Adjoint *a posteriori* error measures for anisotropic mesh optimisation

P.W. Power [a,*], C.C. Pain [a], M.D. Piggott [a], F. Fang [a], G.J. Gorman [a], A.P. Umpleby [a], A.J.H. Goddard [a], I.M.Navon [b]

[a] *Applied Modelling and Computation Group, Department of Earth Sciences and Engineering, Imperial College London London, SW7 2BP. UK.*

[b] *Department of Mathematics Florida State University Tallahassee FL, 32306-4120. USA.*

Abstract

In this paper an adjoint (or sensitivity) based error measure is formulated which measures the error contribution of each solution variable to an overall *goal*. The goal is typically embodied in an integral functional e.g. the solution in a small region of the domain of interest. The resulting *a posteriori* error measures involve the solution of both primal and adjoint problems. A comparison of a number of important *a posteriori* error measures is made in this work. There is a focus on developing relatively simple methods that refer to information from the discretized equation sets (often readily accessible in simulation codes) and do not explicitly use equation residuals. This method is subsequently used to guide anisotropic mesh adaptivity of tetrahedral finite elements. Mesh adaptivity is achieved here with a series of optimization heuristics of the landscape defined by mesh quality. Mesh quality is gauged with respect to a Riemann metric tensor embodying an *a posteriori* error measure, such that an ideal element has sides of unit length when measured with respect to this metric tensor. This results in meshes in which each finite element node has approximately equal (subject to certain boundary conforming constraints and the performance of the mesh optimization heuristics) error contribution to the functional (goal).

* Corresponding author.
Email address: philip.power@imperial.ac.uk (P.W. Power).

Preprint submitted to Computers & Mathematics with Applications30 September 2005

1 Introduction

Mesh adaptation or optimisation algorithms require the derivation of an appropriate error measure. This error measure 'guides' the adaptivity algorithm, or in other words decides how the mesh is to be modified. Numerous error measures to serve this purpose have been presented in the literature. For example, methods developed to measure error with respect to a given energy norm [1, 2, 3]; interpolation based methods [4, 5, 6, 7] calculate an *a priori* measure of the error based on both the local mesh size and some higher order derivative of the exact solution, which must typically be obtained in practice using recovery from the numerical solution; there also exist various types of explicit and implicit *a posteriori* error measures [8, 9] as well as the implicit equation residual approach [10, 11, 12, 13, 14, 15, 16] in which the same set of equations are solved for the errors, with sources given by the residuals of the governing equations. These are the only methods which can correctly propagate the errors through the domain.

The literature contains various approaches to mesh optimization for tetrahedral elements, for example [17, 18, 19, 20, 21]. The Imperial College Ocean Model (ICOM) [22, 23, 24] utilizes dynamic adaptation of a fully unstructured tetrahedral mesh in three-dimensions (3-D), as presented in [21]. This technique uses a form of *h-refinement* (or mesh optimization) to adapt the mesh, changing the size, shape and location of tetrahedral elements to optimize the mesh according to specific criteria, as defined by an error measure. The algorithm is based on a series of mesh connectivity and node position searches, defining the mesh quality. A Riemannian metric tensor reflecting the error measure is used to calculate the desired element size and, importantly, shape. A functional is used to gauge the mesh quality, this functional embodies both element size and shape with respect to the metric tensor. A local based search strategy is adopted to carry out the adaptation operations: node smoothing; edge and face-edge swapping; and edge splitting and collapsing, to minimize the functional. The algorithm is robust, produces high quality anisotropic meshes, and has a time complexity which varies linearly with the number of elements, see [21]. The anisotropy of the method offers both substantial computational improvements over fixed mesh methods and feature-following opportunities.

Other approaches to mesh adaptation can also be applied using the error measures presented. Unstructured tessellation methods can deal with geometries of arbitrary complexity and render themselves naturally to adaptivity. There are three main approaches to the generation of unstructured tetrahedral meshes, Quadtree/Octree methods [25], the Advancing Front technique [26, 27] and Delaunay schemes [19]. In general, good quality 2-D meshes can be created through Delaunay schemes which can lead to mesh adapting schemes for trian-

gular [28] or quadrilateral [29] elements; however in 3-D element aspect ratios can become very large [19]. Further work on 3-D Delaunay methods can be found in [30, 31, 32, 33, 34].

The error measure utilized in this work is one based upon both the curvature of the solution (which provides directional information) and a required specification of an appropriate interpolation error, derived from a goal-based method. The required interpolation error varies in time and space through the simulation. Scaling of the resulting metric tensor [21, 32] allows phenomena acting at various scales to be resolved. The overall approach effectively uses the current solution of the problem in hand to adapt the mesh to reflect future 'activity' in the flow.

The motivation for the incorporation of sensitivity analysis in error measure design stems from the assumption that it is ineffective and undesirable (computationally speaking) to adapt the mesh every time step (or few time steps); hence in a dynamically evolving flow the mesh could be said to be 'behind the flow', and therefore not in an optimal form to resolve features and provide the best solution. The method presented here is our first step toward being able to 'predict' the flow's future movement (at least to highlight possibly important areas), and adapt the mesh accordingly.

Sensitivity analysis deals with the calculation of the gradients of a model forecast with respect to model parameters, where these parameters might be initial conditions, boundary conditions or other physical inputs. Use of an adjoint model, first introduced in [35], can identify regions where changes to variables or parameters has the largest impact. Put simply, an area with high 'sensitivity' is one within which small perturbations can strongly influence the growth of errors in the overall solution. Sensitivity analysis has been applied in a variety of fields, including the control of water through irrigation channels [36] and contaminant releases in rivers [37], applications to the shallow water equations [38], as well as various meteorological applications [39, 40, 41]. Theoretical considerations have been presented by [42, 43], with later applications to optimal control, 3/4D-VAR data assimilation and error estimation [44].

Adjoint models can be derived with either of two approaches: the continuous approach based on the Euler-Lagrange equations, see [45, 46] or the discrete approach in which the discrete representation of the non-linear problem is differentiated, see [47, 48]. It is only the latter which is completely consistent with the discretized representation of the forward equations, and as such is the ideal. Some discretization methods however are complex (such as non-linear methods, non-linear Petrov-Galerkin [49] or flux limiting methods [50]) and as such an Euler-Lagrange approach may be more convenient. Indeed this is also the case if a different mesh is used for the forward and adjoint computations when the meshes are adapted independently to optimize the accuracy of the

forward and adjoint solutions individually, see [51].

Although in principle automatic differentiation [52, 53] may be used to form gradients (or sensitivities), some aspects of the solution method may be hard (or even impossible e.g. decision branches in the code) to differentiate. A big advantage of using the consistent discrete approach is that the correct boundary conditions are automatically applied to the adjoint equations. Although error measures are developed in this work from the discrete equations (see section 3), it will been seen that the resulting methods are amenable to either approach.

An issue for *real* problems is time dependence. Problems are typically solved using time stepping methods which march forward in time while not necessarily storing all the previous time levels. The sensitivity approach described here requires access to all time levels. For example, if the mesh is to be adapted to optimize the accuracy of the model at an observation point then there is a need to march forward all the way through time recording the model result at the observation point as a time series, then propagating, with an adjoint model, backward in time with the observations from the end of the time domain to the start of the time domain. Complications arise because of the fact that the adjoint model needs access to the forward solution (e.g. the velocities) in order to back propagate information. High width trees via checkpointing can provide a memory efficient alternative to storing the forward solution at every time level. However, this does require solving the forward problem several times in order to obtain solution variable sensitivity information from which the mesh can be adapted. For many problems the super-convergence (see [54]) properties of the target functional may well warrant an approach like this, but it may be complicated by multiple solutions and the chaotic nature of such flows (exponential divergence of the flow behavior following a small perturbation in the solution, such as would be provided by a different mesh).

Since the adjoint solution is typically used in data assimilation (correlating the model with observations to obtain suitable initial conditions say) and optimization (industrial plant based optimization of the efficiency of process technology) then this work provides an additional use for this adjoint information. That is, to adapt the mesh and provide indicators of the accuracy of the functional or goal (e.g. the efficiency of a process). These methods provide the framework for optimizing the accuracy of the inverse problem as this is equivalent to optimizing the accuracy of the functional (goal) in the primal solution.

The aim of the work presented here is to take the emphasis off the numerical analyst in designing error norms and put it in the domain of the engineer or physicist who has a firm understanding of the problem in hand. An engineer or physicist would typically want to extract a particular quantity from the

numerical simulation, for example the solution at a range of points or an integral quantity. The aim of goal-based error measure design is to take a measure of what is deemed 'important' in a problem and design an error measure, and consequently mesh adaptation scheme, to optimize the accuracy of this quantity. A bound on the required accuracy for this quantity (goal) can be set, and the method presented in this paper yields a mesh which achieves this level of accuracy with minimal computational resources. A method of this type is particularly important for problems with a number of solution variables where it is typically unclear what priority to put on resolving each of the solution fields. The method developed here provides a systematic way of doing this.

The error contribution to each of the nodal solution variables can be determined and used to substantially improve the accuracy of the goal functional, see [54, 55]. In addition, sensitivity information may also be used to provide a bound for the error in this functional, which can be invaluable to any model, see [2]. An example of an application area of this type of technique is in the adaptation of a mesh to optimize a quantity of interest in a fluid simulation such as the drag or lift past an aerofoil [54]. The particular interest here is for an ocean model, as described by [22, 23, 24]. The goal function in such a model could be an observation or some measure of the dynamics of the system, for example some integral of vorticity, the strength of the thermohaline circulation or some more novel measures and diagnostics as explored by [56, 57, 58].

The tendency of an adapting mesh to refine down to a scale able to resolve molecular viscosity in large scale turbulent flows (e.g. an ocean) can be reduced, as the adaptation technique can be guided by the measure of what is deemed important in the model. Although there are serious complications associated with the non-linearity of systems, such as the Boussinesq equations, permitting multiple and chaotic solutions, these error measures will still provide useful mesh adaptivity guides.

What distinguishes this work from previous work on goal-based error measures and adaptivity, for example in [59, 60, 61, 62], is the use of readily accessible (in simulation codes) discretized equations, the application to transient problems and the use of a metric tensor obtained from sensitivity analysis to adapt three-dimensional meshes of anisotropic unstructured tetrahedral elements. The metric tensor is used to gauge both shape and size quality of each tetrahedral element which forms the basis of a mesh adaptivity/optimization procedure. Since the adjoint solution is typically used in data assimilation, control, and optimization problems, then the method presented in this paper provides an additional use for the adjoint information.

By adapting the resolution of the mesh in space and direction to optimize the functional (in this case the quantity of interest or goal) the overall aim of this

work can be achieved. The method is demonstrated for simple 3-D advection-diffusion problems in which the solution accuracy at certain positions of the domain is optimized. Issues associated with solving problems with multiple solution variables are investigated here by solving a number of advection-diffusion problems on a single mesh.

This paper is organized as follows: the next section gives a brief overview of the adaptivity algorithm, followed by the derivation of two dual equations for the functional sensitivity in section 3; metric tensors and residual calculation methods are presented in sections 4 and 5; section 6 contains a summary of the computational aspects of the approach; sections 7 and 8 consider some results on simple test problems; finally conclusions are drawn.

2 Anisotropic Mesh Adaptivity Method

For completeness a brief overview of the mesh optimization algorithm employed in this work is given in this section. A more detailed description of the method is given by [21].

2.1 Metric tensor

The mesh optimization method presented requires as a precursor an error measure in the form of a nodally defined metric tensor [19]. This positive definite matrix defines, anisotropically, the desired mesh edge lengths at each node. The desired edge length, h_i, in the direction of the i^{th} eigenvector, $\mathbf{e}_i$, of the symmetric metric tensor $\mathbf{M}$, is defined as $h_i = 1/\sqrt{\Lambda_i}$, where Λ_i is the eigenvalue associated with $\mathbf{e}_i$. In general the metric tensor is used to calculate distances during mesh optimization via $\|\mathbf{v}\| = \mathbf{v}^T\mathbf{M}_\mathbf{v}\mathbf{v}$ where $\mathbf{M}_\mathbf{v}$ is the average metric tensor along vector $\mathbf{v}$. This approach may be viewed as a discretization of a Riemannian geometry constructed from an error norm. As the directionality of the solution is encoded in the metric tensor, anisotropic solutions will generally lead to anisotropic meshes that balance accuracy with computational efficiency. For example, the fact that the method seeks to attain a specific error means that the procedure will also remove surplus elements, thus improving computation efficiency.

When forming a suitable metric tensor a number of generic operations are regularly applied. These include: limiting the maximum and minimum desired edge lengths so as not to specify unrealistic modelling goals; limiting the aspect ratio of principal directions of an element; combining several metric tensors from different solution fields; global scaling of the metric tensor in order to

limit the computational resources the model may request (i.e. limit the number of degrees of freedom in the model); and metric tensor gradation control. Most of these operations are discussed extensively in the literature (e.g. [21, 19]) and so will not be elaborated on here.

2.2 Element functional

To define a mesh optimization problem an element functional is defined in terms of the metric tensor and of the properties a good mesh should exhibit for modelling. Trials are then performed on the local mesh connectivity and node position — in the case of a minimization problem being defined the aim is to find a local configuration that reduces the functional value. Defining the mesh functional as

$$\mathcal{F} = ||\boldsymbol{F}||_\infty, \tag{1}$$

where $\boldsymbol{F}$ is the vector of element functionals for the whole mesh, the process terminates when $\mathcal{F}$ falls below some tolerance. There are many possible choices for the definition of the local functional $\mathcal{F}$ (see [63] for a review). The element functional used here is geometrical based:

$$F_e = \frac{1}{2} \sum_{\ell \in \mathcal{L}_e} (r_\ell - 1)^2 + \mu \left(\frac{\alpha}{\rho_e} - 1 \right)^2 . \tag{2}$$

Here r_ℓ is the length, with respect to the edge centred metric tensor, $\mathbf{M}_\ell$, of edge ℓ; $\mathcal{L}_e$ is the set of edges of element e; ρ_e is the radius, with respect to the element centred $\mathbf{M}_e$, of the inscribed sphere (insphere) of element e; and α is the radius of the inscribed sphere of an ideal equilateral element. In 3-D the ideal element is defined in metric space as an equilateral tetrahedron with sides of unit length.

The trade-off between size and shape is controlled by the parameter μ, which has been chosen to be unity for the work here. The first term in the functional becomes zero as all of the edge lengths approach unity (measured with respect to the metric tensor $\mathbf{M}$), and the second term becomes zero as the in-sphere radius approaches α (measured with respect to the metric tensor $\mathbf{M}$). In this way the functional gives a measure of the quality of an element in terms of both its size and shape.

2.3 Mesh adaptations

For a given metric tensor and objective functional, the tetrahedral mesh is adapted through a combination of [21]:

- node insertion/deletion via edge splitting and collapsing,
- face-to-edge, edge-to-face and edge-to-edge swapping, [34]
- laplacian smoothing (in metric space),
- optimization-based node positioning.

It is worth noting that: edge collapsing reduces the number of elements and nodes thereby coarsening the mesh; edge splitting increases the number of elements and nodes, thereby refining the mesh; face and edge swapping, and mesh smoothing primarily serve to modify the shape of elements and do not generally alter the number of elements or nodes.

For the mesh optimization method employed here the surface geometry is modelled using discrete elements, i.e. the domain is represented using a piecewise linear approximation whose integrity is maintained throughout a simulation. This is to avoid conservation issues associated with using adaptive mesh methods with parametric surface descriptions.

Some of the adaptive operations are constrained when applied close to a geometrical boundary (whether an external surface or an internal boundary). For example, edge collapsing is normally effected by replacing the two nodes of an edge with another node at the mid-point of the edge. If one of those nodes lies on a geometrical boundary then the edge must be collapsed to that node. If both nodes lie on such a boundary then the edge cannot be collapsed.

Similarly, a face cannot be swapped to an edge if it forms part of such a boundary, and neither can an edge be swapped for another edge if the new edge would intersect a boundary. With regards to the node movement, if the node is located on a geometrical boundary then it's movement is constrained to that surface. Alternatively, if a node is on a geometrical edge (an intersection of boundaries) then it is constrained to move on that edge. If the node is on a geometrical corner (intersection of two or more boundaries) then it cannot move at all.

The optimization method visits each element of the mesh in turn. The adaptive operations, associated with an element, as listed above, are only proposed if the worst element is sufficiently 'bad' (i.e. if the maximum element functional is greater than some pre-determined value, 0.15 is used in the examples presented here). Once a local mesh change has been proposed, the maximum element functionals are compared before and after the change. If there is not sufficient reduction then the face/edge/node is flagged so that the change is not proposed again later on. Otherwise the proposed alteration is made to the mesh, the surrounding elements/edges/nodes (that have been affected by the change) are un-flagged (since their local situation has changed and they should be checked again), and the algorithm moves on to propose the next change to another face, edge or node.

This multi-pronged approach is similar to that of [17] except that the objective function described above is used rather than purely Euclidean measures of element quality.

3 An adjoint based sensitivity measure

In this section two equivalent error measures are derived based on the forward and adjoint solutions, leading to two alternate definitions for the metric tensor $\mathbf{M}$.

3.1 A goal based error measure

Suppose a differential equation to be solved is

$$\mathscr{L}\psi_{exact} - s = 0, \tag{3}$$

for source s, linear operator $\mathscr{L}$ (the extension to non-linear operators is relatively straightforward although it can involve considerable algebra) and the exact solution is $\psi_{exact} \equiv \psi_{exact}(\mathbf{x})$. This solution is approximated with a finite element scheme as

$$\psi \equiv \psi(\mathbf{x}) = \sum_{j=1}^{\mathcal{N}} N_j(\mathbf{x})\,\Psi_j, \tag{4}$$

with $\mathbf{x}$ being the spatial coordinates; $N_j(\mathbf{x})$ is the finite element basis function associated with node j; $\Psi \equiv (\Psi_1, \Psi_2, ..., \Psi_{\mathcal{N}})^T$ is the discrete solution vector; and $\mathcal{N}$ is the number of nodes in the finite element mesh. The equation residual is:

$$\mathscr{R}(\psi) \equiv \mathscr{L}\psi - s. \tag{5}$$

The aim is to make this residual small in some sense. This is done by multiplying equation 5 by a weighting function. In the Bubnov-Galerkin method this weighting function is chosen to be the basis function $N_i(\mathbf{x})$. However the equations are discretized a matrix equation and residual vector is obtained:

$$r(\psi) = \mathbf{A}\Psi - S = 0, \tag{6}$$

for matrix $\mathbf{A}$ and discretized source S. In a practical implementation with inexact arithmetic and possibly the use of iterative solution methods, $r(\psi)$ may not be identically zero. It is assumed insignificantly small in this work. This assumption may be relaxed by retaining $r(\psi)$ in the following equations. In addition, note that in the analysis it is assumed that $r(\psi)$ is in some sense a discretized representation of the residual $\mathscr{R}(\psi)$ multiplied by a representative volume of each cell or node, as achieved in typical finite element Petrov-Galerkin or control volume methods.

3.2 The functional or goal

Suppose that the functional whose accuracy is to be optimized is represented as $F \equiv F(\psi)$, and

$$F(\psi) = \int_{\Omega} f(\psi)\, dV, \tag{7}$$

where Ω is the solution domain. $F(\psi)$ may be any derived quantity of the solution ψ. Applying a first order Taylor series analysis, the gradient $\frac{\partial f}{\partial \psi}$ near the exact solution ψ_{exact} can be obtained from

$$\frac{\partial f}{\partial \psi}(\psi_{exact} - \psi) \approx f(\psi_{exact}) - f(\psi), \tag{8}$$

or in discrete form,

$$\left(\frac{\partial F}{\partial \Psi}\right)^T (\Psi_{exact} - \Psi) \approx F\left(\widetilde{\psi}_{exact}\right) - F(\psi), \tag{9}$$

in which $\Psi_{exact} \equiv (\Psi_{exact_1}, \Psi_{exact_2}, \ldots, \Psi_{exact_{\mathcal{N}}})^T$ is a vector containing the exact solution at the $\mathcal{N}$ finite element nodes (or control volume cells) and

$$\widetilde{\psi}_{exact} = \sum_{j=1}^{\mathcal{N}} N_j(\mathbf{x})\, \Psi_{exact_j}. \tag{10}$$

That is, $\widetilde{\psi}_{exact}$ is a finite element (or other numerical) interpolant of the exact solution ψ_{exact}.

3.3 Continum error measure

In a similar manner to the previous sub-section, applying a first order Taylor series results in:

$$\frac{\partial \mathscr{R}}{\partial \psi}(\psi_{exact} - \psi) \approx \mathscr{R}(\psi_{exact}) - \mathscr{R}(\psi). \tag{11}$$

Since $\mathscr{R}(\psi_{exact}) = 0$,

$$(\psi_{exact} - \psi) \approx -\left(\frac{\partial \mathscr{R}}{\partial \psi}\right)^{-1} \mathscr{R}(\psi). \tag{12}$$

Equations 8 and 12 can be combined to obtain

$$f(\psi_{exact}) - f(\psi) \approx -\frac{\partial f}{\partial \psi}\left(\frac{\partial \mathscr{R}}{\partial \psi}\right)^{-1} \mathscr{R}(\psi). \tag{13}$$

Integrating this expression over the domain Ω, using Green's theorem and ignoring the resulting surface integrals leads to

$$\begin{aligned} F(\psi_{exact}) - F(\psi) &\approx -\int_\Omega \frac{\partial f}{\partial \psi}\left(\frac{\partial \mathscr{R}}{\partial \psi}\right)^{-1} \mathscr{R}(\psi)\, dV \\ &= -\int_\Omega \mathscr{R}(\psi)\left(\frac{\partial \mathscr{R}^*}{\partial \psi^*}\right)^{-1} \frac{\partial f}{\partial \psi}\, dV \\ &= -\int_\Omega \mathscr{R}(\psi)\,\psi^*\, dV, \end{aligned} \tag{14}$$

in which

$$\left(\frac{\partial \mathscr{R}^*}{\partial \psi^*}\right)\psi^* = \frac{\partial f}{\partial \psi}, \tag{15}$$

which is typically solved for ψ^* using a finite element method. In addition

$$\mathscr{L}^* = \left(\frac{\partial \mathscr{R}^*}{\partial \psi^*}\right). \tag{16}$$

Equation 14 can now be used directly to determine an improved prediction of F, that is to obtain $F(\psi_{exact})$.

3.4 Discrete error measure

Suppose a non-singular matrix $\mathbf{A}_{exact}$ exists such that

$$r_{exact}\left(\widetilde{\psi}_{exact}\right) = \mathbf{A}_{exact}\Psi_{exact} - S = 0, \tag{17}$$

and

$$r^*_{exact}\left(\widetilde{\psi}^*_{exact}\right) = \left(\mathbf{A}_{exact}\right)^T \Psi^*_{exact} - \frac{\partial F}{\partial \Psi} = 0. \tag{18}$$

Define $\Psi^*_{exact} \equiv \left(\Psi^*_{exact_1}, \Psi^*_{exact_2}, \ldots, \Psi^*_{exact_{\mathcal{N}}}\right)^T$ which is associated with the finite element representation

$$\widetilde{\psi}^*_{exact} = \sum_{j=1}^{\mathcal{N}} N_j(\mathbf{x})\,\Psi^*_{exact_j}. \tag{19}$$

The discrete analogue of equation 11 is:

$$\begin{aligned} \frac{\partial\left(r_{exact}\left(\widetilde{\psi}_{exact}\right)\right)}{\partial \Psi_{exact}}\left(\Psi_{exact} - \Psi\right) &\approx r_{exact}\left(\widetilde{\psi}_{exact}\right) - r_{exact}(\psi) \\ &\approx r\left(\widetilde{\psi}_{exact}\right) - r(\psi). \end{aligned} \tag{20}$$

Combining this with equation 17 differentiated w.r.t. Ψ_{exact} results in,

$$(\Psi_{exact} - \Psi) \approx (\mathbf{A}_{exact})^{-1} \left(r\left(\widetilde{\psi}_{exact}\right) - r\left(\psi\right)\right). \tag{21}$$

This can then be combined with equations 9 and 18 to obtain

$$\begin{aligned} F\left(\widetilde{\psi}_{exact}\right) - F\left(\psi\right) &\approx \left(\frac{\partial F}{\partial \Psi}\right)^{T} (\mathbf{A}_{exact})^{-1} \left(r\left(\widetilde{\psi}_{exact}\right) - r\left(\psi\right)\right) \\ &= \left(r\left(\widetilde{\psi}_{exact}\right) - r\left(\psi\right)\right)^{T} (\mathbf{A}_{exact})^{-T} \left(\frac{\partial F}{\partial \Psi}\right) \\ &= \left(r\left(\widetilde{\psi}_{exact}\right) - r\left(\psi\right)\right)^{T} \Psi^{*}_{exact}. \end{aligned} \tag{22}$$

This expression can also be used to obtain an improved prediction of F, that is to find $F(\widetilde{\psi}_{exact})$, once estimates of $r\left(\widetilde{\psi}_{exact}\right)$ and Ψ^{*}_{exact} have been obtained.

3.5 Discrete functional correction

Although the error estimates are obtained from equation 22, the product

$$\left(r\left(\widetilde{\psi}_{exact}\right) - r\left(\psi\right)\right)^{T} \Psi^{*}_{exact}$$

is not known and cannot be used directly to correct the functional F. To do this the following relation is employed

$$\begin{aligned} F\left(\widetilde{\psi}_{exact}\right) - F\left(\psi\right) &\approx \left(r\left(\widetilde{\psi}_{exact}\right) - r\left(\psi\right)\right)^{T} \Psi^{*} \\ &\approx \left(\hat{r}\left(\psi\right) - r\left(\psi\right)\right)^{T} \Psi^{*}. \end{aligned} \tag{23}$$

$\hat{r}\left(\psi\right)$ is approximately equal to $r\left(\widetilde{\psi}_{exact}\right)$ and is obtained from the residual calculation method outlined in section 5. Equation 23 can also be used to re-normalise the error measures to correctly obtain the desired error in F.

3.6 Dual continuum error measure

An alternative dual error measure can be obtained which utilises $\mathscr{L}\psi^{*}$ instead of $\mathscr{R}\left(\psi\right)$. Since

$$\mathscr{R}\left(\psi_{exact}\right) \equiv \mathscr{L}\psi_{exact} - s = 0, \tag{24}$$

with the application of Green's Theorem and again ignoring the resulting surface integrals, equation 14 becomes

$$
\begin{aligned}
F\left(\psi_{exact}\right) - F\left(\psi\right) &\approx -\int_{\Omega} \mathscr{R}\left(\psi\right) \psi^{*}\, dV \\
&= \int_{\Omega} \left(\left(\mathscr{L}\psi_{exact} - s\right) - \left(\mathscr{L}\psi - s\right)\right) \psi^{*}\, dV \\
&= \int_{\Omega} \left(\mathscr{L}\left(\psi_{exact} - \psi\right)\right) \psi^{*}\, dV \\
&= \int_{\Omega} \left(\psi_{exact} - \psi\right) \mathscr{L}^{*}\psi^{*}\, dV. \qquad (25)
\end{aligned}
$$

3.7 Dual discrete error measure

Following the method of section 3.6 the analogous result in the discrete sense is, from equation 22,

$$
\begin{aligned}
F\left(\psi_{exact}\right) - F\left(\psi\right) &\approx \left(r\left(\widetilde{\psi}_{exact}\right) - r\left(\psi\right)\right)^{T} \Psi^{*}_{exact} \\
&= \left(\left(\mathbf{A}\Psi_{exact} - S\right) - \left(\mathbf{A}\Psi - S\right)\right)^{T} \Psi^{*}_{exact} \\
&= \left(\mathbf{A}\left(\Psi_{exact} - \Psi\right)\right)^{T} \Psi^{*}_{exact} \\
&= \left(\Psi_{exact} - \Psi\right)^{T} \mathbf{A}^{T} \Psi^{*}_{exact} \\
&= \left(\mathbf{A}^{T}\Psi^{*}_{exact}\right)^{T} \left(\Psi_{exact} - \Psi\right). \qquad (26)
\end{aligned}
$$

This is used to form a directional error measure based on the forward solution.

3.8 Alternative error measures

The dual error measures derived in section 3.7 can be expressed, assuming $r\left(\psi\right) = 0$, as

$$
\begin{aligned}
F\left(\psi_{exact}\right) - F\left(\psi\right) &\approx r\left(\widetilde{\psi}_{exact}\right)^{T} \Psi^{*}_{exact} \\
&= r\left(\widetilde{\psi}_{exact}\right)^{T} \Psi^{*} + r\left(\widetilde{\psi}_{exact}\right)^{T} \left(\Psi^{*}_{exact} - \Psi^{*}\right). \qquad (27)
\end{aligned}
$$

From equation 27 a useful relashionship, which has an analagous continuous form is

$$
F\left(\psi_{exact}\right) - F\left(\psi\right) \approx \left(\frac{\partial F}{\partial \psi}\right)^{T} \left(\Psi_{exact} - \Psi\right) + r\left(\widetilde{\psi}_{exact}\right)^{T} \left(\Psi^{*}_{exact} - \Psi^{*}\right). \quad (28)
$$

Equation 28 is used to form a directional error measure based on the adjoint solution. Similar results to 27 have been presented by [62], in which the latter

part of equation 28, that is

$$r\left(\tilde{\psi}_{exact}\right)^T\left(\Psi^*_{exact}-\Psi^*\right),$$

is used to gauge mesh adaptivity isotropically [64] and anisotropically [65]. In [54] a defect correction method, similar in form, is presented to increase the accuracy of estimation. [55] use a similar approach to that described here, but using an isotropic error measure. Note that equation 27 can also be expressed for linear operators as

$$\begin{aligned}
F\left(\psi_{exact}\right)-F\left(\psi\right) &\approx \left(\mathbf{A}^T\left(\Psi^*_{exact}-\Psi^*+\mathbf{A}^{-T}\frac{\partial F}{\partial \psi}\right)\right)^T\left(\Psi_{exact}-\Psi\right)\\
&=\left(\mathbf{A}^T\left(\Psi^*_{exact}-\Psi^*\right)+\frac{\partial F}{\partial \psi}\right)^T\left(\Psi_{exact}-\Psi\right)\\
&=\left(\Psi^*_{exact}-\Psi^*\right)^T\mathbf{A}\left(\Psi_{exact}-\Psi\right)\\
&\qquad+\left(\frac{\partial F}{\partial \psi}\right)^T\left(\Psi_{exact}-\Psi\right). \qquad (29)
\end{aligned}$$

Estimates of the solution errors $(\Psi^*-\Psi^*_{exact})$ and $(\Psi-\Psi_{exact})$ can readily be obtained using interpolation theory as outlined in section 5.

4 Derivation of a metric tensor

In section 3 the error measures were derived. In this section these error measures are modified slightly by approximating the unknown quantities $\mathbf{A}^T\Psi^*_{exact}$ and $r\left(\tilde{\psi}_{exact}\right)$ with known $\widehat{\mathbf{A}^T\Psi^*_{exact}}$ and $\hat{r}(\tilde{\psi}_{exact})$ respectively. Full details of how this is implemented are outlined in section 5. The error measures (equations 26 and 28) then become,

$$F\left(\psi_{exact}\right)-F\left(\psi\right)\approx\left(\widehat{A^T\Psi^*_{exact}}\right)^T\left(\Psi_{exact}-\Psi\right), \qquad (30)$$

$$\begin{aligned}
F\left(\psi_{exact}\right)-F\left(\psi\right) &\approx \left(\frac{\partial F}{\partial \psi}\right)^T\left(\Psi_{exact}-\Psi\right)\\
&\qquad+\hat{r}\left(\tilde{\psi}_{exact}\right)^T\left(\Psi^*_{exact}-\Psi^*\right). \qquad (31)
\end{aligned}$$

These equations require a measure of the error in the forward solution (equation 30) and both forward and adjoint solutions (equation 31).

A metric tensor (see section 2) can be defined:

$$\bar{\mathbf{M}} = \frac{\gamma}{|\epsilon|}|\mathbf{H}|. \tag{32}$$

Here $\mathbf{H}$ is the Hessian matrix, ϵ is the required level of error and γ a scalar constant. The absolute value of the symmetric Hessian metrix is defined as,

$$|\mathbf{H}| = \mathbf{V}\,|\mathbf{\Lambda}|\,\mathbf{V}^T. \tag{33}$$

Where the matricies $\mathbf{V}$ and $\mathbf{\Lambda}$ contain the eigenvectors $\mathbf{e}_i$ and eigenvalues Λ_i of the Hessian matrix $\mathbf{H}$ respectively and are defined as,

$$\mathbf{V} = \begin{pmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{pmatrix}, \quad |\mathbf{\Lambda}| = \begin{pmatrix} |\Lambda_1| & 0 & 0 \\ 0 & |\Lambda_2| & 0 \\ 0 & 0 & |\Lambda_3| \end{pmatrix}.$$

Ideal elements then have sides of length unity when measured with respect to the metric tensor. The discrete (nodal) form of ϵ, from equation 30 is

$$\epsilon_i = \frac{\widetilde{\delta F}}{\left|\left(\widehat{\mathbf{A}^T\Psi^*_{exact}}\right)_i\right|}. \tag{34}$$

Here $\left(\widehat{\mathbf{A}^T\Psi^*_{exact}}\right)_i$ is the i^{th} entry of the vector $\widehat{\mathbf{A}^T\Psi^*_{exact}}$ and

$$\epsilon = (\psi_{exact} - \psi) = \sum_{j=1}^{\mathcal{N}} N_j\,(\mathbf{x})\,\epsilon_j.$$

Suppose δF is the acceptable error in F; then assuming the error contribution to F is the same for each node, then define

$$\widetilde{\delta F} = \frac{\delta F}{\mathcal{N}}. \tag{35}$$

Predicting, and potentially fixing the available number of nodes is an alternative to specifying δF, and useful when computational resources are restricted in some way. The number of available nodes can be fixed and the metric scaled in the following way. The volume of an optimal tetrahdral element can be taken to be $\gamma = \left(\frac{1}{\sqrt{72}}\right)$. If it is assumed that after the mesh is adapted all

the elements have the ideal volume γ, then as the domain is a fixed volume the number number of elements can be calculated from

$$\theta E_{\text{new}} = \frac{\sum_{e=1}^{E_{\text{old}}} V_e}{\gamma}, \tag{36}$$

where E_{old} is the number of elements in a mesh before it is adapted and E_{new} the number of elements after. V_e is the volume of an element e. A new metric $\mathbf{M}_{\text{new}} = \beta \mathbf{M}$ replaces $\mathbf{M}$ for some scalar β. $\mathbf{M}$ is obtained from the method above (see equation 32) with $\widetilde{\delta F}$ arbitarily taken as $\widetilde{\delta F} = 1$. Mapping the node-wise values of the metric tensor to the element-wise metric tensor $\mathbf{M}_e$ is achieved by taking the average of the metric tensor values at the nodes of element e.

The scalar θ acknowledges the fact that equation 36 is not exact, a value of $\theta \approx 0.85$ is found to be appropriate in [21]. To make the replacement of $\mathbf{M}$ with $\mathbf{M}_{\text{new}}$ then the required number of elements is θE_{new}, and substituting into equation 36 leads to

$$\theta E_{\text{new}} = \frac{\sum_{e=1}^{E_{\text{old}}} V_e \sqrt{\det\left(\beta \mathbf{M}_e\right)}}{\gamma} = \frac{\sum_{e=1}^{E_{\text{old}}} \beta^{3/2} V_e \sqrt{\det\left(\mathbf{M}_e\right)}}{\gamma}, \tag{37}$$

and so

$$\beta = \left(\frac{\gamma \theta E_{\text{new}}}{\sum_{e=1}^{E_{\text{old}}} V_e \sqrt{\det\left(\mathbf{M}_e\right)}} \right)^{2/3}. \tag{38}$$

4.2 Application to multiple field problems

For each solution variable l and at each node i it is possible to define two Hessians: $\mathbf{H}_i^l$ associated with the forward solution ψ_i^l and $\mathbf{H}^{*l}_i$ associated with the adjoint solution ψ^{*l}_i. The Hessian matrices are defined as

$$\mathbf{H}_i^l \equiv \left(\nabla^T \nabla \psi^l \right)_i, \quad \mathbf{H}^{*l}_i \equiv \left(\nabla^T \nabla \psi^{*l} \right)_i. \tag{39}$$

To calculate the Hessians, the method presented in [21] is followed. Galerkin projections are repeated applied to calculate the first derivatives. Consider the forward Hessian constructed from the forward solution ψ^l, consequently at node i:

$$\left. \frac{\partial \psi^l}{\partial x} \right|_i \approx q_{x_i} = \mathsf{M}_{\mathsf{L}_i}^{-1} \int N_i \left(\frac{\partial \psi^l}{\partial x} \right) dV$$

and in a similar manner for $q^l_{y_i}$ and $q^l_{z_i}$. M_{L} is the row summed lumped mass matrix, see [66]. The second order terms which form the Hessian for an equa-

tion solution variable l centred on node i,

$$\mathbf{H}_i^l = \begin{pmatrix} q_{xx_i}^l & q_{xy_i}^l & q_{xz_i}^l \\ q_{yx_i}^l & q_{yy_i}^l & q_{yz_i}^l \\ q_{zx_i}^l & q_{zy_i}^l & q_{zz_i}^l \end{pmatrix},$$

are calculated in a similar way, for example

$$q_{xx_i}^l = \mathsf{M}_{\mathsf{L}_i}^{-1} \int N_i \frac{\partial q_x^l}{\partial x} dV, \quad q_{xy_i}^l = \mathsf{M}_{\mathsf{L}_i}^{-1} \int N_i \frac{\partial q_x^l}{\partial y} dV.$$

If there are $\mathcal{M}$ solution variables per node then an averaged Hessian $\bar{\mathbf{H}}_i$ associated with node i can be defined as

$$\bar{\mathbf{H}}_i = \frac{1}{\sum_{l=1}^{\mathcal{M}} \left|\lambda_i^l\right|} \sum_{l=1}^{\mathcal{M}} \left|\lambda_i^l\right| \left|\mathbf{H}_i^l\right|, \tag{40}$$

where

$$\lambda_i^l = \left(\widehat{\mathbf{A}^T \Psi_{exact}^*}\right)_i^l.$$

Absolute values of the Hessian matricies $\left|\mathbf{H}_i^l\right|$ are as given by equation 33. This technique uses equation 30 as the basis for this average. Using the error measure defined by equation 30 the metric tensor field can be found. An interpolation error $\bar{\epsilon}_i$ at node i can be defined as:

$$\bar{\epsilon}_i = \frac{\widetilde{\delta F}}{\sum_{l=1}^{\mathcal{M}} \left|\lambda_i^l\right|}, \tag{41}$$

then the nodal Metric Tensor $\bar{\mathbf{M}}_i$ is obtained from

$$\bar{\mathbf{M}}_i = \frac{\gamma}{\left|\bar{\epsilon}_i\right|} \left|\bar{\mathbf{H}}_i\right|. \tag{42}$$

Using the adjoint error measure defined by equation 31 a second metric tensor can be obtained based on the forward and adjoint solutions. To that end, suppose that

$$\lambda^{+l}{}_i = \left(\frac{\partial F}{\partial \psi}\right)_i^l, \quad \lambda^{*l}{}_i = \hat{r}\left(\psi\right)_i^l,$$

then a combined averaged Hessian $\bar{\mathbf{H}}_i^+$ associated with node i can be defined as:

$$\bar{\mathbf{H}}_i^+ = \frac{1}{\sum_{l=1}^{\mathcal{M}} \left(\left|\lambda^{+l}{}_i\right| + \left|\lambda^{*l}{}_i\right|\right)} \sum_{l=1}^{\mathcal{M}} \left(\left|\lambda^{+l}{}_i\right| \left|\mathbf{H}_i^l\right| + \left|\lambda^{*l}{}_i\right| \left|\mathbf{H}^{*l}{}_i\right|\right). \tag{43}$$

An adjoint based interpolation error $\bar{\epsilon}_i^*$ can be defined as:

$$\bar{\epsilon}_i^+ = \frac{\widetilde{\delta F}}{\sum_{l=1}^{\mathcal{M}} \left(\left| \lambda^{+l}_i \right| + \left| \lambda^{*l}_i \right| \right)}. \tag{44}$$

So a new adjoint based metric tensor $\bar{\mathbf{M}}_i^+$ at node i is obtained from

$$\bar{\mathbf{M}}_i^+ = \frac{\gamma}{\left| \bar{\epsilon}_i^+ \right|} \left| \bar{\mathbf{H}}_i^+ \right|. \tag{45}$$

As shall be seen in some of the examples of this method in section 8.1, it may be valuable to consider a modified adjoint metric tensor constructed in a similar manner to equation 45, but omitting λ^+ terms. Taking this into consideration, a modified averaged adjoint Hessian, $\bar{\mathbf{H}}_i^*$ say, can be defined,

$$\bar{\mathbf{H}}_i^* = \frac{1}{\sum_{l=1}^{\mathcal{M}} \left| \lambda^{*l}_i \right|} \sum_{l=1}^{\mathcal{M}} \left| \lambda^{*l}_i \right| \left| \mathbf{H}^{*l}_i \right|. \tag{46}$$

Consequently a modified adjoint based interpolation error $\bar{\epsilon}_i^*$ can be defined as:

$$\bar{\epsilon}_i^* = \frac{\widetilde{\delta F}}{\sum_{l=1}^{\mathcal{M}} \left| \lambda^{*l}_i \right|}. \tag{47}$$

So a modified adjoint based metric tensor $\bar{\mathbf{M}}_i^*$ for node i is obtained from

$$\bar{\mathbf{M}}_i^* = \frac{\gamma}{\left| \bar{\epsilon}_i^* \right|} \left| \bar{\mathbf{H}}_i^* \right|. \tag{48}$$

4.3 Incorporation of multiple metric tensors

In practice, one metric tensor $\mathbf{M}$ is required to guide the mesh adaptivity algorithm. A single metric tensor field can be obtained by either ignoring the metric tensors $\bar{\mathbf{M}}^+$ or $\bar{\mathbf{M}}^*$ and using only $\bar{\mathbf{M}}$ as the basis for mesh adaptivity, thus the overall metric tensor $\mathbf{M}_i$ at node i is obtained from

$$\mathbf{M}_i = \bar{\mathbf{M}}_i, \tag{49}$$

or alternatively by including aspects of more than one metric tensor. One method could take the form an of averaging, similar to that used in equation 43. The average metric tensor takes the form,

$$\bar{\bar{\mathbf{M}}}_i = \frac{\gamma}{\left| \bar{\bar{\epsilon}}_i \right|} \left| \bar{\bar{\mathbf{H}}}_i \right|. \tag{50}$$

where the interpolation error $\bar{\bar{\epsilon}}$ and averaged Hessian $\bar{\bar{\mathbf{H}}}_i$ bring in components from the forward and modified adjoint metric tensors, where

$$\bar{\bar{\epsilon}}_i = \frac{\widetilde{\delta F}}{\sum_{l=1}^{\mathcal{M}} \left(\left| \lambda_i^l \right| + \left| \lambda^{*l}_i \right| \right)} \tag{51}$$

and

$$\bar{\bar{\mathbf{H}}}_i = \frac{1}{\sum_{l=1}^{\mathcal{M}} \left(\left| \lambda_i^l \right| + \left| \lambda^{*l}_i \right| \right)} \sum_{l=1}^{\mathcal{M}} \left(\left| \lambda_i^l \right| \left| \mathbf{H}_i^l \right| + \left| \lambda^{*l}_i \right| \left| \mathbf{H}^{*l}_i \right| \right) \tag{52}$$

Alternatively a method of superposition can be used. If metric tensors $\bar{\mathbf{M}}$ and $\bar{\mathbf{M}}^*$ are to be superimposed then

$$\bar{\bar{\mathbf{M}}}_i^{\mathcal{G}} = \mathcal{G} \left(\bar{\mathbf{M}}_i, \bar{\mathbf{M}}_i^* \right). \tag{53}$$

In this case the operator $\mathcal{G}(\mathbf{A}, \mathbf{B})$ superimposes the two metric tensors $\mathbf{A}$ and $\mathbf{B}$ by the method presented in [21].

5 Determining equation residuals

In this section the forward and adjoint equation residuals are estimated, which enables the metric tensor to be obtained by the method described in section 4. If the differential equation under consideration is first order then it is a simple matter of substituting the finite element solution ψ^* into the operator $\mathscr{L}^*(\psi^*)$ into the final result of equation 25 or $\mathscr{R}(\psi)$ into the first result of equation 25. It may be possible for a second order operator $\mathscr{L}(\psi)$, using quadratic and higher order elements, to obtain estimates of $\mathscr{L}^*(\psi^*)$ and $\mathscr{R}(\psi)$; however, in this work only linear elements are considered. Moreover, since second order operators are often solved in their weak form and since it may be inconvenient to write additional code to determine the residual through the residuals of the equations, estimates of $\mathscr{L}^*(\psi^*)$ and $\mathscr{R}(\psi)$ are calculated from the discretized equations. It should also be noted that the equation residual $\mathscr{R}(\psi)$ or similarly $\mathscr{L}^*(\psi^*)$ can be quite oscillatory, see [54] and have been smoothed in [62] to obtain residuals that can be used in the equations above.

The overall approach that will be used here for calculating equation residuals will be based on forming course grid approximations to the discretised equations on the current fine mesh. It will be seen that this can be equivalent to using a high order Taylor series expansion to find the equation residuals or truncation errors and is thus accurate. However, since interpolating theory is used throughout this work In calculating metric tensors this approach is also demonstrated by using it to estimate equation residuals.

The residuals can be obtained by finding an estimate of the error in the solution at the nodes using interpolation theory. For a finite element e the interpolation errors are:

$$|\Psi_{exact} - \Psi| \approx \gamma_c \max_{\ell \in \mathcal{L}_e} \left\{ \mathbf{v}_\ell^T \, |\mathbf{H}| \, \mathbf{v}_\ell \right\},$$
$$|\Psi^*_{exact} - \Psi^*| \approx \gamma_c \max_{\ell \in \mathcal{L}_e} \left\{ \mathbf{v}_\ell^T \, |\mathbf{H}^*| \, \mathbf{v}_\ell \right\}.$$

where ℓ is an edge of an element (or side of a control volume), $\mathbf{v}_\ell$ is the vector along edge ℓ and with magnitude equal to its length, and $\mathcal{L}_e$ is the set of edges of the element e. The Hessian matrices remain as previously defined.

5.2 Residual estimation

In this section approximations $\widehat{\mathbf{A}^T \Psi^*_{exact}}$ and $\hat{r}\left(\widetilde{\psi}_{exact}\right)$, for a particular node i, are obtained to the quantities $\mathbf{A}^T \Psi^*_{exact}$ and $r\left(\widetilde{\psi}_{exact}\right)$ respectively. This is achieved by finding a coarse grid representation of the discrete matrix equations at each node i; the residual is obtained when an approximation to the adjoint solution Ψ^* is evaluated at the coarse grid nodes and used in conjunction with the resulting coarse grid matrix. The coarse grid residuals are also obtained algebraically, so that no access to the governing equations is necessary. This takes a step toward the possibility of developing a method (and associated computer implementation) that is independent of the governing equations. With knowledge of how the residual varies with the grid resolution one can determine an estimate of the residual.

Matrices and vectors are calculated to obtain an estimate of the residual at a node i, the set of nodes directly connected to i is taken to be $\mathcal{S}_i$. Vectors and matrices used in this section should, strictly, have a subscript i indicating this, however for simplicity of the notation this has not been included, but note must be taken that the following must be performed for each node i in turn in order to obtain the (nodal) residual vector. Suppose the following system of equations is to be solved:

$$\mathbf{A}\Psi = S,$$

where

$$\Psi = \begin{pmatrix} \Psi_f \\ \Psi_c \end{pmatrix}, \quad S = \begin{pmatrix} S_f \\ S_c \end{pmatrix}.$$

Ψ is the vector of unknowns associated with the nodal basis and S is the source. Let a vector of unknowns associated with a hierarchical basis be:

$$\hat{\Psi} = \begin{pmatrix} \Delta\Psi_f \\ \Psi_c \end{pmatrix}.$$

The subscripts f and c refer to the fine and coarse grid variables respectively and $\Delta\Psi_f$ are the fine grid correction terms. $\hat{\Psi}$ and Ψ are related through

$$\Psi = \mathbf{J}\hat{\Psi}, \tag{54}$$

where

$$\mathbf{J} = \begin{pmatrix} \mathbf{I} & \mathbf{J}_{12} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}. \tag{55}$$

Consequently

$$\begin{pmatrix} \Psi_f \\ \Psi_c \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{J}_{12} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \Delta\Psi_f \\ \Psi_c \end{pmatrix}. \tag{56}$$

The subscripts f and c refer to the fine and coarse grid variables respectively. From equation 55 it can be said

$$\mathbf{J}^{-1} = \begin{pmatrix} \mathbf{I} & -\mathbf{J}_{12} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}. \tag{57}$$

The matrix formed with the hierarchical basis is defined by

$$\widehat{\mathbf{A}} = \mathbf{J}^T\mathbf{A}\mathbf{J}, \tag{58}$$

thus

$$\widehat{\mathbf{A}} = \begin{pmatrix} \mathbf{I} & 0 \\ \mathbf{J}_{12}^T & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{J}_{12} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{Q}_{12} \\ \mathbf{Q}_{21} & \mathbf{A}_c \end{pmatrix}, \tag{59}$$

where

$$\mathbf{Q}_{12} = \mathbf{A}_{11}\mathbf{J}_{12} + \mathbf{A}_{12}, \tag{60}$$

$$\mathbf{Q}_{21} = \mathbf{J}_{12}^T\mathbf{A}_{11} + \mathbf{A}_{21}, \tag{61}$$

$$\mathbf{A}_c = \mathbf{J}_{12}^T\mathbf{A}_{11}\mathbf{J}_{12} + \mathbf{J}_{12}^T\mathbf{A}_{12} + \mathbf{A}_{21}\mathbf{J}_{12} + \mathbf{A}_{22}. \tag{62}$$

The matrix $\widehat{\mathbf{A}}$ is spectrally equivalent to the matrix $\mathbf{A}$ and can be formed using the shape functions associated with the coarse grid. The residual can now be estimated from

$$\gamma_c\left(\mathbf{A}_c\Psi_c - \widehat{S}_c\right) = \gamma_c\mathbf{Q}_{21}\Delta\Psi_f, \tag{63}$$

in which the coarse grid source $\widehat{S}_c$ takes the form

$$\hat{S}_c = \mathbf{J}_{12}^T S_f + S_c. \tag{64}$$

It is worth noting that $\Delta\Psi_f$ may be used directly in equation 30 to help improve the estimation of F. In this work the equation for the residual at node i is:

$$\left(r\left(\widetilde{\psi}_{exact}\right)\right)_i \approx \left(\hat{r}(\psi)\right)_i = \left(\gamma_c \mathbf{Q}_{21} \Delta\Psi_f\right)_i, \tag{65}$$

where

$$\Delta\Psi_f = \Psi_f - J_{12}\Psi_c. \tag{66}$$

The scalar γ_c has been used to take into account the fact that the residual is estimated on a mesh that is $2h$ (see sections 5.3 and 5.4) as opposed to h in element size. Supposing that the residual converges with h^2, then $\gamma_c = \frac{1}{4}$ as this is the factor required to give a residual estimation on the current mesh.

5.3 $\mathbf{J}_{12}$ *definition for structured meshes*

The matrix vector multiplication $\mathbf{J}_{12}\Psi_c$ acts to interpolate the coarse grid solution variables Ψ_c to obtain solutions on the other (fine grid) nodes. The matrix $\mathbf{J}_{12}$ possesses the property that it has a row sum of unity and mostly positive entries. The fact that $\mathbf{J}_{12}\Psi_c$ acts to interpolate coarse grid variables reiterates the definition of $\mathbf{J}^{-1}$ in equation 57, that $\Delta\Psi_f$ are fine mesh correction terms.

A fine grid stencil is constructed from the nodes directly connected to node k say; these nodes form the set $\mathcal{S}_k$ and are shown as filled in circles in figure 1. The fine grid stencil is extended by a factor of two in each direction to give the location of the coarse grid nodes. These points are shown as open circles in figure 1. The solution at node k is obtained by interpolating the surrounding coarse grid nodes. For example, suppose the node at which the residual is sort is (i, j), on a regular $N_x \times N_y$ mesh, as illustrated in figure 1. The node (i, j) is the central node, and forms the centre of both the fine and coarse grid stencils, then the the solution (interpolated from the coarse grid solution variables) at the fine node $(i, j+1)$ is

$$\hat{\Psi}_{i,j+1} = \frac{\Psi_{i,j} + \Psi_{i,j+2}}{2},$$

and at the fine grid node $(i+1, j+1)$

$$\hat{\Psi}_{i+1,j+1} = \frac{\left(\Psi_{i,j} + \Psi_{i,j+2} + \Psi_{i+2,j} + \Psi_{i+2,j+2}\right)}{4}.$$

This results in a 9-point stencil in the coarse mesh in matrix A_c, which is used

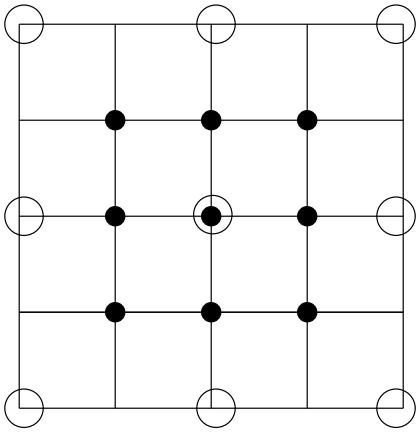


Fig. 1. Diagram showing how the coarse grids are obtained for a structured mesh, as outlined in section 5.3. Filled circles represent fine grid nodes; open circles indicate coarse grid nodes.

in section 7. Alternatively,

$$\hat{\Psi}_{i+1,j+1} = \frac{(\Psi_{i,j+2} + \Psi_{i+2,j})}{2},$$

which results in the same 5-point stencil as the original, as indicated by equation 84. Consequently $\Delta\Psi_f$ can be taken to be

$$\Delta\Psi_f = \hat{\Psi} - \Psi_f, \tag{67}$$

with indices (i, j) omitted for simplicity.

5.4 $\mathbf{J}_{12}$ *definition for unstructured meshes*

Extending the method to an unstructured mesh requires some extra work, as the coarse grid nodes do not generally lie in the same positions as the fine grid nodes. A simple method of mesh coarsening and quadratic interpolation is employed to enable computation of the fine grid correction $\Delta\Psi_f$ and approximations $\widehat{\mathbf{A}^T\Psi^*_{exact}}$ and $\hat{r}\left(\widetilde{\psi}_{exact}\right)$ to $\mathbf{A}^T\Psi^*_{exact}$ and $r\left(\widetilde{\psi}_{exact}\right)$ respectively for each node i of the mesh. Fine and coarse grid stencils are established for each node.

A fine grid stencil is constructed from the nodes directly connected to the node i; these nodes form the set $\mathcal{S}_i$ and are shown as filled circles in figure 2. The fine grid stencil is indicated by solid bold lines in figure 2. The fine grid stencil is extended by a factor of two in each direction to give the location of coarse grid nodes. The coordinates of the coarse nodes $\mathbf{x}_c$, related to the nodes $j \in \mathcal{S}_i$ are defined as

$$\mathbf{x}_c = \mathbf{x}_i + 2\left(\mathbf{x}_j - \mathbf{x}_i\right). \tag{68}$$

These points are shown with solid open circles in figure 2, the extension of the stencil is shown with a bold dashed line. To apply quadratic interpolation a further point must be defined, $\mathbf{x}_d$ say. Similarly, the coordinates of the coarse nodes $\mathbf{x}_d$, related to the nodes $j \in \mathcal{S}_i$ are defined as

$$\mathbf{x}_d = \mathbf{x}_i - 2\left(\mathbf{x}_j - \mathbf{x}_i\right). \tag{69}$$

These points are shown with dashed open circles in figure 2, the extension of the stencil shown with a standard weight dashed line. This process results in points $\mathbf{x}_c$ and $\mathbf{x}_d$, related to each node $j \in \mathcal{S}_i$. These points generally lie inside elements of the fine (original) mesh. The elements are identified by means of a local vicinity search, initiated from one of the elements surrounding node i. The values of a field, for the purposes of this work the solution to the adjoint problem, Ψ^{*l} at points $\mathbf{x}_c$ and $\mathbf{x}_d$ can then be found by interpolating from the known values of Ψ^{*l} at the nodes of the element in the points lies, thus yielding ${\Psi^*}_c^l$ and ${\Psi^*}_d^l$. The objective is to find an interpolated value of Ψ^{*l} at the fine grid node j. Computing this is now a simple matter of applying a quadratic interpolation scheme, such that

$${\Psi^*}_{exact\,j}^{\;\;l} = \frac{-{\Psi^*}_d^l + 6{\Psi^*}_i^l + 3{\Psi^*}_c^l}{8}. \tag{70}$$

This process is repeated for each of the nodes j surrounding node i, the interpolated value at node i, the centre of the discretization, is taken to be ${\Psi^*}_i^l$. Figure 2 shows the stencil extensions required for a node i with three surrounding nodes j.

Generally the points $\mathbf{x}_c$ and $\mathbf{x}_d$ lie within elements of the fine (original) mesh; there are however some special cases to consider when the nodes lie close to the boundaries of the domain. As demonstrated in figure 3. It is likely that the stencil will be extended beyond the boundaries of the domain. In the case that only $\mathbf{x}_c$ lies outside the domain or both $\mathbf{x}_c$ and $\mathbf{x}_d$ lie outside the domain then

$${\Psi^*}_{exact\,j}^{\;\;l} = {\Psi^*}_j^l. \tag{71}$$

Alternatively if $\mathbf{x}_d$ lies outside the domain then linear interpolation is applied such that

$${\Psi^*}_{exact\,j}^{\;\;l} = \frac{{\Psi^*}_j^l + {\Psi^*}_c^l}{2}. \tag{72}$$

The interpolated values of Ψ^* at the nodes surrounding node i are enough to compute the i^{th} row of $\widehat{\mathbf{A}^T\Psi^*_{exact}}$, required by equation 34. It is sufficient to take the interpolated values of Ψ^* in this case as the values of Ψ^*_{exact}.

5.4.1 Residual Smoothing

It should also be noted that the equation residual can be quite oscillatory, see [54] and has been smoothed in [62] to obtain residuals that can be used

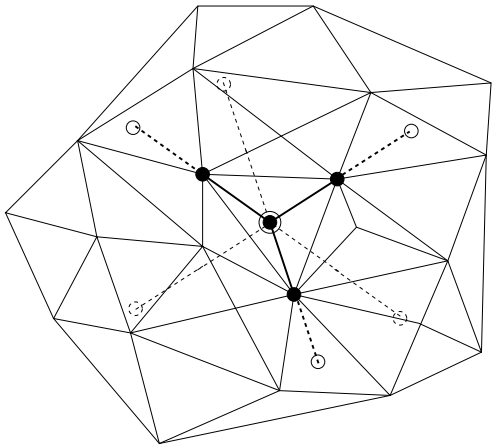


Fig. 2. Diagram showing how the coarse grid is obtained for an unstructured mesh. Filled circles represent fine grid nodes (Ψ_j); solid open circles indicate coarse grid nodes (Ψ_c) and dashed open circles represent the extra points required for interpolation (Ψ_d).

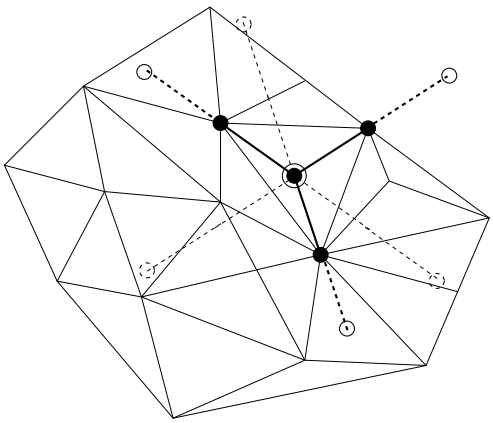


Fig. 3. Diagram showing how the coarse grid is obtained for an unstructured grid near to the domain boundary. Definitions identical to those given by figure 2. Note that it is possible to extend the stencil beyond the domain boundary, the methodology for deadling with this is detailed in section 5.

in the equations above. As indicated in the previous section, smoothing is applied to the residuals calculated using the above method. The smoothing method employed uses a weighted average of residual values at the node in question as well as those nodes directly connected to it. For simplicity, let $\hat{r}$ be the approximation to the residual that has calculated; $\widehat{\mathbf{A}^T\Psi^*_{exact}}$ or $\hat{r}\left(\widetilde{\psi}_{exact}\right)$. This is defined at the nodes. Then a smoothed value of this, $\hat{r}_i^{\mathsf{S}}$ say, at a node i, is defined as,

$$\hat{r}_i^{\mathsf{S}} = \frac{1}{2}\hat{r}_i + \frac{1}{2\mathsf{s}_i}\sum_{j\in\mathcal{S}_i}\hat{r}_j \tag{73}$$

where $\mathcal{S}_i$ is the set of nodes directly connected to node i, excluding node i itself and s_i is the number of nodes in set $\mathcal{S}_i$. Applying this proceedure to all the nodes in turn creates a new smoothed field, the process is repeated a number of times (in the examples presented here 5 times where $\widehat{\mathbf{A}^T\Psi^*_{exact}}$ is amended in equation 30 using this approach), producing increasingly smoother fields.

5.4.2 Metric relaxation

A form of metric relaxation is also employed to avoid excessive refinement of the mesh. The mesh is typically adapted a number of times, generally between 25 and 35, using the same definition of metric tensor in each case (i.e. forward, superimposed e.t.c.). For the first adaptation of the mesh the metric tensor is used 'as is'. For subsequent adaptations the metric used to adapt the mesh

is defined as

$$\mathbf{M} = (1 - \varrho)\,\mathbf{M}^{(k)} + \varrho \mathbf{M}^{(k-1)} \tag{74}$$

where $\mathbf{M}^{(k)}$ is the metric tensor calculated at adaptation k, $\mathbf{M}^{(k-1)}$ is the metric tensor calculated upon the previous mesh interpolated onto the current mesh and ϱ is some relaxation parameter. $\mathbf{M}^{(1)}$ is simply the metric tensor calculated from the initial grid.

For the examples given in this paper ϱ is taken to be 0.9. As the process is repeated the change in number of nodes between adaptations becomes very small. For this work adaption ceases when the change in number of nodes becomes $< 0.5\%$.

6 Numerical basis of model

The advection-diffusion equation is solved for the examples given in section 8. For the forward model:

$$\boldsymbol{u} \cdot \nabla C - \nabla \boldsymbol{\kappa} \nabla C = S, \tag{75}$$

and for the corresponding adjoint model,

$$-\boldsymbol{u} \cdot \nabla C^* - \nabla \boldsymbol{\kappa} \nabla C^* = S^*. \tag{76}$$

Here $\boldsymbol{u} = (u, v, w)$ represents the 3D velocity, u, v and w are the velocity components in the x-, y- and z-directions respectively. C is the concentration field and the diffusivity tensor is represented by $\boldsymbol{\kappa}$; in this model the $\boldsymbol{\kappa}$ is diagonal and isotropic (all diagonal values are equal). S and S^* are the sources in the forward and adjoint models.

The equations in 75 are solved using "the standard" Petrov-Galerkin method and linear tetrahedral elements. In this way a matrix equation involving the $\mathbf{A}$ is obtained for the solution of the forward problem and the matrix $\mathbf{A}^T$ is obtained for the adjoint problem. These matricies allow then enable the error measures defined in the previous section to be used. More details of the numerical basis of the model can be found in [22, 24].

6.1 Definition of a functional

A functional can be defined to optimize the accuracy of the solution in a certain region of the domain, let this region be $\Omega_{\mathcal{D}}$. In section 8 test cases are considered where the adaptivity algorithm is based upon the temperature field only, it is appropriate therefore to consider a functional based upon increasing

the accuracy of the solution of the temperature field in a certain region of the domain. In section 8 the region $\Omega_{\mathcal{D}^*}$ is referred to as a 'detector' and the region $\Omega_{\mathcal{D}}$ is referred to as the source region (the region containing a non-zero source a box shape in the example problems). Recalling the previous definition for the functional,

$$F = \int_{\Omega} f\left(\psi\right)\, dV,$$

a functional can now be defined such that,

$$F = \int_{\Omega} G^* C \, dV, \tag{77}$$

where $G^* = \sum_{j=1}^{\mathcal{N}} N_j G_j^*$ is a function with a finite element representation defined such that,

$$G_i^* = \begin{cases} \gamma^* & : \quad \text{if} \quad \mathbf{x}_i \in \Omega_{\mathcal{D}^*}; \\ 0 & : \quad \text{otherwise} \quad \mathbf{x}_i \notin \Omega_{\mathcal{D}^*}, \end{cases} \tag{78}$$

where $\mathbf{x}_i$ is the position vector of node i and γ^* is defined such that

$$\int G^* \, dV = G_v^*. \tag{79}$$

where G_v^* is a constant. This definition of G^* allows an unstructured mesh to be used while maintaining the volume integral $\int G^* \, dV$ to be constant of the detector defined by equation 6.1. Thus the adjoint source terms S_i^* for node i in equation 76 with $C_i = \Psi_i$ is calculated from:

$$S_i^* = \frac{\partial F}{\partial C_i} = \int N_i G^* dV. \tag{80}$$

Similarly, the volume integral under the source of the forward problem given in the discretised forward solution vector S must have a constant volume integral G_v say. Then the source terms S_i for node i in equation 76 is calculated from:

$$S_i = \int N_i G dV \tag{81}$$

where $G = \sum_{j=1}^{\mathcal{N}} N_j G_j$ is a function with a finite element representation defined such that,

$$G_i = \begin{cases} \gamma & : \quad \text{if} \quad \mathbf{x}_i \in \Omega_{\mathcal{D}}; \\ 0 & : \quad \text{otherwise} \quad \mathbf{x}_i \notin \Omega_{\mathcal{D}}, \end{cases} \tag{82}$$

where γ is defined such that

$$\int G \, dV = G_v. \tag{83}$$

for some scalar G_v.

7 Application of residual calculation to a diffusion equation on a structured grid

In this section residual calculations are tested for a second order problem on a structured grid; the diffusion equation $\nabla^2\psi = s$ is solved. A source s of unit strength is introduced in the central 0.2×0.2 region of domain $x \in [0, 1]$, $y \in [0, 1]$. At the boundaries a zero potential ψ boundary condition is enforced. Three simulations with finite element grids of resolution 50×50, 100×100 and 200×200 elements are used to demonstrate how the residual calculated using different methods converge with increased resolution. For simplicity a five point finite difference operator was used to discretize the diffusion equation and solve for the potential at the nodes denoted by the indices i, j, that is:

$$\text{Area}\left(\frac{-4\Psi_{i,j} + \Psi_{i,j-1} + \Psi_{i,j+1} + \Psi_{i-1,j} + \Psi_{i+1,j}}{h^2} - S_{i,j}\right) = 0, \qquad (84)$$

in which h is the width or size of the elements and Area is the area of each element, here Area $= h^2$. The truncation error may be obtained by dividing this equation by the area Area. The residual from the solution of this equation is calculated from a rotated operator:

$$r_{i,j} = \text{Area}\left(\frac{-4\Psi_{i,j} + \Psi_{i-1,j-1} + \Psi_{i-1,j+1} + \Psi_{i+1,j-1} + \Psi_{i+1,j+1}}{2h^2} - S_{i,j}\right). \qquad (85)$$

The indices i and j refer to the i^{th} and j^{th} cell measures from the bottom left $x = y = 0$ corner of the domain. The method of calculating the residual using interpolation theory is explained in section 5.3 and the multi-grid method defined by equation 65.

The purpose of this section is to validate these methods by comparing their residuals with the residual calculated using a high-order Taylor series expansion as presented by [47, 48] and the rotated operator residual in equation 85. The graphs in figures 4 to 7 show the residuals or truncation errors along a line $y = 0.5$ through the centre of the domain.

Figure 4 shows the potential ψ obtained using a mesh of 50×50 elements. The results for the other finer meshes are difficult to distinguish visually from this result.

The residuals, or more specifically the absolute value of the residuals, however are quite different. The residuals obtained for the high order and the interpolation theory method, are relatively large in the local vicinity of the interface, with the source as shown in figure 5. This figure shows that the magnitudes do not decrease with increased resolution, but the large residual magnitude regions do become narrower. Figures 6 & 7 compare the different residual calculation methods. Notice that the residual magnitude found by interpolation

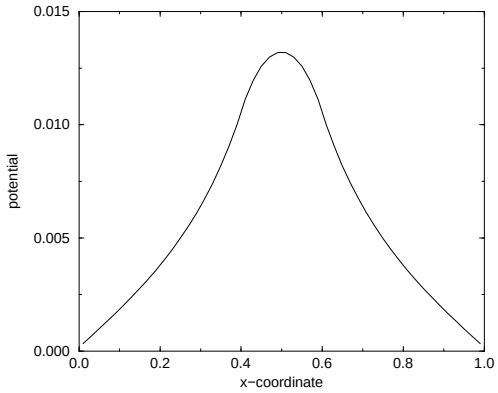


Fig. 4. Potential field generated with 50×50 elements. Cut-through at $y = 0.5$.

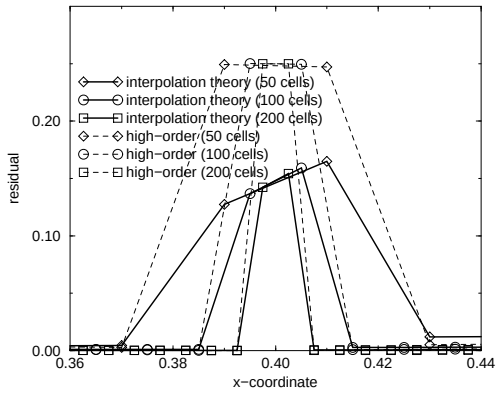


Fig. 5. Absolute value of the truncation errors (residuals/h^2) generated by interpolation theory and high-order Taylor series expansion at the interface of the source region. Cut-through at $y = 0.5$.

theory is in general rather conservative in producing larger residuals than the other methods. It is particularly larger in the source region because the source itself generates curvature, making this roughly twice the magnitude that is should be. We assume the value it should be is governed by the high-order Taylor series expansion. The magnitude of the residual calculated by the rotated operator is almost exactly 4 times smaller than that obtained from the Taylor series. Note should also be taken that the magnitude of the residuals (by comparing figures 6 & 7), increases with h^2 and is consequently 16 times smaller for the 200×200 element result than for the 50×50 result. In general, the interpolation theory approach to calculating residuals does an adequate job even in the difficult case of a second order operator and thus provides a simple alternative to the multi-grid approach. The multi-grid method produces virtually identical residuals (figures 5, 6 & 7) to the high order Taylor series method. The difference in these results is a maximum of about 2%. The only place where there is a significant difference is near the corners of the source and this is largely due to that fact that the multi-grid stencil is a 3×3 9-point stencil; and the high order Taylor series is a 13-point stencil (a 7-point 1-D stencil in each direction). This validates the use of the multi-grid method in these applications.

Note that the multi-grid method produces visually identical residuals (in these graphs) to the high order Taylor series method. The truncation errors shown are approximate equation truncation errors, that is $\frac{r_{i,j}}{\text{Area}}$.

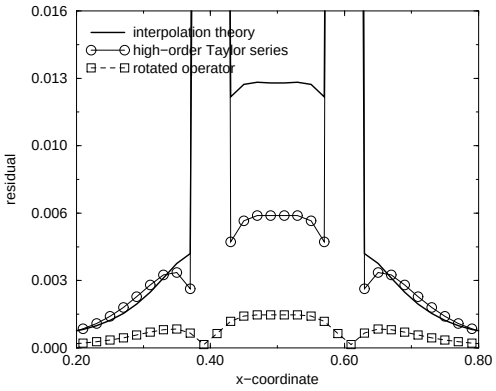


Fig. 6. Comparison of the absolute value of the truncation errors (residuals/h^2) using the 3 methods with a grid of 50×50 elements. Cut-through at $y = 0.5$.

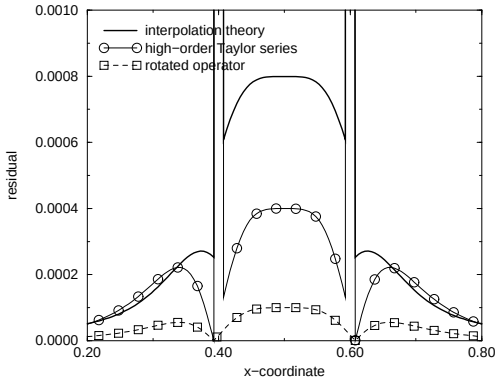


Fig. 7. Comparison of the absolute value of the truncation errors (residuals/h^2) using the 3 methods with a grid of 200×200 elements. Cut-through at $y = 0.5$.

8 Application of residual calculation to a diffusion equation on an unstructured grid

8.1 Simple source/detector problem

An application similar to that investigated in section 7 is explored here. In this example we employ a cuboidal domain $x \in [0, 4], y \in [0, 4], z \in [0, 0.02]$ with an initial grid resolution of $40 \times 40 \times 1$ tetrahedral finite elements. A source of unit strength, cuboidal in shape, with edge lengths 0.2, 0.2 and 0.02 in the x- y- and z-directions respectively, is centred at $(1.0, 2.0, 0.01)$, with a corresponding receiver centred at $(3.0, 2.0, 0.01)$ and of the same size. The integral within the source region is 8×10^{-4}, giving a corresponding potential value of $\psi = 1$ within that region.

A functional F is considered (as described in sub-section 6.1) where the mesh in the vicinity of the detector region is to be optimized; consequently the source for the adjoint problem $\left(\frac{\partial F}{\partial \psi}\right)$ is equal to approximately unity (corrected to take make the volume integral in the source and detector regions constant, see section 6.1) within the detector region and zero elsewhere. The integral over the source region is maintained at a constant value while the meshes are adapting.

The inlet boundaries where $\mathbf{n} \cdot \mathbf{u} < 0$ have a zero potential ψ condition enforced otherwise a zero normal derivative condition is applied as a natural boundary condition. The initial mesh contains 3362 nodes and 9600 elements. A velocity field $\mathbf{u} = (15.0, 0, 0)$ is also Imposed with a unit isotropic diffusion coefficient, see equation 75. The solutions to the forward and adjoint problems are shown

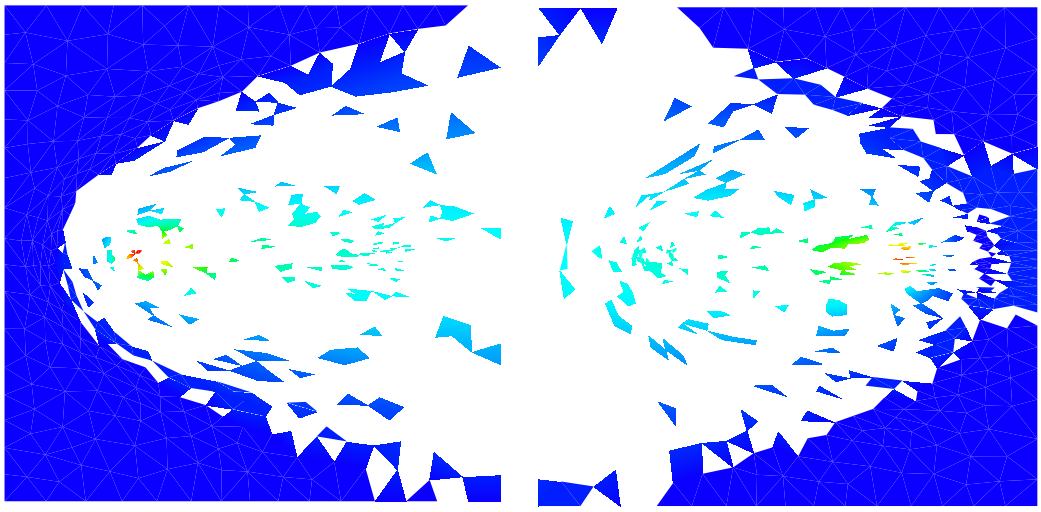


Fig. 8. Solution to the forward problem.

Fig. 9. Solution of the adjoint problem.

in figures 8 and 9 respectively. The forward residual corresponding to these solutions is shown in figure 17. The adaptivity algorithm is constrained so that the domain maintains one element in the vertical. New element edge lengths are limited to being in the region $l \in [0.001, 0.25]$ (in the horizontal). In this case the number of nodes (and consequently the number of elements) is effectively unrestricted to avoid any scaling of the element sizes. The parameter δF is taken to be 5×10^{-8}.

Figure 10, shows the mesh adapted using the forward metric tensor, defined by equation 42. Areas highlighted by both the Hessian of the forward solution (equation 40) and the residual can be seen to be receiving mesh refinement, while in other areas the mesh is coarsened.

Figure 11 shows the mesh adapted using the adjoint metric tensor, defined by equation 45. Areas highlighted by both the Hessian of the adjoint solution (equation 43) and the residual can be seen to be receiving substantial mesh refinement, while again, in other areas the mesh is coarsened. An interesting feature of the mesh is a very distinct area of fine resolution centred around the location of the detector. This results from the inclusion of the λ^+ terms in the definition of the *adjoint* metric tensor (see equation 45). This area is hard to distinguish from figure 11, so a blowup of the mesh is given in figure 12. Figure 12 shows an area of mesh refinement directly resulting from the inclusion of the forward Hessian matrix $\mathbf{H}$ and source terms for the adjoint problem. This pattern of refinement results from the type of functional being employed, and it may be valuable to not consider its contribution to the overall metric tensor. A *modified adjoint* metric tensor (equation 48) is applied to the same problem. Figure 14, shows the mesh adapted using this *modified adjoint* metric tensor, the result is similar to that of figure 11, but without the additional square form refinement as highlighted previously. Figure 14 is also symmetrical under

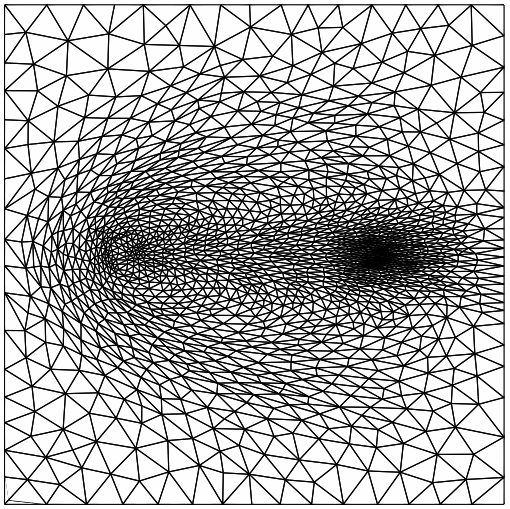


Fig. 10. Mesh resulting from the use of the forward metric tensor to adapt the initial mesh. 3296 nodes, 10467 elements.

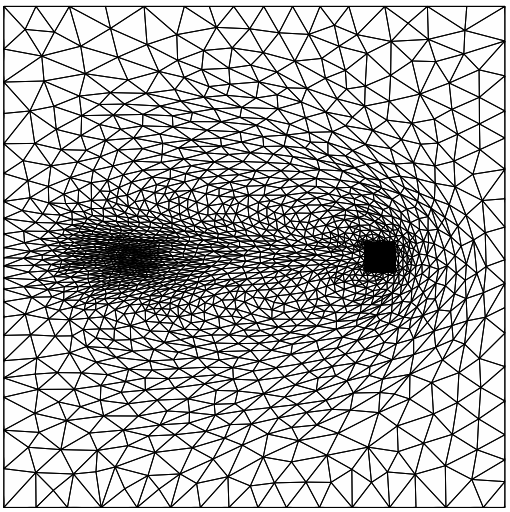


Fig. 11. Mesh resulting from the use of the adjoint metric tensor to adapt the initial mesh. 8061 nodes, 27217 elements.

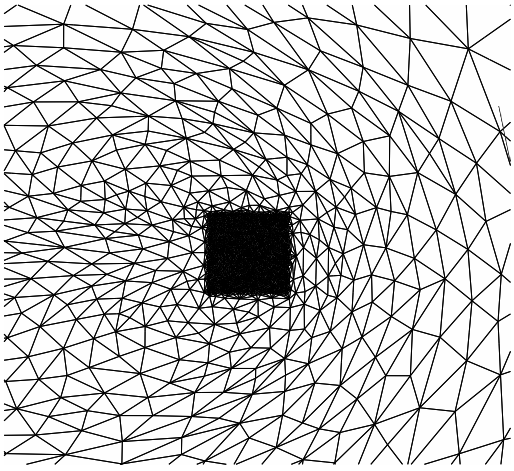


Fig. 12. Blowup of the mesh shown in figure 11 around point $(3.0, 1.5)$ showing the squareform adaptation.

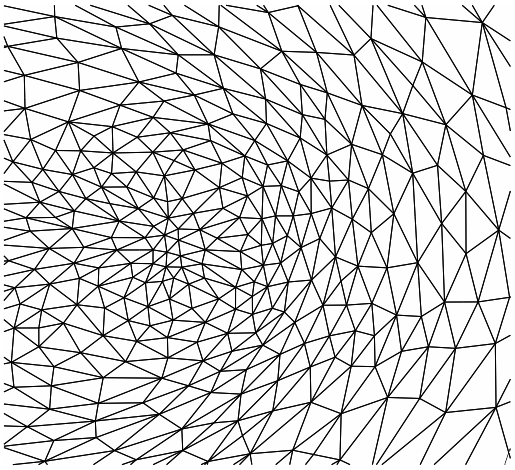


Fig. 13. Blowup of the mesh shown in figure 14 around point $(3.0, 1.5)$.

a 180° rotation, as expected. For completeness figure 13 shows a blowup of the same region as figure 12

Figure 15 shows the mesh adapted using a metric tensor $\bar{\mathbf{M}}^G$ resulting from the superposition of the *forward* and *modified adjoint* metric tensors (see section 4.3 for details). Anisotropic elements dictated by both component metric tensors can be seen in the adapted mesh. For comparison figure 16 shows the mesh adapted using the averaged metric tensor (see also section 4.3). Additional anisotropic structure can be observed in this mesh.

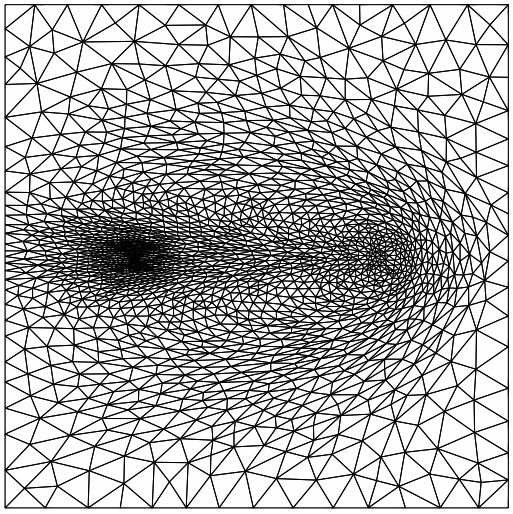


Fig. 14. Mesh resulting from the use of the modified adjoint metric tensor to adapt the initial mesh. 3216 nodes, 10150 elements.

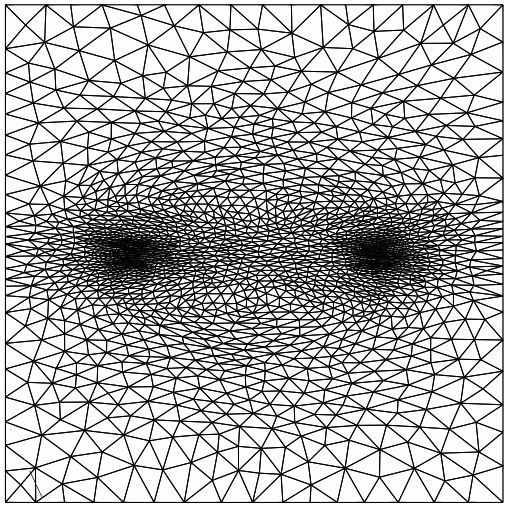


Fig. 15. Mesh resulting from the use of a superposition of forward and modified adjoint metric tensors to adapt the initial mesh. 3576 nodes, 11023 elements.

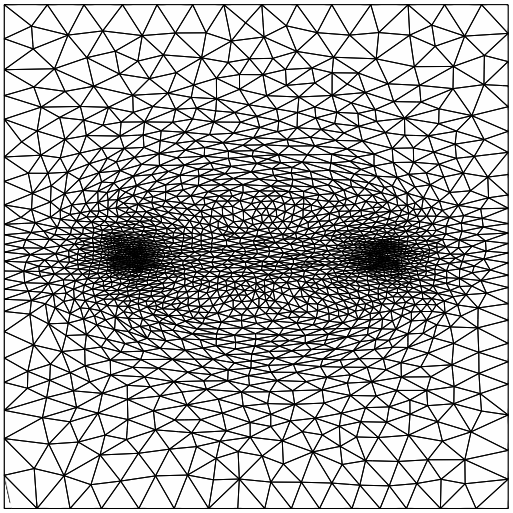


Fig. 16. Mesh resulting from the use of the averaged metric tensor ($\bar{\bar{\mathbf{M}}}$, see section 50) to adapt the initial mesh. 3712 nodes, 11287 elements.

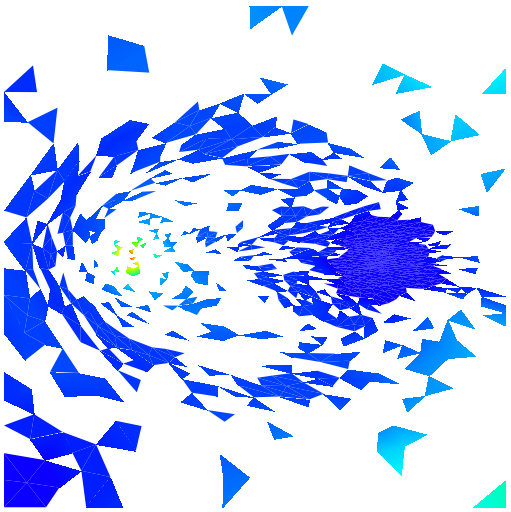
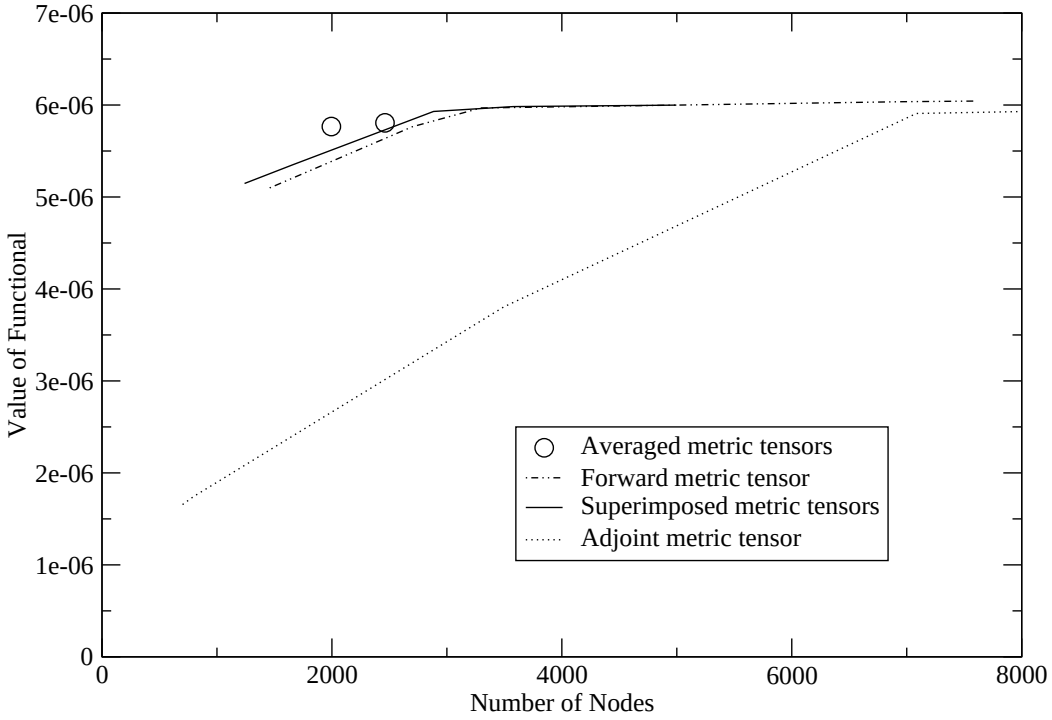


Fig. 17. Forward residual corresonding to the solution given in figure 10.

Figures 18 and 19 compare the different metric tensors for this problem which are compared visually in figures 10 to 16. The graphs plot values of the functional to be optimized for the different metric tensors, a blowup is given in figure 19. The graph indicates the failings of the adjoint metric tensor for achieving a set functional value within a set number of nodes, compared to the other options. The best method appears to be the averaging of metric tensors, from figure 16 it is possible to see both fine resolution and directional information for this choice of metric tensor.



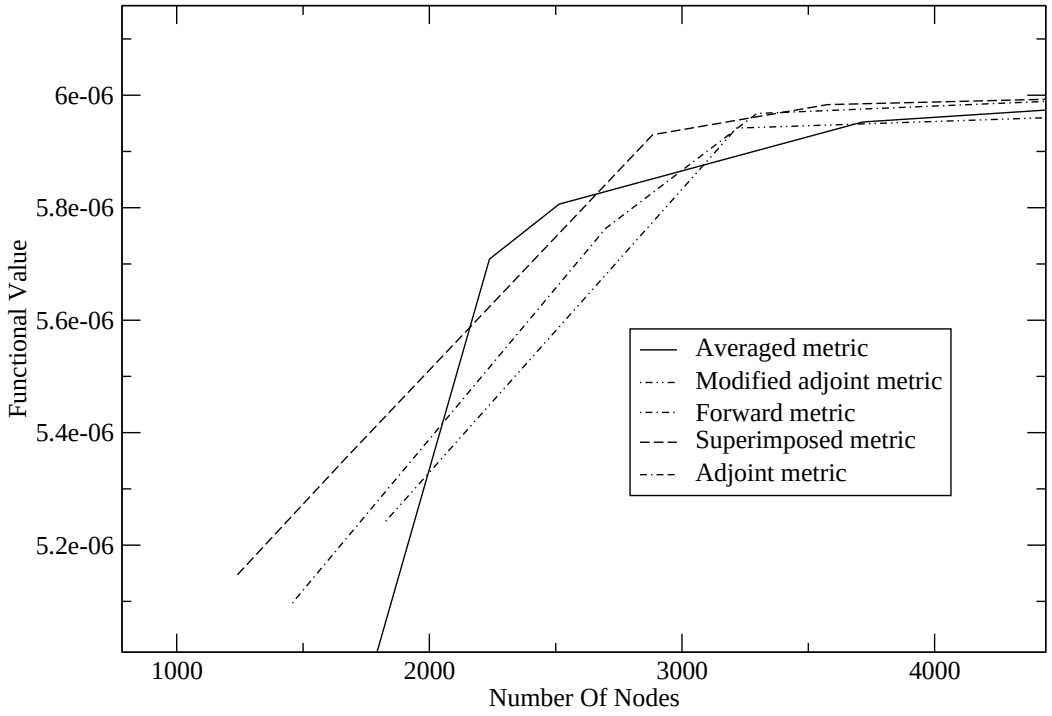


Fig. 19. Graph comparing the value of the functional for mesh adapted using different metric tensors, blowup around area of 2000–4000 nodes.

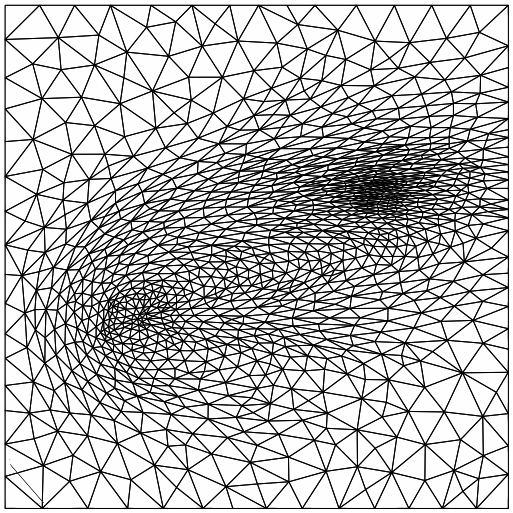


Fig. 20. Mesh adapted using forward metric tensor for an offset source and detector. 2578 nodes, 8453 elements.

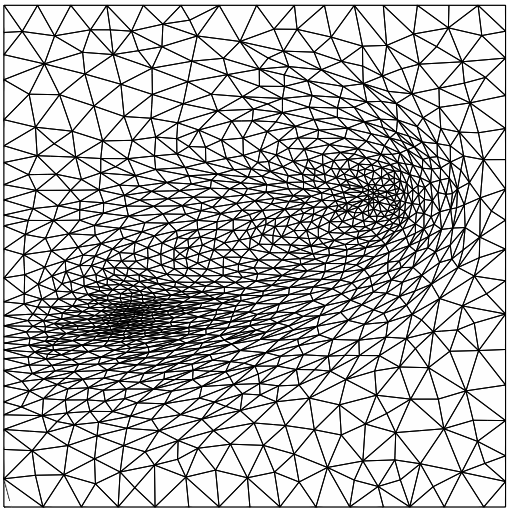


Fig. 21. Mesh adapted using modified adjoint metric tensor for an offset source and detector. 2643 nodes, 8655 elements.

8.2 Off set source and detector

Following the same principles as the problem discussed in the previous section a slight change can be made to test the method. Off setting the source and detector ensures the resulting metric tensors loose some of the symmetry seen in the results of section 8.1. The source is now located at $(1.0, 1.5, 0.01)$, with the receiver at $(3.0, 2.5, 0.01)$. All other parameters remain the same. Figure 20 shows the mesh adapted using the forward metric tensor, figure 21 using the modified adjoint metric tensor and figure 22 using a superposition of the forward and modified adjoint metric tensors. Clear directional information resulting from the forward and adjoint Hessians can be seen in figures 20 and 21 respectively.

8.3 Two field problem

The definitions of metric tensors in section 4 extend to multiple fields, to demonstrate the combining of information from more than one field an idealised test case is considered here. Figures 24 and 25 show the forward solutions for a problem with two temperature fields. A single source in placed in each of the two fields at $(1.0, 1.5, 0.01)$ and $(1.0, 2.5, 0.01)$, with corresponding detectors at $(3.0, 1.5, 0.01)$ and $(3.0, 2.5, 0.01)$. All other parameters are as defined in section 8.1 The result is two independent solutions, for which information can be combined in constructing a metric tensor to adapt the mesh. Figure 23 shows a mesh adapted using a metric tensor based on a superposition of

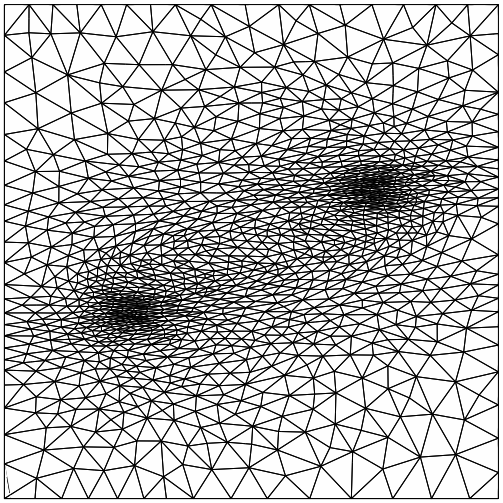


Fig. 22. Mesh adapted using superimposed forward and modified adjoint metric tensors, for offset source and detector. 2747 nodes, 8699 elements.

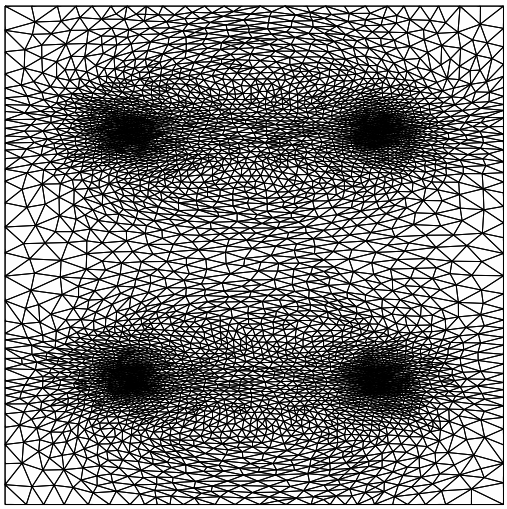


Fig. 23. Mesh adapted using superimposed forward and modified adjoint metric tensors, for 2 solution fields each with a source and detector. 9382 nodes, 30261 elements.

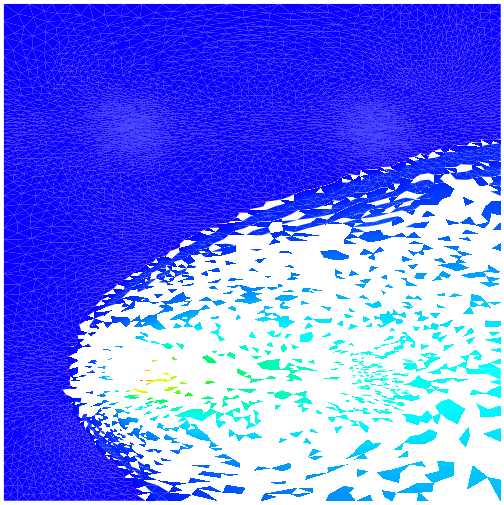


Fig. 24. Solution for first of two temperature fields.

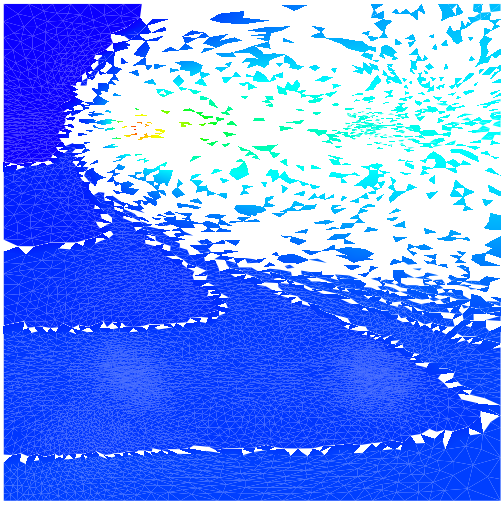


Fig. 25. Solution for second of two temperature fields.

forward and modified adjoint metric tensors.

8.4 Three dimensions

The same method can be applied in three dimensions, with the domain changed to become a cube of length 4.0, with an initial grid resolution of $40 \times 40\times$ tetrahedral elements. The initial mesh contains 68921 nodes and 384000 ele-

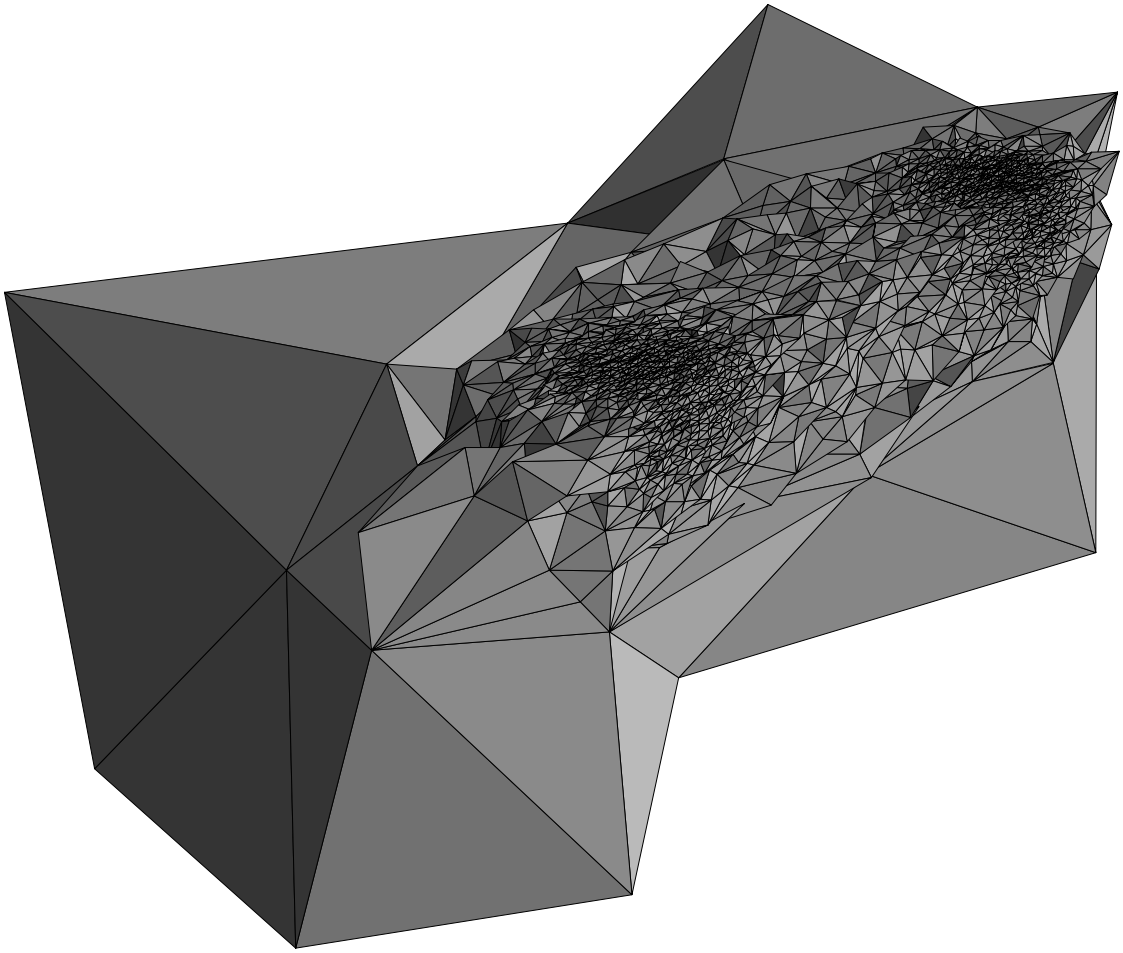


Fig. 26. Adaptation in three-dimensions, elements where $x > 2$ and $z > 2$ are removed. 28974 nodes, 165049 elements.

ments. In this case the source and detector are located at $(1.0, 2.0, 2.0)$ and $(3.0, 2.0, 2.0)$ respectively and have their size adjusted to be a cube with sides of length 0.2. The integral of the source/detector is consequently adjusted to be 0.008. To account for the change to three dimensions and the increased number of nodes the parameter δF is changed to $\delta F = 5 \times 10^{-7}$. The maximum edge length allowable when the mesh is adapted is also increased to 2.0.

Figure 26 shows a cut away of the domain adapted using a superposition of the forward and modified adjoint metric tensors. The figure excludes elements containing any point with $x > 2$ and $z > 2$ so the mesh refinement can be seen. The adaptation pattern in the three dimensional sense is easy to determine, exhibiting the same characteristics as the previous results.

9 Conclusions

In this paper a sensitivity or goal based error norm is defined. This results in a metric tensor from which 3-D anisotropic mesh adaptivity with tetrahedral elements is performed. The advantage of this approach is three fold: the super-convergence of the goal with mesh adaptivity; the ability to assign bounds on the accuracy of this goal; and the possibility of improving the accuracy of the goal. The application to relatively simple advection-diffusion problems demonstrate the robustness of this approach and a number of metric tensor based error measures are assessed. The method that using a superposition of metric fields obtained from the forward and adjoint Hessians performs the best closely followed by the method that uses the forward Hessian and the method that takes an average of forward and adjoint Metrics and then the method based on the adjoint Hessian. There is a focus on developing relatively simple methods that refer to information from the discretized equation sets (often readily accessible in simulation codes) and do not explicitly use equation residuals. Thus a number of methods based on a multi-grid approach have been outlined here. This general approach was shown to be highly accurate and resulting in identical residuals for a simple diffusion problem to that obtained from a high order Taylor series analysis.

Future work will concentrate on defining appropriate integral measures of the dynamics of various non-linear fluid problems from which anisotropic mesh adaptivity can be performed; as well as the application to time dependent non-linear and inverse problems.

Acknowledgements

The authors would like to thank Prof. C.R.E. de Oliveria for a major contribution in helping to construct the mesh adaptivity method method used in this work. The authors would also like to acknowledge the support of the UK Natural Environment Research Council (Grant NER/A/S/2000/00375).

References

[1] R. Rannacher and F.T. Suttmeier. *a-posteriori* error control in finite element methods via duality techniques: Application to perfect plasticity. *Comput. Mech.*, 21:123–133, 1998.

[2] M. Paraschivoiu and A. Patera. A hierarchical duality approach to bounds for the outputs of partial differential equations. *Comput. Meth. Appl. Mech. Eng.*, 158:389–407, 1998.

[3] J. Peraire and A.T. Patera. *Advances in adaptive computational methods in mechanics*, pages 199–215. Elsevier, 1998.

[4] J. Peraire, M. Vahdati, K. Morgan, and O.C. Zienkiewicz. Adaptive remeshing for compressible flow computations. *J. Comput. Phys.*, 72:449–466, 1987.

[5] J. Wu, J.Z. Zhu, J. Szmelter, and O.C. Zienkiewicz. Error estimation and adaptivity in Navier-Stokes incompressible flows. *Comput. Mech.*, 6:259–270, 1990.

[6] R. Lohner, K. Morgan, and O.C. Zienkiewicz. An adaptive finite element procedure for compressible high speed flows. *Comput. Meth. Appl. Mech. Eng.*, 51:441–465, 1985.

[7] M.D. Piggott, C C. Pain, G.J. Gorman, P.W. Power, and A.J.H. Goddard. *h, r* and *hr* adaptivity with applications in numerical ocean modelling. *Ocean Modelling*, 10:95–113, 2005.

[8] T. Strouboulis and J.T. Oden. *a-posteriori* error estimation of finite element approximations in fluid mechanics. *Comput. Meth. Appl. Mech. Eng.*, 78:201–242, 1990.

[9] M. Ainsworth and J.T. Oden. *a-posteriori* error estimation in finite element analysis. *Comput. Meth. Appl. Mech. Eng.*, 142:1–88, 1997.

[10] M. Ainsworth and J.T. Oden. A unified approach to *a-posteriori* error estimation using element residual methods. *Numer. Math.*, 65:23–50, 1993.

[11] I. Babuska and W.C. Rheinboldt. *a-posteriori* error estimates for the finite element method. *Int. J. Num. Meth. Eng.*, 12:1597–1615, 1978.

[12] R.E. Bank and R.K. Smith. *a-posteriori* error estimates based on hierarchical bases. *SIAM J. Numer. Anal.*, 30:921–935, 1993.

[13] R.E. Bank and A. Weiser. Some *a-posteriori* error estimators for elliptic partial differential equations. *Math. Comp.*, 44:283–301, 1985.

[14] R. Verfurth. *A review of a-posteriori error estimation and adaptive mesh-refinement techniques.* Wiley-Teubner, 1996.

[15] O.C. Zienkiewicz and J.Z. Zhu. The superconvergent patch recovery and *a-posteriori* error estimates. Part 1: the recovery technique. *Int. J. Num. Meth. Eng.*, 33:1331–1364, 1992.

[16] O.C. Zienkiewicz and J.Z. Zhu. The superconvergent patch recovery and *a-posteriori* error estimates. Part 2: error estimates and adaptivity. *Int. J. Num. Meth. Eng.*, 33:1365–1382, 1992.

[17] L.A. Freitag and C. Ollivier-Gooch. Tetrahedral mesh improvement using swapping and smoothing. *Int. J. Num. Meth. Eng.*, 40:3979–4002, 1997.

[18] G.C. Buscaglia and E.A. Dari. Anisotropic mesh optimization and its application in adaptivity. *Int. J. Num. Meth. Eng.*, 40:4119–4136, 1997.

[19] P.L. George. *Delaunay Triangulation and Meshing: Application to Finite Elements.* HERMES, Paris, 1998.

[20] P.L. George, F. Hecht, and E. Saltel. Automatic mesh generator with a specified boundary. *Comput. Meth. Appl. Mech. Eng.*, 92:269–288, 1991.

[21] C.C. Pain, A.P. Umpleby, C.R.E. de Oliveira, and A.J.H. Goddard. Tetra-

hedral mesh optimisation and adaptivity for steady-state and transient finite element calculations. *Comput. Meth. Appl. Mech. Eng.*, 190:3771–3796, 2001.

[22] R. Ford, C.C. Pain, M.D. Piggott, A.J.H. Goddard, C R.E. de Oliveira, and A P. Umpleby. A non-hydrostatic finite element model for three-dimensional stratified ocean flows. Part I: Model formulation. *Monthly Weather Rev.*, 132:2816–2831, 2004.

[23] R. Ford, C.C. Pain, M.D. Piggott, A.J.H. Goddard, C R.E. de Oliveira, and A P. Umpleby. A non-hydrostatic finite element model for three-dimensional stratified ocean flows. Part II: Model validation. *Monthly Weather Rev.*, 132:2832–2844, 2004.

[24] C.C. Pain, M.D. Piggott, A.J.H. Goddard, F. Fang, G.J. Gorman, D.P. Marshall, M.D. Eaton, P.W. Power, and C.R.E. de Oliveira. Three-dimensional unstructured mesh ocean modelling. *Ocean Modelling*, 10:5–33, 2005.

[25] Y. Kallinderis and P. Vijayan. Adaptive refinement-coarsening scheme for three-dimensional unstuctured meshes. *AIAA J.*, 31:1440–1447, 1993.

[26] R. Lohner and P. Parikh. Generation of three-dimensional unstructured grids by the advancing-front method. *Int. J. Num. Meth. Fluids*, 8:1135–1149, 1988.

[27] P. Moller and P. Hansbo. On advancing front mesh generation in three dimensions. *Int. J. Num. Meth. Eng.*, 38:3551–3569, 1995.

[28] X. Xu, C.C. Pain, A.J.H. Goddard, and C.R.E. de Oliveria. An automatic adaptive meshing technique for delaunay triangulations. *Comput. Meth. Appl. Mech. Eng.*, 161:297–303, 1998.

[29] H. Borouchaki and P.J. Frey. Adaptive triangular-quadrilateral mesh generation. *Int. J. Num. Meth. Eng.*, 41:915–934, 1998.

[30] H. Borouchaki and S.H. Lo. Fast Delaunay triangulation in three dimensions. *Comput. Meth. Appl. Mech. Eng.*, 128:153–167, 1995.

[31] H. Borouchaki and P.L. George. Optimal Delaunay point insertion. *Int. J. Num. Meth. Eng.*, 39:3407–3437, 1996.

[32] M.J. Castro-Diaz, F. Hecht, B. Mohammadi, and O. Pironneau. Anisotropic unstructured mesh adaption for flow simulations. *Int. J. Num. Meth. Fluids*, 25:475–491, 1997.

[33] N.P. Weatherill. Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints. *Int. J. Num. Meth. Eng.*, 37:2005–2039, 1994.

[34] B. Joe. Three-dimensional triangulations from local transformations. *SIAM J. Sci. Stat. Comput.*, 10:718–741, 1989.

[35] D.G. Cacuci. Sensitivty theory for non-linear systems. I: Nonlinear function analysis approach. *J. Math. Phys.*, 22:2794–2802, 1981.

[36] B.F. Sanders and N.D. Katopodes. Control of canal flow by adjoint sensitivity method. *J. Irrigation Drainage Eng.*, 125:287–297, 1999.

[37] M. Piasecki and N. Katopodes. Control of contaminant releases in rivers. I: Adjoint sensitivity analysis. *J. Hydr. Engrg.*, 123:488–492, 1997.

[38] B.F. Sanders and N.D. Katopodes. Adjoint sensitivity analysis for shallow water wave control. *J. Eng. Mech.*, 126:909–919, 2000.

[39] S. Zhang, X. Zou, J. Ahlquist, I.M. Navon, and J.G. Sela. Use of differentiable and non-differentiable optimization algorithms for variational data assimilation with discontinuous cost functions. *Monthly Weather Rev.*, 128:4031–4044, 2000.

[40] M. Gunzburger. Sensitivities, adjoints and flow optimization. *Int. J. Num. Meth. Fluids*, 31:53–78, 1999.

[41] R.H. Langland, R.L. Elsberry, and R.M. Errico. Evaluation of physical processes in an idealized extratropical cyclone using adjoint sensitivity. *Q. J. R. Meteorol. Soc.*, 121:1349–1386, 1995.

[42] C. Homescu and I.M. Navon. Numerical and theoretical considerations for sensitivity calculation of discontinuous flow. *Systems and Control Letters*, 48:253–260, 2003.

[43] D.G. Cacuci. *The forward and adjoint methods of sensitivity analysis*, chapter 3, pages 71–144. CRC Press, Inc., 1988.

[44] A.K. Alekseev and I.M. Navon. *a-posteriori* pointwise error estimation for compressible fluid flows using adjoint parameters and Lagrange remainder. *Int. J. Num. Meth. Fluids*, 47:45–74, 2005.

[45] F-X. Le Dimet, I.M. Navon, and D.N. Daescu. Second-order information in data assimilation. *Monthly Weather Rev.*, 130:629–648, 2002.

[46] A.K. Alekseev and I.M. Navon. The analysis of an ill-posed problem using multi-scale resolution and second-order adjoint techniques. *Comput. Meth. Appl. Mech. Eng.*, 190:1937–1953, 2001.

[47] I.M. Navon. Variational data assimilation with an adiabatic version of the NMC spectral model. *Monthly Weather Rev.*, 120:1433–1446, 1992.

[48] F-X. Le Dimet and I.M. Navon. Technical Report: Early Review on Variational data assimilation. SCRI Technical Report, 1988.

[49] T.J.R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: IV. A discontinuity-capturing operator for multidimensional advective-diffusive systems. *Comput. Meth. Appl. Mech. Eng.*, 58:329, 1986.

[50] B.P. Leonard. The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection. *Comput. Meth. Appl. Mech. Eng.*, 88:17, 1991.

[51] F. Fang, M.D. Piggott, C.C. Pain, G.J. Gorman, and A.J.H. Goddard. An adaptive mesh adjoint data assimilation method for coastal flows. Submitted to Ocean Modelling, 2005.

[52] A. Griewank. Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation. *Optimization Methods and Software*, 1:35–54, 1992.

[53] S.K. Nadarajah and A. Jameson. A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization. In *AIAA 38th Aerospace Sciences Meeting, Reno, NV.*, 2000.

[54] N.A. Pierce and M.B. Giles. Adjoint recovery of superconvergent func-

tionals from PDE approximations. *SIAM Review*, 42:247–264, 2000.

[55] J.-D. Müller and M.B. Giles. Solution adaptive mesh refinement using adjoint error analysis. 15th Computational Fluid Dynamics Conference, American Institute of Aeronautics and Astronautics, June 2001.

[56] R. Hide. A note on helicity. *Geophys. Astrophys. Fluid Dyn.*, 7:157–161, 1976.

[57] R. Hide. Superhelicity, helicity and potential vorticity. *Geophys. Astrophys. Fluid Dyn.*, 48:69–79, 1989.

[58] R. Hide. Helicity, superhelicity and weighted relative potential vorticity: Useful disgnostic pseudoscalars? *Q. J. R. Meteorol. Soc.*, 128:1759–1762, 2002.

[59] J.T. Oden and S. Prudhomme. Goal-oriented error estimation and adaptivity for the finite element method. *Comp. & Math. with Applic.*, 41:735–756, 2001.

[60] S. Prudhomme and J.T. Oden. On goal-oriented error estimation for elliptic problems. *Comput. Meth. Appl. Mech. Eng.*, 179:313–331, 1999.

[61] F. Cirak and E. Ramm. *a-posteriori* error estimation and adaptivity for linear elasticity using the reciprocal theorem. *Comput. Meth. Appl. Mech. Eng.*, 156:351–362, 1998.

[62] D.A. Venditti and D.L. Darmofal. Adjoint error estimation and grid adaption for functional outputs: Application to quasi-one-dimensional flow. *J. Comput. Phys.*, 164:204, 2000.

[63] P.M. Knupp. Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. Part II — A framework for volume mesh optimisation and the condition number of the Jacobian matrix. *Int. J. Num. Meth. Eng.*, 48:1165–1185, 2000.

[64] D.A. Venditti and D.L. Darmofal. Grid adaption for functional outputs: Application to two-dimensional inviscid flows. *J. Comput. Phys.*, 176:40–69, 2002.

[65] D.A. Venditti and D.L. Darmofal. Anisotropic grid adaption for functional outputs: Application to two-dimensional viscous flows. *J. Comput. Phys.*, 187:22–46, 2003.

[66] O.C. Zienkiewicz and R.L. Taylor. The Finite Element Method, Volume 1. 5th Edition, 2000.