{4}\Delta\varphi_{\alpha i}\right), \tag{23}$$

where b, t, l, r represent the bottom, top, left and right neighbouring elements of α.

Momentum advection on the staggered grid is performed by constructing a dual mesh [5,8]. Nodal masses are calculated from the surrounding element masses

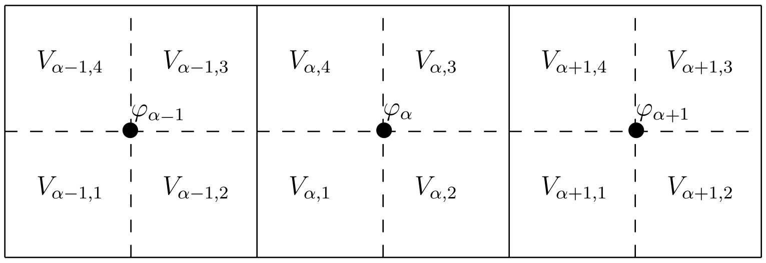


Fig. 1. Diagram of the three elements used to calculate the slopes and their partial volumes.

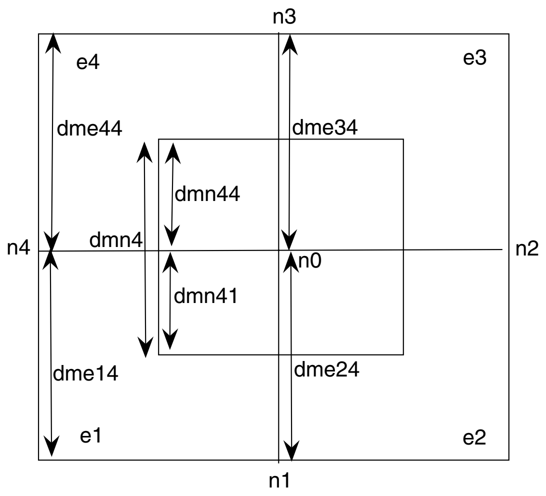


Fig. 2. Variables for nodal mass flux calculation.

$$M_n = \sum_{i=1}^{4} \frac{1}{4} M_{e_i}. \tag{24}$$

The nodal mass flux through a dual cell side s is calculated as the sum of the nodal mass fluxes through the two dual lines that make up the dual cell side

$$dmn_s = dmn_{sk_1} + dmn_{sk_2}, \tag{25}$$

where k_1 and k_2 denote the quadrant that the dual line is in. The nodal mass flux through a dual line is calculated as the average of the two surrounding element mass fluxes dme_{is} over only half the length, for example referring to Fig. 2 we have

$$\begin{aligned} dmn_{44} &= \frac{1}{2}\frac{(dme_{34} + dme_{44})}{2}, \\ dmn_{41} &= \frac{1}{2}\frac{(dme_{24} + dme_{14})}{2}, \end{aligned} \tag{26}$$

so that using Eq. (25) we have that the nodal flux through dual cell side 4 is

$$dmn_4 = dmn_{44} + dmn_{41} = \frac{1}{4}(dme_{34} + dme_{44} + dme_{24} + dme_{14}). \tag{27}$$

Velocity slopes can be calculated on the dual mesh and then the new velocity values are evaluated in an analogous way to the cell centred values.

We now extend the ALE method by considering local refinement of the mesh.

3. Isotropic and anisotropic refinement

The main subject of this paper is the inclusion of a cell by cell refinement technique into the ALE scheme. The cell by cell strategy requires fewer refined elements than structured AMR [2,17,4], which selects rectangular groups of elements to be refined. A coarse element is refined by subdividing along the bisectors of the element sides. In contrast to subdivision on fixed Cartesian meshes, where the refinement follows the spatial coordinates, our refinement follows the element's local coordinates ξ and η. The ALE formulation approximately aligns elements with the flow, by bisecting the element sides, the refinement will also align with the flow. The subdivision occurs in one direction for anisotropic refinement, or in both directions for isotropic refinement.

The change in density $\Delta\rho = \Delta\rho_\xi\hat{\xi} + \Delta\rho_\eta\hat{\eta}$ is used as the refinement criteria in this work as the flow is inviscid. The change in density in the element's ξ direction is given by

$$|\Delta\rho_\xi| = \mathrm{MAX}[|\rho_\mathrm{r} - \rho_i|, |\rho_i - \rho_\mathrm{l}|], \tag{28}$$

where ρ_i is the coarse cell density, ρ_l is the coarse density in the left neighbour cell and ρ_r is the coarse density in the right cell. A similar equation applies for the η direction.

If $|\Delta\rho|$ is greater than the refinement radius then a coarse element is refined. If $|\Delta\rho|$ becomes lower than the derefinement radius then fine elements are derefined. The refinement and derefinement radii must differ otherwise elements may alternate between refinement and derefinement over the time steps, this is referred to as 'blinking'.

The ratio of the density differences $|\Delta\rho_\eta|$ and $|\Delta\rho_\xi|$ is used to decide whether anisotropic or isotropic refinement is required [1]

$$\begin{aligned} \frac{|\Delta\rho_\eta|}{|\Delta\rho_\xi|} &< \tan(30^\circ) \Rightarrow \xi\text{-refinement}, \\ \tan(30^\circ) < \frac{|\Delta\rho_\eta|}{|\Delta\rho_\xi|} &< \tan(60^\circ) \Rightarrow \text{isotropic refinement}, \\ \frac{|\Delta\rho_\eta|}{|\Delta\rho_\xi|} &> \tan(60^\circ) \Rightarrow \eta\text{-refinement}. \end{aligned} \tag{29}$$

Isotropic to anisotropic derefinement may also occur

$$\begin{aligned} \frac{|\Delta\rho_\eta|}{|\Delta\rho_\xi|} &< \tan(25^\circ) \Rightarrow \text{derefine isotropic to } \xi\text{-refined}, \\ \frac{|\Delta\rho_\eta|}{|\Delta\rho_\xi|} &> \tan(65^\circ) \Rightarrow \text{derefine isotropic to } \eta\text{-refined}, \end{aligned} \tag{30}$$

where the derefinement angles 25° and 65° are used rather than 30° and 60° to prevent 'blinking'.

In summary $\Delta\rho$ can be considered as lying on an anisotropic refinement quadrant map [1] as illustrated in Fig. 3. The magnitude of $\Delta\rho$ dictates whether the element refines or derefines completely, the angle of $\Delta\rho$

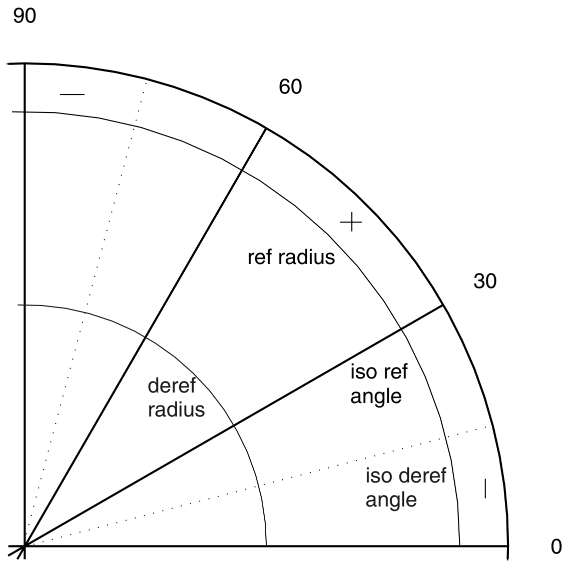


Fig. 3. Refinement quadrant showing refinement radius, derefinement radius, refinement angle and derefinement angle (dotted line). The refinement regions are indicated as + for isotropic, | for ξ-refined and − for η-refined.

dictates the type of refinement or derefinement. An element is never allowed to refine from one anisotropic direction straight to the other anisotropic direction.

Whenever an anisotropic element is isotropically refined the anisotropic dynamic element values are interpolated rather than derefining them to coarse values first. For cell centred variables, the new values are the piecewise constant values of the dynamic element being subdivided, i.e. φ_{a_1} or φ_{a_2} in Fig. 4 (for clarity we illustrate the steps of the method on orthogonal grids, in reality the grids will consist of distorted quadrilaterals).

For derefinement, the fine cell centred values are conservatively averaged to give the new coarse values. For the isotropic to anisotropic coarsening shown in Fig. 4,

$$\begin{aligned}\varphi_{a_1} &= \frac{\varphi_1 X_1 + \varphi_4 X_4}{X_1 + X_4},\\ \varphi_{a_2} &= \frac{\varphi_2 X_2 + \varphi_3 X_3}{X_2 + X_3},\end{aligned} \tag{31}$$

where X represents either volume or mass.

New nodal values are linearly interpolated from the coarse nodal values. When an anisotropic element is isotropically refined the positions and velocities from the anisotropic element are retained. Referring to Fig. 5, showing the isotropic refinement of an η-refined element, the existing positions of nodes $a1$ and $a2$ are retained and these are not necessarily at the mid-points of the left and right coarse sides. The new node in the middle, shown in Fig. 5, takes the position of the intercept between the retained anisotropic line and the new perpendicular bisector,

$$\begin{aligned}x_m &= \frac{(x_{f2} - x_{f1})(x_{a2}y_{a1} - x_{a1}y_{a2}) - (x_{a2} - x_{a1})(x_{f2}y_{f1} - x_{f1}y_{f2})}{(x_{a2} - x_{a1})(y_{f2} - y_{f1}) - (x_{f2} - x_{f1})(y_{a2} - y_{a1})},\\ y_m &= \frac{(y_{a2} - y_{a1})(x_{f1}y_{f2} - x_{f2}y_{f1}) - (y_{f2} - y_{f1})(x_{a1}y_{a2} - x_{a2}y_{a1})}{(x_{a2} - x_{a1})(y_{f2} - y_{f1}) - (x_{f2} - x_{f1})(y_{a2} - y_{a1})}.\end{aligned} \tag{32}$$

The velocity for the middle node is given by interpolating between the existing anisotropic velocities by an amount given by the ratio of the new middle node distance, l_m, to the side's full length L

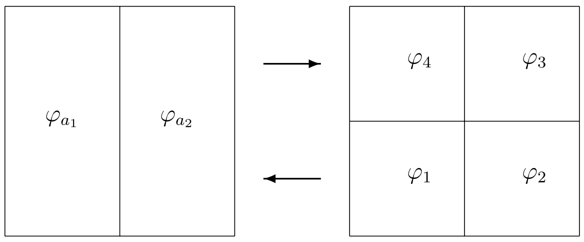


Fig. 4. Diagram showing refinement of anisotropic elements and derefinement to anisotropic elements.

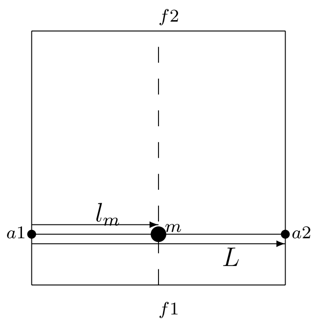


Fig. 5. Diagram showing the interpolation procedure for the middle velocity.

$$\mathbf{u}_m = \mathbf{u}_{a1} + \frac{l_m}{L}(\mathbf{u}_{a2} - \mathbf{u}_{a1}), \tag{33}$$

where

$$\begin{aligned} l_m &= \sqrt{(x_m - x_{a1})^2 + (y_m - y_{a1})^2}, \\ L &= \sqrt{(x_{a2} - x_{a1})^2 + (y_{a2} - y_{a1})^2}. \end{aligned} \tag{34}$$

The nodes $a1$ and $a2$, shown in Fig. 5, are those that already existed on the anisotropic line. When derefinements take place the redundant nodal values are simply removed, while the positions and velocities of the nodes still required for the anisotropic elements are unaltered.

During refinement and derefinement mass is conserved, however the linear interpolation of velocities during refinement does not conserve momentum. A second order solution transfer method has been developed for isotropic refinement that conserves both mass and momentum. The cell centred variables were refined using conservative interpolation [2]. This approach was then generalised for nodal variables with a refinement ratio of two [16]. However, the conservative second order method has not improved the solutions for the radial Sod problem or Riemann problem significantly, and it requires a large increase in code complexity. Further work on second order transfer and a generalisation for anisotropic refinement may be beneficial in the future.

For unsteady problems buffering is required to ensure that during a time step the shocks do not move outside of the fine regions, since this could cause oscillations. Buffering involves the refinement of unflagged elements that surround the refined regions [9]. To include buffering with our anisotropic method the following steps are implemented (an element is referred to as primary if its directional density changes are above the refinement thresholds for that type of element).

- If a buffer element is only surrounded by one type of refinement it becomes an element of that type.
- Any primary isotropic element has 8 isotropic cells around it. This may mean altering a primary anisotropic element to isotropic or setting a buffer cell to isotropic even though it may also have anisotropic neighbours, as shown in Fig. 6.
- If a buffer cell has both types of anisotropic neighbour, see Fig. 7, its type depends on where the change in density lies on the anisotropic refinement quadrant map.
- If one type of primary anisotropic refinement is sandwiched between two elements of the other type of primary anisotropic refinement then all three elements become isotropic, see Fig. 8.

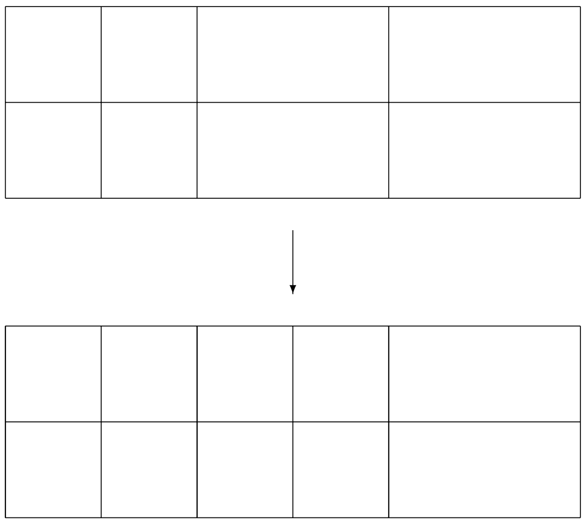


Fig. 6. Buffering for a primary isotropic element can change an already anisotropic element.

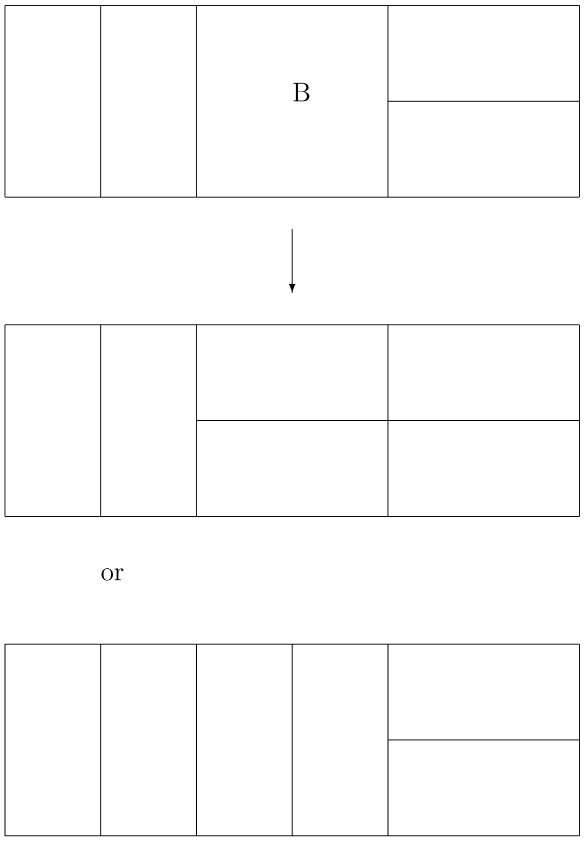


Fig. 7. Type of buffer element B decided by change in density because it has two different anisotropic neighbours.

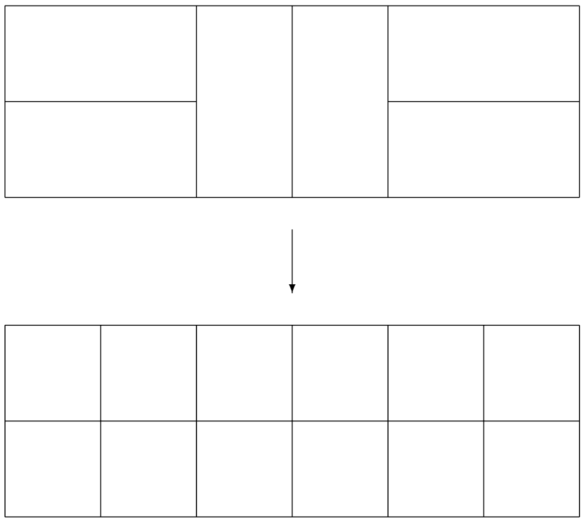


Fig. 8. Buffering to stop anisotropic elements being sandwiched.

Buffering helps to reduce the possibility of isolated refinement regions; these have not been seen in our calculations to date. These steps also help to prevent isolated types of refinement, such as individual anisotropic elements. Smoothing the refinement criteria, for example as done by Khokhlov [14], may be beneficial for future work involving multi-material unsteady flows but at present this has not been implemented.

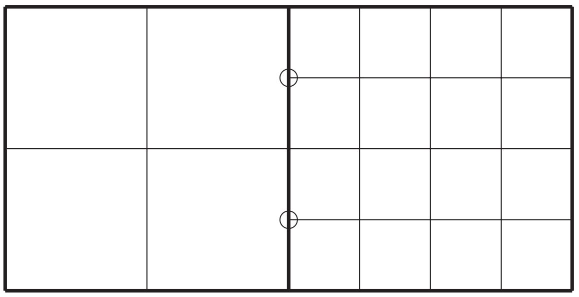


Fig. 9. Diagram of a course-to-fine interface. Empty circles indicate disjoint nodes.

The new elements are inserted into the mesh forming a dynamic mesh that changes with time. The data structure for the adaptive mesh technique involves separate connectivity arrays for each level similar to a hierarchical data structure. The original coarse element–node, element–element, and node–node connectivity arrays are retained. Additional dynamic connectivity arrays describing the dynamic mesh are introduced. The different levels of connectivity arrays are linked using an array that records for each original coarse element the new dynamic element number if the element remains coarse, the two new anisotropic element numbers, or the four new isotropic element numbers. Further arrays record the neighbouring nodes for disjoint nodes, and the element connectivity at a disjoint node where a coarse element has two fine neighbours on a side.

The solution is obtained on the dynamic mesh containing coarse, isotropic and anisotropic elements and not separately on each level of refinement. This can be viewed as obtaining a solution on an unstructured mesh, where an element may have more than four neighbouring elements due to the different levels of refinement. This strategy should be significantly less expensive than solving on every level separately, while still preserving the benefits of a hierarchical data structure.

The adaptive mesh technique requires minimal alterations to the Lagrangian step. Disjoint or hanging nodes are introduced where the change in refinement requires nodes with only three neighbours, as illustrated in Fig. 9. Each disjoint node is slaved to remain the same ratio, r, along the line joining the neighbouring non-disjoint nodes on the interface,

$$\begin{aligned} \mathbf{x}_{n_D} &= r\mathbf{x}_{n_2} + (1-r)\mathbf{x}_{n_1}, \\ \mathbf{u}_{n_D} &= r\mathbf{u}_{n_2} + (1-r)\mathbf{u}_{n_1}, \end{aligned} \tag{35}$$

where $r = \frac{1}{2}$ in this work.

Disjoint nodes are treated as non-dynamic points that have no mass or momentum, so the mass (and force) gathered at these points must be redistributed to the neighbouring dynamic points

$$\begin{aligned} M_{n_1} &= M_{n_1} + (1-r)M_{n_D}, \\ M_{n_2} &= M_{n_2} + rM_{n_D}. \end{aligned} \tag{36}$$

Numerical results illustrating the accuracy of the disjoint nodes and highlighting the success of the adaptive scheme with buffering are presented in Section 6.

Alterations to the Christensen artificial viscosity [5] are required to treat the unstructured nature of the mesh. At a coarse-to-fine interface the correct neighbouring element edges must be identified and any missing velocities are constructed by interpolation.

4. Anisotropic equipotential relaxation

In this section the equipotential mesh relaxation is generalised to make it compatible with anisotropic refinement. In Anderson [2] the relaxation of the levels is carefully interwoven with the time refinement. When the levels are at the same time the finest mesh is relaxed (using a fixed boundary condition at the resolution transition), its new nodal positions are "injected" into the mesh one level coarser, the other nodes on this coarser level are now relaxed. In contrast our work does not include time refinement and considers mesh relaxation on the combined dynamic mesh by altering the mesh relaxation equations to take the differences in nodal length spacings into account. This avoids the complexity of time refinement and is potentially cheaper since the dynamic mesh is considered altogether, rather than considering individual levels using resolution transition boundary conditions and "injection".

A mesh relaxation strategy that does not cause the areas of finer meshing to spread into the areas of coarser meshing is required. All disjoint nodes are slaved (their nodal positions are interpolated from the non-disjoint nodes) and do not require nodal stencils since the relaxation formulae are not applied to them.

The approach used is to take the nodal length spacings into account by giving each node a length value, $C_{\phi,\psi+1}$ etc., denoting if it is coarse or finely spaced from the centre node. Taylor Series expansions involving these length values are then used to discretise the derivatives. The resulting relaxation equation weights depend on the length values.

In the isotropic case only one length value is required to express the nodal spacing of a node from the centre node in both directions. This will not be the case with anisotropic refinement as shown in Fig. 10. The $(\phi+1, \psi+1)$ node, the $(\phi+1, \psi-1)$ node, the $(\phi-1, \psi+1)$ node and the $(\phi-1, \psi-1)$ node can have coarse or fine nodal spacings from the centre node that are different in each direction. Separate C's were set for the horizontal and vertical spacings for the nodes at the corners of the 9 point stencil. The weights for the x-coordinate relaxation were calculated using the horizontal Cx's and the weights for the y-coordinate relaxation used the vertical Cy's.

The non-mixed second order derivatives $\frac{\partial^2 x}{\partial\phi^2}, \frac{\partial^2 y}{\partial\phi^2}, \frac{\partial^2 x}{\partial\psi^2}$ and $\frac{\partial^2 y}{\partial\psi^2}$ and the first order derivatives all involve three nodes that lie on the same equipotential line, so that for the four nearest neighbour nodes only one C is required. The Taylor Series expansion of $x_{\phi+1,\psi}$ is

$$x_{\phi+1,\psi} = x_{\phi,\psi} + C_{\phi+1,\psi} l\left(\frac{\partial x}{\partial\phi}\right) + \frac{C^2_{\phi+1,\psi} l^2}{2}\left(\frac{\partial^2 x}{\partial\phi^2}\right) + \frac{C^3_{\phi+1,\psi} l^3}{6}\left(\frac{\partial^3 x}{\partial\phi^3}\right)\dots \tag{37}$$

and the $\frac{\partial^2 x}{\partial\phi^2}$ term becomes

$$\frac{\partial^2 x}{\partial\phi^2} = \frac{2}{l^2(C^2_{\phi-1,\psi} + C_{\phi-1,\psi}C_{\phi+1,\psi})}x_{\phi-1,\psi} - \frac{2}{l^2(C_{\phi-1,\psi}C_{\phi+1,\psi})}x_{\phi,\psi} + \frac{2}{l^2(C^2_{\phi+1,\psi} + C_{\phi-1,\psi}C_{\phi+1,\psi})}x_{\phi+1,\psi}. \tag{38}$$

For the mixed derivative, the weights for the x-coordinate must be derived using the following type of Taylor Series expansion for the corner nodes

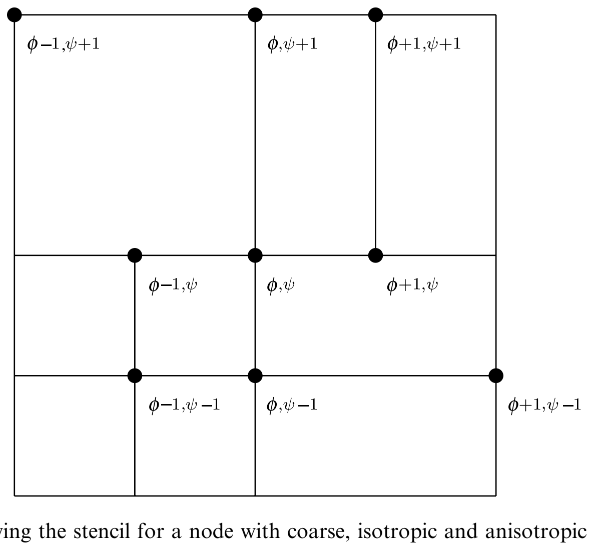


Fig. 10. Diagram showing the stencil for a node with coarse, isotropic and anisotropic neighbouring elements.

$$x_{\phi+1,\psi+1} = x_{\phi,\psi} + Cx_{\phi+1,\psi+1} l\left(\frac{\partial x}{\partial \phi}\right) + Cx_{\phi+1,\psi+1} h\left(\frac{\partial x}{\partial \psi}\right) + \frac{Cx^2_{\phi+1,\psi+1} l^2}{2}\left(\frac{\partial^2 x}{\partial \phi^2}\right) + Cx^2_{\phi+1,\psi+1} lh\left(\frac{\partial^2 x}{\partial \psi^2}\right) + \frac{Cx^2_{\phi+1,\psi+1} h^2}{2}\left(\frac{\partial^2 x}{\partial \psi^2}\right)\ldots. \tag{39}$$

Although only the Cx's are required for the diagonal nodes, these must appear with both the ϕ and ψ terms in the Taylor Series expansion. Otherwise the balance of the diagonal nodes will be lost and the relaxation equation of a node with uniformly coarse nodal spacings in both directions will not reduce to the equipotential relaxation equations.

Using the Taylor expansion to discretise the x-coordinate mixed derivative results in

$$\begin{aligned}\frac{\partial^2 x}{\partial \phi \psi} = &-\frac{Cx_{\phi+1,\psi-1}Cx_{\phi-1,\psi+1} - Cx_{\phi+1,\psi+1}Cx_{\phi-1,\psi-1}}{2lhCx_{\phi+1,\psi+1}Cx_{\phi+1,\psi-1}Cx_{\phi-1,\psi+1}Cx_{\phi-1,\psi-1}} x_{\phi,\psi} + \frac{x_{\phi+1,\psi+1}}{2lh(Cx_{\phi-1,\psi-1}Cx_{\phi+1,\psi+1} + Cx^2_{\phi+1,\psi+1})} \\ &- \frac{x_{\phi-1,\psi+1}}{2lh(Cx_{\phi-1,\psi+1}Cx_{\phi+1,\psi-1} + Cx^2_{\phi-1,\psi+1})} - \frac{x_{\phi+1,\psi-1}}{2lh(Cx_{\phi-1,\psi+1}Cx_{\phi+1,\psi-1} + Cx^2_{\phi+1,\psi-1})} \\ &+ \frac{x_{\phi-1,\psi-1}}{2lh(Cx_{\phi-1,\psi-1}Cx_{\phi+1,\psi+1} + Cx^2_{\phi-1,\psi-1})},\end{aligned} \tag{40}$$

where the l's and h's will cancel out in the inverse equation.

The new α's, β's and γ's are related to the previous central difference values $\alpha_{\phi,\psi}$, $\beta_{\phi,\psi}$ and $\gamma_{\phi,\psi}$ (Eqs. (14)–(16)) by

$$\alpha = \frac{4\alpha_{\phi,\psi}}{(C_{\phi,\psi+1} + C_{\phi,\psi-1})^2}, \tag{41}$$

$$\beta = \frac{4\beta_{\phi,\psi}}{(C_{\phi+1,\psi} + C_{\phi-1,\psi})(C_{\phi,\psi+1} + C_{\phi,\psi-1})}, \tag{42}$$

$$\gamma = \frac{4\gamma_{\phi,\psi}}{(C_{\phi+1,\psi} + C_{\phi-1,\psi})^2}. \tag{43}$$

Substituting into the inverse equation results in the following formula for the new x nodal coordinate

$$\begin{aligned}x_{\phi,\psi} = &\frac{\alpha_{\phi,\psi}\left(\frac{x_{\phi-1,\psi}}{C_{\phi-1,\psi}} + \frac{x_{\phi+1,\psi}}{C_{\phi+1,\psi}}\right)}{D(C_{\phi,\psi+1} + C_{\phi,\psi-1})^2(C_{\phi-1,\psi} + C_{\phi+1,\psi})} + \frac{\gamma_{\phi,\psi}\left(\frac{x_{\phi,\psi-1}}{C_{\phi,\psi-1}} + \frac{x_{\phi,\psi+1}}{C_{\phi,\psi+1}}\right)}{D(C_{\phi+1,\psi} + C_{\phi-1,\psi})^2(C_{\phi,\psi-1} + C_{\phi,\psi+1})} \\ &+ \frac{\beta_{\phi,\psi}\left(\frac{\left(\frac{x_{\phi-1,\psi+1}}{Cx_{\phi-1,\psi+1}} + \frac{x_{\phi+1,\psi-1}}{Cx_{\phi+1,\psi-1}}\right)}{(Cx_{\phi-1,\psi+1} + Cx_{\phi+1,\psi-1})} - \frac{\left(\frac{x_{\phi+1,\psi+1}}{Cx_{\phi+1,\psi+1}} + \frac{x_{\phi-1,\psi-1}}{Cx_{\phi-1,\psi-1}}\right)}{(Cx_{\phi+1,\psi+1} + Cx_{\phi-1,\psi-1})}\right)}{D(C_{\phi+1,\psi} + C_{\phi-1,\psi})(C_{\phi,\psi+1} + C_{\phi,\psi-1})},\end{aligned} \tag{44}$$

where D is given by

$$\begin{aligned}D = &\frac{\alpha_{\phi,\psi}}{(C_{\phi,\psi+1} + C_{\phi,\psi-1})^2(C_{\phi-1,\psi}C_{\phi+1,\psi})} + \frac{\gamma_{\phi,\psi}}{(C_{\phi+1,\psi} + C_{\phi-1,\psi})^2(C_{\phi,\psi-1}C_{\phi,\psi+1})} \\ &+ \frac{\beta_{\phi,\psi}(Cx_{\phi+1,\psi+1}Cx_{\phi-1,\psi-1} - Cx_{\phi+1,\psi-1}Cx_{\phi-1,\psi+1})}{\Pi(C_{\phi+1,\psi} + C_{\phi-1,\psi})(C_{\phi,\psi+1} + C_{\phi,\psi-1})},\end{aligned} \tag{45}$$

with

$$\Pi = Cx_{\phi+1,\psi+1}Cx_{\phi+1,\psi-1}Cx_{\phi-1,\psi+1}Cx_{\phi-1,\psi-1}. \tag{46}$$

The expression for the y-coordinate has the same form except that all of the Cx's are replaced by Cy's in both the $y_{\phi,\psi}$ equation and the expression for D.

The method automatically reduces to the isotropic case if the Cx's are equal to the Cy's and reduces to the original Winslow–Crowley relaxation equations if the nodal spacings are also all the same. The method is formally only first order when the stencil has mixed coarse and fine nodal spacings. However, the same approach is applied to all nodes (except disjoint nodes) and the node's nearest neighbours are always used reducing the possibility that a node will be moved too far outside its neighbouring nodes.

The variables must now be updated to correspond to the new relaxed dynamic mesh. We now extend the advection method so that it can be applied on the dynamic mesh.

5. Adaptive mesh advection

Since the adaptive meshes are unstructured the advection algorithms need to be generalised to include elements that have more than four neighbouring fluxes, see Fig. 11.

Consider an interface between a coarse element and two fine neighbours, if the fine neighbours are outfluxing then the required influx to the coarse element must be

$$\Delta\varphi_{\mathrm{c}} = \Delta\varphi_{f1} + \Delta\varphi_{f2}. \tag{47}$$

If the coarse element outfluxes into the two fine elements, the influx for each fine element must be calculated. If the variable is considered approximately constant over the coarse side (an approximation made in the original advection flux interpolation) then the required fine fluxes are given by

$$\begin{aligned} \Delta\varphi_{f1} &= \frac{\Delta V_{f1}\Delta\varphi_{\mathrm{c}}}{\Delta V_{f1} + \Delta V_{f2}}, \\ \Delta\varphi_{f2} &= \frac{\Delta V_{f2}\Delta\varphi_{\mathrm{c}}}{\Delta V_{f1} + \Delta V_{f2}}. \end{aligned} \tag{48}$$

The slope calculation is complicated at coarse-to-fine interfaces because the variable positions are not in the same place in coarse and fine elements. Furthermore, the width in the direction orthogonal to the slope will have an unwanted affect because the slopes are calculated using volume coordinates and the three cells do not line up.

A conservative average is used to give a variable position in line with the others. The volume that aligns with the element whose slope is being calculated is used in the volume coordinates. For the case in Fig. 12, the slope of a coarse element next to a fine, the value of the required variable for Eq. (18) is given by

$$\varphi_{\alpha+1} = \frac{(\varphi_A V_A + \varphi_B V_B)}{(V_A + V_B)}, \tag{49}$$

and the partial volumes used for $\Delta x_{\alpha+1}$ in Eq. (20), when calculating the parabolic slope, are then

$$\begin{aligned} V_{\alpha+1,1} &= V_{A,1} + V_{A,4}, \\ V_{\alpha+1,4} &= V_{B,1} + V_{B,4}. \end{aligned} \tag{50}$$

For the case in Fig. 13 the value of the required variable for Eq. (18) is given by

$$\varphi_{\alpha-1} = \varphi_C, \tag{51}$$

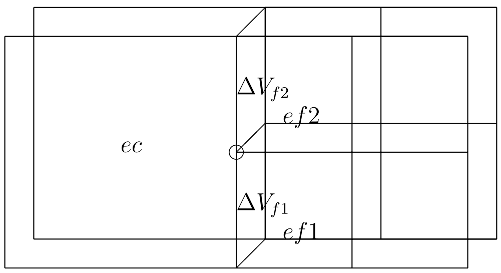


Fig. 11. Diagram of a course-to-fine interface to show overlap volumes. The elements are shown before and after mesh relaxation. The empty circle indicates the disjoint node.

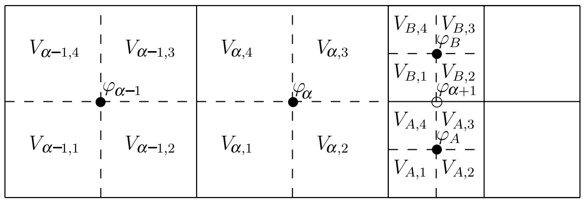


Fig. 12. Diagram of a coarse cell with coarse and fine neighbours to show slope variables. The empty circle indicates the interpolated value.

where for first order interpolation the value in the element is constant. The partial volumes used for Δx_α in Eq. (19) are then

$$V_{\alpha-1,2} + V_{\alpha-1,3} = V_{C,2}. \tag{52}$$

Once slopes and neighbouring fluxes have been obtained the new cell values can be calculated using Eqs. (17) and (23).

The disjoint nodes are slaved (their velocities are interpolated from the non-disjoint nodes) and so we need only consider the momentum advection of non-disjoint nodes. The nodal mass that would have been gathered to the disjoint nodes must be redistributed to the neighbouring non-disjoint nodes, thus the dual mesh lines pass straight through the disjoint nodes. The nodal mass weights are a quarter of the surrounding element's mass plus any other mass contributions passed to the node from other neighbouring nodes of that element being disjoint. The total mass is then the sum of the weights for each node. This corresponds exactly to the dual mesh shown in Fig. 14.

The nodal mass fluxes are easier to calculate if the contributions to them are calculated individually. From Eq. (25), for side 2 of node n we have

$$dmn_2 = dmn_{2,2} + dmn_{2,3}, \tag{53}$$

where the second subscript denotes the quadrant the dual line is associated with, and we can calculate $dmn_{2,2}$ and $dmn_{2,3}$ separately.

A dual mesh line that passes straight through the disjoint node lies along one of the element mass fluxes but is only half as long. Referring to Fig. 15, f_a the contribution to the nodal mass flux is given by

$$f_a = dmn_{1,2} = \frac{1}{2} dme_{2,1}. \tag{54}$$

The dual mesh line not passing through the disjoint node is similar to the usual dual mesh line except that it cuts the whole of the element not just half, therefore the contribution to the nodal mass flux is

$$f_b = dmn_{2,2} = \frac{1}{2}(dme_{2,2} + dme_{2,4}). \tag{55}$$

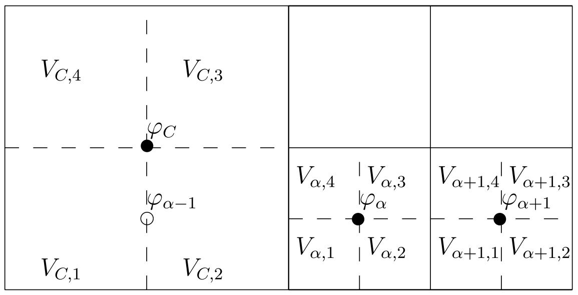


Fig. 13. Diagram of a fine cell with coarse and fine neighbours to show slope variables. The empty circle indicates the interpolated value.

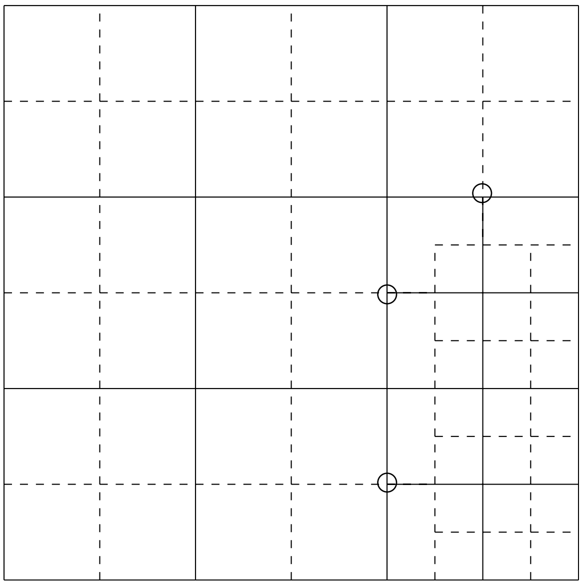


Fig. 14. Diagram of the chosen dual mesh. Dotted lines are the dual mesh, empty circles indicate disjoint nodes.

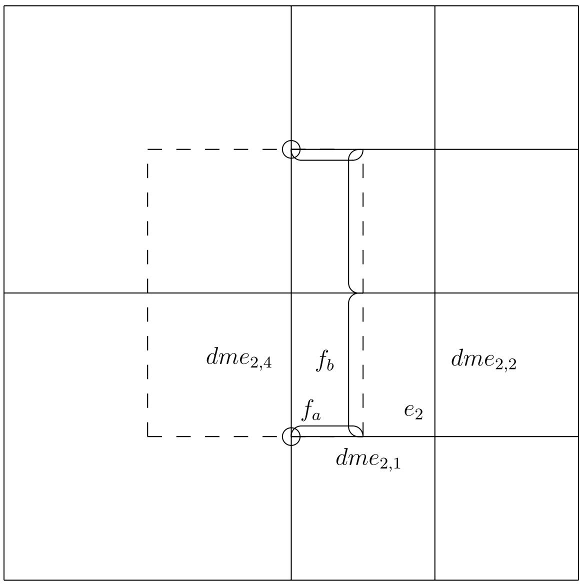


Fig. 15. Diagram of the chosen dual mesh around an interface showing the mass fluxes.

In the case of a corner shown in Fig. 16, $D1$ and $D2$ represent diagonal fluxes to a diagonal neighbour dual cell and are calculated as the usual contributions.

To be able to calculate the post advection nodal mass we require the influxes from the neighbours. Therefore we must consider between which nodes fluxes are passed and the value of the neighbouring nodal mass fluxes. When an element's nodal mass flux contribution is split between two neighbouring nodes it is split equally, half is passed to each.

The easiest and most efficient way of calculating the neighbour fluxes is in two parts, the first two terms are from the usual direct neighbouring node on that side and then there are perturbations caused by the unstructured mesh adding neighbouring fluxes or reducing the flux as it passes to another neighbour. From Fig. 17 the

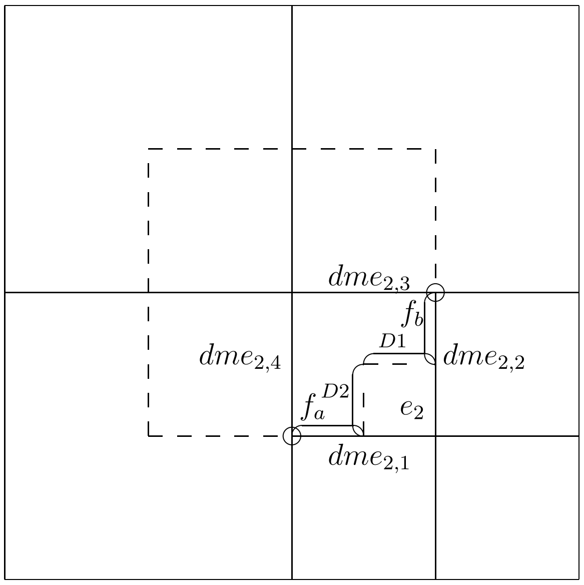


Fig. 16. Diagram of the chosen dual mesh around a corner showing the mass fluxes.

neighbouring flux for side 2 of node n can be seen to be the sum of all the neighbouring fluxes for side 2 $dmn2_{4,1}$, $dmn2_{4,4}$, $dmnb_{4,4}$, and $dmnt_{4,1}$. This can be split into the direct neighbouring node contribution and the perturbation, so that

$$\begin{aligned} fnb_{n,2} &= dmn2_{4,1} + dmn2_{4,4} + pert \\ pert &= dmnb_{4,4} + dmnt_{4,1}. \end{aligned} \tag{56}$$

For a corner, involving an isotropic element, diagonal fluxes must also be considered and their neighbouring fluxes calculated.

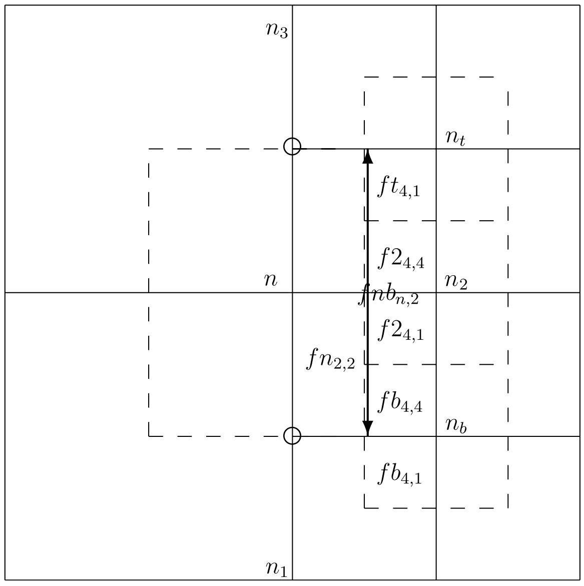


Fig. 17. Diagram of the neighbour nodes, n's, and the neighbour mass fluxes, $f = dmn$, for an interface. The arrow indicates the extent of $fnb_{n,2}$ which is the sum of $ft_{4,1}$, $f2_{4,4}$, $f2_{4,1}$ and $fb_{4,4}$.

Finally the post advection nodal mass is calculated using

$$M_n^+ = M_n^- + \sum_{i=1}^{4} \Delta M_{n,i} + \sum_{i=1}^{4} \Delta diag1_{n,i} + \sum_{i=1}^{4} \Delta diag2_{n,i}, \tag{57}$$

where for node n for example

$$\Delta M_{n,2} = -dmn_2 + fnb_{n,2} \tag{58}$$

is the difference between the dual cell's outflux on side 2, dmn_2, and the neighbouring flux through side 2 into the dual cell of node $n, fnb_{n,2}$. Similar differences, $\Delta diag1_{n,i}$ and $\Delta diag2_{n,i}$, are calculated for the diagonal fluxes.

Velocity slopes are required to calculate the new velocities. If the dual mesh is defined without disjoint node dual cells then the fine node next to the disjoint node is missing an aligning dual cell. The approach taken in calculating slopes for coarse nodes on coarse-to-fine interfaces is to calculate slopes on a dual mesh defined using the coarse nodal stencil. Slopes for nodes in the fine area of meshing are then calculated using a different dual mesh that includes volumes around disjoint nodes. Due to the use of volume coordinates the difference between the widths perpendicular to the slope and the width of the node's original dual cell must be taken into account. It should be noted that the limiting here may not be strictly between the node and it's flux neighbour but this appears to suffice in all the test problems tried.

It is easier to calculate the contributions to the momentum fluxes separately, rather than for the whole fluxing side. The difference between the flux dual mesh and the slope dual mesh must be taken into account as a ratio that corrects the width over which the slope was calculated. The velocity is interpolated out as far as the flux dual mesh line. For the corner diagonal fluxes, the only interpolation required is in the direction perpendicular to the flux side as the original scheme assumes the value is constant along the flux side.

For example the case of a corner, shown in Figs. 18 and 19 with $\Delta un_{2,2} = k2$ and $\Delta un_{1,2} = k1$, has contributions to the momentum fluxes given by

$$\Delta un_{2,2} = -dmn_{2,2}\left(u + u_\eta\left(\frac{Wc_{n,3} + Wc_{n,2}}{\frac{2}{3}W_{n,2}}\right)\left(\frac{2}{3}W_{n,2} - \frac{1}{2}dmn_{2,2}\right)\right), \tag{59}$$

$$\Delta un_{1,2} = -dmn_{1,2}\left(u - u_\xi\left(\frac{Wc_{n,1} + Wc_{n,2}}{\frac{2}{3}W_{n,2}}\right)\left(\frac{2}{3}W_{n,2} - \frac{1}{2}dmn_{1,2}\right)\right), \tag{60}$$

where the $Wc_{n,i}$ are the nodal mass weights of node n used in the slope calculation and the $W_{n,i}$ are the nodal mass weights of node n used in the flux calculation.

The total diagonal momentum fluxes from Fig. 20 are

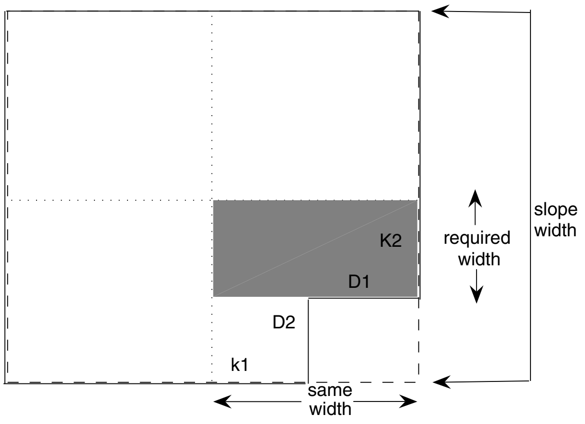


Fig. 18. Calculation of momentum flux k_2 for a corner, dotted boundary is the slope dual cell, line boundary is the flux dual cell.

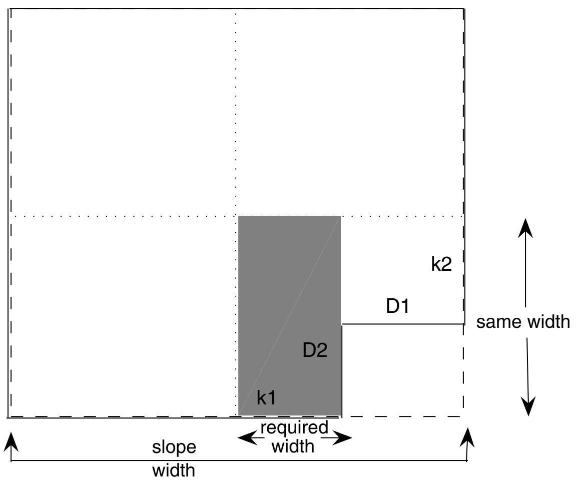


Fig. 19. Calculation of momentum flux k_1 for a corner, dotted boundary is the slope dual cell, line boundary is the flux dual cell.

$$\Delta udiag1_{n,2} = -D1\left(u - u_\xi\left(\frac{Wc_{n,1} + Wc_{n,2}}{\frac{2}{3}W_{n,2}}\right)\left(\frac{W_{n,2}}{3} - \frac{\mathrm{D1}}{2}\right)\right), \tag{61}$$

$$\Delta udiag2_{n,2} = -D2\left(u + u_\eta\left(\frac{Wc_{n,3} + Wc_{n,2}}{\frac{2}{3}W_{n,2}}\right)\left(\frac{W_{n,2}}{3} - \frac{\mathrm{D2}}{2}\right)\right). \tag{62}$$

The calculation for neighbour momentum fluxes follows exactly the same procedure as for nodal mass fluxes. Finally the new post advection velocity is found using the momentum fluxes, their neighbour momentum fluxes, and the pre and post advection nodal masses.

This generalisation of the advection method conserves mass, internal energy and momentum at resolution transitions in the dynamic mesh. The changes to the advection procedures are implemented by looping through the disjoint nodes and are thus valid for anisotropic elements. It should be noted that anisotropic elements never cause the diagonal fluxes seen in the isotropic element corner case.

The anisotropic adaptive mesh ALE code has been validated on a range of test problems we now present results for three of these test problems.

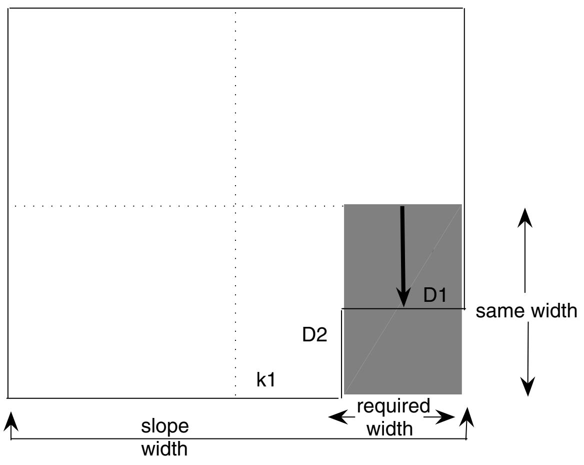


Fig. 20. Calculation of diagonal momentum flux D_1 for a corner, dotted boundary is the slope dual cell, line boundary is the flux dual cell.

6. Results

6.1. Piston problem

A piston problem with a horizontal velocity of 1.0 on the left boundary and $\rho = 1$, $p = 1$ in a 0.4×0.1 domain was run to $t = 0.15$ to assess the effect of a shock crossing a static refinement interface. Calculations were performed with a fine-to-coarse interface at $x = 0.2$ initially, a coarse-to-fine interface at $x = 0.2$, and a horizontal interface at $y = 0.04$. The three types of refinement (isotropic, x-refinement and y-refinement) were tested for each case. Convergence results were obtained and compared to convergence results for a uniformly fine and a dynamic isotropic calculation with buffering.

The convergence of the dynamically adaptive scheme was comparable to that of the uniformly fine calculation, and the buffering ensures that the shock remains within the refined region. However, if the shock was allowed to cross from a fine region to a coarse region, the convergence for the isotropic and x-refinement cases was reduced to less than that of a uniformly coarse calculation, see Fig. 21. This was due to the spurious reflections and transmissions caused as the shock crosses the interface. The y-refinement calculation, that introduced disjoint nodes but had no resolution transition in the x direction, retained the accuracy of the coarse calculation.

In Fig. 22, results for a coarse-to-fine interface are presented, these show that for the x-refinement and isotropic calculations (where the resolution in the x direction has increased) the accuracy is between that of the uniformly fine and coarse calculations. The y-refinement has no resolution increase in the x direction, so the coarse accuracy is retained. The coarse-to-fine interfaces cause much smaller spurious reflections and transmissions than those seen with a fine-to-coarse interface.

Fig. 23 shows that the convergence results for the horizontal interface have comparable accuracy to the uniformly coarse calculation for all three types of refinement. Running a shock along an interface causes differences in the shock width and disturbances due to the impedance mismatch and the vorticity this generates.

In summary it is the change in resolution and not the disjoint nodes that lead to spurious reflections and transmissions as shocks pass through interfaces. Allowing shocks to pass through or along interfaces should

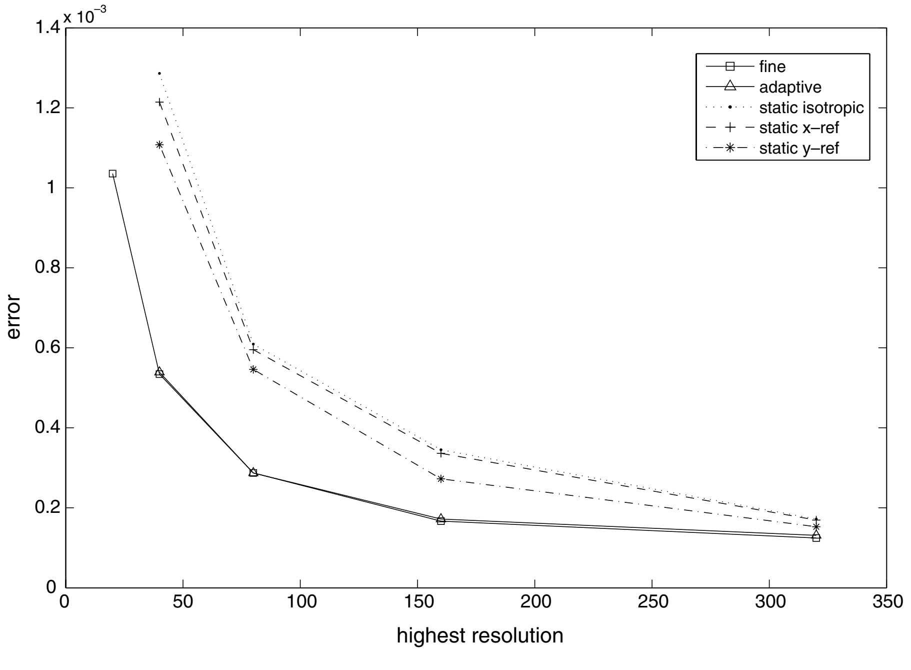


Fig. 21. Convergence of norm 1 with highest resolution for fine, adaptive isotropic, static isotropic, static anisotropic x-refinement and static anisotropic y-refinement for fine-to-coarse piston problem.

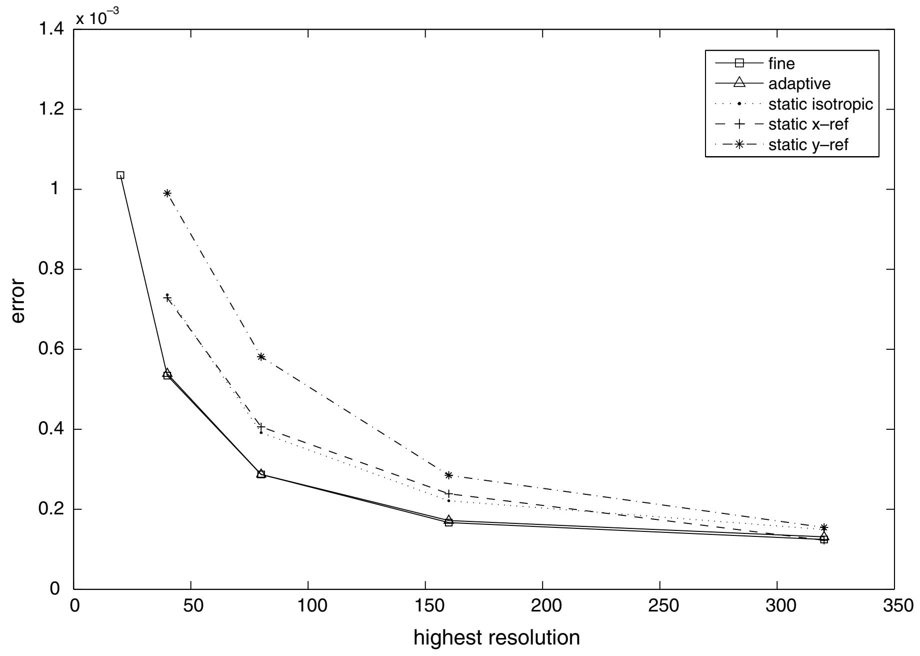


Fig. 22. Convergence of norm 1 with highest resolution for fine, adaptive isotropic, static isotropic, static anisotropic x-refinement and static anisotropic y-refinement for coarse-to-fine piston problem.

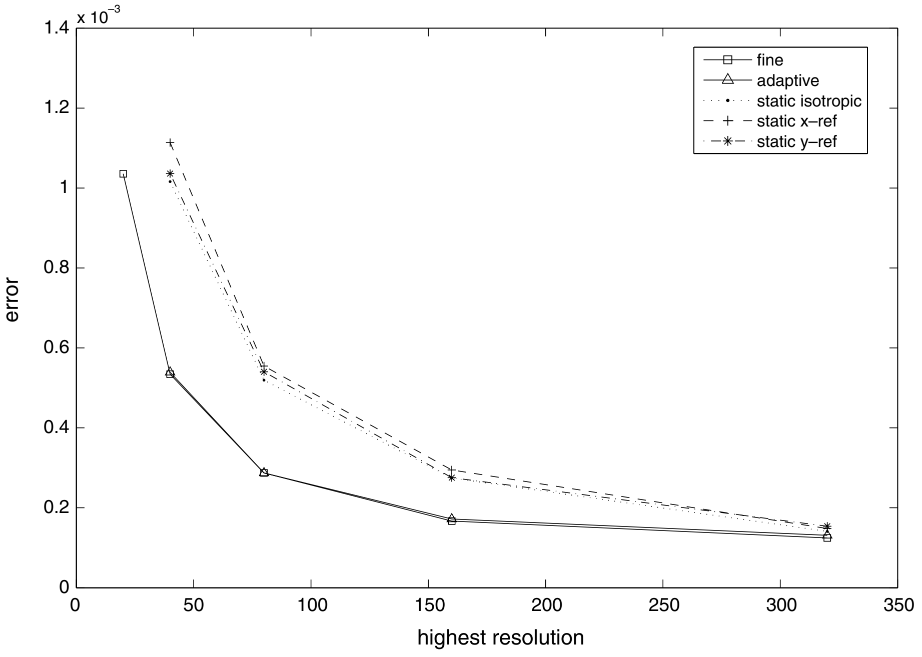


Fig. 23. Convergence of norm 1 with highest resolution for fine, adaptive isotropic, static isotropic, static anisotropic x-refinement and static anisotropic y-refinement for horizontal interface piston problem.

be avoided by employing adaptive refinement and including buffering; accuracy comparable to the uniformly fine calculations can then be achieved. Reducing the spurious effects produced when shocks interact with refinement interfaces remains a challenge for the future.

6.2. Radial Sod problem

A quarter radial Sod problem [20] was run to $t = 0.25$. Within a 0.4 radius of (0,0) the initial conditions were $\rho = 1$, $p = 1$ and zero velocity, elsewhere the initial conditions were $\rho = 0.125$, $p = 0.1$ and zero velocity. The refinement threshold was 0.04 and the derefinement value was 0.025. For the anisotropic calculation the isotropic refinement angle was 30° and the derefinement angle was 25° from each axis. The problem was run with anisotropic and isotropic refinement on a 50×50 initial mesh, only isotropic refinement on a 50×50 mesh, a uniform 100×100 fine mesh, and a uniform 50×50 coarse mesh. The adaptive calculations were run with only one level of mesh refinement as the data structures used have yet to be generalised from two

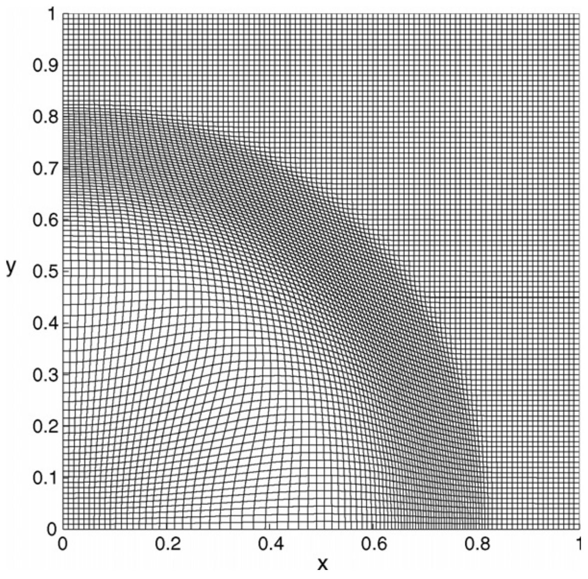


Fig. 24. Uniformly fine mesh for Radial Sod problem $t = 0.25$.

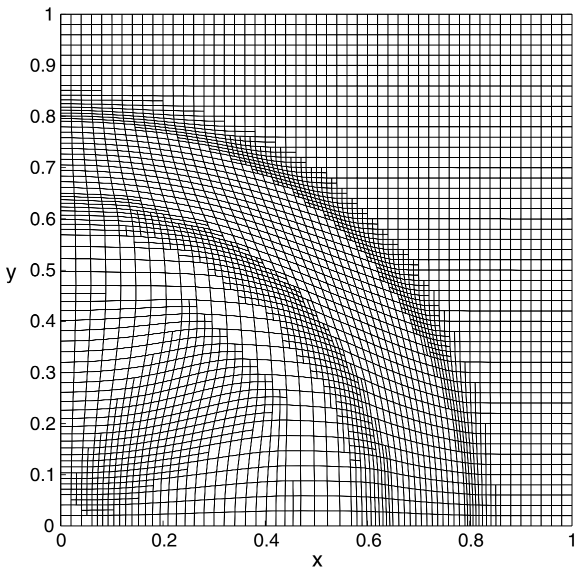


Fig. 25. Anisotropic mesh for Radial Sod problem $t = 0.25$.

levels to many refinement levels. However, the anisotropic refinement and ALE methods presented in this paper would generalise to many refinement levels as long as the refined elements of each level are properly nested. This is a coding issue and generalisation of the data structures and code to many levels, together with an investigation of the results, is an area for future work.

Refined regions occurred around the rarefaction fan, contact and shock, with derefinement in between these features. In the anisotropic calculation, see Fig. 25, isotropic refinement only occurred within 15–20 degrees of

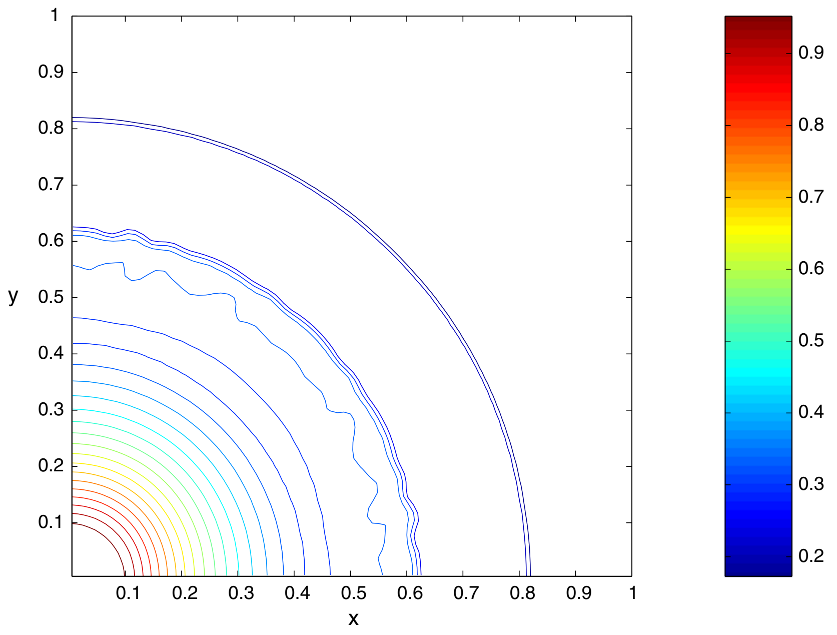


Fig. 26. Fine mesh density contours for Radial Sod problem $t = 0.25$.

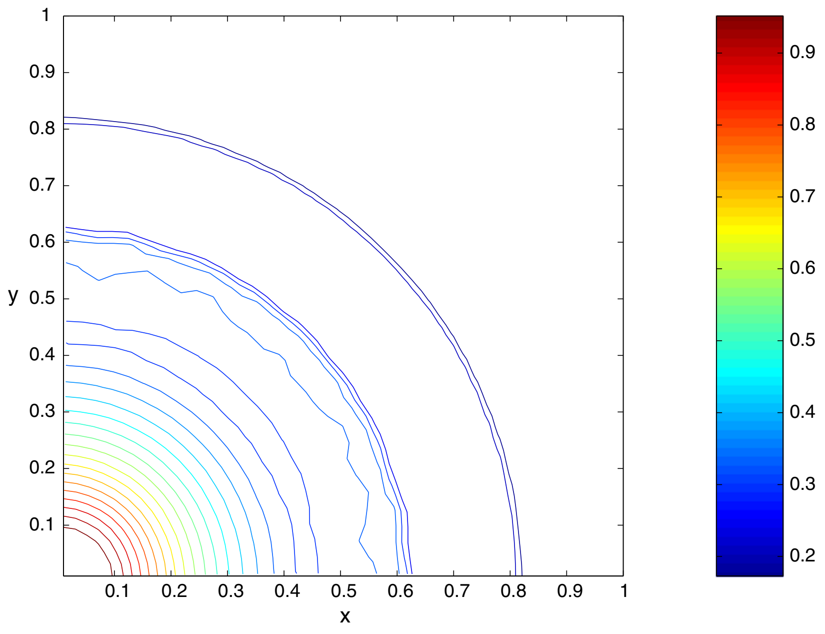


Fig. 27. Anisotropic refinement density contours for Radial Sod problem $t = 0.25$.

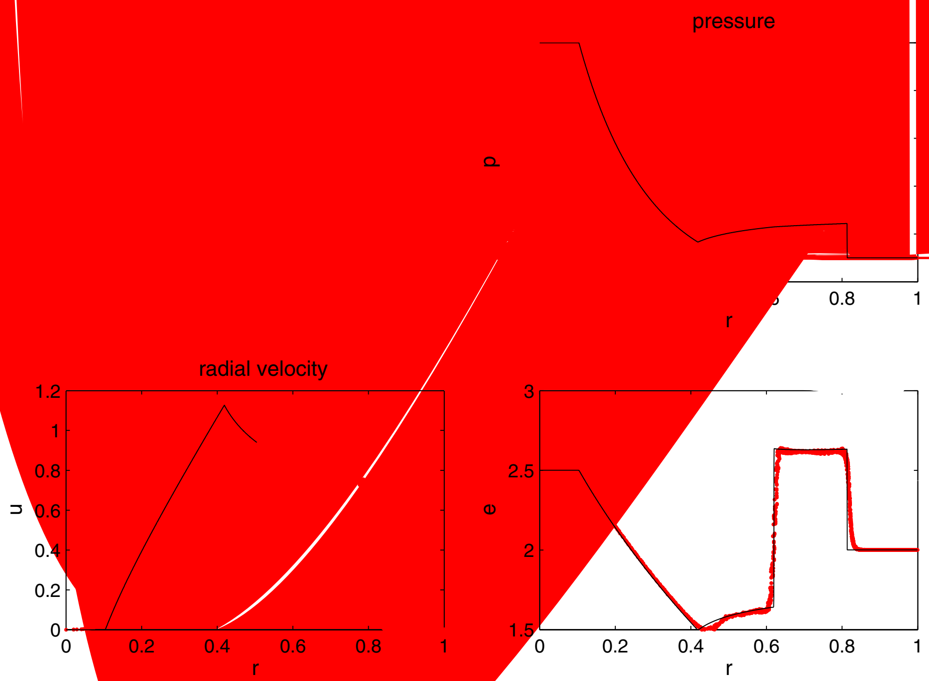


Fig. 28. Radial Sod problem density, pressure, radial velocity and specific internal energy for anisotropic solution at $t = 0.25$.

the diagonal, where the gr[illegible]ted to vary in both directions. The meshes in Figs. 24 and 25 are smooth and untangled. T[illegible]efits greatly from the mesh relaxation as the mesh becomes highly distorted if the problem is run as [illegible] Lagrangian. The density contours are very similar for the anisotropic calculation, see Fig. 27, and the fine calculation, see Fig. 26, with only small differences occurring at the edges of the rarefaction fan and contact. The loss in radial symmetry and noise experienced between the rarefaction fan and contact is due to the implementation of the initial conditions on the quadrilateral mesh forming a stepped radius rather than a perfect quarter circle.

A very fine one-dimensional approximate solution, with 5000 points, can be obtained using a Roe approximate Riemann solver, with the radial terms included as a source term as explained in [24]. This fine approximation is compared against the adaptive mesh ALE results in Fig. 28 and is used to obtain a measure of the error. The 1-norm error for a calculation with N elements of volume V_i is

$$\|e\|_1 = \sum_{i=1}^{N} [|\rho_{\mathrm{amr}} - \rho_{\mathrm{s}}| V_i], \tag{63}$$

whilst the 2-norm error is

$$\|e\|_2 = \sqrt{\sum_{i=1}^{N} [(\rho_{\mathrm{amr}} - \rho_{\mathrm{s}})^2 V_i]}. \tag{64}$$

Table 1
Errors for fine, isotropic, anisotropic and coarse Radial Sod calculations $t = 0.25$

Calculation	$\|e\|_1$	$\|e\|_2$
Fine	0.0044	0.0103
Isotropic	0.0046	0.0109
Anisotropic	0.0046	0.0107
Coarse	0.0072	0.0139

Table 1 shows that the anisotropic calculation has comparable errors to the isotropic calculation. The anisotropic calculation runs 7.8 times faster than the uniformly fine calculation, where the isotropic calculation only runs 6.6 times faster. Both the isotropic and anisotropic calculation errors are nearly as good as the fine calculation error, and are much better than the coarse error. Fig. 29 compares the convergence of the 1 norm error in the anisotropic calculations (plotted at the highest resolution) and the uniformly fine calculations. The anisotropic results converge only slightly slower than the uniformly fine results, confirming that the

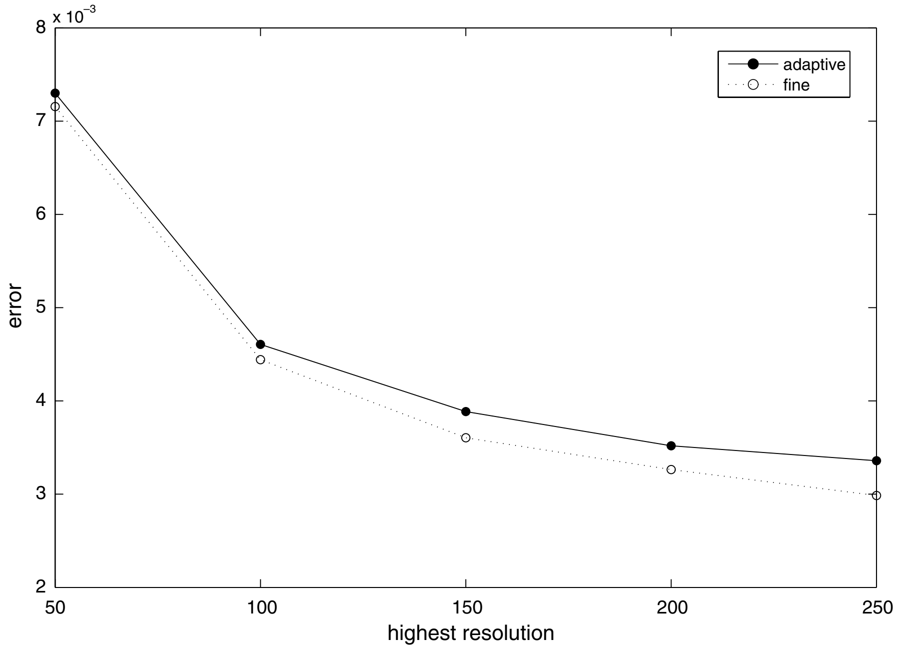


Fig. 29. Convergence of norm 1 with highest (finest) resolution for fine and anisotropic Radial Sod calculations.

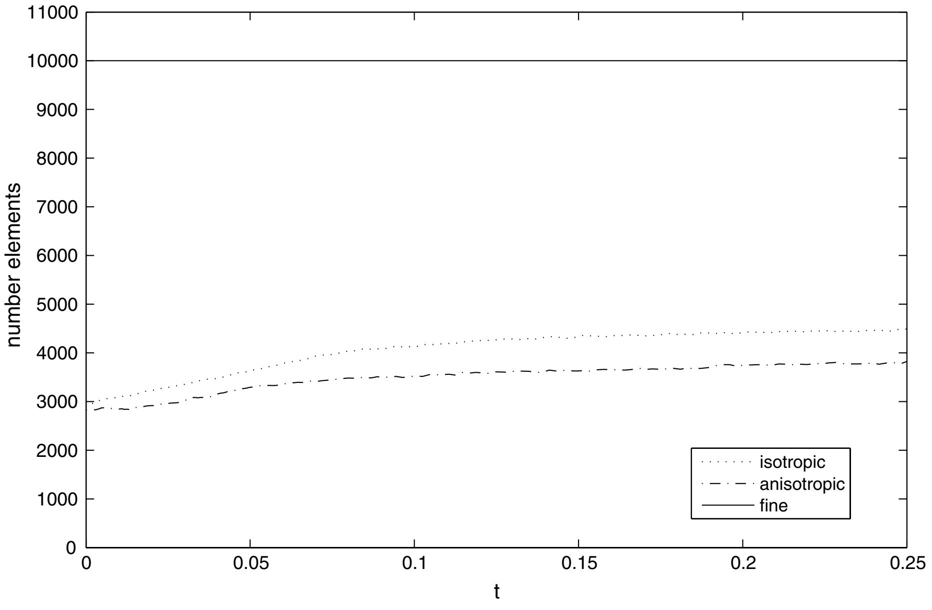
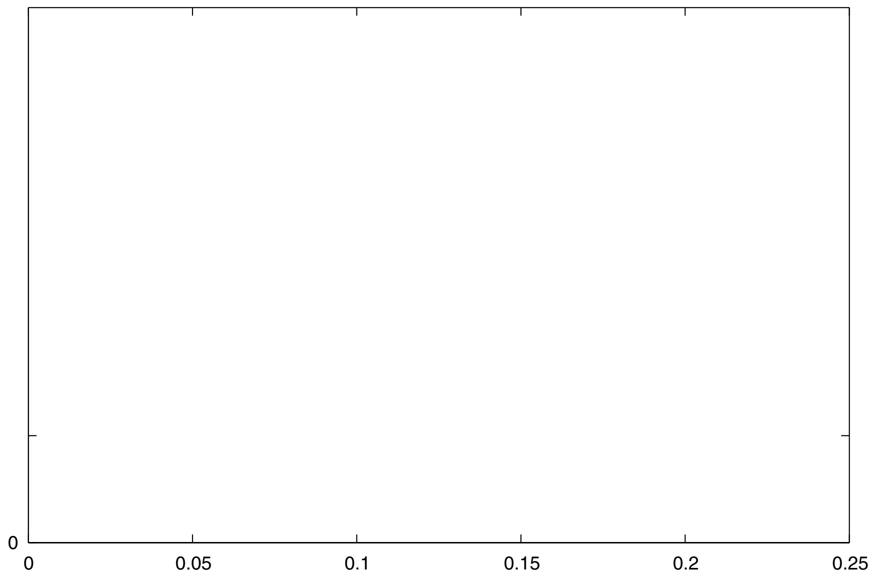


Fig. 30. Number of elements over time for Radial Sod problem.

anisotropic method's convergence is of a comparable order to that of uniform meshing, and is dictated by the finer resolution and not the coarse resolution.

The anisotropic calculation requires fewer elements every time step, see Fig. 30. The types of refinement occurring over the calculation time are plotted in Fig. 31 and this shows that the majority of the elements remain coarse. Even in this problem, which contains a shock that curves rather than aligns with the mesh, the amount of anisotropic refinement outweighs the number of isotropic refinements.



6.3. Two-dimensional Riemann problem

In this example different constant initial conditions are given in each quarter of a square domain. The quarters 1,2,3 and 4 are defined respectively as $x > 0.5$ and $y > 0.5$, $x < 0.5$ and $y > 0.5$, $x < 0.5$ and $y < 0.5$ and finally $x > 0.5$ and $y < 0.5$. The initial conditions for the 4 shock configuration given in Kurganov and Tadmor [15] are

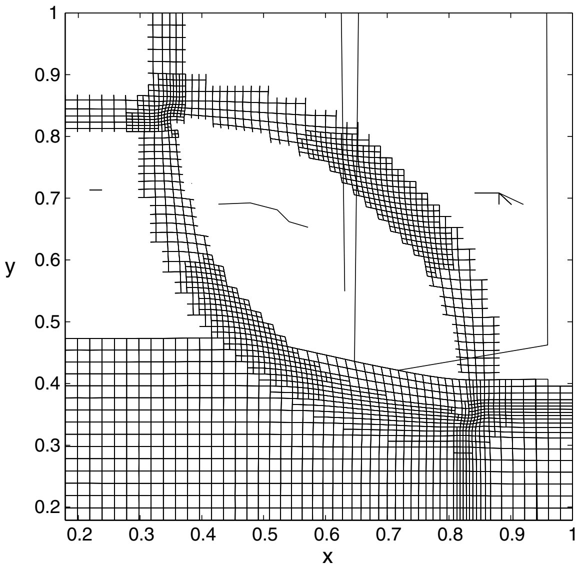


Fig. 33. Anisotropic mesh for 2D Riemann problem $t = 0.2$.

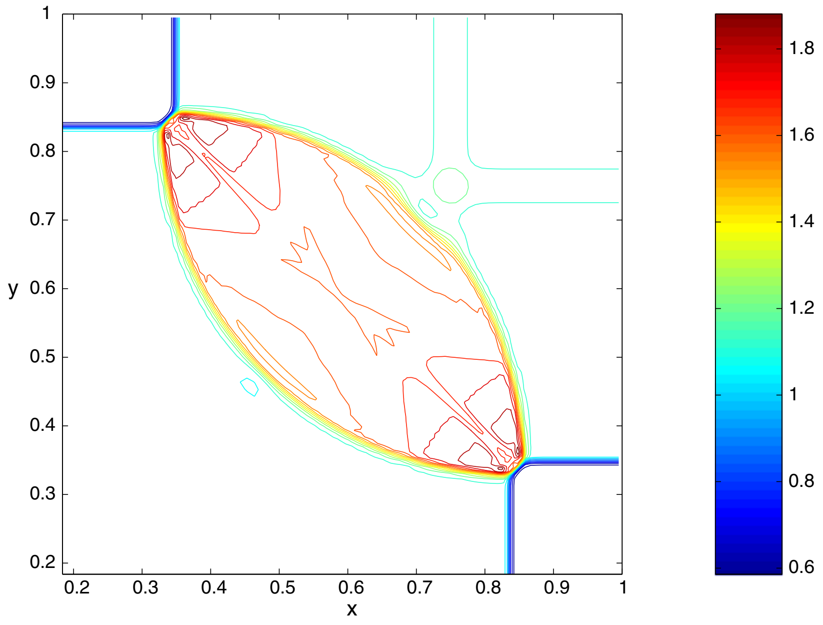


Fig. 34. Uniformly fine calculation density contours for 2D Riemann problem $t = 0.2$.

$$
\begin{array}{llll}
p_2 = 0.3500 & \rho_2 = 0.5065 & p_1 = 1.1000 & \rho_1 = 1.1000 \\
u_2 = 0.8939 & v_2 = 0.0000 & u_1 = 0.0000 & v_1 = 0.0000 \\
p_3 = 1.1000 & \rho_3 = 1.1000 & p_4 = 0.3500 & \rho_4 = 0.5065 \\
u_3 = 0.8939 & v_3 = 0.8939 & u_4 = 0.0000 & v_4 = 0.8939.
\end{array}
$$

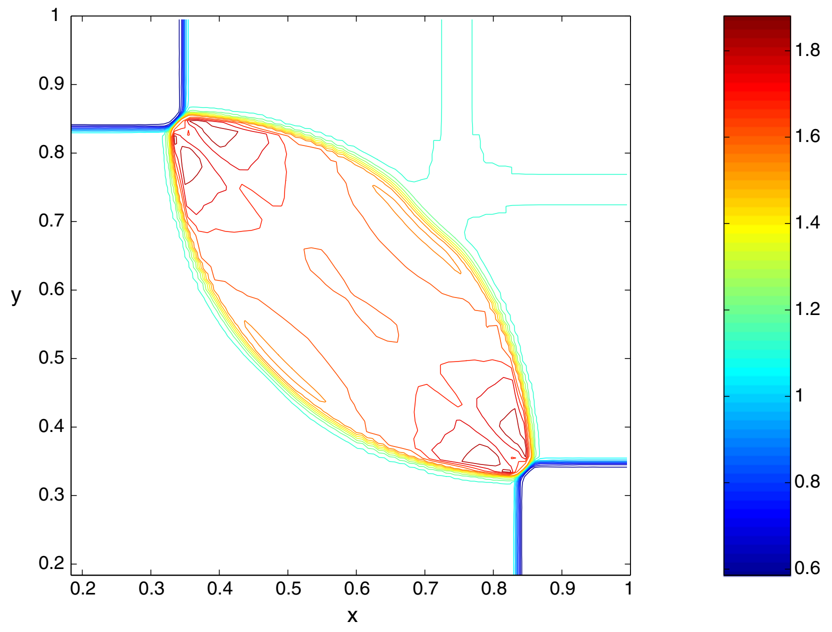


Fig. 35. Anisotropic calculation density contours for 2D Riemann problem $t = 0.2$.

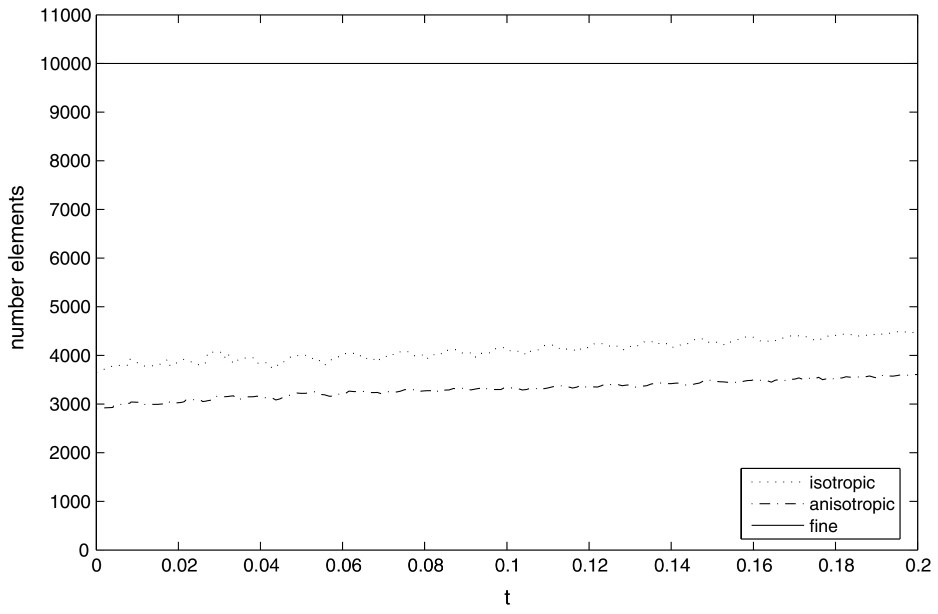


Fig. 36. Number of elements over time for 2D Riemann problem.

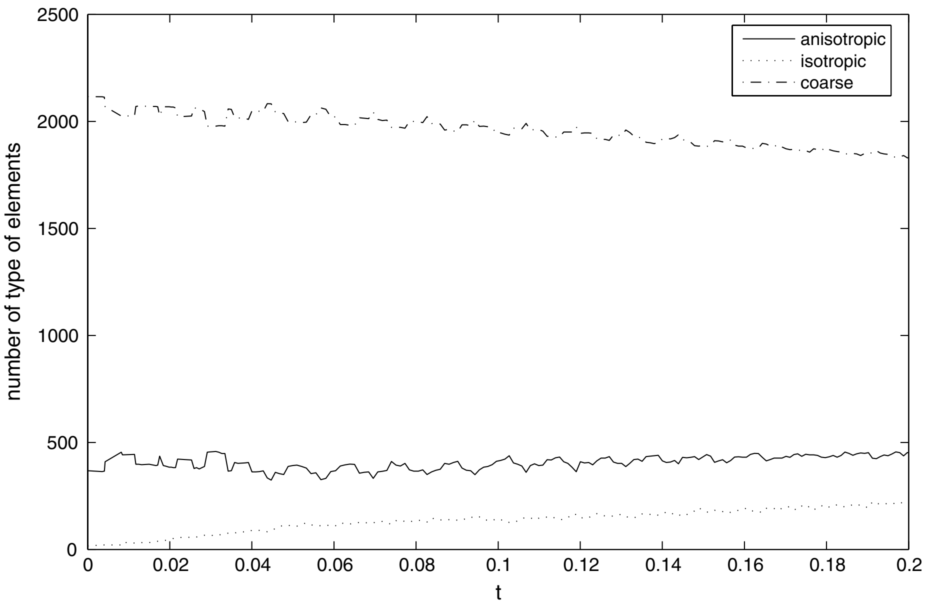


Fig. 37. Number of each type of element over time for 2D Riemann problem.

The refinement threshold was 0.15 and the derefinement value was 0.13. For the anisotropic calculation the isotropic refinement angle was 30° and the derefinement angle was 25° from each axis. The problem was run with anisotropic and isotropic refinement on a 50×50 initial mesh.

The anisotropic method performs very well on this problem, as illustrated by comparing Fig. 32 with Fig. 33. The majority of the refinement is anisotropic. Isotropic refinement only occurs where the shocks intersect and where the circumference of the oval does not align with the direction of the mesh. Within the high density oval cells have derefined completely. The density contours, see Fig. 35, are comparable with those achieved from a uniformly fine calculation, see Fig. 34.

Fig. 36 shows that the anisotropic calculation requires significantly less elements per time step. The majority of the refinement was anisotropic, see Fig. 37. The number of isotropic refinements increased slightly as the oval became larger. This problem highlights the significant reduction in calculation time that can be achieved with anisotropic refinement, the isotropic calculation runs 5.5 times faster than the uniformly fine calculation, while the anisotropic calculation runs 8.6 times faster.

7. Conclusions

This paper has highlighted the feasibility and practicality of combining a cell by cell anisotropic adaptive mesh technique with an ALE code. The Lagrangian, equipotential mesh relaxation and advection steps have been generalised to be applied on the dynamic mesh containing coarse and fine elements. Excellent results were achieved in an eighth of the original time for a radial Sod problem and a two-dimensional Riemann problem. Proposed future work includes the generalisation of the data structure and code to many refinement levels.

Acknowledgement

We thank Dr. Chris Powell for his help on the isotropic adaptive mesh code. This work was funded by A.W.E. Aldermaston.

References

[1] M. Aftosmis, Upwind method for simulation of viscous flow on adaptively refined meshes, AIAA J. 32 (2) (1994) 268.

[2] R.W. Anderson, N.S. Elliott, et al., An Arbitrary Lagrangian–Eulerian method with adaptive mesh refinement for the solution of the Euler equations, J. Comput. Phys. 199 (2004) 598.

[3] T. Apel, S. Grosman, P. Jimack, A. Meyer, A new methodology for anisotropic mesh refinement based upon error gradients, Appl. Numer. Math. 50 (2004) 329.

[4] S. Baden, Structured Adaptive Mesh Refinement (SAMR) Grid Methods, IMA Volumes in Mathematics and Its Applications, vol. 117, Springer, 2000.

[5] A. Barlow, An adaptive multi-material Arbitrary Lagrange Eulerian algorithm for computational shock hydrodynamics, Ph.D. thesis, Swansea, 2002.

[6] D. Benson, Computational methods in Lagrangian and Eulerian hydrocodes, Comput. Methods Appl. Mech. Engrg. 99 (1992) 235.

[7] D. Benson, An efficient, accurate simple ALE method for nonlinear finite element programs, Comput. Methods Appl. Mech. Engrg. 72 (1989) 305.

[8] D. Benson, Momentum advection on a staggered mesh, J. Comput. Phys. 100 (1992) 143.

[9] M.J. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1989) 64.

[10] M.J. Berger, Adaptive mesh refinement for hyperbolic partial differential equations, Ph.D. thesis, Stanford University, 1982.

[11] C. Hirt, A. Amsden, et al., An Arbitrary Lagrange-Eulerian computing method for all flow speeds, J. Comput. Phys. 14 (3) (1974) 227.

[12] Y. Kallinderis, J. Baron, Adaptation methods for a new Navier–Stokes algorithm, AIAA J. 27 (1) (1989) 37.

[13] W. Keats, F. Lien, Two-dimensional anisotropic Cartesian mesh adaptation for the compressible Euler equations, Int. J. Numer. Methods Fluids 46 (2004) 1099.

[14] A. Khokhlov, Fully threaded tree algorithms for adaptive refinement fluid dynamics, J. Comput. Phys. 143 (1998) 519.

[15] A. Kurganov, E. Tadmor, Solution of two-dimensional Riemann problems for gas dynamics without Riemann problem solvers, Numer. Methods Partial Differ. Equat. 18 (5) (2002) 584.

[16] J.M. Morrell, A cell by cell anisotropic adaptive mesh Arbitrary Lagrangian Eulerian method for the numerical solution of the Euler equations, Ph.D. thesis, Reading, 2007.

[17] T. Plewa, T. Linde, V. Weirs, Adaptive mesh refinement – Theory and applications, Lecture Notes Comput. Sci. Eng. 41 (2005).

[18] J.J. Quirk, An adaptive algorithm for computational shock hydrodynamics, Ph.D. thesis, Cranfield, 1991.

[19] J.F. Thompson, F.C. Thames, C.W. Mastin, TOMCAT – A code for numerical generation of boundary-fitted curvilinear coordinate systems on fields containing any number of arbitrary two-dimensional bodies, J. Comput. Phys. 24 (1977) 274.

[20] E.F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, second ed., Springer-Verlag, 1999.

[21] J. van der Vegt, H. van der Ven, Discontinuous Galerkin finite element method with anisotropic local grid refinement for inviscid compressible flows, J. Comput. Phys. 141 (1998) 46.

[22] van Leer, Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection, J. Comput. Phys. 23 (1977) 276.

[23] van Leer, Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method, J. Comput. Phys. 32 (1979) 101.

[24] B. Wells, A moving mesh finite element method for the numerical solution of partial differential equations and systems, Ph.D. thesis, University of Reading, 2004.

[25] A.M. Winslow, Numerical solution of the quasilinear poisson equation in a non-uniform triangular mesh, J. Comput. Phys. 1 (1966) 149.