Electronic Transactions on Numerical Analysis.
Volume 15, pp. 78-93, 2003.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

COMPARISON OF NON-LINEAR SOLVERS FOR THE SOLUTION OF RADIATION TRANSPORT EQUATIONS*

LINDA STALS†

Abstract. We compare the performance of an inexact Newton-multigrid method and Full Approximation Storage multigrid when solving radiation transport equations. We also present an adaptive refinement algorithm and explore its impact on the solution of such equations.

Key words. FAS, Multigrid, Newton Method, Radiation Transport

AMS subject classifications. 35K55, 65M55, 65M60, 49M15

1. Introduction. Interest in the solution of radiation transport equations stems from the modeling of applications found in, for example, combustion, astrophysics and laser fusion. Features such as strong nonlinearities and large jumps in the coefficients make these equations difficult to solve and they can consume considerable computational resources if efficient solution techniques are not used. Two examples of efficient solution methods are the inexact Newton-multigrid method and the Full Approximation Storage (FAS) multigrid; both of which are nonlinear multigrid techniques.

The solution of radiation transport equations usually contains a wave front. Adaptively refined grids are well suited to capture the information along the front and thus give high resolution results.

In this paper we compare the performance of an inexact Newton-multigrid method and the FAS method for the solution of time-dependent radiation transport equations. We also explore the use of adaptive refinement techniques.

2. Physical Model. Under certain assumptions, such as isotropic radiation, optically thick material and temperature equilibrium, radiation transport may be modeled by the equation:

$$\frac{\partial E}{\partial t} - \nabla.(D(E)\nabla E) = 0 \qquad \text{on } \Omega \times I, \tag{2.1}$$

where E is the radiation energy.

More physically meaningful models of radiation transport are represented by systems of equations like those described in [4, 12, 14]; however, Equation (2.1) contains many of the features seen in the more general system, such as strong nonlinearities and large jumps in the coefficients, and therefore is a good place to start our investigation into different solution techniques.

One definition of the diffusivity term, $D(E)$, is:

$$D_1(E) = Z^{\alpha} E^{\beta}, \tag{2.2}$$

with $\alpha < 0$, $0 \leq \beta \leq 1$. Typically α is taken to be between -1 and -3 while β is taken to be $1/4$ or $3/4$. Z is the atomic mass number and may vary within the domain to reflect inhomogeneities in the material. The constant β controls the strength of the nonlinearity while α affects the size of the jumps in the coefficients.

*This research was supported by the U.S. Department of Energy under the ASCI ASAP program (subcontract B347882 from the Lawrence Livermore National Laboratory). Received May 22, 2001. Accepted for publication October 20, 2001. Recommended by Van Henson.

†Department of Computer Science, Old Dominion University, Norfolk, VA 23529-0162, USA AND ICASE, NASA Langley Res. Ctr, Hampton, VA 23681-2199, USA.

ETNA
Kent State University
etna@mcs.kent.edu

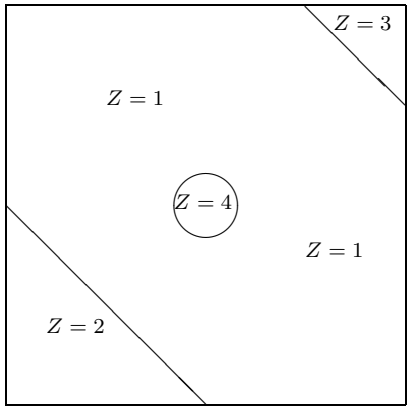

FIG. 2.1. *The values for the atomic mass number Z depend on the topology of the material. In our model problem, we define Z as shown above.*

The definition of $D(E)$ given in Equation (2.2) may produce results that shows the energy moving through the system at a rate faster than the speed of light. Consequently a flux limiter is included to slow down the movement and the diffusivity term is rewritten as:

$$D_2(E) = \left(\frac{1}{Z^\alpha E^\beta} + \frac{|\nabla E|}{E} \right)^{-1}. \tag{2.3}$$

The domain, Ω, is a square domain ($[0,1] \times [0,1]$) with the following mixture of Newton and Neumann boundary conditions;

$$\begin{array}{ll} \partial E/\partial n = 0 & \text{on } \Gamma_N \times I, \\ n^T D(E) \nabla E + E/2 = 2 & \text{on } \Gamma_{F_0} \times I, \\ n^T D(E) \nabla E + E/2 = 0 & \text{on } \Gamma_{F_1} \times I, \end{array}$$

where Γ_N represents the lines $y = 0$ and $y = 1$, Γ_{F_0} is the line $x = 0$ and Γ_{F_1} is the line $x = 1$, n is the outward unit normal and I is the time interval.

In our model problem we take Z to be 1.0 except in the following regions: $\sqrt{(x-0.5)^2 + (y-0.5)^2} \leq 0.125$, $y \leq 0.5 - x$ and $y \geq 1.75 - x$ where $Z = 4$, $Z = 2$ and $Z = 3$ respectively. See Figure 2.1.

Initially, the energy, E is set to be the constant value $E = 10^{-5}$.

Additional papers that look at these equations include [3, 4, 5, 7, 12, 16] and their accompanying references. The radiation transport model described here is very similar to that presented in [7, 12, 16], except that we use the finite element method instead of the finite difference method, as this more readily allows the use of adaptive refinement.

3. Discretization. To solve Equation (2.1), we use the following variational formulation, similar to that given in [6]: Find $u(t) \in V = H^1(\Omega)$, $t \in I$, such that

$$(\dot{u}(t), v)_\Omega + a(D(u(t)); u(t), v)_\Omega = < g, v >_{\partial\Omega} \qquad \forall v \in V, \tag{3.1}$$

where

$$a(D(u); v, w)_\Omega = \int_\Omega D(u) \nabla v \nabla w \, d\Omega + \int_{\Gamma_{F_0}} \frac{vw}{2} \, d\Gamma + \int_{\Gamma_{F_1}} \frac{vw}{2} \, d\Gamma,$$

ETNA
Kent State University
etna@mcs.kent.edu

$$< v, w >_{\partial\Omega} = \int_{\partial\Omega} vw \, d\Omega,$$

and

$$g = \begin{cases} 0 & \text{on } \Gamma_N \times I, \\ 2 & \text{on } \Gamma_{F_0} \times I, \\ 0 & \text{on } \Gamma_{F_1} \times I. \end{cases}$$

The time derivative is dealt with by either an implicit Euler or the Crank-Nicolson method.

4. Node-Edge Data Structure. A node-edge data structure similar to the one described in [15, 17, 18, 20] is used to store the finite element grid. In this data structure, a grid $\mathcal{M}^m$ is defined in terms of its geometrical, topological and algebraic attributes. The geometrical and topological attributes are simply the set of *nodes*, $\mathcal{N}^m$, and *edges*, $\mathcal{E}^m$. The stiffness matrix is associated with the algebraic attribute and is stored in the set of connections, $\mathcal{C}_A^m$, as a graph.

The node-edge data structure does not *explicitly* contain any information about the elements in the grid. Consequently, the same data structure can be used to store triangular, quadrilateral or tetrahedral grids. Information about the elements may be extracted by looping through the nodes and edges if necessary. The advantage of such a data structure is its flexibility. Although we concentrate on serial results in this paper, the code has been implemented in parallel and those results will be presented in a future paper.

We use a refinement algorithm to build the hierarchy of grids needed by our solution techniques (multigrid methods). In notational form this nested sequence of grids is represented by $\mathcal{M}^1 \subset \mathcal{M}^2 \subset ... \subset \mathcal{M}^n$. A more thorough description of this refinement algorithm is given in Section 6.

Information is moved from the coarse grid $\mathcal{M}^{m-1}$ to the fine grid $\mathcal{M}^m$ by the linear interpolation matrix $\mathbf{I}_{m-1}^m$. The restriction matrix $\mathbf{I}_m^{m-1}$ is defined to be the transpose of the interpolation matrix and moves the information from the fine grid $\mathcal{M}^m$ to the coarse grid $\mathcal{M}^{m-1}$. This extra algebraic information is stored in the table of connections. That is, if $\mathcal{C}_I^m$, $\mathcal{C}_R^m$ and $\mathcal{C}_A^m$ hold the interpolation, restriction and stiffness matrices, respectively, then the algebraic information for a multigrid grid is the set of *connections*, $\mathcal{C}^m$, where

$$\mathcal{C}^m = \mathcal{C}_A^m \cup \mathcal{C}_I^m \cup \mathcal{C}_R^m,$$

for $1 < m < n$, $\mathcal{C}^1 = \mathcal{C}_A^1 \cup \mathcal{C}_I^1$ and $\mathcal{C}^n = \mathcal{C}_A^n \cup \mathcal{C}_R^n$.

Finally, the grid $\mathcal{M}^m$ is given by $\mathcal{M}^m = \{\mathcal{N}^m, \mathcal{E}^m, \mathcal{C}^m\}$.

Conceptually the user views $\mathcal{E}^m$ as a set of edges, but internally they are stored as sets of neighbor endpoints. Given a node N_i it's set of neighbor endpoints is defined to be:

$$\mathcal{B}(\mathcal{E}, N_i) = \{N_j : (N_i, N_j) \in \mathcal{E}\}.$$

The neighbor endpoints readily allow us to find information about the triangles in the grid. For example, to build a table of triangles, $\mathcal{T}$, the following algorithm can be used:

```
for N_i ∈ N
    for N_j ∈ B(E, N_i)
        for N_k ∈ B(E, N_i)
            if (N_j, N_k) ∈ E
                T ← {N_i, N_j, N_k} ∪ T
```

ETNA
Kent State University
etna@mcs.kent.edu

5. Solution Techniques. We compare two different solution techniques: the inexact Newton-multigrid method and the Full Approximation Storage (FAS) multigrid. Note that Newton's method relies on a global linearisation sweep whereas the FAS scheme uses local linearisations.

5.1. Inexact Newton-Multigrid. Our implementation is a standard implementation of Newton's method, but we have included a brief description below to aid in the discussion of the numerical results.

Suppose we want to solve the nonlinear system $\mathbf{N}[\mathbf{x}] = \mathbf{b}$ where $\mathbf{N}$ is a nonlinear operator. Let $\mathbf{F}[\mathbf{x}] = \mathbf{b} - \mathbf{N}[\mathbf{x}]$ and take an initial guess $\mathbf{x}_0$. A high level algorithm for the inexact Newton-multigrid method is;

While $|\mathbf{F}[\mathbf{x}_k]|| >$ a given tolerance
  Calculate the Jacobian $\mathbf{F}'(\mathbf{x}_k)$
  Construct the coarse grid operator
  Solve the linear system $\mathbf{F}'(\mathbf{x}_k)\mathbf{y} = -\mathbf{F}[\mathbf{x}_k]$ by using the multigrid method
  Find the scaling factor γ
  Set $\mathbf{x}_{k+1} = \gamma\mathbf{y} + \mathbf{x}_k$

The method is inexact because the linear system is not solved exactly; we just require that

$$||\mathbf{F}[\mathbf{x}_k] + \mathbf{F}'(\mathbf{x}_k)\mathbf{y}|| < ||\mathbf{F}[\mathbf{x}_k]||/10.$$

The scaling factor is defined as

$$\gamma = \min\left\{1, 1/||\mathbf{c}||_2\right\}^{\frac{1}{4}}, \tag{5.1}$$

where $\mathbf{c}_i = \mathbf{y}_i/(\mathbf{x}_k)_i$. It is necessary to include a scaling factor when solving the time-dependent problems as the solution changes rapidly near wave fronts. This is similar to the scaling factor defined in [16].

When using the multigrid algorithm to solve a linear system $\mathbf{A}^n\mathbf{y} = \mathbf{f}^n$ given on a fine grid, the coarse grid operators are defined by the equation;

$$\mathbf{A}^{m-1} = \mathbf{I}_m^{m-1}\mathbf{A}^m\mathbf{I}_{m-1}^m, \tag{5.2}$$

where $\mathbf{A}^m$ $(1 < m \leq n)$ is the matrix defined on the grid $\mathcal{M}^m$, and $\mathbf{I}_m^{m-1}$ and $\mathbf{I}_{m-1}^m$ are the restriction and interpolation matrices introduced in Section 4.

A nested iteration scheme was used to obtain the initial guess $\mathbf{x}_0$.

5.2. FAS Scheme. A high level algorithm of the FAS scheme is given below. A more thorough description can be found in, for example, [1, 2, 8].

While $|\mathbf{F}[\mathbf{x}_k]|| >$ a given tolerance
  If not at coarsest grid
    Apply a nonlinear smoother μ_1 times to the system $\mathbf{N}^m[\mathbf{x}^m] = \mathbf{b}^m$
    Compute the defect $\mathbf{d}^m = \mathbf{b}^m - \mathbf{N}^m[\mathbf{x}^m]$
    Restrict the defect $\mathbf{d}^{m-1} = \mathbf{I}_m^{m-1}\mathbf{d}^m$
    Restrict the current approximation $\mathbf{x}^{m-1} = \widehat{\mathbf{I}}_m^{m-1}\mathbf{x}^m$
    Compute the approximation to
      $\mathbf{N}^{m-1}[\mathbf{y}^{m-1}] = \mathbf{d}^{m-1} + \mathbf{N}^{m-1}[\mathbf{x}^{m-1}]$
      by calling the FAS Scheme again using $\mathbf{x}^{m-1}$ as an initial guess

ETNA
Kent State University
etna@mcs.kent.edu

Calculate the correction $\widehat{\mathbf{x}}^{m-1} = \mathbf{y}^{m-1} - \mathbf{x}^{m-1}$
Interpolate the correction $\widehat{\mathbf{x}}^{m} = \mathbf{I}_{m-1}^{m}\widehat{\mathbf{x}}^{m-1}$
Correct the current approximation $\mathbf{x}^{m} = \mathbf{x}^{m} + \widehat{\mathbf{x}}^{m}$
Apply a nonlinear smoother μ_2 times to the system $\mathbf{N}^{m}[\mathbf{x}^{m}] = \mathbf{b}^{m}$
else
Solve the coarse grid problem $\mathbf{N}^{1}[\mathbf{x}^{1}] = \mathbf{b}^{1}$ by using Newton's method.

Notice that the Jacobian matrix is not formed, except on the coarsest grid, so the FAS scheme requires less storage than Newton's method.

The matrix $\widehat{\mathbf{I}}_{m}^{m-1}$ is a restriction operator, but not necessarily the same as $\mathbf{I}_{m}^{m-1}$. We used injection for this operation.

The nonlinear SOR method [13] was used as a smoother by the FAS scheme. The linearisation phase is incorporated in the smoother by applying a point-Newton method to a given grid node after 'fixing' the value at all of the other nodes. To apply the point Newton method, the diffusivity term (and its derivative) must be evaluated, which is expensive.

6. Grid Refinement. The refinement algorithm is based on the newest node bisection method. In this method, the triangles are subdivided by bisecting the edges that sit opposite the newest nodes. For example, if the dark points in Figure 6.1 are the newest nodes, then the resulting grid after one and two refinement sweeps are shown in Figure 6.2.

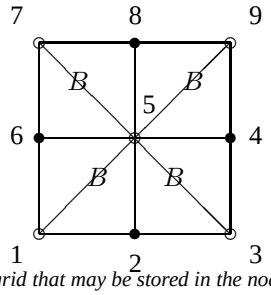

FIG. 6.1. *Example grid that may be stored in the node-edge data structure.*

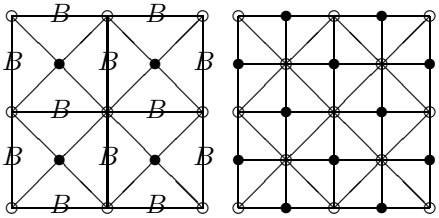

FIG. 6.2. *Resulting grid after two non-adaptive refinement sweeps of the grid in Figure 6.1. Note that the edges have been bisected along the base edges marked by a B.*

In terms of the node-edge data structure, it is easier to work with the base edges rather then the newest nodes, where the base edges are the edges that sit opposite the newest nodes, such as those marked by B in Figure 6.1.

6.1. Controlling the order of refinement. The most difficult part of the adaptive refinement routine is ensuring that the edges are bisected in the correct order to avoid long thin triangles. For example, suppose a triangle in Figure 6.1 is refined along one of the base edges as shown in Figure 6.3; then several of the triangles in the resulting grid will have two base edges. If the edges B_1 and B_2 are bisected during the next refinement sweep, then it is not

ETNA
Kent State University
etna@mcs.kent.edu

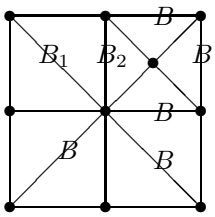

FIG. 6.3. *Result after bisecting the grid in Figure 6.1 along one of the base edges.*

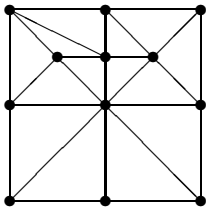

FIG. 6.4. *If the wrong edge (B_1) is bisected first, then the triangles can become long and thin.*

clear which base edge should be bisected first. If the wrong edge is chosen, as in Figure 6.4, we get long thin triangles that reduce the grids efficiency.

To determine the order in which to bisect the edges, we use a method similar to Mitchell's Compatibly Divisible Triangles [9, 10, 11].

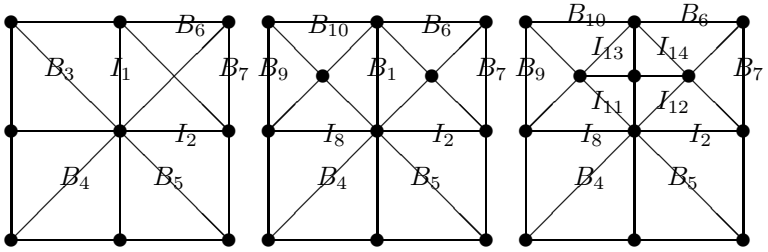

FIG. 6.5. *The interface-base edges marked by I sit between two different levels of refinement. Once an interface-base edge has been updated to a base edge, it may be bisected.*

We define an interface-base edge to be an edge that sits between two different levels of refinement. For example, in Figure 6.5, we have redrawn the grid from Figure 6.3 and marked the interface-base edges by an I. The neighboring coarse triangles must be refined before the interface-base edges are bisected. So B_3 in Figure 6.5 must be bisected before I_1. Note that it may be necessary to refine more then one neighboring triangle. For example, to bisect edge I_{11} in Figure 6.5, edges B_4 and I_8 should be bisected first.

6.2. Error Indicator (Stationary Problem). The idea behind our error indicator is to determine if the addition of a new node will significantly reduce the error.

Let $\mathbf{v}^m$ be the current approximation to the system of equations $\mathbf{N}^m[\mathbf{u}^m] = \mathbf{b}^m$, where $\mathbf{N}^m$ and $\mathbf{b}^m$ are the stiffness matrix and load vector defined on the current grid, $\mathcal{M}^m$. Then, each base edge and interface-base edge is assigned an error indicator that is equal to a weighted residual calculated at its midpoint. That is, if node i is the midpoint of a edge then the error indicator e^m assigned to that edge is:

ETNA
Kent State University
etna@mcs.kent.edu

$$e^m = \frac{\mathbf{r}_i^{m+1}}{\mathbf{N}_{i,i}^{m+1}[\mathbf{I}_m^{m+1}\mathbf{v}^m]}, \tag{6.1}$$

where

$$\mathbf{r}^{m+1} = \mathbf{b}^{m+1} - \mathbf{N}^{m+1}[\mathbf{I}_m^{m+1}\mathbf{v}^m].$$

$\mathbf{N}^{m+1}$ and $\mathbf{b}^{m+1}$ are the resulting stiffness matrix and load vector that would result if the edge was bisected at node i to form a new set of triangles. Note that it is not necessary to construct the whole stiffness matrix (or load vector); we only need the row corresponding to node i.

The motivation behind the error indicator is the question: how much will the error be reduced if we add node i? In regions of the domain where the solution is well approximated by the coarse grid we would expect the residual $\mathbf{r}^{m+1}$ to be small. In other regions where the solution is rapidly changing, and not well approximated by the coarse grid, the residual will be large. We divide by $\mathbf{N}_{i,i}^{m+1}[\mathbf{I}_m^{m+1}\mathbf{v}^m]$ to normalize the residual.

This error indicator is similar to the error indicator described by Mitchell [9, 10, 11], Rüde [15] or Villegas [21].

6.3. Moving Grids. When modeling non-stationary problems it is advantageous to adjust the shape of the grid to match the movement of the wave front. By noting that we store a sequence of nested grids, not just a single FEM grid, we are able to implement a very cheap de-refinement procedure. We simply shuffle the grids up one level, i.e. set $\bar{\mathcal{M}}^m$ to $\mathcal{M}^{m-1}$ where $\bar{\mathcal{M}}^m$ is the de-refined grid. The interpolation and restriction information at the coarsest and finest grids has to be updated, but otherwise this is a simple copy. Once all of the levels have been updated, we can apply the refinement algorithm described above.

6.4. Error Indicator (Non-Stationary Problem). The next issue is how to calculate the error indicator when taking the time derivative into account. To calculate the indicator at the next time-step, we need an approximation of the solution at that time-step. Applying an implicit algorithm to approximate the solution is cost-prohibitive, so we use an explicit method instead. We only use an explicit method to find the error indicator; once the grid has been refined, we use an implicit method to calculate the solution. Recall that the explicit Euler method is

$$\mathbf{M}^m\bar{\mathbf{v}}^m = \mathbf{M}^m\mathbf{v}^m + \Delta t(\mathbf{b}^m - \mathbf{N}^m[\mathbf{v}^m]),$$

where $\mathbf{M}^m$ is the mass matrix, $\mathbf{v}^m$ is the solution at the current time-step, $\bar{\mathbf{v}}^m$ is the solution at the next time-step and Δt is the step size. Based on this equation we then define the error indicator for non-stationary problems to be:

$$\begin{aligned} e^m &= \frac{\mathbf{r}_i^{m+1}}{\left(\mathbf{M}_{i,i}^{m+1}\right)^{1/4}}, \\ \mathbf{r}^{m+1} &= \mathbf{g}^{m+1} - \mathbf{M}^{m+1}\mathbf{I}_m^{m+1}\bar{\mathbf{v}}^m, \\ \mathbf{g}^{m+1} &= \mathbf{I}_m^{m+1}\left(\mathbf{M}^m\mathbf{v}^m + \Delta t(\mathbf{b}^m - \mathbf{N}^m[\mathbf{v}^m])\right) \end{aligned} \tag{6.2}$$

Note that this indicator only needs to evaluate $\mathbf{N}^m$ once on the original de-refined grid, to form the right-hand-side, and is thus a lot cheaper than the indicator used for the stationary problem.

ETNA
Kent State University
etna@mcs.kent.edu

Figures 6.6, 6.7 and 6.8 show examples of the moving grid, if $\alpha = -3$, $\beta = 3/4$. The values of Z are as given in Figure 2.1 and $D(E) = D_1(E)$. The step size is 0.5/1024, and the figures show the results at time step 110, 220 and 440 respectively. The indicator should pick up the regions where the solution is changing rapidly, which it does.

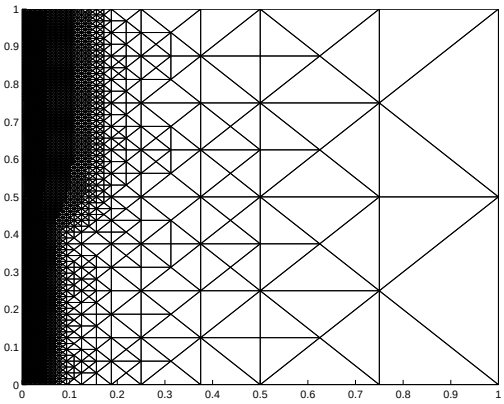

FIG. 6.6. *Example grid at time step* $110\Delta t$ *(*$\Delta t = 0.5/1024$*), which contains 5457 nodes. This grid was used to obtain the results given in Figure 7.6.*

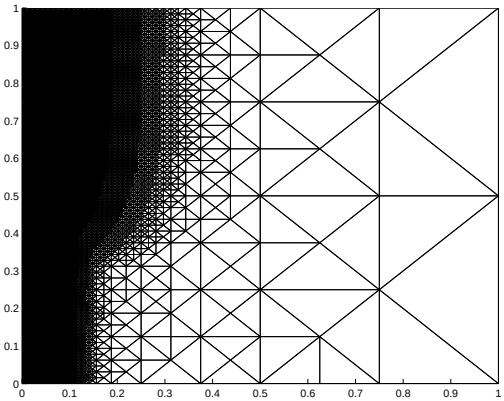

FIG. 6.7. *Example grid at time* $220\Delta t$ *(*$\Delta t = 0.5/1024$*), which contains 12713 nodes. This grid was used to obtain the results given in Figure 7.7.*

7. Results. All of the test examples were run on 1.7 GHz Pentium 4 processors with 1024 MB of 400 MHz RAM. Further particulars of the machine can be found at `http://www.icase.edu/`.

7.1. Solution of Stationary Problem. To better understand the behavior of the two nonlinear multigrid methods we firstly considered the stationary problem.

7.1.1. Test Problems. We looked at four different test problems, `low`, `low J`, `high` and `high J`.

In the first two examples, `low` and `low J`, $\alpha = -1$ and $\beta = 1/4$. The value of Z is fixed at 2 throughout the whole domain for `low`, but in `low J` it varies as shown in Figure 2.1. Figure 7.1 shows examples solutions of `low` and `low J`. For the other two examples, `high` and `high J`, the nonlinearity and jump size were set at $\alpha = -3$ and $\beta = 3/4$. Again,

ETNA
Kent State University
etna@mcs.kent.edu

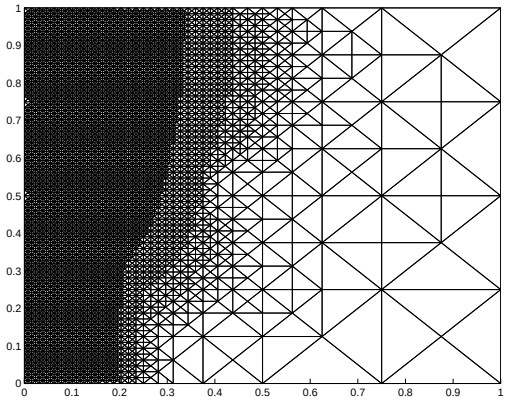

FIG. 6.8. *Example grid at time* $440\Delta t$ *(*$\Delta t = 0.5/1024$*), which contains 5583 nodes. This grid was used to obtain the results given in Figure 7.8.*

Z is fixed at 2 for `high`, but is spatially dependent for `high J`. Figure 7.2 shows example solutions of `high` and `high J`.

The flux limiter was included in the test problems so the definition of the diffusivity term is as shown in Equation (2.3).

The initial value for the energy was $E = 10^{-5}$.

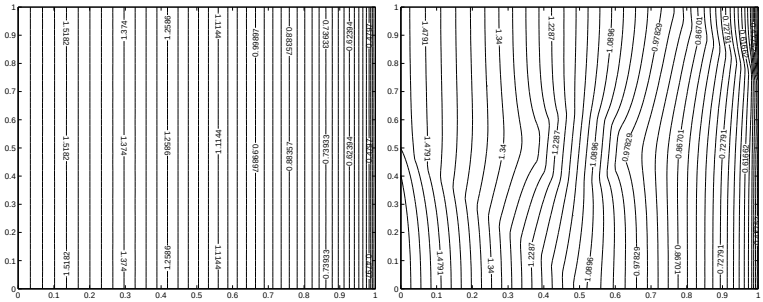

FIG. 7.1. *Example solutions of the test problems* `low` *and* `low J` *respectively.*

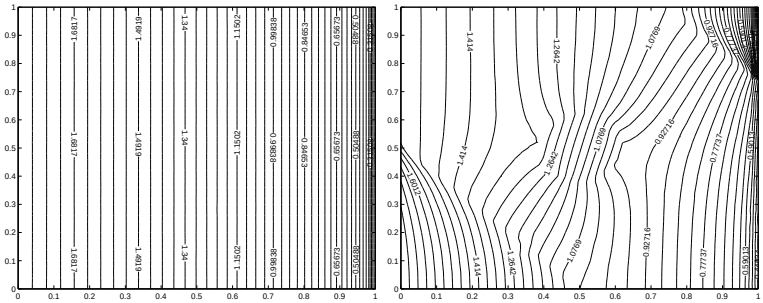

FIG. 7.2. *Example solutions of the test problems* `high` *and* `high J` *respectively.*

7.1.2. Newton's method. The first sets of results we present are those for Newton's method, which are given in Tables 7.1 and 7.2.

The timing results have been broken down into the total time (Total Time), time required to solve the nonlinear system using Newton's method (Newton), the time spent solving the

ETNA
Kent State University
etna@mcs.kent.edu

TABLE 7.1
Solution time, in seconds, when solving the test problems using Newton's method on an uniformly refined grid.

	low	low J	high	high J
No. Nodes	66049	66049	66049	66049
No. It.	7	7	7	7
Total Time	160.7	169.3	148.1	175.0
Newton	152.4	161.0	140.0	166.8
V-cycle	55.6	62.8	45.6	66.5
Grid Refine	3.7	3.8	3.7	3.7
FEM	4.5	4.5	4.4	4.4

TABLE 7.2
Solution time, in seconds, when solving the test problems using Newton's method on an adaptively refined grid.

	low	low J	high	high J
No. Nodes	51583	35664	64490	14911
No. It.	8	8	9	9
Total Time	376.9	240.7	334.7	75.1
Newton	353.6	223.7	304.1	68.3
V-cycle	243.7	145.7	161.7	35.4
Grid Refine	19.1	14.0	25.3	5.6
FEM	4.2	3.0	5.3	1.2

linearized system with the aid of the V-cycle (V-cycle), the time needed to build the nested sequence of grids (Grid Refine) and the time to build the load vector (FEM). Strictly speaking we do not have to evaluate the load vector, we could just automatically set the values to zero. But this will not be the case when we look at systems of equations, so for functionality reasons we still evaluate the load vector. The time given for Newton's method includes the time to solve the linearized system of equations ('V-cycle'). So, Total Time = Newton + Grid Refine + FEM.

The scaling factor given in Equation 5.1, was also included in the algorithm. This was necessary otherwise the `high J` test example diverged when solving the problem on coarser grids. The factor generally evaluated to 1 on the finer grids.

Six pre-smoothers and six post-smoothers were used in the V-cycle solver.

As part of Newton's method, the solution of the linearized system is added to the current approximation to get the new approximation (see the Newton-multigrid algorithm given in Section 5.1). After applying this update some of the energy values may be negative. This is not physically meaningful and it causes a numerical error when the negative energy values are passed into the diffusivity functions, $D_1(E)$ or $D_2(E)$. Physically, the energy should not go below 10^{-5} (the initial energy value), so after adding the solution of the linear system we sweep through the grid and change any negative values to 10^{-5}. As a consequence, if the grid is not fine enough Newton's method may stall.

Recall that the initial guess on the fine grid was obtained by the use of nested iteration. Such an approach is especially appropriate in the case of adaptive refinement where we need to approximate the solution on each grid before we can calculate the error indicator. However, as explained above, Newton's method stalls if the grid is too coarse. Therefore, when using the coarse grid to find an initial guess we limit the maximum number of iterations to 10. We chose 10 because the coarse grid solves either stalled or converged within 10 iterations.

ETNA
Kent State University
etna@mcs.kent.edu

TABLE 7.3
Solution time, in seconds, when solving the test problems using the FAS scheme on an uniformly refined grid.

	low	low J	high	high J
No. Nodes	66049	66049	66049	66049
No. It.	10	14	6	15
Total Time	404.7	555.0	272.0	600
FAS	396.7	546.8	263.9	591.7
Nonlin. SOR	308.5	422.9	203.8	460.4
Grid Refine	3.7	3.8	3.8	3.9
FEM	4.2	4.4	4.2	4.4
Newton	1.1	1.4	0.7	2.0

TABLE 7.4
Solution time, in seconds, when solving the test problems using FAS scheme on an adaptively refined grid

	low	low J	high	high J
No. Nodes	51583	35664	64490	14911
No. It.	22	21	10	11
Total Time	1683.9	1020.6	946.4	247.7
FAS	1662.0	1004.0	916.5	240.8
Nonlin. SOR	1485.7	896.0	818.0	216.4
Grid Refine	17.7	13.6	24.7	5.7
FEM	4.2	3.0	5.3	5.7
Newton	1.2	1.0	0.7	0.8

The times given in the tables is the overall time for the nested iteration, not just the time required to solve the fine grid problem.

The number of iterations, labeled as 'No. It.', is the number of iterations required to solve the problem on the finest grid level. The iterations were terminated when the residual $\mathbf{F}[\mathbf{x}_k]$ was less than 10^{-10}. Setting the tolerance to 10^{-10} meant that both the FAS and Newton's method required several iterations on the fine grid to converge. This challenged the two solvers and better allowed us to compare the performance.

Six levels of uniformly refined grids were used to obtain the results presented in Table 7.1. The initial coarse grid contained 25 nodes. The number of nodes on the finest grid was 66049.

Five levels of adaptively refined grids form the basis of the results given in Table 7.2. The initial coarse grid contained 25 nodes. The number of nodes on the finest grid level ranges from 64490 to 14911, which is an artifact of the refinement routine. If the error indicator is reduced by a factor of four the refinement algorithm exits, which does not necessarily imply that the number of nodes in one grid level to the next increased by a factor of four.

When using adaptive refinement the number of nodes in certain regions of the domain may vary greatly from one grid level to the next. Consequently the convergence rate for the linear solver decreased when using adaptive grids. This is why the percentage of time spent in the V-cycle is high for the adaptively refined grids. The convergence rate of the V-scheme when solving the `low J` test problem, for example, was better than 0.1 for uniform grids but decreased to 0.6 for the adaptively refined grids.

7.1.3. FAS. Tables 7.3 and 7.4 give the timing results when the FAS scheme was used. Much of the discussion given in Section 7.1.2 on how the timing results were obtained also

ETNA
Kent State University
etna@mcs.kent.edu

applies here.

The results labeled 'Nonlin. SOR' show the amount of time spent in the nonlinear SOR routine. Two pre-smoothers and two post-smoothers were used for the uniform grids and six pre-smoothers and six-post smoothers were used for the adaptive grids. It was necessary to apply more pre and post smoothers for the adaptive grids to ensure that the method converged.

The convergence rate for the FAS method when solving `low J` was 0.4 with uniform grids and 0.5 with adaptive grids. The rate for the uniform grids is less than that reported for Newton's method, but we only used two pre and two post smoother for the FAS scheme where as we used six pre and six post smoothers for Newton's method.

To choose the number of pre and post smoothers we took the `low J` test problem and tried different numbers of pre and post smoothers and choose the combination that gave the smallest run time for each method (Newton or FAS).

We used Newton's method as the coarse grid solver in the FAS routine. The results labeled 'Newton' show how long it took to solve the problem on the coarse grid, which contained 25 nodes.

The time spent in the nonlinear SOR routine and Newton's method contribute to the time reported for the FAS scheme. Hence Total Time = FAS + Grid Refine + FEM.

The nonlinear SOR routine, as with Newton's method, may produce negative values for the energy. If this happened, the negative values were changed to 10^{-5}. If the grid is too coarse the FAS scheme stalled, as with Newton's method. Hence the maximum number of times the FAS scheme was called to solve the problem on a given grid was 30.

The times for the FAS scheme are slower than Newton's method. We have also observed this for other test problems. This does not mean that the FAS scheme performed badly in terms of the number of updates per node. As a specific example, consider the test problem `low J` on a uniformly refined grid. Fourteen iterations of the FAS-scheme were required to solve this problem. When Newton's method was used a total of fourteen V-cycles were also called to solve the linearized system. Taking into account that we had two pre and post smoothers for the FAS scheme, but used six pre and post smoothers for the V-cycle, we see that in terms of number of node updates the FAS scheme was cheaper for this particular example.

The main reason why FAS is slower is the high cost of calculating the effect of the diffusivity term, and it's derivative. Every time the value of a node is changed $a(D(u); v, w)$ has to be reevaluated. This must be done by looping over all of the triangles that have the node as one of their vertices, forming the appropriate basis functions and calculating an integral. Our data structure is not triangle based, so there is some extra overhead in finding the triangles around a given node, although we use the neighbor endpoints and have been very careful in the way we implement the data structure so the overhead is very small. Irrespective of what data structure is implemented, updating the nodal values when using a linear smoother is much cheaper than updating the nodal values when using a non-linear smoother. In the first case we simply update the nodal values by finding some weighted average of the neighboring nodes. In the second case we have, at least, the additional cost of evaluating an integral by the use of some quadrature rule. Such characteristics have also been observed when using other codes, see [7].

Note that the radiation transport model used here is fairly simple, so the diffusivity term would become more complex if some of the underlying assumptions were removed. In which case we would expect the cost of evaluating the diffusivity term to become an even bigger bottleneck.

7.2. Non-Stationary. The Crank-Nicolson method was used in all of the test runs.

ETNA
Kent State University
etna@mcs.kent.edu

7.2.1. Jump Size and Non-Stationary Solutions. We now look at how the jump size affects the movement of the energy wave. We used the inexact Newton-multigrid method to solve the problem.

In the first example $\alpha = -1$ and $\beta = 3/4$. The values of Z are as given in Figure 2.1. For simplicity the flux limiter was not used. The step sizes have to be greatly reduced if the limiter is included and we leave this calculation to our report on the parallel performance. We took time steps of size 0.5/1024 and the results in Figures 7.3, 7.4 and 7.5 show the energy at time steps 110, 220 and 440 respectively. The nonlinear iterations were terminated when the residual was less than 10^{-6}. We chose this stopping criteria as it is small enough to get physically meaningful results.

We have observed that for non-stationary problems it is important that the grid is fine enough otherwise Newton's method (and the FAS scheme) tends to stall. Hence the use of adaptive refinement is beneficial, especially in the first few time steps where the energy values near the left boundary increase rapidly.

Four levels of adaptive refinement was used to build the grid levels. The coarsest grid in this case contained 9 nodes.

Recall that the atomic mass number in the lower left corner of the domain is higher than that in the upper left corner. We can clearly see how this influences the movement of the wave front.

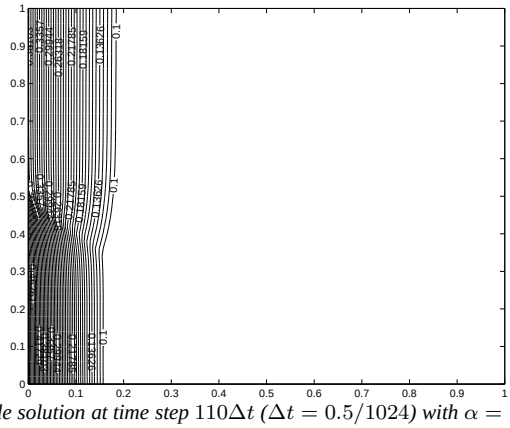

FIG. 7.3. *Example solution at time step* $110\Delta t$ *(*$\Delta t = 0.5/1024$*) with* $\alpha = -1$ *and* $\beta = 1/4$.

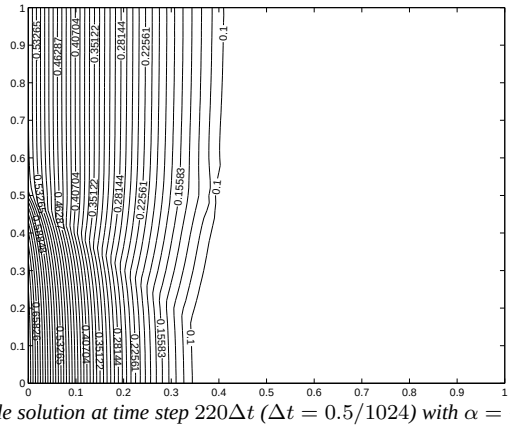

FIG. 7.4. *Example solution at time step* $220\Delta t$ *(*$\Delta t = 0.5/1024$*) with* $\alpha = -1$ *and* $\beta = 1/4$.

ETNA
Kent State University
etna@mcs.kent.edu

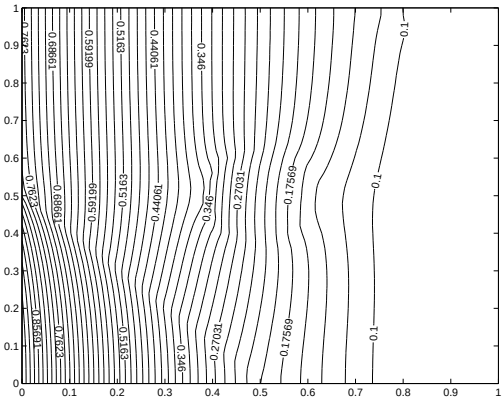

FIG. 7.5. *Example solution at time step* $440\Delta t$ *(*$\Delta t = 0.5/1024$*) with* $\alpha = -1$ *and* $\beta = 1/4$*.*

In the next set of examples we set $\alpha = -3$ and $\beta = 3/4$. All of the other parameters are the same as those given above. The results in Figures 7.6, 7.7 and 7.8 show the energy at time steps 110, 220 and 440 respectively.

These graphs show how an increase in the mass number slows the movement of the wave front.

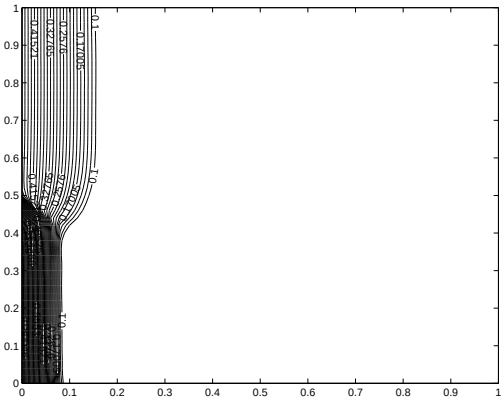

FIG. 7.6. *Example solution at time step* $110\Delta t$ *(*$\Delta t = 0.5/1024$*) with* $\alpha = -3$ *and* $\beta = 3/4$*.*

8. Conclusion. The experiments that we carried out imply that the inexact Newton-multigrid method is faster than the FAS scheme when solving radiation transport equations. The reason is that calculating the diffusivity term;

$$a(D(u); v, w)_\Omega = \int_\Omega D(u) \nabla v \nabla w \, d\Omega + \int_{\Gamma_{F_0}} \frac{vw}{2} \, d\Gamma + \int_{\Gamma_{F_1}} \frac{vw}{2} \, d\Gamma,$$

is relatively expensive. In Newton's method we only have to evaluate the diffusivity term once for each node on the finest grid level to form the Jacobian, but with the FAS scheme it must be evaluated several times for each node on each grid level.

The cost of evaluating the diffusivity term may not be so influential when using, say, the finite difference method. However we believe that even in this case it will still become an issue if the diffusivity term was made more complex to include more physics (remove some of the current simplifications).

ETNA
Kent State University
etna@mcs.kent.edu

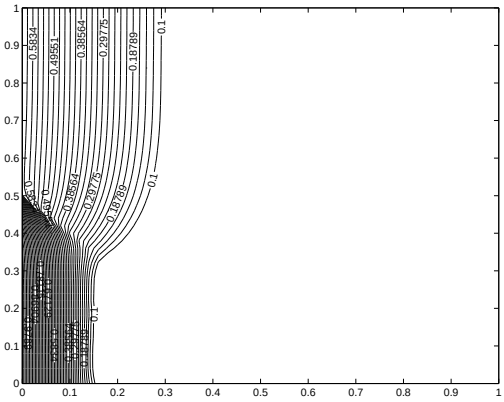

FIG. 7.7. *Example solution at time step* $220\Delta t$ *(*$\Delta t = 0.5/1024$*) with* $\alpha = -3$ *and* $\beta = 3/4$.

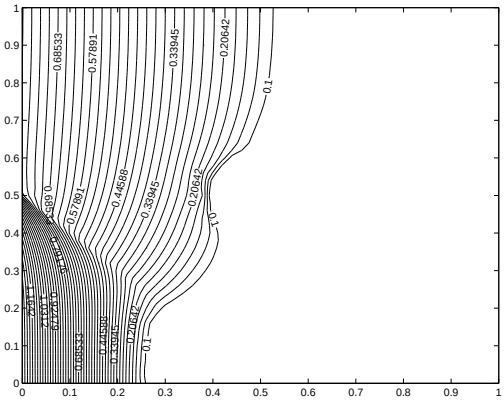

FIG. 7.8. *Example solution at time step* $440\Delta t$ *(*$\Delta t = 0.5/1024$*) with* $\alpha = -3$ *and* $\beta = 3/4$.

The finite element method is also better suited to the use of adaptive refinement. We also showed that adaptive refinement helps to capture the information along the wave front.

Note that we only presented serial results in this report and our initial study into the parallel performance suggests that the FAS scheme scales better.

9. Future Work. Both of the solvers have difficulty converging during the first few time steps, so we plan to investigate some adaptive time-stepping techniques.

We also plan to study the parallel scalability of the solvers.

REFERENCES

[1] A. BRANDT, Multi-level adaptive solutions to boundary-value problems. *Math Comput.*, 31(138):333–390, April 1977.

[2] W. L. BRIGGS, V. E. HENSON, AND S. F. MCCORMICK, *A Multigrid Tutorial*. SIAM, 2000.

[3] P N. BROWN, B. CHANG, F. GRAZIANI AND C. S. WOODWARD, *Implicit solution of large-scale radiation-material energy transfer problems*, Technical Report UCRL-JC-132831, Lawrence Livermore National Laboratory, Jan. 1999. To appear in Proceedings of the Fourth IMACS International Symposium on Iterative Methods in Scientific Computations.

[4] P N. BROWN AND C. S. WOODWARD, *Preconditioning Strategies for Fully Implicit Radiation Diffusion with Material-Energy Transfer*, Technical Report UCRL-JC-139087, Lawrence Livermore National Laboratory, May. 2000. To appear in SIAM J. Sci. Comput.

ETNA
Kent State University
etna@mcs.kent.edu

[5] D. A. KNOLL, W. J. RIDER AND G. L. OLSON, *An efficient nonlinear solution method for non-equilibrium radiation diffusion*, Technical Report LA- UR-98-2154, Los Alamos National Laboratory, 1998. Submitted to J. Quant. Spec. and Rad. Trans.

[6] M. KŘÍŽEK AND L. LIU, *Finite element approximation of a nonlinear heat conduction problem in anisotropic media*, Technical Report 4/1997, Laboratory of Scientific Computing, University of Jyväskylä, Finland, 1997.

[7] D. J. MAVRIPLIS,*Multigrid approaches to non-linear diffusion problems on unstructured meshes*, Technical Report ICASE Report No. 2001-3, (NASA/CR-2001-210660), February 12, 2001, 16 pages. Submitted to the Journal of Numerical Linear Algebra with Applications.

[8] S. F. MCCORMICK, *Multigrid Methods*. SIAM Frontiers In Applied Mathematics, 1987.

[9] W. F. MITCHELL, *Unified Multilevel Adaptive Finite Element Methods For Elliptic Problems*. PhD thesis, Department Of Computer Science, University Of Illinois at Urbana-Champaign, Urbana, IL, 1988. Technical Report UIUCDCS-R-88-1436.

[10] W. F. MITCHELL, A comparison of adaptive refinement techniques for elliptic problems. *ACM Trans. Math. Software*, 15(4):326–347, December 1989.

[11] W. F. MITCHELL, Optimal multilevel iterative methods for adaptive grids. *SIAM J. Sci. Stat. Comput*, 13(1):146–167, January 1992.

[12] V. A. MOUSSEAU, D. A. KNOLL AND W. J. RIDER, Physics-based preconditioning and the Newton-Krylov method for non-equilibrium radiation diffusion. *Journal of Computational Physics*, 160:743-765, 2000.

[13] J .M. ORTEGA AND W .C. RHEINBOLDT, *Iterative solution of nonlinear equations in several variables*, Academic Press, New York and London, 1970.

[14] G. C. POMRANING, *The equations of radiation hydrodynamics*, Pergamon, New York, 1973.

[15] U. RÜDE, *Mathematical and computational techniques for multilevel adaptive methods*, SIAM, Philadelphia, 1993.

[16] W. J. RIDER, D. A. KNOLL AND G. L. OLSON, *A multigrid Newton- Krylov method for multimaterial equilibrium radiation diffusion*, Technical Report LA-UR-98-2153, Los Alamos National Laboratory, 1998, 34 pages.

[17] L. STALS, *Adaptive multigrid in parallel*, in Proceedings of Seventh SIAM Conference on Parallel Processing for Scientific Computing, D. Bailey, P. Bjørstad, J. Gilbert, M. Mascagni, R. Schreiber, H. Simon, V. Torczon and L. Watson, eds, SIAM, Philadelphia, 1995, pp. 367-372.

[18] L. STALS, *Parallel multigrid on unstructured grids using adaptive finite element methods*, PhD thesis, Department of Mathematics, Australian National University,Canberra, Australia, 1995.

[19] L. STALS, *Implementation of multigrid on parallel machines using adaptive finite element methods*, in Proceedings of 9th International Conference on Domain Decomposition, P. Bjørstad, M. Espedal and D. Keyes, eds, 1998, pp. 488-496.

[20] L. STALS, *A flexible data structure for the adaptive refinement of unstructured grids in parallel*, , in The Proceedings of The 14th Kiel GAMM Seminar on Concepts of Numerical Software, January, 1997. To appear.

[21] J. C. A. VILLEGAS, *Anisotropic Adaptive Refinement Algorithms for Finite Element Methods*. PhD thesis, Graduate School of Arts and Science, Department of Mathematics, New York University, September 2000.