THE HIERARCHICAL BASIS MULTIGRID METHOD AND INCOMPLETE LU DECOMPOSITION

RANDOLPH E. BANK* AND JINCHAO XU†

Abstract. A new multigrid or incomplete LU technique is developed in this paper for solving large sparse algebraic systems from discretizing partial differential equations. By exploring some deep connection between the hierarchical basis method and incomplete LU decomposition, the resulting algorithm can be effectively applied to problems discretized on completely unstructured grids. Numerical experiments demonstrating the efficiency of the method are also reported.

Key words. Finite element, hierarchical basis, multigrid, incomplete LU.

AMS subject classifications. 65F10, 65N20

1. Introduction. In this work, we explore the connection between the methods of sparse Gaussian elimination [8][13], incomplete LU (ILU) decomposition [9][10] and the hierarchical basis multigrid (HBMG) [16][4].

Hierarchical basis methods have proved to be one of the more robust classes of methods for solving broad classes of elliptic partial differential equations, especially the large systems arising in conjunction with adaptive local mesh refinement techniques [5][2]; they have been shown to be strongly connected to space decomposition methods and to classical multigrid methods [14][15][4][9]. As with typical multigrid methods, classical hierarchical basis methods are usually defined in terms of an underlying refinement structure of a sequence of nested meshes. In many cases this is no disadvantage, but it limits the applicability of the methods to truly unstructured meshes, which may be highly nonuniform but *not* derived from some grid refinement process. A major goal of our study is to generalize the construction of hierarchical bases to such meshes, allowing HBMG and other hierarchical basis methods to be applied. Some work on multigrid methods on non-nested meshes is reported in Bramble, Pasciak and Xu [6], Xu [14], and Zhang [17].

In Section 2, we develop a simple graph elimination model for classical hierarchical basis methods on sequences of nested meshes. This elimination model can be interpreted as a particular ILU decomposition where certain fillin edges, namely those corresponding the element edges on a coarser mesh, are allowed. This graph elimination model can be generalized in a very simple and straightforward fashion to the case of completely unstructured meshes, providing a simple mechanism for defining hierarchical bases on such meshes. The key concept is that of *vertex parents* of a given vertex v_i in the mesh. In the case of a sequence of refined meshes, the parents of v_i are just the endpoints of the triangle edge which was bisected when v_i was created. By generalizing this notion slightly, we are able to define vertex parents for vertices in an unstructured mesh, which in effect, supplies us with a heuristic procedure for systematically unrefining the unstructured mesh.

In Section 3, we describe algebraic aspects of HBMG and its application to completely unstructured meshes. In the classical case, vertices are ordered (blocked) by refinement level and we apply symmetric block Gauss-Seidel to the linear system rep-

* Department of Mathematics, University of California at San Diego, La Jolla, CA 92093. The work of this author was supported by the Office of Naval Research under contract N00014-89J-1440.

† Department of Mathematics, Penn State University, University Park, PA 16802. The work of this author was supported by National Science Fundation.

resented in the hierarchical basis. We note that it is the transformation of the stiffness matrix from nodal to hierarchical basis which has a strong connection to ILU, and this is completely algebraic in nature once the transformation is defined. In turn, the transformation matrix relies upon the vertex parents function to determine its sparsity structure and blocking, and upon the geometric properties of the refinement procedure to determine the numerical values of the nonzero elements. Since we define vertex parents on unstructured meshes as part of our ILU graph elimination algorithm, we can determine the structure of the transformation matrix just as in the case of nested meshes. The numerical values are selected by examining the geometry of the mesh, in a fashion similar to the nested mesh case. At the level of implementation, HBMG and other iterations based on hierarchical bases are algebraically identical for the cases of structured and unstructured meshes. Indeed, we made only slight changes to our HBMG routines to implement the new method; essentially all new coding was devoted to the graph elimination process and the hueristics for determining vertex parents.

In Section 4, we present a numerical illustration of the method and make a few concluding remarks.

2. Graph Theoretical Properties of Hierarchical Bases. In this section, we explore the connection between the HBMG method and ILU decomposition in terms of graph theory. We will consider first the standard Gaussian elimination and classical ILU factorization. We then progress to the HBMG method, first considering the triangular meshes generated through a process of grid refinement and then considering completely unstructured triangular meshes. To begin, we briefly review the process of Gaussian Elimination from a graph theoretical point of view. A more complete discussion of this point can be found in Rose [13] or George and Liu [8].

Corresponding to a sparse, symmetric, positive definite $N \times N$ matrix A, let $\mathcal{G}(X, E)$ be the graph that consists of a set of N ordered vertices $v_i \in X$, $1 \leq i \leq N$, and a set of edges E such that the edge (connecting vertices v_i and v_j) $e_{ij} \in E$ if and only if $a_{ij} \neq 0$, $i \neq j$. Note that edges in the graph $\mathcal{G}$ correspond to the nonzero off diagonal entries of A. For the case of interest here, A is the stiffness matrix for the space of continuous piecewise linear polynomials represented in the standard nodal basis. Then $\mathcal{G}$ is just the underlying triangulation of the domain (with some possible minor modifications due to the treatment of Dirichlet boundary conditions). If we view $\mathcal{G}$ as an *unordered* graph, then the graph corresponds to the class of matrices of the form P^tAP, where P is a permutation matrix; that is, reordering the vertices of the graph corresponds to forming the product P^tAP for a suitable permutation matrix P.

For convenience, we shall need a little additional terminology from graph theory. Let $v_i \in X$; the set of adjacent vertices $adj(v_i)$is given by

$$adj(v_i) = \{v_j \in X | \, e_{ij} \in E\}.$$

Roughly speaking, the set $adj(v_i)$ corresponds to the set of column indices for the nonzero entries in row i of matrix A (or the set of row indices for nonzero entries of column i of A), with the exception of the diagonal entry a_{ii}.

A *clique* $\mathcal{C} \subseteq X$ is a set of vertices which are all pairwise connected; that is $v_i, v_j \in \mathcal{C}, i \neq j \Rightarrow e_{ij} \in E$. If A is a dense $N \times N$ matrix, then its graph is a clique on N vertices. More generally, with a proper ordering of the vertices, cliques correspond to dense submatrices of A.

With these definitions, we can define the graph theoretic equivalent of Gaussian elimination of A. First, in terms of matrices let

$$A = \begin{bmatrix} a_{11} & r^t \\ r & B \end{bmatrix},$$

where r is an $N-1$-vector and B is an $(N-1)\times(N-1)$ matrix. The first step of Gaussian elimination consists of the factorization

$$\begin{aligned} A &= \begin{bmatrix} 1 & 0 \\ r/a_{11} & I \end{bmatrix} \begin{bmatrix} a_{11} & 0 \\ 0 & B - rr^t/a_{11} \end{bmatrix} \begin{bmatrix} 1 & r^t/a_{11} \\ 0 & I \end{bmatrix} \\ &= L_1 D_1 L_1^t. \end{aligned}$$

The matrix $A' = B - rr^t/a_{11}$ is a symmetric, positive definite matrix of order $N-1$ to which the factorization can be inductively applied. Note that A' may be less sparse than B due to the *fillin* caused by the outer product rr^t/a_{11}.

In graph theoretic terms, eliminating vertex v_1 from $\mathcal{G}$ transforms $\mathcal{G}(X,E)$ to a new graph $\mathcal{G}'(X',E')$, corresponding to matrix A', as follows

1. Eliminate vertex v_1 and all its incident edges from $\mathcal{G}$. Set $X' = X - \{v_1\}$. Denote the resulting set of edges $E_1 \subseteq E$.
2. Create the set F of *fillin* edges as follows: For each distinct pair $v_j, v_k \in adj(v_1)$ in $\mathcal{G}$, add the edge e_{jk} to F if is not already present in E_1. Set $E' = E_1 \cup F$.

Note that the set $adj(v_1)$ in $\mathcal{G}$ becomes a clique in $\mathcal{G}'$.

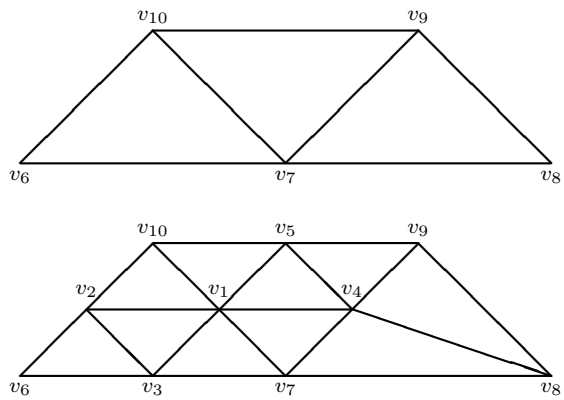
Within this framework, an ILU factorization is one in which all the fillin called for in step 2 above is not allowed. The classical form of ILU is to allow *no* fillin, that is, no new edges are added in step 2 ($E' = E_1$). That forces the resulting matrix A', (which is now not necessarily equal to $B - rr^t/a_{11}$) to have the same sparsity pattern as B. The effect of the neglected fillin elements in terms of the numerical values of entries in A' varies, and is not considered here; at the moment, our concern is with the sparsity pattern itself.

Now let us view HBMG on a set of nested meshes in terms of ILU. For convenience, we will restrict to the case of only two levels. Let $\mathcal{T}_c$ be the coarse triangulation, and $\mathcal{T}_f$ be the fine triangulation, where some elements $t \in \mathcal{T}_c$ are refined into four elements in $\mathcal{T}_f$ by pairwise connecting the midpoints of the three edges of t (regular refinement). Some elements near the boundary of the refined region can be bisected (*green* refinement), while others are left unrefined. The details of such a refinement algorithm can be found in [5] [2] and are not of great interest to the current discussion. An example is shown in Figure 1. Let X be the set of vertices in $\mathcal{T}_f$, and $X_c \subset X$ be the set of vertices in $\mathcal{T}_c$. Denote by X_f the set of fine grid vertices not in X_c ($X_f = X \setminus X_c$).

For each vertex $v_i \in X_f$, there are a (unique) pair of vertices $v_j, v_k \in X_c$ such that v_i is the midpoint of the edge connecting v_j and v_k in the coarse grid $\mathcal{T}_c$. This pair of vertices are called the *vertex parents* of v_i. The vertex parents for the set X_f for our example are given in Figure 1.

Suppose now that we choose an ordering in which all the vertices in X_f are ordered first, followed by those in X_c. We then consider eliminating the vertices in X_f as follows:

1. Eliminate vertex v_1 and all its incident edges from $\mathcal{G}$. Set $X' = X - \{v_1\}$. Denote the resulting set of edges $E_1 \subseteq E$.



vertex	parents
v_1	v_7, v_{10}
v_2	v_6, v_{10}
v_3	v_6, v_7
v_4	v_7, v_9
v_5	v_9, v_{10}

FIG. 1. *The coarse grid $\mathcal{T}_c$ and the fine grid $\mathcal{T}_f$ Vertices $\{v_i\}_{i=6}^{10} = X_c$, while $\{v_i\}_{i=1}^{5} = X_f$.*

2. Add *one* fillin edge connecting the vertex parents of v_i, say $v_j, v_k \in X_c$. Set $E' = E_1 \cup \{e_{jk}\}$.

It is easy to see this is an ILU algorithm in that only selected fillin edges are allowed, namely those connecting vertex parents. It also is important to note that the triangulation $\mathcal{T}_f$ is the graph for the original stiffness matrix A represented in the standard nodal basis. After all the vertices in X_f are eliminated, the resulting graph is just the triangulation $\mathcal{T}_c$; that is, the sparity structure of the coarse grid matrix corresponds to the coarse grid triangulation. One of the important properties of HBMG is that the corresponding coarse grid matrix is just the stiffness matrix with respect to the nodal basis of the coarse grid (e.g., the hierarchical basis). For this to occur requires a particular (but natural) choice of numerical values for the multipliers used in computing the ILU. This is a topic for the next section. A comparison of the elimination graphs for vertex v_1, using regular Gaussian elimination, classical ILU and HBMG is shown in Figure 2.

This brings us to HBMG for completely unstructured, nonnested meshes. So far we have considered two different algorithms. We begin with the simpler of the two. Suppose that for a vertex $v_i \in X$, we denote a pair of distinct vertices $v_j, v_k \in adj(v_i)$ as the *tentative vertex parents* of v_i. Generally, in selecting tentative vertex parents, we try to emulate the case of nested HBMG as closely as possible; that is, we want v_i to be "close" in some sense to being the midpoint of the straight line connecting v_j and v_k. Not all vertices can be assigned reasonable parents; those that cannot will be called "corners" of the region. Among those vertices that do have tentative parents, we chose the vertex v_i which best optimizes the hueristic criteria used in selecting tentative parents, and eliminate it as follows:

1. Eliminate v_i and all its incident edges. Set $X' = X - \{v_i\}$.
2. If not already present, add one fillin edge connecting the tentative vertex parents of v_i, say v_j and v_k. These become the *permanent* vertex parents of

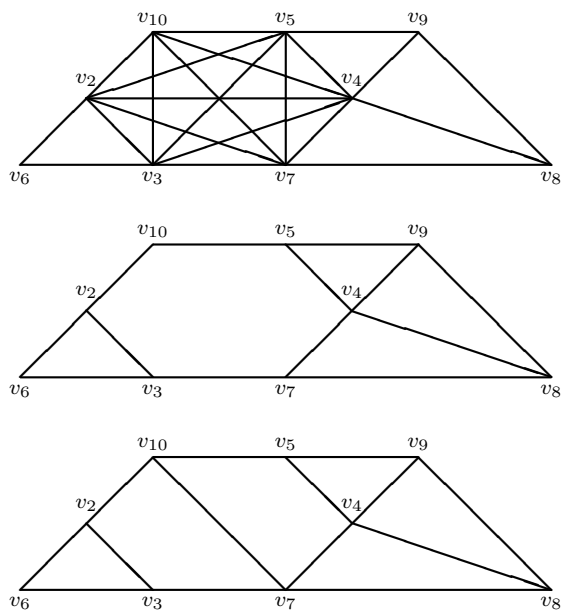


FIG. 2. *The elimination graphs $\mathcal{G}'$ generated by eliminating vertex v_1. Standard Gaussian elimination is shown at the top, classical ILU in the center, and HBMG at the bottom.*

v_i. Set $E' = E_i \cup \{e_{jk}\}$.

3. For each $v_\ell \in adj(v_i)$ in $\mathcal{G}$, compute new tentative vertex parents based on the new graph $\mathcal{G}'$.

The third step is essential, since v_i could have been a tentative parent in $\mathcal{G}$ of any the vertices in $adj(v_i)$, including its permanent vertex parents v_j and v_k. Furthermore, the additional edge connecting v_j and v_k allows them to be considered as tentative vertex parents of each other in $\mathcal{G}'$. This algorithm continues inductively until no more vertices can be eliminated (all remaining vertices are corners).

After the first vertex is eliminated, the remaining elimination graphs are not necessarily triangulations of the domain, but typically contain polygonal elements of various orders. The second variant of the algorithm addresses this issue. In this algorithm, a vertex v_i can have either two or three tentative vertex parents, again selected from among the vertices in $adj(v_i)$. The first two are chosen as in the first variant. If v_i is "too far" from being colinear with its tentative parents v_j and v_k, then a third tentative parent, say $v_m \in adj(v_i)$ is chosen such that the resulting triangle with vertices v_j, v_k and v_m is optimized with respect to shape regularity. Once again, not all vertices may have tentative vertex parents, although in practice, allowing for the possibility of three parents tends to reduce the number of corners.

In any event, the vertex v_i which best optimizes the criteria, is then eliminated as follows:

1. Eliminate v_i and all its incident edges. Set $X' = X - \{v_i\}$.
2. If not already present, add a fillin edge connecting the two principal vertex parents of v_i. Add additional fillin edges as required, such that the resulting graph remains a triangulation of the domain. Denote this set of edges by F and set $E' = E_i \cup F$. The two or three tentative vertex parents of v_i become permanent.
3. For each $v_\ell \in adj(v_i)$ in $\mathcal{G}$, compute new tentative vertex parents based on

the new graph $\mathcal{G}'$.

This algorithm is aesthetically more pleasing than the first. Since all the elimination graphs are triangulations, it seems easier to handle in terms of the mathematical analysis. However, it generally allows for more fillin than the first algorithm, and requires more complex data structures based on linked lists to represent the graphs. The additional fillin and the possibility of having vertices with three vertex parents makes the resulting HBMG algorithm more expensive (see next section). Of course, this would be justified if the resulting HBMG algorithm performed significantly better, but so far in our experience, both algorithms perform comparably well in terms of convergence rate. Thus, at present we don't see a justification for the more expensive version.

As a final point in this section, we consider the assignment of vertex levels. Each vertex in the mesh has a unique level; this level is used to partition the stiffness matrix in HBMG and other hierarchical basis iterations. It is not such an important point for the current discussion, since we have assumed only two levels, but it is very important for the case of more than two levels. In the classical HBMG using a sequence of nested meshes, the level ℓ_i of vertex v_i is defined as follows. All vertices on the coarse grid are assigned $\ell_i = 1$. Thereafter, the remaining vertices are assigned levels in terms of the levels of their parents, according to

$$\ell_i = \max(\ell_k, \ell_j) + 1, \tag{1}$$

where v_k and v_j are the parents of vertex v_i. In the case of unstructured meshes, equation (1) can still be used, modified appropriately for the case of vertices with three vertex parents. All vertices without parents (corners) are assigned $\ell_i = 1$, and then (1) uniquely determines the level of the remaining vertices. In computing vertex levels, one should process the vertices in the *reverse* order of elimination, so that the level of all parents of vertex v_i will be defined prior to the processing of v_i.

3. Algebraic HBMG and ILU. In this section we consider the algebraic aspects of the HBMG method, and its relation to Gaussian elimination. Again for convenience we will consider the case of only two levels. Let A denote the stiffness matrix for the fine grid, and consider the block partitioning

$$A = \begin{bmatrix} A_f & C \\ C^t & A_c \end{bmatrix}, \tag{2}$$

where A_f corresponds to the nodal basis functions in X_f, A_c corresponds to the (fine grid) nodal basis functions in X_c, and C corresponds to the coupling between the two sets of basis functions. We consider transformations of the form $A' = SAS^t$, where S has the block structure

$$S = \begin{bmatrix} I & 0 \\ R & I \end{bmatrix}. \tag{3}$$

From (2)-(3), we obtain

$$SAS^t = \begin{bmatrix} A_f & A_f R^t + C \\ RA_f + C^t & \hat{A}_c \end{bmatrix}, \tag{4}$$

where

$$\hat{A}_c = RA_f R^t + C^t R^t + RC + A_c. \tag{5}$$

Different algorithms can be characterized by different choices of R. For example, in classical block Gaussian elimination $R = -C^t A_f^{-1}$, and $\hat{A}_c = A_c - C^t A_f^{-1} C$ is the Schur complement. In this case, the off diagonal blocks are reduced to zero, but at the cost of having fairly dense matrices R and $\hat{A}_c$.

In the case of classical HBMG, the matrix R^t is sparse, and contains information about changing from the nodal to hierarchical basis [4],[1]. Each row of R^t is zero except for two entries which are equal to 1/2. For the row corresponding to vertex $v_i \in X_f$, the two nonzero entries are in the columns corresponding to $v_j \in X_c$ and $v_k \in X_c$, where v_j and v_k are the vertex parents of v_i. In this case, the matrix $\hat{A}_c$ is just the stiffness matrix for the coarse grid represented in the *coarse* grid nodal basis. Although we know a priori that the graph for $\hat{A}_c$ is just the coarse grid triangulation $\mathcal{T}_c$, we can formally compute this graph by applying the symbolic ILU elimination process described in the last section. The matrix $RA_f + C^t$ is not zero as in the case of Gaussian elimination; indeed it is less sparse than C^t. However, the matrix is small in some sense; the usual Cauchy inequality estimate [3] [1] [7] written in this notation is:

$$|x^t(RA_f + C^t)y| \leq \gamma(x^t \hat{A}_c x)^{1/2}(y^t A_f y)^{1/2},$$

where $\gamma < 1$ is the constant in the strengthened Cauchy inequality. It is worth commenting that in implementation, the matrix $RA_f + C^t$ is never formed explicitly; all that is required to implement HBMG and other iterations using hierarchical bases (either additive or multiplicative variants) are the matrices A_f, C, $\hat{A}_c$ and R. For our current discussion, R is the critical matrix. A_f and C are of course just parts of the nodal stiffness matrix, and $\hat{A}_c$ is explicitly computed from (5) once R is known.

The sparsity pattern of R is completely determined by the vertex parents function; in the case of classical HBMG on nested meshes, it follows from the refinement structure of the mesh. The numerical values of the coefficients (the "weights" or the "multipliers") are all equal to 1/2 for the nested case; the 1/2 arises naturally from the geometry of the refinement process, stating that a vertex created at the midpoint of an edge of a coarse grid triangle is midway between its vertex parents.

Now let us consider HBMG on nonnested meshes; from the algebraic point of view, nothing changes (!!) from the nested case once the matrix R is defined, and to define R we need only two things: the vertex parents function to define the sparsity pattern, and the weights to define the numerical values. From this point of view, it should be clear that this process will (implicitly) construct linear combinations of the fine grid nodal basis functions, whose energy inner products appear as matrix elements in the matrix $\hat{A}_c$ just as in the nested case. The difference is that in the nested case, these complicated linear combinations reduce to simple nodal basis functions for the coarse mesh. For nonnested meshes, they remain complicated linear combinations of the fine grid basis functions. On the other hand, one never need explicitly deal with these basis functions (except in the mathematical analysis), since the iteration itself is completely algebraic. The critical issue for HBMG is *not* that one obtains simple coarse grid nodal basis functions, but rather that the *support* of the basis functions which are obtained is increasing at the proper rate, and as long as the complicated basis functions have that property, one should see the expected convergence rates. The use of modified hierarchical basis functions appears in the work of Hoppe and Kornhuber [11] and Kornhuber [12] in connection with the the solution of obstacle problems.

We now consider the construction of R for the two heuristic algorithms for un-

structured meshes described in the last section. In both cases, the vertex parents function (sparsity pattern for R) is determined by the graph elimination algorithms discussed in the last section. Each column of R^t will have either two or three nonzeroes, corresponding to its permanent vertex parents.

As for the weights, for the algorithm which allows just two vertex parents, let vertex v_i have parents v_j and v_k, and let $v_i' = \theta v_j + (1-\theta) v_k$ be the orthogonal projection of v_i onto the straight line connecting v_j and v_k. Then we take the corresponding weights (numerical values of the nonzeroes in R) to be θ and $1-\theta$.

In the case of a vertex with three vertex parents, let v_j and v_k be the two principle parents and v_ℓ the third. In this case, we compute the barycentric coordinates of v_i with respect to the the triangle with vertices v_j, v_k and v_ℓ, and these barycentric coordinates became the numerical values used in R. Since v_i was supposed to be close to the line connecting v_j and v_k, the barycentric coordinate corresponding to v_ℓ should generally be small in comparison with the others, which in turn, should be close to the form described above for method using only two vertex parents. Allowing for the possibility of three permanent parents means that the matrix R will be less sparse, which in turn means $\hat{A}_c$ will be less sparse, a situation which is of course compounded as recursion adds more levels. Overall, this leads to more work (a bigger constant, but apparently not a change in the order of magnitude), but as far as we can tell from our early experience with the algorithm, does not significantly improve the rate of convergence.

4. Numerical Illustrations and Conclusions. In this section, we present a simple example of our algorithm. This example was developed using the finite element package $PLTMG$ [2]. We consider a square domain Ω with a circular hole. We triangulate this domain using 684 triangles and 398 vertices as shown in Figure 3. The mesh is unstructured, in that it was not generated through the refinement of a coarser mesh.

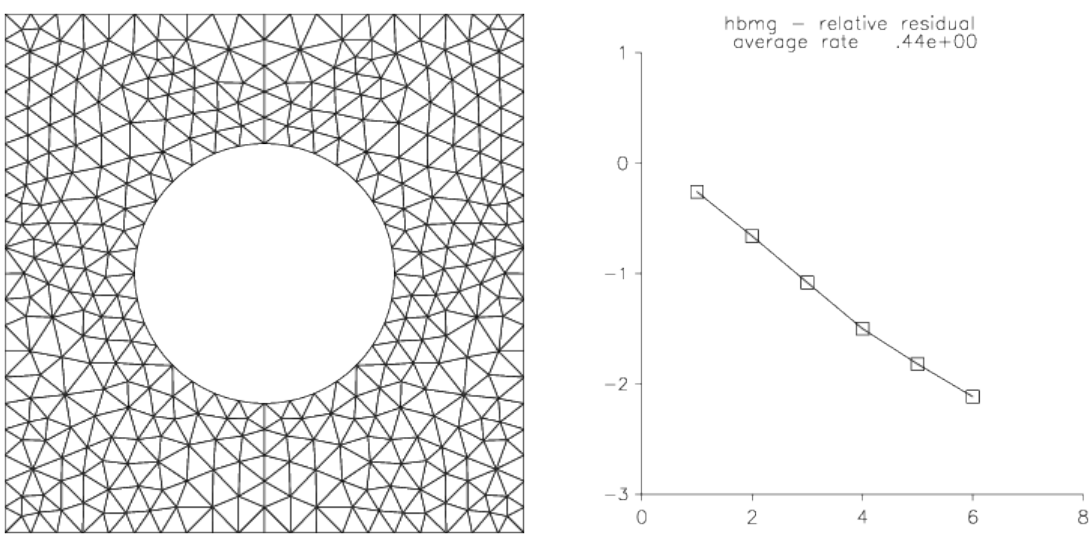


FIG. 3. *The triangulation and the convergence history for HBMG.*

We solved the equation $-\Delta u = 1$ in Ω with a combination of homogeneous Dirichlet and Neumann boundary conditions on $\partial\Omega$. Applying the algorithms outlined in Sections 2-3, we created an algebraic hierarchical basis with 10 levels, and 95 vertices on the coarsest level. Levels were determined using (1). In Figure 3 we show the convergence history of the multiplicative (symmetric Gauss-Seidel) hierarchical basis iteration, starting from a zero initial guess. The quantity plotted is

$$\sigma_k = \log \left\{ \frac{\| r_k \|}{\| r_0 \|} \right\},$$

where r_k is the residual at iteration k and $\| \cdot \|$ is the ℓ^2 norm. From the data points, we estimate by least squares that the convergence rate is approximately .44, which is fairly typical of this particular iterative method applied to a similar problem on a sequence of refined meshes.

We close with several remarks. First, the success of the method obviously depends rather crucially on the method for choosing tentative vertex parents, and the criteria which determines which vertex to eliminate next in the symbolic (graph) elimination phase of the algorithm. These are of course both hueristics, which are based on emulating the case of hierarchical basis for a sequence of refined meshes. At the moment, we do not think our hueristics are optimal, and we expect them to significantly improve as we gain further insights through the mathematical analysis of the iteration.

Second, we think that our scheme for choosing vertex parents, and that of the classical hierarchical basis multigrid method, are generally appropriate for self adjoint Laplace like operators. We anticipate that as we gain more experience with the method, variations more suitable for anisotropic problems or convection dominated problems will be developed. For example, one can imagine adding weights and/or directions to the edges in the graph and incorporating this information into the hueristics used to select vertex parents.

REFERENCES

[1] R. E. Bank, *Hierarchical preconditioners for elliptic partial differential equations*, in Proceedings of the SERC Summer Workshop in Numerical Analysis, 1992.

[2] ———, *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations, Users' Guide 7.0*, Frontiers in Applied Mathematics, SIAM, Philadelphia, 1994.

[3] R. E. Bank and T. F. Dupont, *Analysis of a two level scheme for solving finite element equations*, Tech. Report CNA-159, Center for Numerical Analysis, University of Texas at Austin, 1980.

[4] R. E. Bank, T. F. Dupont, and H. Yserentant, *The hierarchical basis multigrid method*, Numer. Math., 52 (1988), pp. 427–458.

[5] R. E. Bank, A. H. Sherman, and A. Weiser, *Refinement algorithms and data structures for regular local mesh refinement*, in Scientific Computing (Applications of Mathematics and Computing to the Physical Sciences) (ed. R. S. Stepleman), North Holland, 1983, pp. 3–17.

[6] J. Bramble, J. Pasciak, and J. Xu, *The analysis of multigrid algorithms with non-imbedded spaces or non-inherited quadratic forms*, Math. Comp., 56 (1991), pp. 1–43.

[7] V. Eijkhout and P. Vassilevski, *The role of the strengthened Cauchy-Buniakowskii-Schwarz inequality in multilevel methods*, SIAM Review, 33 (1991), pp. 405–419.

[8] A. George and J. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice Hall, Englewood Cliffs, NJ, 1981.

[9] W. Hackbusch, *Multigrid Methods and Applications*, Springer-Verlag, Berlin, 1985.

[10] W. Hackbusch and G. Wittum, eds, *Incomplete Decompositions - Algorithms, Theory and Applications*, Notes on Numerical Fluid mechanics, vol. 41, Vieweg, Braunschweig, 1993.

[11] R. H. W. Hoppe and R. Kornhuber, *Adaptive multilevel methods for obstacle problems*, SIAM J. Numer. Anal., (to appear).

[12] R. Kornhuber, *Monotone multigrid methods for variational inequalities I*, Numer. Math., (to appear).

[13] D. J. Rose, *A graph theoretic study of the numeric solution of sparse positive definite systems*, in Graph Theory and Computing, Academic Press, New York, 1972.

[14] J. Xu, *Theory of Multilevel Methods*, PhD thesis, Cornell University Report AM-48, 1989.

[15] ———, *Iterative methods by space decomposition and subspace correction*, SIAM Review, 34 (1992).

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

[17] S. Zhang, *Multilevel Iterative Techniques*, PhD thesis, Pennsylvania State University, Department of Mathematics Report 88020, 1988.