Anisotropic Adaptivity for Finite Element Solutions of 3-D Convection-Dominated Problems

M. Walkley P. K. Jimack M. Berzins
Computational PDE Unit, School of Computing, University of Leeds

1 Introduction

In this work we investigate the use of anisotropic mesh refinement algorithms for the adaptive finite element (FE) solution of three-dimensional convection-dominated flow problems. Conventional mesh adaptivity, based upon *a posteriori* error estimation and regular h-refinement for example, is not always efficient for this class of problem; where solutions often exhibit strongly directional features such as sharp layers or shocks. In these regions of the solution isotropic mesh refinement results in an unnecessary level of resolution parallel to the flow feature being captured in order to deliver the required resolution across the feature. In many such situations an appropriate anisotropic mesh can yield an equivalent accuracy using substantially fewer degrees of freedom, and therefore at a potentially significantly reduced computational cost.

The fundamental issue that is considered in this paper is that of how to automatically adapt a three-dimensional tetrahedral mesh in order to obtain appropriate anisotropic meshes for convection-dominated problems. Our approach is to combine a standard local h-refinement algorithm, [8], with the use of local node movement in order to either drive down a norm of the residual, [7], or a local *a posteriori* error estimate, [2]. Details of these techniques are provided in the following two sections. Following this, in section 4, we present a modification of the node movement algorithm which permits nodes at different levels of the mesh hierarchy to be moved independently. An example is presented which demonstrates the advantage of being able to move the nodes on the coarsest mesh only, dragging higher level nodes with them. The paper concludes with a brief discussion of how we are currently developing this work to permit the solution of time-dependent equations and systems.

2 Least-square residual minimization

Here we present a three-dimensional generalization of previous work in two dimensions, [4, 7, 9], which aims to combine node movement (frequently referred to as r-refinement) with local h-refinement in order to drive down a given functional (which is bounded below). For simplicity we describe the application of our technique to a simple linear hyperbolic model problem of the form

$$(\underline{a} \cdot \underline{\nabla})u = f \quad \text{in } \Omega = (0,1)^3, \tag{2.1}$$

$$u = \begin{cases} 1 - x/\delta & x < \delta \\ 0 & x \geq \delta \end{cases} \quad \text{on } \Gamma_{in} = \{\underline{x} \in \partial\Omega : \underline{a} \cdot \underline{n}(\underline{x}) < 0\}, \tag{2.2}$$

where $\underline{n}(\underline{x})$ is the unit outward normal to the boundary $\partial\Omega$. Figure 1 illustrates the nature of the solution of this problem when $f(\underline{x}) = 0$, $\underline{a} = (2,1,1)^T$ and $0 < \delta \ll 1$. The figure shows the isosurface $u(\underline{x}) = 0.5$ and the solution changes rapidly from 0 to 1 as one moves across this isosurface. In all of the calculations which follow we use $\delta = 0.01$ and the above choices of $f(\underline{x})$ and $\underline{a}$.

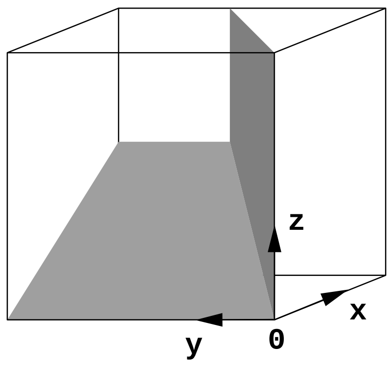


Figure 1: The strongly directional layer that is present in the solution of the model problem

In order to define a suitable functional for minimization we consider the residual of (2.1) for a piecewise linear approximation to u, u^h say, on a tetrahedral mesh which covers Ω:

$$r^h = f - (\underline{a} \cdot \underline{\nabla}) u^h \; . \tag{2.3}$$

Following [7] we now introduce the functional

$$R(u^h) = \frac{1}{2} \int_{\Omega} (r^h)^2 \, d\underline{x} = \frac{1}{2} \sum_k \int_{\Omega^k} (r^h)^2 \, d\underline{x} \; , \tag{2.4}$$

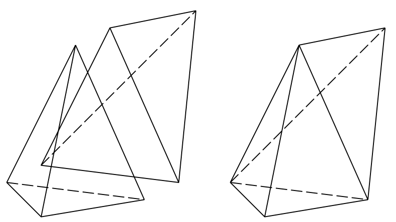
where the summation is over all tetrahedra Ω^k in the given mesh. The optimization scheme that we apply is as follows.

1. Order the nodes of the mesh according to their distance downstream of the inflow boundary.
2. For each node i (located at $\underline{x}_i$, say) in this ordering:
   (a) find $\frac{\partial R}{\partial \underline{x}_i}$ (details of this calculation are given in Appendix A),
   (b) perform a 1-d minimization of R in this direction of steepest decent,
   (c) solve a local version of (2.1) on the elements surrounding node i to update the solution values at this node and those on the downstream boundary of this patch.

   Repeat this step a fixed number of times (typically 3 times in our examples).
3. For each internal face in the mesh consider reconnecting the union of the two tetrahedra sharing this face into three tetrahedra as shown in Figure 2: the topology yielding the lower value of (2.4) being accepted.
4. Perform h-refinement on those elements with a local L^2 residual greater than 20% (say) of the maximum residual on any element and recompute the global least-squares solution.

These steps may be repeated until a desired value of the least-squares residual is obtained or a maximum number of elements has been reached. The need for the simple ‘edge-swapping’ sweep (step 3) is illustrated in 2-d in [4, 7, 10] for example. This assists with the alignment of the edges in the mesh with the flow features that we wish to capture.

Table 1 illustrates some sample results obtained when the proposed scheme is applied to our linear model problem. It is apparent that, measured either in terms of the residual or the exact error, the hr-refinement approach provides a significant improvement over the use of local h-refinement alone (in this calculation a similar accuracy is obtained with just a fifth of the number of node points (Np) for example).


(a) Two initial tetrahedra

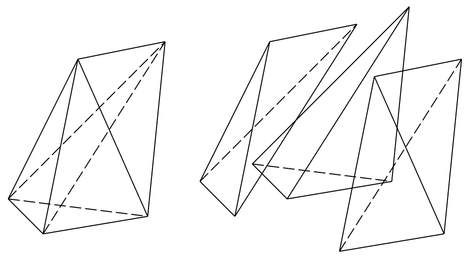

(b) Three modified tetrahedra

Figure 2: Local reconnection operation for the union of 2 tetrahedra

	hr-refinement sequence				
	Solve	Optimize	Adapt	Solve	Optimize
Np	729		3072		
$R(u^h)$	1.016	0.381	0.381	0.189	0.130
$\|\|e\|\|_1$	0.075	0.056		0.034	0.031

	h-refinement		
	Solve	Solve	Solve
Np	729	3313	15353
$R(u^h)$	1.016	0.428	0.180
$\|\|e\|\|_1$	0.075	0.049	0.031

Table 1: Sample results using the residual minimization scheme

3 Minimization of estimated *a posteriori* error

As an alternative to controlling the mesh adaptivity process through the local residual we now consider the use of an *a posteriori* error estimate. Provided that the error can be estimated sufficiently accurately it is clearly more desirable to control this quantity. Our approach is similar to that of [2], for example, where it is observed that subtraction of (2.1) from (2.3) yields a hyperbolic error equation

$$(\underline{a} \cdot \underline{\nabla})e = r^h \tag{3.1}$$

for $e = u - u^h$. This PDE is solved subject to an exact inflow boundary condition on Γ_{in} but using either a different mesh or numerical scheme from that applied to obtain u^h. In this work we use a stable streamline-diffusion linear finite element method, [6], to calculate u^h and a cell-centred finite volume scheme, [1], to compute e^h (an estimate of e) on each element. The adaptivity algorithm that we then apply is as follows.

1. For each node in the mesh:

   (a) find
   $$\underline{x}_i^{av} = \frac{\sum_{k\in\Omega_i} |e_k^h| \underline{x}_k^c}{\sum_{k\in\Omega_i} |e_k^h|}$$
   where $\Omega_i = \{k : \underline{x}_i \in \overline{\Omega^k}\}$, e_k^h is the computed error in cell k and $\underline{x}_k^c$ is the position of the centroid of cell k,

   (b) set $\underline{x}_i := (1-\gamma)\underline{x}_i + \gamma \underline{x}_i^{av}$, where $\gamma \in (0,1)$ is an under-relaxation parameter (typically $\gamma = 0.5$).

   Repeat this step a fixed number of times (e.g. 2).

2. Perform h-refinement on those elements with an estimated error greater than 20% (say) of the maximum error on any element and recompute the global solution and error estimate.

As in the previous section, these steps may be repeated until a desired value of the estimated error is obtained or a maximum number of elements has been reached. Note that, unlike for the residual minimization approach there is no straightforward mechanism for incorporating edge-swapping into this algorithm.

Table 2 presents some sample results when this algorithm is used to solve the linear model problem previously considered. In order to allow the dependency on our choice of solver for (3.1) to be assessed additional statistics are presented to show how the algorithm performs when the exact error, which is known for this problem, is used rather than the estimated error e^h. In both cases the hr-refinement scheme again outperforms standard h-refinement, with similar errors requiring about a third the number of node points. An indication of the contrasting meshes produced by these two forms of adaptivity is provided in Figure 3.

h-refinement				hr-refinement			
Driven by e		Driven by e^h		Driven by e		Driven by e^h	
Np	$\|\|e\|\|_1$	Np	$\|\|e\|\|_1$	Np	$\|\|e\|\|_1$	Np	$\|\|e\|\|_1$
729	0.111	729	0.111	729	0.111	729	0.111
					0.104		0.104
					0.094		0.102
2761	0.070	2799	0.079	2826	0.052	3021	0.069
					0.050		0.065
					0.046		0.062
12026	0.045	11437	0.051	12412	0.038	12260	0.043
					0.033		0.039
					0.029		0.037
52871	0.027	44105	0.032	50737	0.020	44051	0.026
					0.018		0.023
					0.016		0.022

Table 2: Comparison of the L^1 norm of the error when refinement is based upon the estimated error

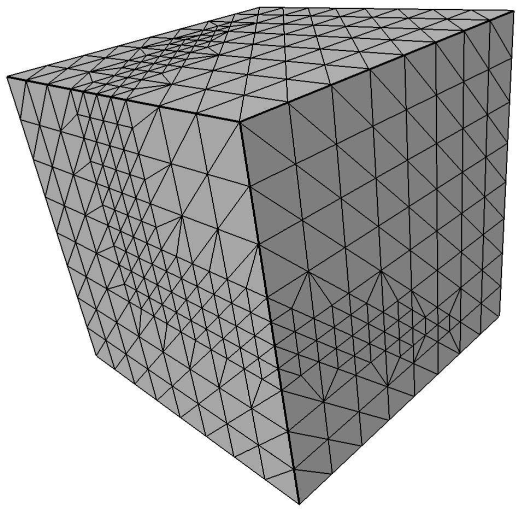
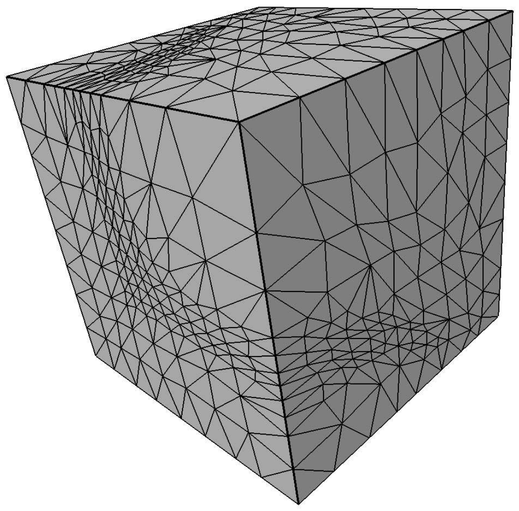



Figure 3: Two sample meshes obtained during the solution of the model problem: the first, obtained using h-refinement, contains 13002 elements but leads to a solution error which is almost twice that of the second grid, containing 13343 elements, obtained using hr-refinement

4 Hierarchical r-refinement

It has been demonstrated that the inclusion of node movement within each of the adaptive algorithms considered leads to better quality meshes than are otherwise obtained. Furthermore, from Figure 3, it may be observed that this node movement allows the element shape to deform in line with the anisotropic features of the solution. It should be noted however that the cost of the node movement grows significantly as the size of the finite element mesh increases since the position of every node must be updated independently at each r-refinement step. The main development that we consider in this section therefore is to utilize the h-refinement hierarchy in order to move only the coarse mesh nodes independently, with the movement of the nodes produced at finer mesh levels being dependent upon this.

The proposed approach is to follow essentially the same algorithms as outlined in the previous sections but, instead of looping through every node in the mesh at the r-refinement stage, only the nodes contained in the initial mesh (i.e. the root mesh, at the lowest level of the h-refinement hierarchy) are visited. When the position of one of these nodes is updated then so too is the position of all nodes in the interior of the patch of root mesh elements surrounding this node. This is illustrated for a two-dimensional example in Figure 4. Note that the dependent node positions are updated so that their barycentric coordinates with respect to the root mesh elements remain unchanged.

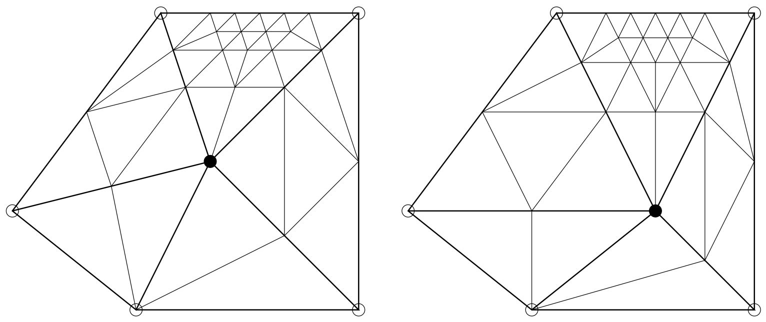


Figure 4: The movement of a root mesh node in two dimensions

For clarity we describe the detail of this approach in the context of the residual minimization procedure of Section 2. In particular, we note that the 'edge-swapping' phase may still be completed in principle although it does add significantly to the overall complexity of the implementation. The calculation of $\frac{\partial R}{\partial \underline{x}_i}$ at step 2(a) is also made more expensive since

$$\frac{\partial R}{\partial \underline{x}_i} = \sum_{j \in \mathcal{P}_i} \theta_{i,j} \frac{\partial R}{\partial \underline{x}_j} , \tag{4.1}$$

where $\mathcal{P}_i = \{j : \underline{x}_j \in \{\text{points in those root mesh elements with a vertex at } \underline{x}_i\}\}$ and $\theta_{i,j}$ is the barycentric coordinate of $\underline{x}_j$ with respect to the vertex of its parent root element that is $\underline{x}_i$. It should be noted however that all expressions (4.1), i.e. for each root node i, may be assembled together in a single loop through the leaf mesh elements (making use of expression (A.4) in Appendix A). Table 3 presents a comparison of using this root node movement approach with the original hr-refinement algorithm for a slightly different test problem to (2.1) which has the solution $u(\underline{x}) = e^{-x_1/\delta}$. When δ is small (0.01 in this example) this solution has a boundary layer next to $x_1 = 0$. It is apparent that, for this particular problem, coarse mesh movement is advantageous since it allows the nodes to move into the boundary layer region near $x_1 = 0$ more quickly than with the original r-refinement approach.

	Original hr-refinement			Modified hr-refinement		
	Np	Functional	$\|e\|_1$	Np	Functional	$\|e\|_1$
Global solve	4913	53.45	0.0237	4913	53.45	0.0237
r-refine/local solve		13.78	0.0059		7.39	0.0028
h-refine/global solve	13536	4.05	0.0016	13155	2.05	0.0007
r-refine/local solve		2.57	0.0010		1.88	0.0007
h-refine/global solve	40186	1.47	0.0004	38606	0.54	0.0002
r-refine/local solve		1.33	0.0004		0.54	0.0002

Table 3: An example where the root node movement strategy proves to be advantageous over the original version of the algorithm in which all node positions are updated independently

5 Discussion

The generalization of both the least-squares solution technique and the *a posteriori* error estimate to linear systems of equations is relatively straightforward and hence the adaptive algorithms introduced in this paper may both be applied to such problems. Clearly, if different components of the solution have different directional behaviour then the scope for using anisotropic meshes will be reduced, however for many physical problems similar directional solution features are present in all components and the benefits of our hr-refinement strategies will again become apparent. Extensions to time-dependent problems are also under development with the aid of the hierarchical r-refinement described in Section 4. By only moving the root mesh nodes independently it is still possible to apply both h-refinement and derefinement (necessary for time-dependent problems, [8]) without disturbing the mesh hierarchy within the adaptive h-refinement code.

Acknowledgments

This work was funded by EPSRC research grant GR/M00077.

References

[1] N.T. Frink. Recent progress towards a three-dimensional unstructured Navier-Stokes flow solver. Technical Report 94-0061, AIAA, 1994.

[2] P. Houston and E. Süli. A posteriori error indicators for hyperbolic problems. Technical Report 97/14, Numerical Analysis Group. Oxford University Computing Laboratory, 1997.

[3] P.K. Jimack. A best approximation property of the moving finite element method. *SIAM J. Num. Anal.*, 33:2286–2302, 1996.

[4] P.K. Jimack and R. Mahmood. A multilevel approach for obtaining locally optimal finite element meshes. In B.H.V. Topping, editor, *Developments in Engineering Computational Technology*, pages 191–197. Civil-Comp Press, 2000.

[5] P.K. Jimack and A.J. Wathen. Temporal derivatives in the finite element method on continuously deforming grids. *SIAM J. Num. Anal.*, 28:990–1003, 1991.

[6] C. Johnson. *Numerical Solutions of Partial Differential Equations by the Finite Element Method.* Cambridge University Press, 1987.

[7] P. Roe. Compounded of many simplices. Reflections on the role of model problems in CFD. In *Barriers and Challenges in Computational Fluid Dynamics*, pages 241–258. Kluwer Academic, 1998.

[8] W. Speares and M. Berzins. A 3d unstructured mesh adaptation algorithm for time-dependent shock dominated problems. *Int. J. Num. Meth. Fluids.*, 25:81–104, 1997.

[9] Y. Tourigny and F. Hülsemann. A new moving mesh algorithm for the finite element solution of variational problems. *SIAM J. Num. Anal.*, 35(4):1416–1438, 1998.

[10] M. Walkley, P.K. Jimack, and M. Berzins. Mesh quality for three-dimensional finite element solutions on anisotropic meshes. In *Proceedings of FEM3D, Univ. of Jyvaskyla, Finland, June 27-30*, 2000.

Appendix A Nodal derivatives of the residual functional

In order to determine the direction of steepest decent in the residual minimization algorithm of Section 2 it is necessary to evaluate expressions of the form $\frac{\partial R}{\partial \underline{x}_i}$ where $\underline{x}_i$ is the location of node i. In order to do this we use the following two results which are proved in [5, Theorem 2.4] and [3, Lemma 3.1] respectively:

$$\frac{\partial u^h}{\partial \underline{x}_i} = -N_i \underline{\nabla} u^h \ , \tag{A.1}$$

$$\frac{\partial V^k}{\partial \underline{x}_i} = V_k \underline{\nabla} N_i \ . \tag{A.2}$$

Here V^k is the volume of any simplex Ω^k which has a vertex at $\underline{x}_i$ and N_i is the usual piecewise linear basis function which has value 1 at $\underline{x}_i$. From (2.3) and (2.4) we have

$$\begin{aligned}
\frac{\partial R}{\partial \underline{x}_i} &= \frac{1}{2} \sum_{k \in \Omega_i} \frac{\partial}{\partial \underline{x}_i} \int_{\Omega^k} (f - (\underline{a} \cdot \underline{\nabla}) u^h)^2 \, d\underline{x} \\
&\quad \text{(where } \Omega_i = \{k : \underline{x}_i \in \overline{\Omega^k}\}\text{)} \\
&= \frac{1}{2} \sum_{k \in \Omega_i} \frac{\partial}{\partial \underline{x}_i} \int_{\Delta} (f - (\underline{a} \cdot \underline{\nabla}) u^h)^2 V^k \, d\underline{\xi} \\
&\quad \text{(where } \Delta \text{ is some reference tetrahedron with unit volume)} \\
&= \sum_{k \in \Omega_i} \int_{\Delta} \left[(f - (\underline{a} \cdot \underline{\nabla}) u^h)(-(\underline{a} \cdot \underline{\nabla}) \frac{\partial u^h}{\partial \underline{x}_i}) V^k + \frac{1}{2} (f - (\underline{a} \cdot \underline{\nabla}) u^h)^2 \frac{\partial V^k}{\partial \underline{x}_i} \right] d\underline{\xi} \\
&= \sum_{k \in \Omega_i} \int_{\Omega^k} \left[(f - (\underline{a} \cdot \underline{\nabla}) u^h)((\underline{a} \cdot \underline{\nabla}) N_i) \underline{\nabla} u^h + \frac{1}{2} (f - (\underline{a} \cdot \underline{\nabla}) u^h)^2 \underline{\nabla} N_i \right] d\underline{x} \qquad \text{(A.3)} \\
&\quad \text{(where (A.1) and (A.2) have been applied).}
\end{aligned}$$

Hence, when $f(\underline{x}) = 0$, as in our example problem, expression (A.3) simplifies further to yield

$$\frac{\partial R}{\partial \underline{x}_i} = \sum_{k \in \Omega_i} V^k ((\underline{a} \cdot \underline{\nabla}) u^h)_k \left(\frac{1}{2} ((\underline{a} \cdot \underline{\nabla}) u^h)_k \underline{\nabla} N_i - ((\underline{a} \cdot \underline{\nabla}) N_i)_k \underline{\nabla} u^h \right) \ , \tag{A.4}$$

where $(\cdot)_k$ denotes the restriction of the quantity within the brackets to element k.