Journal of Computational and Applied Mathematics 36 (1991) 65–78
North-Holland

Adaptive refinement for arbitrary finite-element spaces with hierarchical bases

William F. Mitchell *

General Electric Advanced Technology Laboratories, Bldg. 145-2, Moorestown Corporate Center, Moorestown, NJ 08057, United States

Received 1 October 1990

Abstract

Mitchell, W.F., Adaptive refinement for arbitrary finite-element spaces with hierarchical bases, Journal of Computational and Applied Mathematics 36 (1991) 65–78.

An adaptive refinement algorithm is presented and interpreted as the selective enrichment of a finite-element space through the hierarchical basis. Each elemental division corresponds exactly to the inclusion of a small number of new basis functions, while existing basis functions remain unchanged. Which bases to add, i.e., which divisions to perform, are determined so that those that make the largest contribution to the function are included first. The space of C^0pth-degree piecewise polynomials over triangles are used for illustration, but the techniques apply to any family of function spaces that can be represented with a hierarchical basis. Numerical examples are presented to show that the technique can regain the optimal (smooth) order of convergence for the solution of partial differential equations with nonsmooth solutions.

Keywords: Finite elements, adaptive refinement, hierarchical basis.

1. Introduction

The use of adaptive refinement to obtain a grid for the discretization of a partial differential equation has been the subject of much research in the past decade [2,4,7,9,10,12,16,17,21]. The idea is to automatically construct a grid which is coarse where the solution is well behaved, fine near singularities, boundary layers, etc., and has a smooth transition between the coarse and fine parts. Such a grid can dramatically reduce the number of nodes needed to obtain an accurate solution for marginally smooth problems, and can recover the optimal order of convergence for nonsmooth problems.

In this paper we present an adaptive refinement algorithm that can be interpreted as the selective enrichment of a finite-element space through the hierarchical basis. The presentation focuses on the generation of a grid for the solution of a partial differential equation, but the

* Work performed while at the Department of Computer Sciences, University of Illinois at Urbana-Champaign, and supported in part by the U.S. Department of Energy under grant DOE DEFG02-87ER25026.

same techniques can be used for other purposes, such as generation of a grid for interpolation with nearly uniform error. From the interpretation in terms of the hierarchical basis, the method is applicable to any finite-element space. In this paper, we focus on the family of C^0pth-degree polynomials over triangles. In [9] extensions to three dimensions (tetrahedron) and C^1 Hermite bicubics over rectangles are considered.

Central to any adaptive refinement algorithm for a finite-element grid is a method for dividing (refining) the elements, triangles in our case. There are two major methods for dividing triangles in adaptive refinement algorithms. Regular division divides a triangle into four similar triangles by connecting the midpoints of the sides. Bank and Sherman [4,5] show how to use regular division in an adaptive refinement algorithm. Bisection division connects one of the vertices of the triangle to the midpoint of the opposite side. Two approaches are in use regarding the selection of the vertex to be divided. Rivara [15,16] chooses the vertex opposite the longest edge. Sewell [17,18] chooses the "newest" vertex. This is the method used here. A new idea is to divide the triangles two at a time, which relates directly to the addition of a basis function to the approximating space.

Another critical element of any adaptive refinement algorithm is the error indicator, which is used to determine which triangles should be divided. Several good error indicators have been proposed [2,4,6,21]. Most of these are based on estimating the discretization error over each triangle. The method used in this paper can be derived as a local error estimate, but it is also an attempt to determine which basis functions that can be added to the approximating space would reduce the discretization error the most. This becomes a type of error indicator for *pairs* of triangles based on the size of the coefficient of the hierarchical basis function over those triangles. Mitchell [10] surveyed several error indicators and triangle division methods and compared them in a numerical experiment. He found that, among the methods considered, there is no universally "best" adaptive refinement method, and that most of the methods performed approximately the same. The methods we use performed well in that experiment.

2. Hierarchical bases

The use of hierarchical bases for finite elements has been considered in [3,9,11,19–21]. The usual nodal basis for a space of piecewise polynomials is defined on a given grid by each basis function being a member of the space with the value 1 at one node and 0 at all other nodes. In contrast, the hierarchical basis is defined using the family of nested grids from the refinement process. The hierarchical basis begins with the nodal basis on the initial (coarsest) grid. As refinement proceeds, with each division one or more new nodes are added, and for each node we add a new basis function defined so that it has the value 1 at the new node and 0 at all other nodes, *but the existing basis functions remain unchanged*. Figure 1 illustrates the nodal and

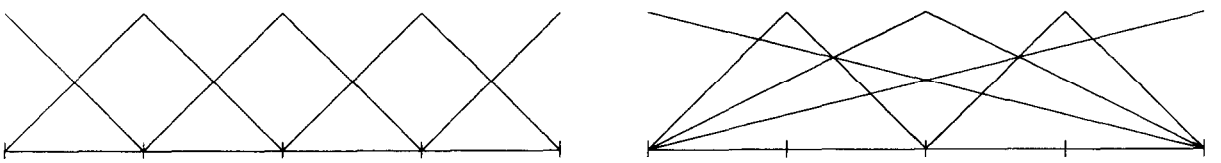

Fig. 1. Nodal and hierarchical bases for piecewise linear functions in one dimension.

hierarchical basis for the simple case of piecewise linear elements in one dimension with a 3-level grid.

When a function is expanded in the nodal basis, the coefficients are the values of the function at the nodes, since at each node there is only one nonzero basis function. But with the hierarchical basis, a node added at the kth level of refinement has nonzero basis functions over it from each of the first k levels. Since the lower level basis functions have not changed, the coefficient of the hierarchical basis is the difference between the function value at the central node and the approximate value from a $k-1$ level grid. This is the key property of hierarchical bases used by the adaptive refinement algorithm. The interested reader is referred to the above references for further properties of hierarchical bases, their use in solving partial differential equations, and the relationship between the hierarchical and nodal bases.

3. Adaptive refinement using newest vertex bisection

The process of adaptive refinement is one of selectively dividing triangles such that

(i) the angles are bounded away from 0 and π to avoid growth in the interpolation error and the condition number of the stiffness matrix [1,8];

(ii) the triangulation is compatible (defined below);

(iii) the grid is fine in the right places to obtain the optimal order of convergence, even in the face of singularities;

(iv) the process requires only O(number of triangles) operations, especially if an optimal order multigrid method will be combined with adaptive refinement as in [9,11].

In this section we present an adaptive refinement algorithm that satisfies these properties, and relate this algorithm to the hierarchical basis. The high level adaptive refinement algorithm is:

```
repeat
   determine which triangle(s) to divide
   divide the triangles and maintain compatibility
until enough refinement has occurred
```

The basic building block of an adaptive refinement algorithm is a method for dividing a triangle. The method we use, which we call *newest vertex bisection*, is nearly identical to a method presented by Sewell [17]. Much of the following terminology is due to Sewell.

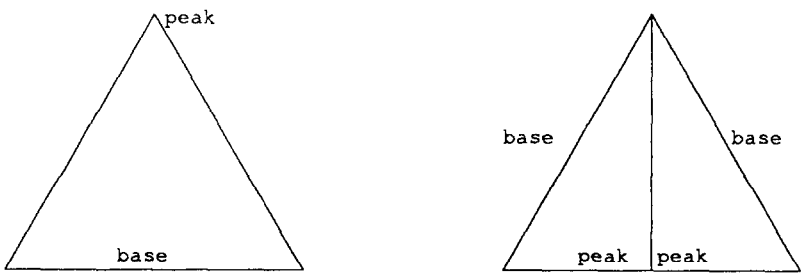

Fig. 2. Propagation of the peak with newest vertex bisection.

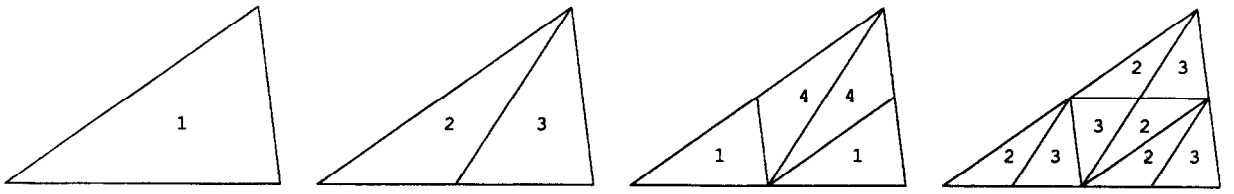

Fig. 3. Four similarity classes of triangles generated by newest vertex bisection.

In bisection division a triangle is divided to form two new triangles by connecting one of the vertices, called the *peak*, to the midpoint of the opposite side called the *base*, as in Fig. 2. The original triangle is called the parent, and the two new triangles are called the *children*. The children are said to have *generation* $i+1$ where i is the generation of the parent. The initial triangle is assigned generation 1. The new vertex created at the midpoint of the base is assigned to be the peak of the children, hence the name newest vertex bisection.

It is easy to show that the angles be bounded away from 0 and π. Sewell [17] showed that there are only four similarity classes of triangles created by this method, as in Fig. 3, which shows the angles are bounded. Further, Mitchell [9] showed that only eight angles arise, as illustrated in Fig. 4. If A, B and C are the angles of the initial triangle and C is the angle at the peak, the other angles are given by

$$E_1 = \tan^{-1}\frac{2\sin A \sin B}{\sin(A-B)}, \qquad E_2 = \pi - E_1, \qquad D_1 = E_2 - A, \qquad D_2 = E_1 - B,$$

$$F = \pi - C.$$

One of the complications in adaptive refinement is that of maintaining compatibility of the triangulation. A triangulation is said to be *compatible* if for any two triangles t_i and t_j, $t_i \cap t_j$ is either empty, a common vertex, or a common side. Other authors, most notably Rivara [16] and Bank et al. (e.g., [5]) have taken the approach of dividing some set of triangles with large error indicators, producing an incompatible triangulation, and then performing a second process to regain compatibility by dividing more triangles. In our approach we never have an incompatible triangulation. Compatibility is maintained *during* the refinement process, rather than after, by dividing *pairs* of triangles rather than individual triangles, eliminating the need for a separate follow-up process to recover compatibility.

A triangle is said to be *compatibly divisible* if its base is either the base of the triangle that shares that side or part of the boundary of the domain. If a triangle is compatibly divisible, then we divide the triangle and the neighbor opposite the peak (if such a neighbor exists) simulta-

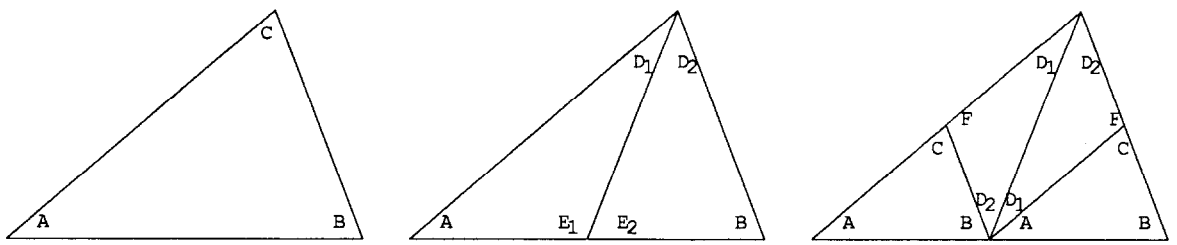

Fig. 4. Angles that arise during bisection.

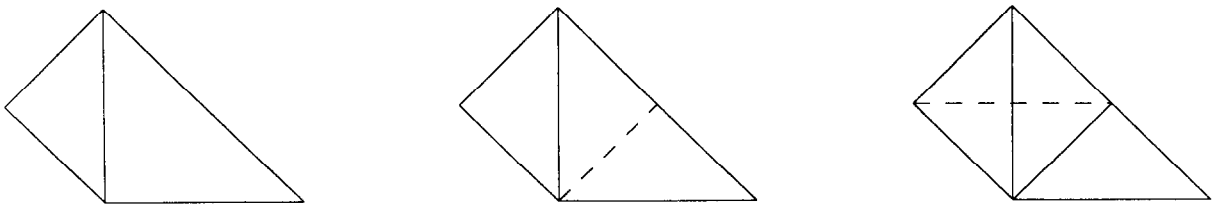

Fig. 5. Maintaining compatibility during refinement.

neously as a pair. If a triangle is not compatibly divisible, then after a single bisection of the neighbor opposite the peak, it will be. So in this case, we first divide the neighbor by the same process, and then divide the triangle and neighbor opposite the peak simultaneously, as illustrated in Fig. 5. We note that this process always divides a pair of compatibly divisible triangles, or a triangle whose base is part of the boundary, and that the triangulation is never incompatible. This leads to the following recursive algorithm, which is easily implemented in FORTRAN by constructing a stack of the triangles that need to be divided.

```
algorithm divide_triangle(t)
if t is not compatibly divisible then
   divide_triangle(neighbor of t opposite peak)
endif
divide the triangle pair t and the neighbor opposite the peak of t
return
```

It is important that this recursion be finite and not very large. Mitchell [9] showed that each recursive call divides a triangle of one earlier generation, and consequently the length of the recursion is bounded by the generation of the triangle. He also proved that given any initial triangulation, there exists a choice of peaks such that every triangle is initially compatibly divisible.

The division of pairs of triangles relates directly to the inclusion of more basis functions in the hierarchical basis. In the case of piecewise linear functions, the division of a pair of triangles adds one new basis function. This function is centered at the new node created by the division, and has support equal to the triangles divided. With the hierarchical basis, the only effect of the division is to add this new basis function, since all old basis functions remain unchanged. Alternatively, one could say that the addition of the new hierarchical basis divides the pair of triangles. From this vantage, each basis function requires that certain other basis functions be in the space before it can be included. Addition of these basis functions corresponds to the recursive chain of triangles divided for compatibility. For higher order $C^0 p$th-degree polynomials, the division of a pair of triangles corresponds to the inclusion of p^2 new basis functions, one for each new node created by the division. This concept generalizes to arbitrary finite-element spaces. A single "refinement" step corresponds to the inclusion of a small set of new hierarchical basis functions. For any given space, there are restrictions on which bases can be added (such as the compatibility requirement), and which bases must be added as a group (such as the p^2 bases for higher-order polynomials).

To guide the adaptive refinement so that the grid is fine in the right places, it is necessary to have some sort of error indicator which determines which triangles should be refined. Many error

indicators have been proposed [2,4,6,21]. Mitchell [10] performed a numerical experiment to compare the effectiveness of several indicators. The method we describe here performed well in those experiments. This method is similar to that proposed by Zienkiewicz et al. [21] for bilinear rectangular elements. Our interpretation of the indicator makes it possible to define an error indicator for any finite-element space. We will concentrate on the spaces of C^0pth-degree polynomials over bisected triangles, first considering linear elements, and then showing how to extend to arbitrary degree.

At this point we must make the distinction between an error indicator and an error estimate. By an *error indicator* we mean a nonnegative real number assigned to each triangle, or small groups of triangles, which has its largest values in the triangles whose refinement would be most beneficial for reducing the discretization error. An *error estimate*, on the other hand, can be defined either locally or globally and should be a good approximation of the discretization error in some norm. An error indicator is used to guide adaptive refinement; an error estimate can be used as a termination criterion for a program, or just to give the user some idea of how accurate the solution is. Usually, if an error estimate is defined locally, it can be used as an error indicator, but it is not clear that error estimates make the best error indicators.

Our error indicator is not an error estimate (although an error estimate based on this error indicator is presented in [9]). Instead of attempting to divide the triangles over which the error is the largest (which is what an error estimate based error indicator does), we attempt to divide the triangles whose divisions make the greatest change in the solution. Since this change in the solution reduces the error, we are attempting to divide the triangles which make the greatest reduction in the error, in other words, reduce the error the fastest for the number of divisions performed.

To approximate how much of a change in the solution will occur, we use the hierarchical basis. Recall that when a function is expanded using the hierarchical basis, the coefficients represent displacements rather than nodal values as with the usual nodal basis. Figure 6 illustrates this for a simple case in one dimension. We consider the interpolation of a function f, defined over the unit interval, by piecewise linear functions f_2 and f_3 which have 2 and 3 nodes, respectively. Let $f_3 = \alpha_1\phi_1 + \alpha_2\phi_2 + \alpha_3\phi_3$ for some basis $\phi = \{\phi_1, \phi_2, \phi_3\}$. If ϕ is the usual nodal basis, $\alpha_2 = f(\frac{1}{2})$. But, if ϕ is the hierarchical basis, $\alpha_2 = f(\frac{1}{2}) - f_2(\frac{1}{2})$, i.e., α_2 represents how much change occurs

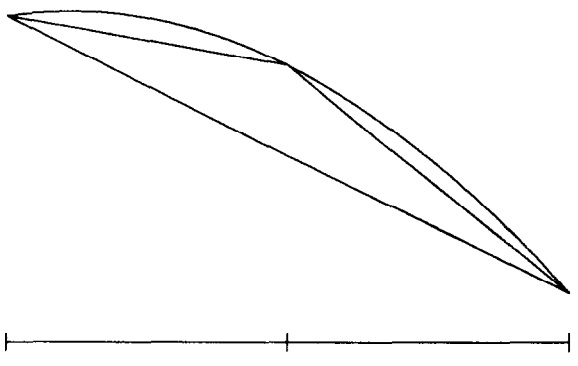

Fig. 6. A smooth function and its 2-node and 3-node linear interpolants.

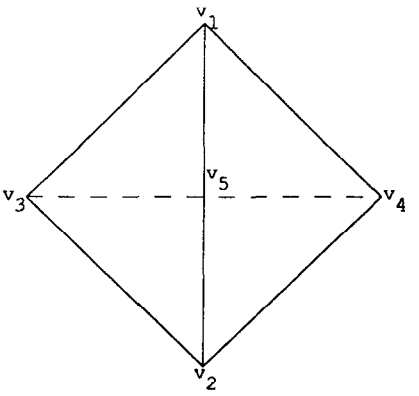

Fig. 7. Triangle pair for error indicator.

in the approximating function when we refine the grid by adding the new node. Then $\| \alpha_2 \phi_2 \|$ is the norm of the change in the approximation. This is the basis of our error indicator. These principles can be extended not only to linear bases over bisected triangles, but to any finite-element space, even rectangles, 3-D, etc. We will consider in detail the spaces of $C^0 p$th-degree polynomials over bisected triangles, starting with the linear case.

The division of a pair of triangles as in Fig. 7 by newest vertex bisection corresponds to the addition of one new basis function. Let v_i and ϕ_i, $i = 1$, 2, 3 and 4, be the vertices of the triangles and corresponding hierarchical basis functions, and v_5 and ϕ_5 be the new vertex and basis function. We wish to approximate how much change would occur if this division were to be performed. If the grid were being generated for interpolation of a known function u, then we would simply define $\alpha_5 = u(v_5) - U(v_5)$, where U is the interpolating function on the grid before refinement, and compute $\| \alpha_5 \phi_5 \|$. But if we are solving a partial differential equation, we must approximate α_5 by assuming that α_i, the coefficients of ϕ_i, remain unchanged for $i = 1$, 2, 3 and 4. Then, for a finite-element method

$$\alpha_5 = \frac{(f, \phi_5) - \sum_{i=1}^{4} \alpha_i \langle \phi_i, \phi_5 \rangle}{\langle \phi_5, \phi_5 \rangle},$$

where $\langle \cdot, \cdot \rangle$ and $(\cdot, \cdot)$ are the usual inner products used to obtain the stiffness matrix and load vector, respectively, and f is the right-hand side of the differential equation. This corresponds to one step of a Gauss–Seidel iteration for the linear system we would have if we divided this pair of triangles. If $\| \cdot \|$ is the energy norm defined by $\| u \|^2 = \langle u, u \rangle$, $\| \alpha_5 \phi_5 \|^2$ is the amount by which the square of the energy norm of the error is reduced by adding $\alpha_5 \phi_5$ to the approximate solution. We use $\| \alpha_5 \phi_5 \|$ as the error indicator for this pair of triangles.

We assumed above that we are computing an error indicator for a compatibly divisible pair of triangles. We must also have error indicators for triangles whose base is on the boundary and triangles that are not compatibly divisible. Boundary triangles are treated conceptually the same: α_5 is determined by a Gauss–Seidel relaxation of the (boundary condition) equation that would be added to the linear system if this triangle were divided. In the case of Dirichlet boundary

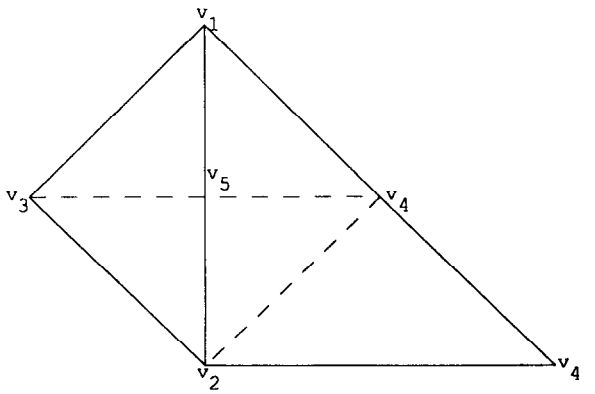

Fig. 8. Triangle pair for error indicator when not compatibly divisible.

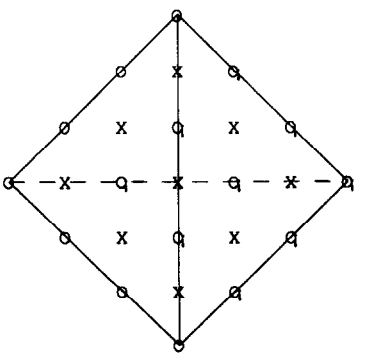

Fig. 9. Old nodes (O) and new nodes (X) associated with a new vertex (cubic elements).

conditions, α_5 is simply the difference between the boundary condition and approximate solution at the prospective new vertex. When one has a triangle that is not compatibly divisible, such as triangle $v_1v_2v_3$ in Fig. 8, we introduce the vertex $\hat{v}_4$ (the vertex which must be added before v_5) with corresponding basis function $\hat{\phi}_4$ and coefficient $\hat{\alpha}_4 = \frac{1}{2}(\alpha_1 + \alpha_4)$ and replace α_4 and ϕ_4 by $\hat{\alpha}_4$ and $\hat{\phi}_4$.

The extension of this error indicator to C^0pth-degree polynomials is straightforward. The only difference is that now there are p^2 new basis functions, one associated with each new node as shown for cubics in Fig. 9, so the amount of change depends on more than one basis function. But we still have a hierarchical basis whose coefficients represent change, and in principle the error indicator is computed the same way. Now the equation for α_5 is replaced by a linear system of p^2 equations in p^2 unknowns. This system can be solved by Cholesky decomposition in an acceptable number of operations, and we can compute $\|\Sigma\alpha_i\phi_i\|$ where the sum is over the p^2 new nodes associated with the prospective new vertex. It can be seen that this error indicator can be further extended to any finite-element space with a hierarchical basis.

The way in which we compute our error indicator presents a very interesting interpretation of the role of the error indicator. We could imagine a situation in which we have an *infinitely* refined uniform grid with the hierarchical basis. Recall that, with the hierarchical basis, the coefficients represent how much change in the solution occurs by including the corresponding basis functions. The process of adaptive refinement is now one of discarding, from our infinite number of basis functions, those whose coefficients are very close to zero, i.e., those basis functions that do not make a significant contribution to the solution. The computation of our error indicator is a form of relaxation for some of the basis functions that have not yet been added to the finite-dimensional subspace. Thus, in effect, we are using a larger approximation space, and are ignoring those basis functions which do not make a significant contribution to the solution.

Given the error indicators for every triangle (or pairs of compatibly divisible triangles) we would ideally want to select the next triangle to divide by choosing the triangle with the largest error indicator. However, for the algorithm to use only O(number of triangles) operations it is imperative that the selection of the next triangle to divide requires only O(1) operations. This means that we do not have time to search every triangle to find one with the largest error indicator. So instead we will be satisfied to find a triangle whose error indicator is close to the largest.

Let e be the largest error indicator at the beginning of the refinement phase. We partition the triangles (only including one triangle from each pair of compatibly divisible triangles) into Q sets such that each set contains all the triangles whose error indicators fall in a certain range. Specifically, for a given $0 < c < 1$, a triangle is in the qth set iff its error indicator is between $c^{q-1}e$ and c^qe for $1 \leqslant q \leqslant Q-1$ and is in the Qth set if its error indicator is less than $c^{Q-1}e$. The first set contains all the triangles whose error indicator is larger than ce, and we will select any one of these triangles as the next triangle to be divided.

To represent the sets, we use a doubly linked list for each set. It is then easy to do any of the following processes in O(1) operations. The head of the first list is selected as the next triangle to divide. If the first list is empty, then e is replaced by ce and all the lists are "shifted to the left" by shifting the head and the tail pointers, for example, head (1) ← head (2). The Qth list is left empty. When a triangle is divided, it is removed from the list. After division, error indicators are computed for the new triangles and these triangles are added to the appropriate list. It is possible

that the new error indicators could be larger than e. In this case e is replaced by e/c and the lists are "shifted to the right" with the $(Q-1)$st and Qth lists merged into one. If the error indicators for neighboring triangles are also updated, these triangles are removed from the lists and inserted back into the correct lists.

By allowing insertion to occur at either the head or tail, we can improve the resolution of the partition, essentially doubling the number of sets. To do this, insert at the head if the error indicator is larger than the midpoint of the range and at the tail if it is smaller than the midpoint.

4. Numerical results

The adaptive refinement method described here has been implemented as part of the FORTRAN program MGGHAT (MultiGrid Galerkin Hierarchical Adaptive Triangles) [9,11] to solve elliptic boundary value problems. Computations were performed on a Pyramid 90x with floating-point accelerator operating under the Pyramid Technology OSx 3.1 Operating System which is a dual port of AT&T Bell Laboratories' System V Release 2.0 and the University of California, Berkeley's 4.2BSD. The Pyramid Technology Optimizing FORTRAN 77 compiler was used with single precision, which has about 7 decimal digits.

For these computations, we use the following problems.

Problem 1. Laplace's equation on the L-shaped domain of Fig. 10(a) with the Dirichlet boundary conditions chosen so that the true solution is $r^{2/3}\sin(\frac{2}{3}\theta)$. Both x and y range from -1 to 1 and the re-entrant corner is located at the origin. Fig. 10(a) shows a sample adaptively refined grid with the initial 6 triangles in bold. The solution exhibits the leading term of the singularity due to the 270° re-entrant corner.

Problem 2. Laplace's equation on the hexagonal domain of Fig. 10(b). The domain has a slit along the positive x-axis. x ranges from -1 to 1 and y from $-\frac{1}{2}\sqrt{3}$ to $\frac{1}{2}\sqrt{3}$. The Dirichlet boundary conditions are chosen so that the true solution is $r^{1/2}\sin(\frac{1}{2}\theta)$. The re-entrant corner is located at the origin. Fig. 10(b) shows a sample adaptively refined grid with the initial 6 triangles in bold. The solution exhibits the leading term of the singularity due to the 360° re-entrant corner.

Problem 3. This is Problem 54 in the elliptic PDE population of [13,14]. The differential equation is

$$\left((1+x^2)u_x\right)_x + \left((1+A^2)u_y\right)_y - \left(1+(8y-x-4)^2\right)u = f$$

on the unit square, where $A = 4y^2 + 0.9$. The right-hand side and Dirichlet boundary conditions are chosen so that the exact solution is

$$\frac{2.25\,x(x-A)^2(1-D)}{A^3} + \frac{1}{1+(8y-x-4)^2},$$

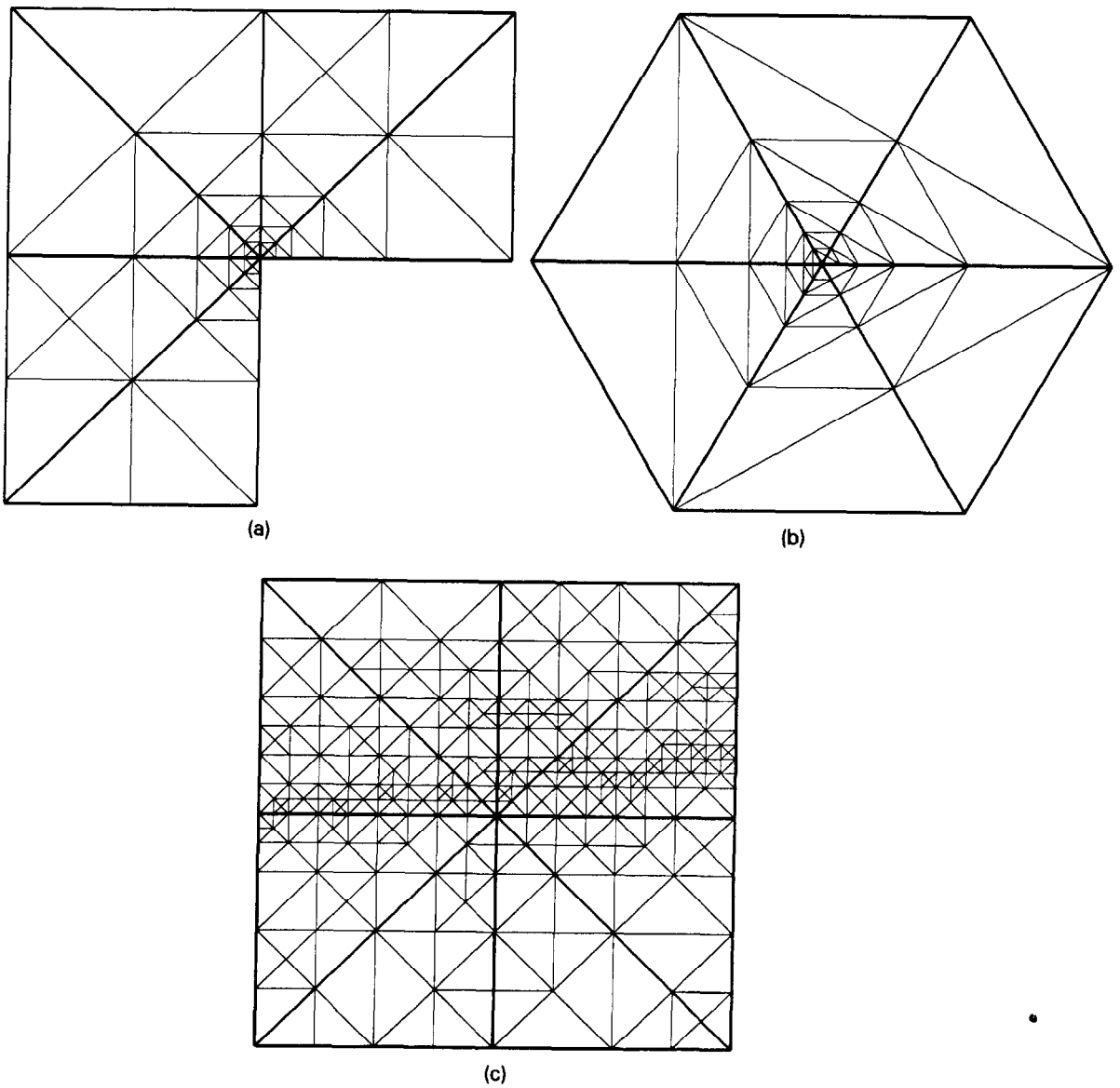

Fig. 10. Domains and sample grids for (a) Problem 1; (b) Problem 2; (c) Problem 3.

where

$$B = \max\left\{0, \left(3 - \frac{x}{A}\right)^3\right\}, \qquad C = \max\{0, x - A\}, \qquad D = \begin{cases} 0 & \text{if } C < 0.02, \\ e^{-B/C} & \text{if } C \geq 0.02. \end{cases}$$

Figure 10(c) shows a sample adaptively refined grid with the initial 8 triangles in bold. A contour plot of the solution can be found in [13] or [14]. The solution has a ridge in the vicinity of $y \approx 0.6$–0.7.

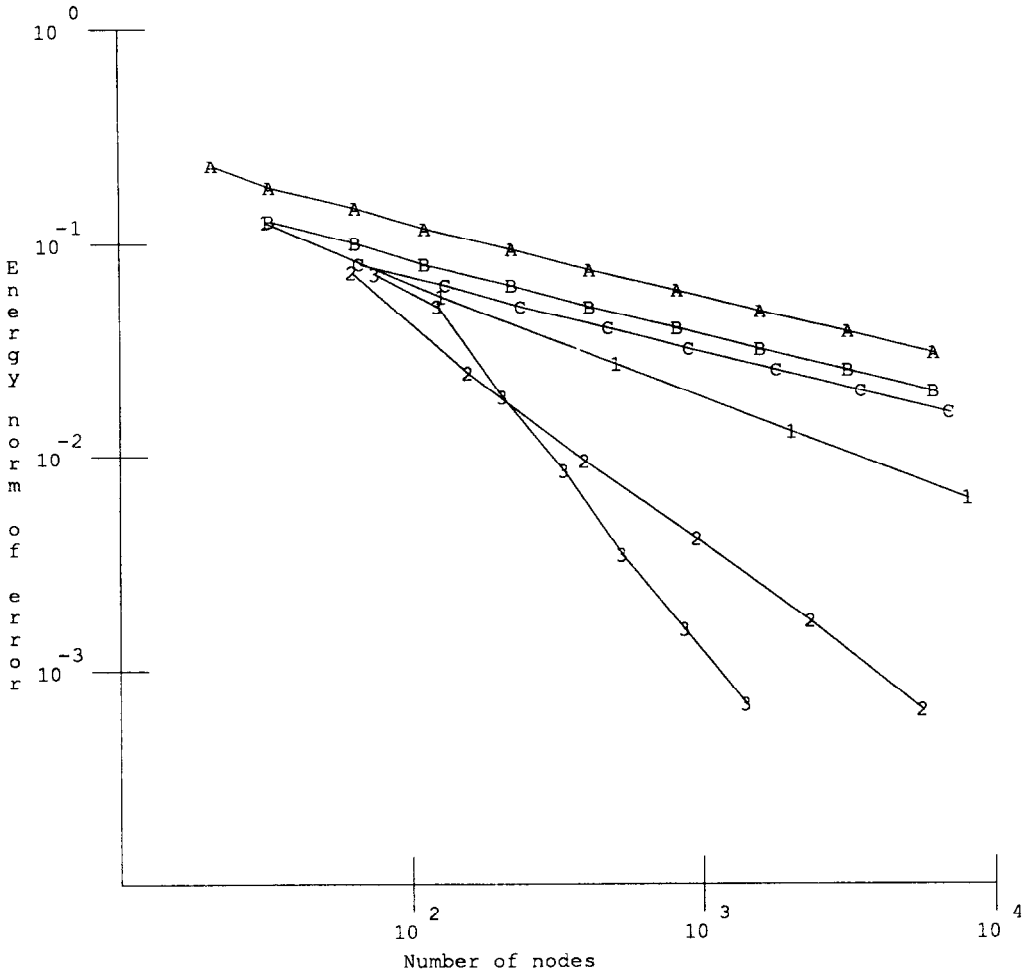

Fig. 11. Results for Problem 1.

For uniform grids, the rate of convergence of the discretization error is usually given in terms of h, a measure of the size of the triangles. A method is said to have order 2α if the energy norm of the error decreases like $O(h^{2\alpha})$ as h gets small. For adaptively refined grids, h is not such a meaningful entity, but we can use N, the number of nodes, to measure the rate of convergence of the discretization error. For a uniform grid, $N = O(h^{-2})$ so the error decreases like $O(N^{-\alpha})$ as N gets large. We define α to be the *rate of convergence of the discretization error* if α is the largest value such that the discretization error is $O(N^{-\alpha})$, i.e., $\| u - u_N \| \sim cN^{-\alpha}$ for some constant c where u is the true solution of the differential equation, u_N is the approximate solution using N nodes, and $\| \cdot \|$ is the energy norm. For smooth functions, $\alpha = \frac{1}{2}$, 1 and $\frac{3}{2}$ for linear, quadratic and cubic elements, respectively.

For Problems 1 and 2, where the solution is not smooth, the best rate of convergence one can hope for with a uniform grid is $\alpha = \frac{1}{3}$ and $\frac{1}{4}$, respectively, no matter what degree polynomials are used. It is possible for adaptive refinement to recover the optimal rate of convergence. The problems were solved using linear, quadratic and cubic elements with both uniform and

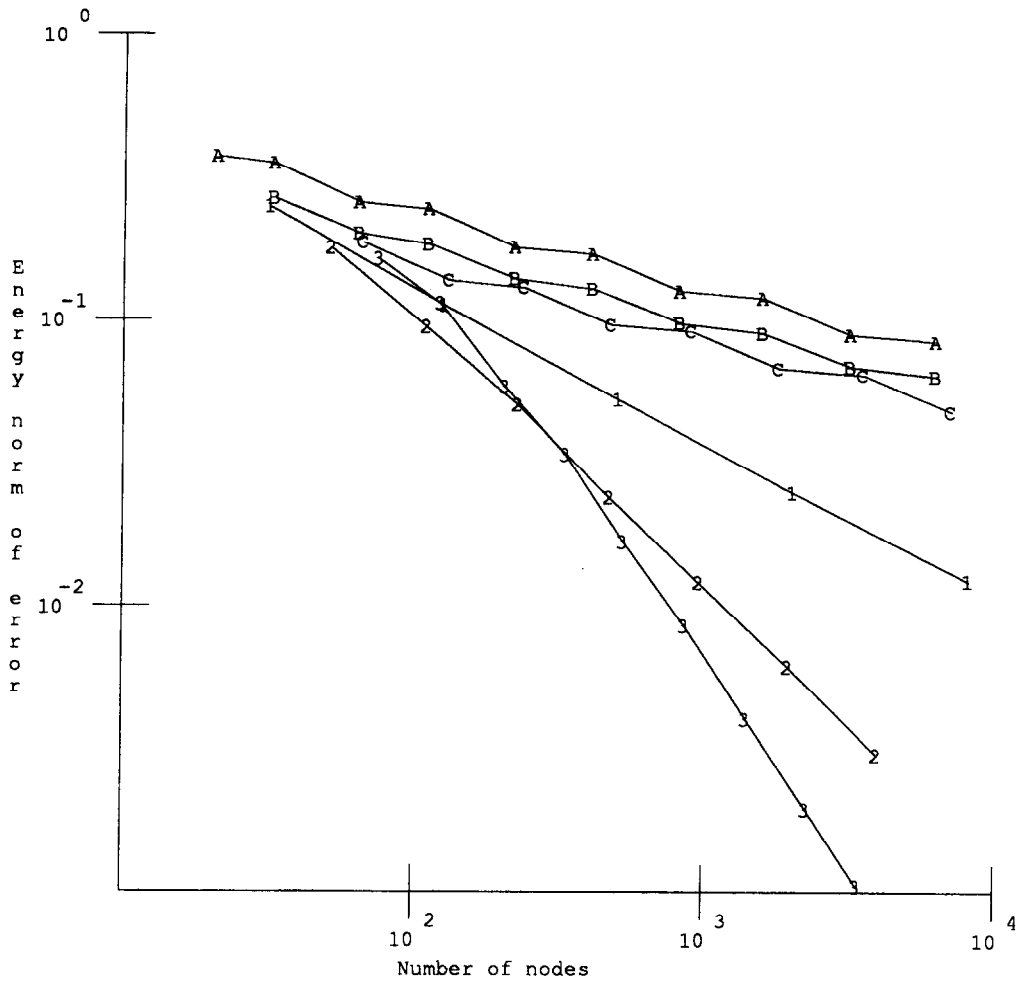

Fig. 12. Results for Problem 2.

adaptively refined grids. The results are presented in Figs. 11, 12 and 13 for Problems 1, 2 and 3, respectively. The data points on the graphs are labeled with A, B and C for linear, quadratic and cubic elements, respectively, for the uniform grids, and 1, 2 and 3 for linear, quadratic and cubic elements, respectively, for the adaptive grids. The observed rate of convergence is given by the slope of a linear least squares fit of the data. When appropriate, we discard some of the first data points in determining the slope. These slopes are given in Tables 1 and 2.

For uniform grids, the rate of convergence is about $\frac{1}{3}$ and $\frac{1}{4}$ for Problems 1 and 2, respectively, for all three polynomial degrees. For adaptive grids, the rate of convergence is about $\frac{1}{2}$, 1 and $\frac{3}{2}$ for linear, quadratic and cubic elements, respectively, for both problems. For Problem 3 the rate of convergence is slightly larger than $\frac{1}{2}$, 1 and $\frac{3}{2}$ for linear, quadratic and cubic elements, respectively, for both uniform and adaptive grids. Although the uniform grids achieve the optimal order of convergence for Problem 3, the adaptive grids have a smaller constant of proportionality.

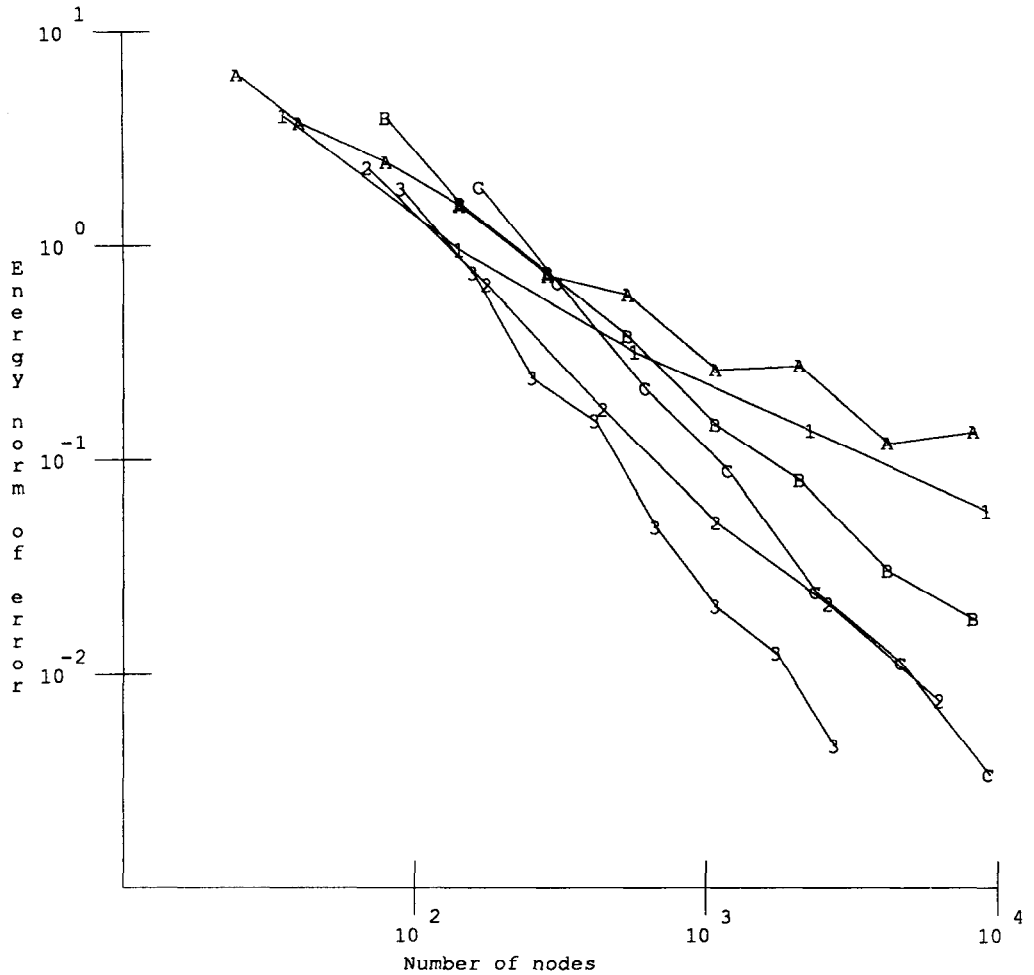

Fig. 13. Results for Problem 3.

In numerical experiments that compare low- and high-order methods with uniform grids for problems with well-behaved solutions (e.g., [13]) it is usually observed that for very low accuracy it is more efficient to use linear elements, but for moderate and high accuracy the high-order elements are more efficient. We observe the same result when using adaptively refined grids.

Table 1
Observed order of convergence with uniform grids

Problem	Linear elements	Quadratic elements	Cubic elements
1	0.355	0.355	0.349
2	0.284	0.273	0.264
3	0.697	1.131	1.555

Table 2
Observed order of convergence with adaptive grids

Problem	Linear elements	Quadratic elements	Cubic elements
1	0.540	1.011	1.633
2	0.542	0.967	1.496
3	0.616	1.159	1.680

References

[1] I. Babuška and A.K. Aziz, On the angle condition in the finite element method, *SIAM J. Numer. Anal.* **13** (1976) 214–226.

[2] I. Babuška and W.C. Rheinboldt, Error estimates for adaptive finite element computations, *SIAM J. Numer. Anal.* **15** (1978) 736–754.

[3] R.E. Bank, T.F. Dupont and H. Yserentant, The hierarchical basis multigrid method, Preprint SC-87-1, Konrad-Zuse-Zentrum für Informationstechnik, 1987.

[4] R.E. Bank and A.H. Sherman, The use of adaptive grid refinement for badly behaved elliptic partial differential equations, in: R. Vichnevetsky and R.S. Stepleman, Eds., *Advances in Computer Methods for Partial Differential Equations III* (IMACS, New Brunswick, NJ, 1979) 33–39.

[5] R.E. Bank and A.H. Sherman, An adaptive multilevel method for elliptic boundary value problems, *Computing* **26** (1981) 91–105.

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

[7] J.P. De, S.R. Gago, D.W. Kelly, O.C. Zienkiewicz and I. Babuška, A posteriori error analysis and adaptive processes in the finite element method: Part II — Adaptive mesh refinements, *Internat. J. Numer. Methods Engrg.* **19** (1983) 1621–1656.

[8] I. Fried, Condition of finite element matrices generated from nonuniform meshes, *AIAA J.* **10** (1972) 219–221.

[9] W.F. Mitchell, Unified multilevel adaptive finite element methods for elliptic problems, Ph.D. Thesis, Report no. UIUCDCS-R-88-1436, Dept. Comput. Sci., Univ. Illinois, Urbana, IL, 1988.

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

[11] W.F. Mitchell, Optimal multilevel iterative methods for adaptive grids, Preliminary Proceedings of Copper Mountain Conference on Iterative Methods, Copper Mountain, CO, 1990; *SIAM J. Matrix Anal. Appl.*, to appear.

[12] W.C. Rheinboldt, On a theory of mesh-refinement processes, *SIAM J. Numer. Anal.* **17** (1980) 766–778.

[13] J.R. Rice and R.F. Boisvert, *Solving Elliptic Problems Using ELLPACK* (Springer, New York, 1985).

[14] J.R. Rice, E.N. Houstis and W.R. Dyksen, A population of linear, second order elliptic partial differential equations on rectangular domains, *Math. Comp.* **36** (1981) 475–484.

[15] M.-C. Rivara, Mesh refinement processes based on the generalized bisection of simplices, *SIAM J. Numer. Anal.* **21** (1984) 604–613.

[16] M.-C. Rivara, Algorithms for refining triangular grids suitable for adaptive and multigrid techniques, *Internat. J. Numer. Methods Engrg.* **20** (1984) 745–756.

[17] E.G. Sewell, Automatic generation of triangulations for piecewise polynomial approximation, Ph.D. Thesis, Purdue Univ., West Lafayette, IN, 1972.

[18] E.G. Sewell, A finite element program with automatic user-controlled mesh grading, in: R. Vichnevetsky and R.S. Stepleman, Eds., *Advances in Computer Methods for Partial Differential Equations III* (IMACS, New Brunswick, NJ, 1979) 8–10.

[19] H. Yserentant, Hierarchical bases give conjugate gradient type methods a multigrid speed of convergence, *Appl. Math. Comput.* **19** (1986) 347–358.

[20] H. Yserentant, On the multi-level splitting of finite element spaces, *Numer. Math.* **49** (1986) 379–412.

[21] O.C. Zienkiewicz, D.W. Kelly, J. Gago and I. Babuška, Hierarchical finite element approaches, error estimates and adaptive refinement, in: J.R. Whiteman, Ed., *The Mathematics of Finite Elements and Applications IV* (Academic Press, New York, 1982) 313–346.