SUPPORT-GRAPH PRECONDITIONERS*

MARSHALL BERN†, JOHN R. GILBERT*, BRUCE HENDRICKSON‡, NHAT NGUYEN§, AND SIVAN TOLEDO¶

Abstract. We present a little-known preconditioning technique, called *support-graph* preconditioning, and use it to analyze two classes of preconditioners. The technique was first described in a talk by Pravin Vaidya, who did not formally publish his results. Vaidya used the technique to devise and analyze a class of novel preconditioners. The technique was later extended by Gremban and Miller, who used it in the development and analysis of yet another class of new preconditioners. This paper extends the technique further and uses it to analyze two classes of existing preconditioners: modified incomplete-Cholesky and multilevel diagonal scaling. The paper also contains a presentation of Vaidya's preconditioners, which was previously missing from the literature.

1. Introduction. This paper presents new applications of a little-known technique for constructing and analyzing preconditioners called *support-graph preconditioning*. The technique was first proposed and used by Pravin Vaidya [11], who described it in a talk in 1991, but did not publish a paper. Vaidya used the technique to design a family of novel preconditioners. Later, Gremban, Miller, and Zagha [5, 6] extended the technique and used it to construct another family of preconditioners. This paper explains the technique, extends it further, and uses it to analyze two classes of *known* preconditioners for model problems. Specifically, we use the extended technique to analyze certain modified-incomplete-Cholesky (MICC) preconditioners (see [8]) and multilevel-diagonal-scaling (MDS) preconditioners (see [10], for example).

The principle goal of this paper is to bring these techniques to the attention of a wider community of researchers. By doing so, we hope to encourage further work in this promising area. The primary original content of this paper, analyzing known preconditioners using the support-graph technique, serves several purposes. First, the analysis of MICC preconditioners establishes bounds that have never been proved before. Second, it shows that the technique is more widely applicable than previously appreciated. Third, we feel that the new proofs provide useful insights into these preconditioners; these insights can be used to improve the preconditioners and to guide heuristics for the construction of additional preconditioners.

A secondary goal of this paper is to provide a complete presentation of the support-graph technique and of Vaidya's preconditioners. Vaidya's important contribution has never been published. Although most of the theory that he uses is presented in Gremban's PhD thesis [6], Vaidya's preconditioners have not been described in any published form. We seek to rectify this situation. Our complete presenta-

*Submitted to SIAM J. Matrix Anal. Appl. This research was supported in part by DARPA contract number DABT63-95-C-0087 and by NSF contract number ASC-96-26298. Hendrickson and Nguyen were supported by the Applied Mathematical Sciences program, U.S. DOE, Office of Energy Research, and performed this work at Sandia National Labs, operated for the U.S. DOE under contract No. DE-AC04-94AL85000. Toledo was partially supported by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities (grant number 572/00 and grant number 9060/99) and by the University Research Fund of Tel-Aviv University. A preliminary version of this paper was presented at the Copper Mountain Conference on Iterative Methods, Copper Mountain, Colorado, March 30–April 3, 1998.

†Xerox Palo Alto Research Center.

‡Sandia National Laboratories.

§Stanford University.

¶Tel Aviv University. Part of this research was performed while Sivan Toledo was with the Xerox Palo Alto Research Center

tion of the support-graph technique is necessary since some important portions of the theory are missing from Gremban's thesis. Specifically, we present a formal proof of the Congestion-Dilation Lemma and we state stronger versions of some important lemmas. We also provide detailed constructions and complete proofs for Vaidya's preconditioners, which are missing from his 1991 manuscript.

The support-graph technique analyzes a preconditioner B for a matrix A by splitting both A and B into $A = A_1 + A_2 + \cdots + A_m$ and $B = B_1 + B_2 + \cdots + B_m$. Proving that $\tau B_i - A_i$ is positive semidefinite for all i shows that $\tau B - A$ is positive semidefinite and hence that the largest finite generalized eigenvalue of the matrix pair (A, B) is bounded by τ. The bound on the smallest generalized eigenvalue is proved by bounding the largest eigenvalue of (B, A) in the same way. The splittings of A and B are guided by their underlying graphs; often this allows us to reduce a complex problem to many problems with simple structures.

This paper has three main parts. The first part of the paper, §2 and §3, describes support-graph theory. The second part of the paper, §4 and §5, describes the preconditioners of Vaidya and of Gremban and Miller. The third part of the paper, §6 and §7, describes support-graph analysis of MICC and MDS preconditioners. Our conclusions from this research are presented in §8.

1.1. A Summary of the Results. This subsection summarizes the results in this paper. We start with a brief discussion of the strengths and weaknesses of the preconditioners of Vaidya and of Gremban and Miller. We also discuss the significance of our condition-number estimates for MICC and MDS preconditioners.

Vaidya proposed two classes of preconditioners. The first class, maximum-weight spanning-tree preconditioners, guarantee a condition-number bound of $O(n^2)$ for any $n \times n$ sparse diagonally-dominant symmetric matrices. They can be constructed and factored at insignificant cost using relatively simple graph algorithms.

Vaidya's second class of preconditioners is based on maximum-weight spanning trees augmented with a few extra edges. They can be constructed at insignificant cost using a slightly more complex algorithm than the first class. The cost of factoring these preconditioners depends on how many edges are added to the tree. Vaidya proposes that the factorization cost be balanced with the iteration costs, and he provides balancing guidelines for some classes of matrices. This class of preconditioners guarantees that the work in the linear solver is bounded by $O(n^{1.75})$ for any sparse M-matrix, and by $O(n^{1.2})$ for M-matrices whose underlying graphs are planar.

The strengths of Vaidya's preconditioners, especially of his second class, is that they are general, easy to construct and provide good condition-number bounds. For example, the work required to solve a model Poisson problem in 2D using Vaidya's preconditioner is $O(n^{1.2})$. This compares favorably with the $O(n^{1.25})$ work required for a solver based on a modified incomplete Cholesky. Coupled with the facts that Vaidya's preconditioners are guaranteed to work well on irregular problems, and that the only numerical assumption they make is that the matrix is an M-matrix, these are impressive results.

The main weaknesses of Vaidya's preconditioners are that they require a high-quality direct solver to factor the preconditioner, that balancing the preconditioner-factorization costs and the iteration costs may be a nontrivial task, and that they are not guaranteed to parallelize well.

The preconditioners that Gremban and Miller proposed are multilevel preconditioners. They are based on a hierarchical partitioning of the matrix, so they may be quite expensive to construct. The cost of preconditioning in every iteration is small, and the preconditioners parallelize well. The condition number of the preconditioned system is similar, for model problems, to the condition numbers offered by modified incomplete factorizations.

Thus, even on model problems, these preconditioners do not offer convergence rates as good as those of other multilevel preconditioners, like multigrid preconditioners. On the other hand, they are guaranteed to parallelize, so they may be preferable to incomplete factorizations on some computers. Gremban and Miller do not present condition number bounds for important classes of matrices other than regular grids with constant coefficients. For such problems their preconditioned systems have condition number bounds of $O(n \log n)$.

Our proofs of the condition-number bounds for modified incomplete factorizations are novel and significant. We show that the condition number for MICC for a model Laplace problem in 2D with either Dirichlet or Neumann or mixed boundary conditions is $O(\sqrt{n})$. The bound for a 3D model problem is $O(n^{1/3})$, but this only holds in the Dirichlet case and in some mixed boundary-condition cases.

To the best of our knowledge, these are the first such bounds for MICC *without diagonal perturbation*. The proofs are relatively simple, requiring only elementary linear algebra.

Our condition-number bound for a multilevel diagonal scaling preconditioner in one dimension is $O(\log n)$, which is good; but the analysis seems difficult to generalize to more realistic problems. Still, it provides some purely algebraic insight as to how this preconditioner works.

2. Basic Support Graph Theory. This section describes the basic linear-algebra tools that Vaidya and Gremban & Miller have developed to analyze their preconditioners. These preconditioners are for *M-matrices*, symmetric, diagonally dominant matrices with nonpositive off-diagonals. Vaidya and Gremban & Miller extended some of their results to symmetric diagonally dominant matrices with mixed off-diagonals. These extensions are specific to the preconditioners that they propose; their extensions and preconditioners are described in §4 and §5. Our own extensions are presented in §3.

The number of iterations of the Conjugate Gradient method for the solution of systems of linear equations $Ax = b$ is bounded above by the square root of the spectral condition number $\kappa(A)$ of A. (The actual number of iterations can be significantly smaller in some cases.) The condition number is the ratio of the extreme eigenvalues of A, $\kappa(A) = \lambda_{\max}(A)/\lambda_{\min}(A)$. The Conjugate Gradient method can be used to solve consistent linear systems with a singular coefficient matrix A when a basis for the null space of A is known. In such cases, the number of iterations is proportional to square root of the ratio of the extreme positive eigenvalues. When a preconditioner B is used in the Conjugate Gradient method, the number of iterations is proportional to the the square root of the ratio of the extreme finite generalized eigenvalues of the pair (A, B), defined below.

DEFINITION 2.1. *The number λ is a finite generalized eigenvalue of the matrix pencil (A, B) if there exists a vector x such that $Ax = \lambda Bx$ and $Bx \neq 0$. We denote the set of finite generalized eigenvalues by $\lambda_f(A, B)$.*

Henceforth whenever we refer to an "eigenvalue" of a matrix pencil, we mean a finite generalized eigenvalue.

To bound the amount of work in the Preconditioned Conjugate Gradient method, we need to bound the finite eigenvalues of (A, B). We need to prove two bounds: an upper bound on $\max \lambda_f(A, B)$ and a lower bound on $\min \lambda_f(A, B)$. We will prove the upper bound directly and the lower bound by proving an upper bound on $\max \lambda_f(B,A) = 1/\min \lambda_f(A,B)$. We therefore only need to show how to prove upper bounds on the $\lambda_f(A,B)$, since the lower bound is proved in essentially the same way for the matrix pencil (B,A).

2.1. The Support Lemma: Bounding Eigenvalues of Matrix Pencils. The main tool that we use to bound $\lambda_f(A,B)$ is the so-called *support* of (A,B), which is the smallest number τ such that $\tau B - A$ is positive semidefinite. Informally, we think of τ as the number of copies of B required to "support" the action of A. If τ is small, B supports A well; if τ is large, B supports A weakly. We denote the support of (A,B) by $\sigma(A,B)$,

$$\sigma(A,B) = \min\{\tau : \tau B - A \text{ is positive semidefinite}\} \; .$$

If there is no τ for which $\tau B - A$ is positive semidefinite, then we take $\sigma(A,B) = \infty$.

The following lemma shows that the support of a pencil bounds its eigenvalues. The lemma is used implicitly by Vaidya without a proof. Gremban states the lemma and gives a proof [6, Lemma 4.4]. We state it under a weaker hypothesis than Gremban, which is nevertheless strong enough for Gremban's proof. A more general version of this lemma can be found in Axelsson [1, Theorem 10.1].

LEMMA 2.2 (Support Lemma [6]). *If $\lambda \in \lambda_f(A,B)$ where B is positive semidefinite and $\mathrm{null}(A) \subseteq \mathrm{null}(B)$, then $\lambda \le \sigma(A,B)$.*

Some pairs of matrices do not have a finite support $\sigma(A,B)$. Let

$$A = \begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix} \quad \text{and let} \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \; .$$

We have $\lambda_f(A,B) = \{\sqrt{2}/2\}$, but $\tau B - A$ has a negative eigenvalue for all τ. (Lemma 2.12 shows how to bound the extreme eigenvalues in some of these cases.) If a pair of matrices has finite support, however, then the Support Lemma is tight.

LEMMA 2.3. *If $\sigma(A,B)$ is finite, then*

$$\sigma(A,B) \in \lambda_f(A,B) \; .$$

Proof. The matrix $\sigma(A,B) \cdot B - A$ has a zero eigenvalue since the eigenvalues of $\tau B - A$ are continuous in τ. Therefore, there is a vector x such that

$$(\sigma(A,B) \cdot B - A)\, x = 0$$

or

$$Ax = \sigma(A,B) \cdot Bx \; .$$

□

We use the Support Lemma to prove an upper bound τ on $\lambda_f(A,B)$ by proving that $\tau B - A$ is positive semidefinite. Much of the rest of the theory consists of tools to prove that a matrix is positive semidefinite.

2.2. The Splitting Lemma: Proving Semidefiniteness by Decomposition. One way to prove that a matrix is positive semidefinite is to split it into a sum of matrices and prove that each term is positive semidefinite. This lemma too was implicitly used by Vaidya and stated and proved in Gremban's thesis [6, Lemma 4.7].

LEMMA 2.4 (Splitting Lemma). *Let $Q = Q_1 + Q_2 + \cdots + Q_m$, where $Q_1, Q_2, \ldots, Q_m$ are all positive semidefinite. Then Q is positive semidefinite.*

We first use the Splitting Lemma to reduce the problem of preconditioning symmetric diagonally dominant matrices to the problem of preconditioning symmetric matrices with zero row sums.

LEMMA 2.5. *Let A be a symmetric diagonally dominant matrix and let A' be the matrix with the same off-diagonal entries but with zero row sums. Let B' be a preconditioner for A' such that both $\beta B' - A'$ and $\alpha A' - B'$ are positive semidefinite and $\alpha, \beta \geq 1$. Let $B = B' + A - A'$ be a preconditioner for A (B has the same off-diagonal entries as B' and the same row sums as A). Then $\beta B - A$ and $\alpha A - B$ are positive semidefinite.*

Proof. We have

$$\begin{aligned}\beta B - A &= \beta(B' + A - A') - A \\ &= (\beta B' - A') + (\beta - 1)(A - A') \ .\end{aligned}$$

Both terms in the last sum are positive semidefinite: the first by the hypothesis, and the second since it is a nonnegative scalar multiple of a diagonal matrix with nonnegative entries. Similarly,

$$\begin{aligned}\alpha A - B &= \alpha A - (B' + A - A') \\ &= (\alpha A' - B') + (\alpha - 1)(A - A')\end{aligned}$$

is positive semidefinite. □

The conditions $\alpha, \beta \geq 1$ do not limit the applicability of the lemma since the condition number is 1 or more. Therefore, if either α or β is less than 1, we scale B' without changing $\alpha\beta$, which is our bound on the condition number of the preconditioned system.

Using this lemma, we assume from now on that both A and B have zero row sums.

2.3. The Congestion-Dilation Lemma: Splitting by Paths in the Graph. Vaidya and Gremban & Miller split $\tau B - A$ in a special way to prove that it is positive semidefinite. We assume that A and B are symmetric. Given a symmetric matrix A, we define its underlying graph.

DEFINITION 2.6. *The* underlying graph $G_A = (V_A, E_A)$ *of an n-by-n symmetric matrix A is a weighted undirected graph whose vertex set is $V_A = \{1, 2, \ldots, n\}$ and whose edge set is $E_A = \{(i,j) : i \neq j, A_{i,j} \neq 0\}$. The weight of an edge (i,j) is $A_{i,j}$. The weight of a vertex i is the sum of elements in row i of A.*

Let G_A be the undirected weighted graph underlying $-A$ and G_B the graph underlying $-B$. Since both A and B have zero row sums, the graph structure and the edge weights determine the matrices exactly, since all the vertex weights are 0. If the off-diagonal elements of A and B are all negative, then the edge weights in G_A and G_B are positive. Vaidya and Gremban & Miller interpret such graphs as resistive networks where the edge weight is the conductance of a resistor. They split $\tau B - A$ into $(\tau B_1 - A_1) + (\tau B_2 - A_2) + \cdots + (\tau B_m - A_m)$ such that each A_i corresponds to exactly one edge in G_A, and each B_i corresponds to a simple path in G_B. Both the A_i's and the B_i's have nonpositive off-diagonals and zero row sums. Each A_i represents the entire weight of one edge, and each corresponding B_i represents a path that can contain fractions of edge weights. The endpoints of the path represented by B_i are the endpoints of the edge represented by A_i. An example of such a splitting is

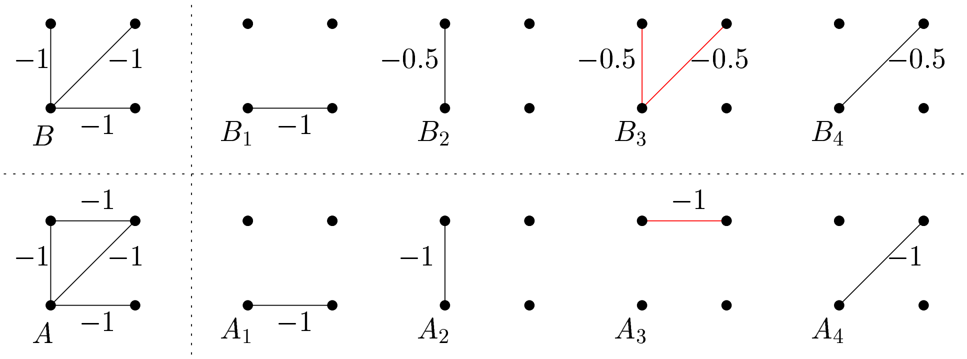

FIG. 2.1. *A graph representation of a splitting of* $A = A_1 + \cdots + A_4$ *and* $B = B_1 + \cdots + B_4$ *such that each* A_i *is represents a single edge and each* B_i *is a path that supports the edge* A_i*. This splitting proves that* $\lambda_f(A, B) \le \sigma(A, B) \le 4$ *since the worst congestion-dilation product is* $2 \cdot 2 = 4$.

shown in Figure 2.1. Both Vaidya and Gremban & Miller use the Congestion-Dilation lemma, which they neither state nor prove, to show that each term $\tau B_i - A_i$ is positive semidefinite.

We prove the Congestion-Dilation Lemma in three steps.

LEMMA 2.7. *Let*

$$A = \begin{pmatrix} a & 0 & \cdots & 0 & -a \\ 0 & 0 & & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & 0 & 0 \\ -a & 0 & \cdots & 0 & a \end{pmatrix}$$

and

$$B = \begin{pmatrix} a & -a & & & \\ -a & 2a & -a & & \\ & & \cdots & & \\ & & -a & 2a & -a \\ & & & -a & a \end{pmatrix}$$

be $(k+1)$*-by-*$(k+1)$ *matrices with* $a > 0$*. Then* $kB - A$ *is positive semidefinite.*

Proof. We prove that $kB - A$ is positive semidefinite by showing that the matrix

$$C = (1/a)(kB - A) = \begin{pmatrix} k-1 & -k & & & 1 \\ -k & 2k & -k & & \\ & & \cdots & & \\ & & -k & 2k & -k \\ 1 & & & -k & k-1 \end{pmatrix}$$

is positive semidefinite.

We show by induction that C is positive semidefinite by performing symmetric Gaussian elimination on rows/columns 2 through $k-1$. The inductive claim is that after we eliminate row and column i (or before we eliminate $i+1$, when $i = 1$), the matrix C becomes

$$\begin{pmatrix} -1+k/i & & & & & -k/i & & & 1 \\ & 2k & & & & & & & \\ & & 3k/2 & & & & & & \\ & & & & & & & & \\ & & & & ik/(i-1) & & & & \\ -k/i & & & & & (i+1)k/i & -k & & \\ & & & & & -k & 2k & & \\ & & & & & & & \ddots & \\ 1 & & & & & & & & 2k \end{pmatrix}.$$

The claim is true before we eliminate row and column 2, since in that case $i = 1$. Assume that the claim is true after we eliminate i but before $i+1$ The elimination of row and column $i+1$ modifies three entries in C: $C_1 1$, $C_{i+2,i+2}$, and $C_{1,i+2} = C_{i+2,1}$. The new values are

$$C_{11} = -1 + \frac{k}{i} - \frac{(-k/i)(-k/i)}{(i+1)k/i} = -1 + \frac{k}{i+1} ,$$

$$C_{i+2,i+2} = 2k - \frac{-k \cdot -k}{(i+1)k/i} = \frac{k(i+2)}{i+1} ,$$

and

$$C_{1,i+2} = 0 - \frac{-k(-k/i)}{(i+1)k/i} = \frac{k}{i+1} ,$$

which proves the inductive claim

Therefore, after we eliminate row k the 2×2 submatrix consisting of the first and last row and column becomes

$$\begin{pmatrix} -1+k/k & 1-k/k \\ 1-k/k & -1+k/k \end{pmatrix} = 0$$

and the remainder of the matrix is positive diagonal, so C is positive semidefinite. □

The combinatorial interpretation of Lemma 2.7 is that A represents a single edge with weight a and B represents a path with the same endpoints and consisting of edges of weight b. The lemma states that the support of A in B is k, the *dilation* of the edge in the path, or simply the length of the path.

The next lemma is slightly more general.

LEMMA 2.8. *Let*

$$A = \begin{pmatrix} a & 0 & \cdots & 0 & -a \\ 0 & 0 & & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & 0 & 0 \\ -a & 0 & \cdots & 0 & a \end{pmatrix}$$

and

$$B = \begin{pmatrix} b & -b & & & \\ -b & 2b & -b & & \\ & & \cdots & & \\ & & -b & 2b & -b \\ & & & -b & b \end{pmatrix}$$

be $(k+1)$-by-$(k+1)$ matrices with $a>0$, $b>0$. Then $(k\cdot a/b)B-A$ is positive semidefinite.

Proof. This case reduces by scaling to Lemma 2.7. □

This lemma states that in the more general case in which the weight of the edge represented by A and the weight of the edges of the path represented by A are not the same, the support is the dilation k multiplied by the ratio a/b of the edge weights.

Finally, we state and prove the full Congestion-Dilation Lemma.

LEMMA 2.9 (Congestion-Dilation Lemma). *Let*

$$A=\begin{pmatrix} a & 0 & \cdots & 0 & -a \\ 0 & 0 & & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & 0 & 0 \\ -a & 0 & \cdots & 0 & a \end{pmatrix}.$$

be a $(k+1)$-by-$(k+1)$ matrix with $a>0$ and let

$$B=\begin{pmatrix} d_1 & -b_1 & & & \\ -b_1 & d_2 & -b_2 & & \\ & & \cdots & & \\ & & -b_{k-1} & d_k & -b_k \\ & & & -b_k & d_{k+1} \end{pmatrix}$$

be a matrix with zero row sums, and with $d_i, b_i > 0$ for all i. Then $(k\cdot a/\min(b_i))B-A$ is positive semidefinite.

Proof. Let $b=\min(b_i)$. We split B into

$$B=B_1+B_2=\begin{pmatrix} b & -b & & & \\ -b & 2b & -b & & \\ & & \cdots & & \\ & & -b & 2b & -b \\ & & & -b & b \end{pmatrix}$$
$$+\begin{pmatrix} d_1-b & -b_1+b & & & \\ -b_1+b & d_2-2b & -b_2+b & & \\ & & \cdots & & \\ & & -b_{k-1}+b & d_{k-1}-2b & -b_k+b \\ & & & -b_k+b & d_k-b \end{pmatrix}.$$

The matrix B_2 is symmetric, diagonally dominant, and has nonpositive off-diagonals, so it is positive semidefinite. We have

$$\begin{aligned}(k\cdot a/\min(b_i))B-A &= (k\cdot a/b)B-A \\ &= [(k\cdot a/b)B_2]+[(k\cdot a/b)B_1-A]\ .\end{aligned}$$

The first term is positive semidefinite since B_2 is positive semidefinite, and the second term is positive semidefinite by Lemma 2.8. Therefore, the sum is positive semidefinite. □

The combinatorial interpretation of the Congestion-Dilation Lemma is that $a/\min(b_i)$ is the *congestion* of the edge represented by A in the path represented by B, and k is the *dilation* of the edge. In the example depicted in Figure 2.1 the congestion of the edge A_3 in the path B_3 is 2 and the dilation is 2, for example.

The proof shows that the congestion-dilation bound $k \cdot (a/\min(b_i))$ on $\sigma(A,B)$ is tight only when all the edges along the path have the same weights; at the other extreme when one edge has small weight b and the rest have very large weights, the actual support $\sigma(A,B)$ is closer to a/b than to $k \cdot (a/b)$.

The Support, Splitting, and Congestion-Dilation Lemmas are the only linear-algebra tools that Vaidya uses in his construction. Given an M-matrix A, Vaidya constructs a preconditioner B whose underlying graph G_B consists of a subset of the edges of G_A and the same set of vertices. Vaidya uses the lemmas above to bound the condition number of the preconditioned system. He splits G_B into paths that support each edge of G_A. Since G_B is a subset of G_A, G_A supports the edges of G_B with paths of length 1 and no congestion, so the small eigenvalue of (A,B) is at least 1. The bound that Vaidya obtains, therefore, is the worst congestion-dilation product for the edges of G_A. The specific constructions that Vaidya proposes are described in §4.

2.4. The Clique-Star Lemma. Gremban and Miller introduce another way of bounding the support of one simple matrix by another. The matrix A being supported represents a fully-connected subgraph, or a *clique*, of size k, and the supporting matrix B represents a k-edge star whose endpoints coincide with the members of the clique.

LEMMA 2.10 (Clique-Star Lemma). *Let*

$$A = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & (k-1)a & -a & \cdots & -a \\ 0 & -a & (k-1)a & \cdots & -a \\ \vdots & & \ddots & & \vdots \\ 0 & -a & -a & \cdots & (k-1)a \end{pmatrix}$$

and

$$B = \begin{pmatrix} kb & -b & -b & \cdots & -b \\ -b & b & 0 & \cdots & 0 \\ -b & 0 & b & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ -b & 0 & 0 & \cdots & b \end{pmatrix}$$

be $(k+1)$-by-$(k+1)$ matrices with $a>0$, $b>0$. Then $(k \cdot a/b)B - A$ is positive semidefinite.

Proof. Let $C = (k \cdot a/b)B - A$. We have

$$\begin{aligned} \frac{1}{a}C &= \begin{pmatrix} k^2 & -k & -k & \cdots & -k \\ -k & k-(k-1) & 1 & \cdots & 1 \\ -k & 1 & k-(k-1) & \cdots & 1 \\ \vdots & & \ddots & & \vdots \\ -k & 1 & 1 & \cdots & k-(k-1) \end{pmatrix} \\ &= \begin{pmatrix} k^2 & -k & -k & \cdots & -k \\ -k & 1 & 1 & \cdots & 1 \\ -k & 1 & 1 & \cdots & 1 \\ \vdots & & \ddots & & \vdots \\ -k & 1 & 1 & \cdots & 1 \end{pmatrix}. \end{aligned}$$

After one step of symmetric Gaussian elimination on the first row and column, the matrix $(1/a)C$ befomes

$$\begin{pmatrix} k^2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} ,$$

which is clearly positive semidefinite. Hence, C is positive semidefinite. $\square$

This lemma provides a stronger bound on the support than the bound that results from splitting the clique into edges and the star into 2-edge paths that support the edges.

2.5. Gaussian Elimination and Generalized Support: Preconditioning in a Larger Space. Gremban and Miller construct their preconditioners in a space of higher dimension (or, on a graph with more vertices) than the original matrix. They introduced a few more linear-algebra tools into the support-graph theory to deal with this.

The next lemma, stated under a stronger hypothesis by Gremban [6, Lemma 4.9], shows that $\lambda_f(A,B)$ is invariant under non-singular transformations that are applied to both A and B.

LEMMA 2.11. *If G and H are nonsingular matrices (not necessarily symmetric) then*

$$\lambda_f(A,B) = \lambda_f(GAH, GBH) \ .$$

We define the *generalized support* $\bar{\sigma}(A,B)$ of (A,B) as

$$\bar{\sigma}(A,B) = \min\left\{\tau : x^T(\tau B - A)x \geq 0 \text{ for all } x \text{ such that } Ax \neq 0 \text{ and } Bx \neq 0\right\} \ .$$

Roughly speaking, $\bar{\sigma}(A,B)$ measures how well B supports A outside their null spaces. Gremban does not use this lemma but a similar one that is tailored more precisely to his technique. We state and prove here the more general case, which is a stronger version of the Support Lemma.

LEMMA 2.12. *If $\lambda \in \lambda_f(A,B)$ where $A \neq 0, B \neq 0$ are symmetric positive semidefinite, then $\lambda \leq \bar{\sigma}(A,B)$.*

Proof. Let $\tau = \bar{\sigma}(A,B)$. Note that the assumption $A \neq 0$ implies $\tau > 0$. Assume for contradiction that there is a $\lambda \in \lambda_f(A,B)$ such that $\lambda > \tau$, and let $\epsilon = \lambda - \tau > 0$. Let y be an eigenvector corresponding to λ. We have $Ay = \lambda By$, so $y^T(A - \lambda B)y = 0$. By definition of $\lambda_f()$, $By \neq 0$. If $By \neq 0$ and $Ay = 0$, then $\lambda = 0 < \tau$, a contradiction. So neither Ay nor By can be zero. By the definition of τ we have

$$\begin{aligned} 0 &\leq y^T(\tau B - A)y \\ &= y^T((\lambda - \epsilon)B - A)y \\ &= y^T(\lambda B - A)y - y^T(\epsilon B)y \\ &= -y^T(\epsilon B)y \ . \end{aligned}$$

Since B is symmetric positive semidefinite and $By \neq 0$, $y^T By > 0$. So $-y^T(\epsilon B)y < 0$, which is a contradiction. $\square$

These two lemmas allow us to use a preconditioner in a space of higher dimension than the original matrix. We embed the original k-by-k matrix A_{11} in an n-by-n matrix A, where n is the order of the preconditioner B.

$$A = \begin{pmatrix} A_{11} & 0 \\ 0 & 0 \end{pmatrix} \quad , \quad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{12}^T & B_{22} \end{pmatrix} . \tag{2.1}$$

We cannot use congestion-dilation arguments directly to bound $\sigma(B,A)$; since the underlying graph of A has $n-k$ disconnected vertices, so paths in A cannot support all the edges of B.

Instead, we use Lemma 2.11 to reduce (2.1) to a simpler case

$$A = \begin{pmatrix} A_{11} & 0 \\ 0 & 0 \end{pmatrix} \quad , \quad \tilde{B} = \begin{pmatrix} \tilde{B}_{11} & 0 \\ 0 & \tilde{B}_{22} \end{pmatrix} \tag{2.2}$$

using Gaussian elimination on the last $n-k$ rows and columns of B. Note that the Gauss transformations must also be applied to A, but they have no effect on it. We complete the analysis of (2.2) using Lemma 2.12. For any τ, the space R^n can be decomposed into two orthogonal subspaces that are invariant under $\tau A - B$. Vectors in one subspace V_1 (represented in the standard basis) have nonzeros only in the first k elements, and vectors in the other subspace V_2 have nonzeros only in the last $n-k$ elements. The subspace V_2 is contained in the null space of A. Therefore, to prove that $\lambda_f(B,A)$ is bounded by τ, we only need to show that $x^T(\tau A - B)x \geq 0$ for $x \in V_1$, which is equivalent to showing that $\tau A_{11} - B_{11}$ is positive semidefinite.

We will see in §5 how Gremban and Miller use this technique to analyze the condition number of his preconditioners. The main drawback of this technique is that the elimination of the last $n-k$ rows and columns of B can significantly fill the leading k-by-k block of B. Unless B is particularly simple, this fill is difficult to analyze. We propose in the next section an alternative technique that leads to a simpler analysis of some preconditioners, since it does not require a complete elimination of the trailing block of B.

3. Support Graph Theory: Extensions.

We now describe *new* tools that extend the support-graph theory developed by Vaidya and Gremban & Miller. (Lemma 2.12 too is an extension of a result of Gremban and Miller's). In particular, these tools enable or simplify the analysis of preconditioners with both positive and negative off-diagonal entries using support-graph theory. The results presented in §3.2 were also derived independently by Guattery [7].

3.1. Stepwise Gaussian Elimination.

The first technique allows us to analyze support graphs that are larger than A, like Gremban and Miller's preconditioners, but without performing a complete elimination of the extra vertices that are in B but not in A. Our technique relies on the following two lemmas. The first lemma is a version of Lemma 2.11 but for generalized support rather than eigenvalues.

LEMMA 3.1. *Let G be a nonsingular matrix. Then $\bar{\sigma}(A,B) = \bar{\sigma}(G^TAG, G^TBG)$.*

Proof. Let $\tau = \bar{\sigma}(A,B)$. By the definition of $\bar{\sigma}(A,B)$ we have

$$x^T(\tau G^TBG - G^TAG)x = (Gx)^T(\tau B - A)(Gx) \geq 0$$

for all x such that $A(Gx) \neq 0$ and $B(Gx) \neq 0$. But $A(Gx) \neq 0$ if and only if $(G^TAG)x \neq 0$, and similarly for G^TBG. Therefore, we have

$$x^T(\tau G^TBG - G^TAG)x \geq 0$$

for all x such that $G^TAGx \neq 0$ and $G^TBGx \neq 0$, so $\bar{\sigma}(G^TAG, G^TBG) \leq \tau = \bar{\sigma}(A, B)$. The opposite inequality is proved in the exactly the same way. □

The second lemma shows that we can subtract certain positive semidefinite matrices from A without increasing $\bar{\sigma}(A, B)$. Subtracting a positive semidefinite matrix C from A makes A easier to support, provided C's null space includes A's.

LEMMA 3.2 (Shifting Lemma). *Let A, B, and C be positive semidefinite matrices such that* $\mathrm{null}(A) \subseteq \mathrm{null}(C)$. *Then*

$$\bar{\sigma}(A - C, B) \leq \bar{\sigma}(A, B) \ .$$

Proof. Let $\tau = \bar{\sigma}(A, B)$. By the definition of $\bar{\sigma}(A, B)$ we have

$$x^T(\tau B - A)x \geq 0$$

for all x such that $Ax \neq 0$ and $Bx \neq 0$. Therefore, we also have

$$x^T(\tau B - (A - C))x = x^T(\tau B - A)x + x^TCx \geq 0$$

for all x such that $Ax \neq 0$ and $Bx \neq 0$. Assume for contradiction that

$$x^T(\tau B - (A - C))x < 0$$

for some x such that $(A - C)x \neq 0$ and $Bx \neq 0$. Since $x^TCx \geq 0$ for all x, we must have

$$x^T(\tau B - A)x < 0 \ ,$$

so $Ax = 0$. But $\mathrm{null}(A) \subseteq \mathrm{null}(C)$ implies $Cx = 0$, contradicting $(A - C)x \neq 0$. Hence,

$$\bar{\sigma}(A - C, B) \leq \tau = \bar{\sigma}(A, B) \ .$$

□

These lemmas allow us to reduce (2.1) to (2.2) in a series of phases. In each phase, we perform one or more steps of Gaussian elimination on A, followed by a subtraction of a negative semidefinite matrix from A (addition of a positive semidefinite matrix). After all of these steps are complete, we prove a bound on the generalized support of the resulting matrices. These lemmas allow us to then retrace the steps that we have taken, performing transformations that reverse the effects of elimination steps and subtracting positive semidefinite matrices, all without changing the bound on the generalized support. Thus, the bound that we prove on the matrices after this series of transformation is also a bound on the original matrix and preconditioner.

3.2. Positive Offdiagonal Elements. Positive off-diagonal entries in B require a modification to the splitting strategy. Recall that the "canonical" use of the Splitting Lemma is to split $\tau B - A$ into $(\tau B_1 - A_1) + \cdots + (\tau B_m - A_m)$, where the B_i's are positive semidefinite. When the Congestion-Dilation Lemma is used, the B_i's are usually paths of negative edges, and the A_i's are negative edges, all with zero row sums (positive row sums can be handled separately as shown in Lemma 2.5). When B has positive off-diagonal entries, this strategy must be modified. One way to prove an upper bound on $\bar{\sigma}(A, B)$ in this case using the congestion-dilation framework is to support both A and the positive edges of B with the negative edges of B. We split $\tau B - A$ into

$$\begin{aligned} \tau B - A = (\tau B_1 - A_1) + \cdots + (\tau B_m - A_m) \\ +(\tau B_{m+1} + \tau B_{m+k+1}) + \cdots + (\tau B_{m+k} + \tau B_{m+2k}) \ , \end{aligned}$$

where each of B_{m+k+1} through B_{m+2k} represents a single positive edge, and B_1 through B_{m+k} represent paths of negative edges. Thus, B_1 through B_m support edges of A, while B_{m+1} through B_{m+k} *support the positive edges of* B.

An important point is that, in this case, increasing τ does not necessarily make all the terms positive semidefinite. Indeed, if $\tau B_{m+j} + \tau B_{m+k+j}$ is indefinite or negative definite, then it remains so for all $\tau \geq 0$. In other words, each positive edge of B must be supported by a path with support at most 1.

We can make the analysis simpler when the precoditioner B can be represented as $B = A - R$ where R is also positive semidefinite. Some common preconditioners that are produced by an incomplete factorization process can be represented in this way, as explained in §6. The following lemma shows how to simplify the analysis.

LEMMA 3.3. *Let $B = A - R$ such that A, B, and R are positive semidefinite. If $\bar{\sigma}(R, A) = \tau' < 1$, then $\kappa(B^{-1}A) \leq 1/(1-\tau')$.*

Proof. Let $\tau = 1/(1-\tau')$. The matrix

$$\begin{aligned} \tau B - A &= \tau A - \tau R - A \\ &= (\tau - 1)A - \tau R \\ &= \frac{\tau'}{1-\tau'} A - \frac{1}{1-\tau'} R \end{aligned}$$

is positive semidefinite since $\bar{\sigma}(R, A) = \tau'$, so $\bar{\sigma}(A, B) \leq \tau$. We also have $\bar{\sigma}(B, A) \leq 1$, since $A - B = A - (A - R) = R$ is positive semidefinite. □

In such cases, the lemma can be interpreted as an application of the strategy described in the previous paragraph. We use a τ' fraction of the negative edges of B to support the positive edges. The negative edges of B are exactly the edges of A. Therefore, we use a $1-\tau'$ fraction of each edge of A to support itself, giving a support bound of $1/(1-\tau')$.

4. Vaidya's Preconditioners. In this section we describe the two families of combinatorial preconditioners developed by Vaidya [11]. Vaidya's first family applies to all symmetric, diagonally dominant matrices; the second family applies only to M-matrices, but Vaidya suggests that it can be extended to all symmetric diagonally dominant matrices. We begin the discussion by assuming that A is an $n \times n$ M-matrix and describe the extension to symmetric diagonally dominant matrices later. Let m be the number of off-diagonal nonzeros in A. If A has rows with positive row sums we increase the diagonal elements of the preconditioner B so that A and B have the same row sums. As shown in Lemma 2.5 in §2, this transformation does not change the condition-number estimates (although it may change the condition number itself).

Both families use preconditioners B whose underlying graphs G_B are subgraphs of G_A (same set of vertices and a subset of the edges), so we can always support an edge of G_B by the corresponding edge of G_A. Therefore, the congestion-dilation bound for $\sigma(B, A)$, and hence for $\max \lambda_f(B, A)$, is 1.

Vaidya's first preconditioner is constructed by finding a maximum-weight spanning tree T in G_A. In other words, T is a connected graph with no cycles (i.e., a spanning tree), and the total weight of its edges is maximal among all spanning trees

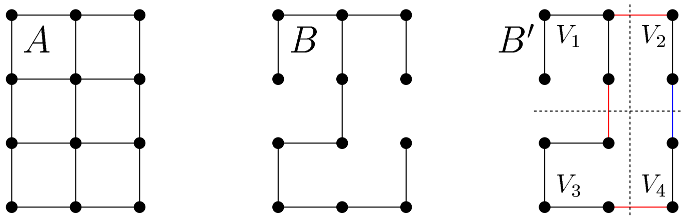

FIG. 4.1. *A graph G_A, a spanning tree G_B of G_A (middle), and a spanning tree augmented with extra edges ($G_{B'}$, left). The augmentation is performed by cutting the tree into subgraphs $V_1, \ldots, V_4$ of roughly equal size, and adding the heaviest edge between each pair of subgraphs.*

of G_A. As illustrated in Figure 4.1, rhe preconditioner B is the M-matrix whose underlying graph is $G_B = T$, and whose row sums are identical to those of A.

Let us analyze the congestion and dilation in T for an edge e of G_A. Since T is a maximum-weight spanning tree, there is exactly one path in T between the endpoints of e. And furthermore, all the edges along the path have heavier edges than e. There are at most $m/2$ edges in G_A, where n is the order of A, so T is split into at most m paths. We allocate a $2/m$ fraction of the weight of each edge of T to every path, so the congestion of an edge-path pair is at most $2/m$. The maximum length of a path is $n-1$, so the dilation is at most $n-1$. Hence, the congestion-dilation product for edge-path pairs is at most $m(n-1)/2 = O(mn)$. By Lemmas 2.4 and 2.9, $\sigma(A, B) \leq O(mn)$, and by Lemma 2.2, $\lambda \leq O(mn)$ for any $\lambda \in \lambda_f(A, B)$. Since the smallest positive eigenvalue of (A, B) is at least 1, we have $\lambda_f(A, B) \in [1, O(mn)]$.

Computing B takes at most $O(m \log n)$ work, using an efficient minimum-weight spanning tree algorithm. Since the underlying graph of B is a tree, B can be factored in time $O(n)$ without producing any fill. Consequently, the costs associated with constructing and factoring the B are insignificant relative to the cost of the iterative linear solver, and the cost of applying it in every iteration is $O(n)$, which is no more expensive than multiplying by A.

The maximum-weight spanning tree preconditioners can be extended to handle any symmetric diagonally dominant matrix by taking G_B to be a maximum-weight basis for G_A rather than a maximum-weight spanning tree. A maximum-weight basis is a generalization of the maximum spanning tree; see [3, Section 17.4 and Problem 17-2] for background. We omit the details from this paper.

Vaidya's second family of preconditioners achieves a better condition number, but it is also more expensive to compute and apply. The construction, also illustrated in Figure 4.1, starts with the same maximum-weight spanning tree T. Let t be an integer parameter. We decompose G_A into a set of t subgraphs $V_1, V_2, \ldots, V_t$ such that each subgraph is spanned by a connected subgraph of T and has at most m/t vertices. We form G_B by adding to T the heaviest edge between V_i and V_j for all i and j (we add nothing if there are no edges between V_i and V_j or if the heaviest edge is already in T). To analyze this preconditioner, we need the additional assumption that no row in A has more than $d+1$ nonzeros for some constant d, which implies that $m \leq dn$. We decompose the augmented tree G_B into a set of paths as follows. If both endpoints of an edge $e \in G_A$ are in the same V_i, we use the single path in T that connects them. If one endpoint belongs to V_i and the other to V_j, the path uses T to get from one endpoint of e to the heaviest edge that connects V_i to V_j, then the path uses this edge, and finally it uses T to get to the other endpoint of e. Again, the edges along a path are all heavier than the edge that is supported by the path. Since the paths now have length at most $1 + (2dn/t)$, and since each edge in G_B carries at most d^2n/t paths, the condition number of (A, B) is bounded by $O(n^2/t^2)$.

What is the cost of factoring B? Let us denote the endpoints of the edges that connect V_i with the other V_j's by U_i. Since the V_i's are disjoint, we have $U_i \cap U_j = \emptyset$. We begin the factorization of B by eliminating all the degree-1 and degree-2 vertices in B, until all the remaining vertices have degree greater than 2. This phase, which we refer to as *contraction*, requires only $O(n)$ work and generates only $O(n)$ fill elements. Once this is done, what remains of each V_i is a tree with no vertices of degree 1 or 2, and whose leaves are all in U_i (these are leaves in V_i, but not in B). It follows that the number of non-leaf vertices in V_i is at most $|U_i|$. Hence, the total number of vertices in the contracted graph is at most $2(|U_1| + \cdots + |U_t|)$. We now factor the contracted graph, exploiting whatever structure it has; when it is planar, for example, we can use nested dissection to factor it. Hence, the total cost of factoring B is $O(n)$ plus the cost of factoring the contracted graph.

In the worst case, each subgraph V_i has a connection to all others, and has $(t-1)$ vertices in U_i. In this case, the contracted graph has no more than $2 \cdot t(t-1)$ vertices, so factoring it requires at most $O((t^2)^3)$ work.

When G_A is planar, G_B is planar and so is the contracted graph. Furthermore, when G_A is planar, the contracted graph has only $O(t)$ vertices. This can be proved by contracting each V_i to a single vertex, which still preserves planarity. Since G_B has at most one edge between between V_i and V_j, these edges do not disappear and are not merged into other edges in the contraction process. This super-contracted planar graph has only t vertices, so it has only $O(t)$ edges. But each one of the edges between the original subgraphs V_i, V_j is still represented in the super-contracted graph, so there are only $O(t)$ of them in B, which proves that there are only $O(t)$ vertices in $|U_1| + \cdots + |U_t|$. The factors of a matrix who whose underlying graph is a planar graph with $O(t)$ vertices have $O(t \log t)$ nonzeros and the factorization can be performed in $O(t^{3/2})$ time.

The following lemma summarizes the discussion above. The result is used in Vaidya's manuscript, but without a proof.

LEMMA 4.1. *The cost of factoring the augmented-maximum-weight-spanning-tree preconditioner B is $O(n + t^6)$ when A is a general M-matrix, and $O(n + t^{1.5})$ when A is planar. The factors of B have $O(n + t^4)$ nonzeros in the general case, and $O(n + t \log t)$ when A is planar.*

The choice of t should balance the costs of constructing and factoring B with the cost of the iterations, which is determined by both the condition number and the cost of applying B. The cost of constructing B is again insignificant. If A has no special nonzero structure (beyond a bound of $d + 1$ nonzeros per row), then the optimal t is $\Theta(n^{0.25})$. Factoring the preconditioner costs $O(t^6) = O(n^{1.5})$. The number of iterations is bounded by $O(\sqrt{n^2/t^2}) = O(n/t)$, and the cost of every iteration is $O(n + t^4)$, so the total cost is $O(n^2/t + nt^3) = O(n^{1.75})$. When A is planar, the cost of factoring B is $O(n + t^{1.5})$, and the cost of every iteration is $O(n + t \log t)$. The cost is minimized near $n^2/t = t^{1.5}$, or $t = n^{0.8}$. The total cost to solve the linear system is $O(n^{1.2})$, versus $O(n^{1.75})$ in the general case.

Vaidya analyzes other cases using other estimates of work and fill during the factorization of various classes of sparse matrices. Vaidya does not show, however, bounds for regular meshes and finite-element grids in 3D that are better than the $O(n^{1.75})$ bound that applies to all bounded-degree graphs.

Vaidya also proposes a recursive scheme that uses the same idea to solve the system $Bz = r$ that must be solved in every iteration. That is, instead of factoring the preconditioner B and performing two triangular solves in every iteration, Vaidya proposes to construct a preconditioner for B that is even sparser than B, and to solve $Bz = r$ iteratively. Similar ideas have been proposed in other contexts, such as domain decomposition solvers where an iterative solver can be used within each subdomain, leading essentially to a multilevel preconditioner. Vaidya does not analyze this idea in any detail.

One potential disadvantage of Vaidya's preconditioners is that they are not guaranteed to parallelize. The maximum-weight trees that are constructed may have a large diameter. The large diameter of the trees creates long chains of dependences in the triangular factors, and these chains limit the parallelism that is available within each iteration of the solver.

We are not aware of any numerical experiments using Vaidya's preconditioners.

5. The Preconditioners of Gremban and Miller. This section presents *support trees*, the family of preconditioners that Gremban and Miller developed. We again assume that A is symmetric, diagonally dominant, and its off-diagonal entries are all non-positive. (Gremban and Miller also show a technique to convert a problem with a symmetric diagonally dominant matrix to a larger problem in which the matrix has only non-positive off-diagonals.) We will again assume that A and the preconditioner both have zero row sums.

Like Vaidya's method, Gremban and Miller's approach is essentially a graph algorithm that constructs $G_{B'}$ given G_A. However, Gremban and Miller construct a graph $G_{B'}$ with more vertices then G_A, so G_A is augmented with additional disconnected vertices so that both graphs use the same vertex set. In matrix terms, the construction embeds A as the leading block of a larger zero matrix,

$$A' = \begin{pmatrix} A & 0 \\ 0 & 0 \end{pmatrix} \quad \text{and} \quad B' = \begin{pmatrix} B_{11} & B_{12} \\ B_{12}^T & B_{22} \end{pmatrix} . \tag{5.1}$$

Gremban uses the Congestion-Dilation Lemma to bound $\sigma(A', B')$. However, there is no way to route all the edges of $G_{B'}$ in $G_{A'}$, since $G_{A'}$ is not connected. In other words, $\sigma(B, A)$ is infinite. Gremban, therefore, develops and uses Lemmas 2.11 and 2.12 to eliminate the extra vertices in $G_{B'}$. Once these nodes are eliminated, Gremban uses the Congestion-Dilation Lemma to bound $\sigma(B_{22} - B_{12}^T B_{11}^{-1} B_{12}, A)$, which provides a lower bound on the smallest positive finite eigenvalue of (A, B).

The construction of $G_{B'}$, illustrated in Figure 5.1, is based on a hierarchical decomposition of G_A. The algorithm removes from G_A a set of edges, known as a separator, that breaks it into a small number of subgraphs $G_1, G_2, \ldots, G_k$. The algorithm then recursively partitions each G_i until the graph is decomposed into single vertices. The separator is chosen so that all the G_i's have roughly the same number of vertices and such that the total weight of the separator is small. A variety of graph-partitioning algorithms can be used to find good edge separators (see, for example, [4]). The process is repeated until each subgraph consists of a single vertex. The graph $G_{B'}$, which is a tree, is constructed using this hierarchical decomposition. The algorithm assigns to each subgraph in the decomposition a vertex of $G_{B'}$. That is, $G_{B'}$ has one vertex that represents all of G_A, a vertex for each subgraph of G_A in the first level of the decomposition, and so on, down to vertices that represent single vertices of G_A, which are the smallest subgraphs in the decomposition. A leaf of $G_{B'}$ represents a single vertex of G_A, and is considered to be the same as that vertex of G_A. The matrices A' and B' are ordered accordingly. A vertex that represents a

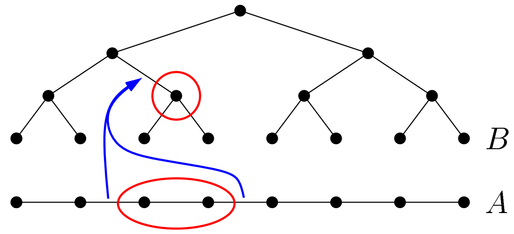

FIG. 5.1. *An illustration of the preconditioners of Gremban and Miller. A is partitioned hierarchically, and the vertices of B represent subgraphs in that partition. The circled vertex of B represents the subgraph of A consisting of the two circled vertices. The weight of an edge of B is the sum of the weights of the edges of A that connect the subgraph to the rest of A.*

subgraph G_i in the decomposition is connected by edges to the subgraphs of G_i in the decomposition $G_{i1}, G_{i2}, \ldots, G_{i\ell}$, and to the subgraph that contains G_i in the previous level of the decomposition. The weight that is assigned to the edge that connects G_i to, say, G_{i_1}, is the total weight of the edges that connect G_{i_1} to the remainder of the graph.

This construction makes it easy to prove a fairly low upper bound on $\lambda_f(A, B)$. We route each edge e of $G_{A'}$ along the unique path in $G_{B'}$ that connects its endpoints. Each edge in this path allocates a weight of w to support e, where w is the weight of e. This is always possible since if the path uses the edge between vertices that represent G_i and G_{i_1} in the decomposition, then e is part of the separator that divides G_{i_1} from the rest of the graph, so the w is included in the weight of each edge in the path. If every subgraph in the decomposition is split into at least k subgraphs whose sizes differ by at most a constant factor, the length of the path is $O(\log_k n)$, where n is the order of A. It follows that the congestion-dilation product is bounded by $O((w/w) \cdot \log_k n) = O(\log_k n)$, which is provides an upper bound on $\lambda_f(A, B)$.

Proving a lower bound on $\lambda_f(A, B)$ is more difficult. As explained above, the bound results from applying the Congestion-Dilation Lemma to the Schur complement $S = B_{22} - B_{12}^T B_{11}^{-1} B_{12}$ and to A. Specifically, Gremban and Miller prove upper bounds on the weights of the edges of S (which is a dense matrix) and show how to route them in A. For regular $n^{1/d} \times \cdots n^{1/d}$ grids in d dimensions with uniform edge weight, Gremban and Miller essentially perform a symbolic elimination to bound the entries of S. The bound that is obtained on $\sigma(S, A)$ is $O(d^2 n)$, leading to an overall condition number bound $O(n \log n)$ for fixed d. They also prove similar bounds for somewhat more general classes of matrices using some additional graph-theoretic tools. These classes of matrices do not include, however, matrices that represent planar graphs or finite element meshes.

Since $G_{B'}$ is guaranteed to be a tree with diameter $O(\log n)$, factoring B' and applying the factors in every iteration requires only $O(n)$ work and $O(\log n)$ parallel steps. The cost of computing $G_{B'}$ depends on the graph-partitioning algorithm that is used and may be substantial in practice.

Gremban and Miller also show how to convert a problem with a symmetric diagonally dominant matrix to a problem with a symmetric diagonally dominant matrix with nonpositive off-diagonals, so that their technique can be used. The graph of the modified problem has twice as many vertices and edges as the original graph. The modified graph represents each vertex of the original graph with two vertices and each edge with two edges. The transformation preserves separators, so if the original graph has a special structure that guarantees small separators, then the modified graph also

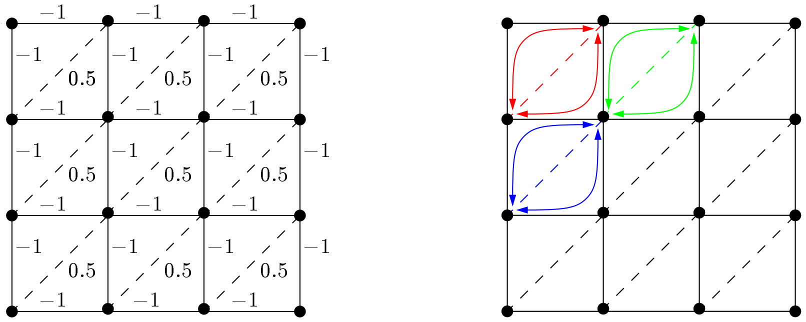

FIG. 6.1. *An incomplete-factorization preconditioner for a model problem with zero row sums (left). The model problem has the same underlying graph, but without the positive dashed edges. The (complete) Cholesky factor of this matrix is the incomplete-Cholesky factor of the model problem. The figure on the right shows the two paths that route each positive edge.*

has good separators and the same algorithm can be used to find them.

Gremban and Miller describe numerical experiments that show that their method outperforms a diagonal-scaling preconditioner and an incomplete-Cholesky preconditioner. On matrices that represent 2D meshes, Gremban and Miller's preconditioner performs fewer iterations and solves systems faster the other preconditioners. On a 3D problem, Gremban and Miller's preconditioner requires more iterations than incomplete Cholesky, but it leads to faster solution times on a vector computer.

6. Analysis of Incomplete Factorizations. This section describes our new analysis of preconditioners based on modified incomplete factorizations. We have recently learned that Guattery [7] has performed a somewhat similar analysis of unmodified incomplete factorizations.

Let $B = LL^T$ be a level-0 modified incomplete factorization of an M-matrix A. The incomplete factor L has the same nonzero structure as the lower triangle of A, and B has the same row sums as A. We can write $B = A - R$, where R consists of the fill elements that are dropped during the factorization plus the diagonal modification that is performed in order to maintain the row sums. Since the elements that are dropped are always negative and since A and B have the same row sums, R is an M-matrix with zero row sums. A modified incomplete factorization of a model problem is shown in Figure 6.1.

6.1. Analysis of a 2-Dimensional Model Problem. We first analyze the modified incomplete Cholesky factorization for a 2-dimensional model problem, a Laplace equation with Neumann boundary conditions on a regular grid. The following result is, to the best of our knowledge, new. It is known, however, that this same asymptotic condition-number bound holds for modified incomplete factorizations of a perturbed matrix $A + c/n$ with $c > 0$ [8]. We are aware of no previous proof for the case when $c = 0$.

Consider the regular grid depicted by the solid lines in Figure 6.1. If we perform an elimination of the vertices in the natural order and discard all fill, then the discarded values will correspond to the dashed diagonals in the figure. If A is the Laplacian matrix and B the modified incomplete Cholesky preconditioner, then $B = A - R$, where R is the matrix of these discarded values. Using Lemma 3.3 we can bound $\kappa(B^{-1}A)$ by supporting R with A. The sketch on the right of Figure 6.1 shows how each entry of R can be supported by two paths of length two within A. If we were to divide the weight of each A edge evenly, using half to support the R edge above it and half to support the R edge below it, we would support every R edge exactly. Unfortunately, this gives $\tau' = 1$ in Lemma 3.3, which does not give a finite bound on $\kappa(B^{-1}A)$. Rather, we must realize that this even division underutilizes the A edges along the boundary of the grid, and use an uneven division that varies from the upper left to the lower right.

We formalize this discussion to prove the following result.

THEOREM 6.1. *Let A represent a Laplace equation with Neumann boundary conditions (i.e., zero row sums) discretized on a $\sqrt{n}$-by-$\sqrt{n}$ grid, as shown in Figure 6.1. Let B be a modified incomplete-Cholesky factorization of A with no fill, using the natural (row-wise) ordering of the grid. Then $\kappa(B^{-1}A) \leq 2\sqrt{n} - 2$.*

Proof. We will assume that both A and B have zero row sums. By the construction of a modified incomplete factorization, they have the same row sums, and by Lemma 2.5, a bound that is obtained for zero row sums also applies to other nonnegative row sums.

We denote by (i,j) the vertex in row i and column j of the grid, and by $(i,j) \leftrightarrow (k,l)$ an edge connecting the vertices (i,j) and (k,l). It is easy to see that B consists of the edges of A plus edges with weight $+1/2$ that connect $v_{i,j}$ with $v_{i+1,j-1}$, as shown in Figure 6.1.

Using Lemma 3.3 we bound $\kappa(B^{-1}A)$ by bounding $\sigma(R, A)$, where $R = A - B$ is the (positive semidefinite) matrix of dropped fill elements, the diagonal dashed lines in the figure. Thus we must use A to support the edges of R. Each R edge is supported by two paths of length 2 in A, as shown in Figure 6.1.

More formally, we split A and R as follows. The matrix A is split into $2(\sqrt{n}-1)^2$ submatrices with the following edge sets, each a path of length 2:

$$\pi_\wedge(i,j) = \{(i,j) \leftrightarrow (i,j+1), (i,j) \leftrightarrow (i+1,j)\} \quad \text{for each } i < \sqrt{n}, j < \sqrt{n} \ ,$$

and

$$\pi_\vee(i,j) = \{(i,j) \leftrightarrow (i,j-1), (i,j) \leftrightarrow (i-1,j)\} \quad \text{for each } i > 1, j > 1 \ .$$

Except along the boundary, each edge of A is divided between one $\pi_\wedge$ submatrix and one $\pi_\vee$ submatrix. The weight of an edge in $\pi_\wedge(i,j)$ that is allocated to that submatrix in the splitting is

$$w_\wedge(i,j) = \frac{2\sqrt{n}-2}{2\sqrt{n}-3} - \frac{i+j-1}{2\sqrt{n}-3} \ ,$$

and the weight allocated to $\pi_\vee(i,j)$ is

$$w_\vee(i,j) = \frac{i+j-3}{2\sqrt{n}-3} \ .$$

By Lemma 2.8, the submatrix $\pi_\wedge(i,j)$ supports exactly a $w_\wedge(i,j)$ fraction of the R edge $(i,j+1) \leftrightarrow (i+1,j)$, and the submatrix $\pi_\vee(i+1,j+1)$ supports a $w_\vee(i+1,j+1)$ fraction of the same edge. Since

$$w_\wedge(i,j) + w_\vee(i+1,j+1) = \frac{i+j-2}{2\sqrt{n}-3} \ ,$$

we can apply Lemma 3.3 with $\tau' = (2\sqrt{n}-3)/(2\sqrt{n}-2)$ to conclude that

$$\kappa(B^{-1}A) = \frac{1}{1-\tau'} = 2\sqrt{n} - 2 \ .$$

We now show that this splitting of A is feasible; that is, that the contribution of each A edge to the paths that support R edges is not more than its weight. Each A edge contributes to either one $\pi_{\pm}$ submatrix (if it is on the boundary of the grid), or to two (if it is in the interior). The total contribution of an interior edge, say $(i,j) \leftrightarrow (i,j+1)$, is

$$w_{\wedge}(i,j) + w_{\vee}(i,j+1) = \frac{2\sqrt{n}-2}{2\sqrt{n}-3} - \frac{i+j-1}{2\sqrt{n}-3} + \frac{i+(j+1)-3}{2\sqrt{n}-3} = \frac{2\sqrt{n}-3}{2\sqrt{n}-3} = 1 \ .$$

The contribution of a boundary edge is at most

$$\max\left\{\frac{2\sqrt{n}-2}{2\sqrt{n}-3} - \frac{1+1-1}{2\sqrt{n}-3}, \frac{\sqrt{n}+\sqrt{n}-3}{2\sqrt{n}-3}\right\} = \frac{2\sqrt{n}-3}{2\sqrt{n}-3} = 1 \ .$$

□

It is easy to see that the same condition-number upper bound holds for the same model problem but with Dirichlet or mixed boundary conditions. The only difference in the structure of A between the Neumann boundary-condition case and the Dirichlet or mixed case is that row sums for vertices on the boundary of the grid may be positive. Since B has the same row sums as A, they both can be split into a zero-row-sum matrix and a positive diagonal matrix. The diagonal parts of A and B support each other with support 1. The zero-row-sum parts are similar to the case that we analyzed, except that the positive edges in B may be smaller than 0.5, but never greater. Hence, it is "easier" to support them, so the same bound holds. That is, we use Lemma 2.9 rather than Lemma 2.8 in the proof above.

The following theorem formalizes this result.

THEOREM 6.2. *Let A represent a model Laplace equation with non-negative row sums discretized on a $\sqrt{n}$-by-$\sqrt{n}$ grid, as shown in Figure 6.1. Let B be a modified incomplete-Cholesky factorization of A with no fill. Then $\kappa(B^{-1}A) \leq 2\sqrt{n}-2$.*

6.2. Analysis of a 3-Dimensional Model Problem. We now analyze a modified incomplete Cholesky preconditioner for a 3-dimensional model problem. As in the 2D case, we first analyze a model problem with boundary conditions that make the analysis as simple as possible, then extend the results.

Consider an $m \times m \times m$ grid, where $m = \sqrt[3]{n}$. A 2-by-2-by-2 example is depicted in Figure 6.2. If we perform elimination in the natural order and discard all the fill, then the discarded values correspond to the dashed edges in Figure 6.2. In contrast to the 2D case, in which each eliminated vertex had degree 2, the vertices that we eliminate in the 3D case can have degree 3, 2 or 1. When we eliminate vertices with degree 3, such as vertex number 1 in the figure, we drop from the Cholesky factor 3 nonzeros that correspond to 3 dashed edges. The vertex that we eliminate and its three neighbors form a three-edge star graph. The discarded nonzeros form a three-edge clique. When eliminate vertices with degree 2, we drop one nonzero, as in the 2D case. Vertices with grid coordinates (i,j,k) where $i,j,k < m$ have degree 3 when they are eliminated. Vertices with grid coordinates (i,j,k) with exactly one index equals to m have degree 2 when they are eliminated.

Our analysis of the 3D case is similar to the analysis of the 2D case, but the splitting is more complex. In the 2D case we supported single edges with paths of length 2. In the 3D case we also use three-edge stars to support three-edge cliques. We analyze the support of a clique by a star using Lemma 2.10. The rest of the proof technique remains the same.

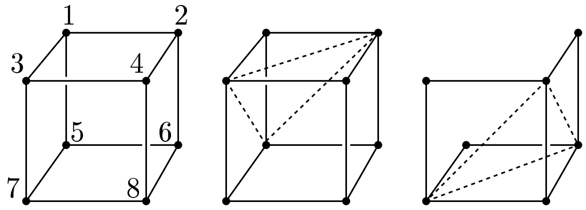

FIG. 6.2. *A 2-by-2-by-2 example of the 3D model problem and its incomplete factorization. At left is the graph of A. The numbers denote the order of elimination. The middle illustration shows the three (dashed) edges that are dropped when we eliminate vertex (row/column) 1. The illustration on the right shows the graph after the elimination of vertex 1 and the three (dashed) edges that are dropped when we eliminated vertices 2, 3, and 5. The elimination of vertices 4, 6, 7, and 8 generates no fill so no edges are dropped.*

The proof of the following theorem analyzes a model problem with boundary conditions that are simple to analyze.

THEOREM 6.3. *Let A represent a model Laplace equation discretized on an m-by-m-by-m grid, where $m = \sqrt[3]{n}$, and with boundary conditions that lead to zero row sums when $i, j, k \le \sqrt[3]{n}$ and row sums of one when i, j, or $k = \sqrt[3]{n}$. Let B be a modified incomplete-Cholesky factorization of A with no fill. Then $\kappa(B^{-1}A) \le 3\sqrt[3]{n} - 3$.*

Proof. We again use Lemma 3.3.

In the 3D model problem, the elimination of vertex (i, j, k), where $i, j, k < m$, eliminates the three edges

$$\Pi_\wedge(i,j,k) = \{(i,j,k) \leftrightarrow (i+1,j,k), (i,j,k) \leftrightarrow (i,j+1,k), (i,j,k) \leftrightarrow (i,j,k+1)\} \qquad \text{for each } i < m, j < m, k < m \ ,$$

and drops three fill edges of weight $-1/3$ which form a triangle. (as in the elimination of vertex number 1 in Figure 6.2.) When exactly one of i, j, or k equals m, situation is analogous to the 2D case, and the elimination of the vertex eliminates 2 edges that we continue to denote by $\pi_\wedge$, and drops one edge of weight $-1/3$. (The weight of the dropped edge is $-1/3$ because we eliminate a degree-2 vertex with row sum 1; recall that when at least one index is m, we set the row sum to 1.)

To support edges in R we use two sets of submatrices, $\Pi_\wedge(i, j, k)$ as defined above, and

$$\Pi_\vee(i,j,k) = \{(i,j,k) \leftrightarrow (i-1,j,k), (i,j,k) \leftrightarrow (i,j-1,k), (i,j,k) \leftrightarrow (i,j,k-1)\} \ .$$

We use the conventions that when i, j, or k equals m, $\Pi_\wedge$ contains less than 3 edges, and that when i, j, or k equals 1, $\Pi_\vee$ is empty. The Π subnmatrices represent star subgraphs of G_A.

The edge weights that are allocated to these submatrices are

$$W_\wedge(i,j,k) = \frac{3m-3}{3m-4} - \frac{i+j+k-2}{3m-4}$$

and

$$W_\vee(i,j,k) = \frac{i+j+k-4}{3m-4} \ .$$

Using Lemma 2.10, each Π submatrix now supports a fraction of the weight of *three* positive edges that form a clique when $i, j, k < m$, and a fraction of one edge when exactly one of i, j, or k equals either 1 or m.

In particular, if $i, j, k < m$, then $\Pi_\wedge(i,j,k)$ supports a $W_\wedge(i,j,k)$ fraction of the weight of the edges $(i,j,k+1) \leftrightarrow (i,j+1,k)$, $(i,j+1,k) \leftrightarrow (i+1,j,k)$, and $(i+1,j,k) \leftrightarrow (i,j,k+1)$. If one of i, j, or k equals m, say $k = 1$, then $\Pi_\wedge(i,j,k)$ supports more than a $W_\wedge(i,j,k)$ fraction of the weight of the edge $(i,j+1,k) \leftrightarrow (i+1,j,k)$. (More precisely, it supports weight $(1/2)W_\wedge(i,j,k)$ of the edge, whose weight is $1/3$.) The submatrix $\Pi_\vee(i,j,k)$ always supports a $W_\vee(i,j,k)$ fraction of the weight of the edges $(i,j,k-1) \leftrightarrow (i,j-1,k)$, $(i,j-1,k) \leftrightarrow (i-1,j,k)$, and $(i-1,j,k) \leftrightarrow (i,j,k-1)$.

It is easy to verify that the entire weight of every R edge is supported by one $\Pi_\wedge$ submatrix and one $\Pi_\vee$ submatrix. For example, a positive edge $(i+1,j,k) \leftrightarrow (i,j+1,k)$, is supported by $\Pi_\wedge(i,j,k)$ and by $\Pi_\vee(i+1,j+1,k)$. The total support for the edge by $\Pi_\wedge(i,j,k)$ and by $\Pi_\vee(i+1,j+1,k)$ is at least a

$$\begin{aligned} W_\wedge(i,j,k) + W_\vee(i+1,j+1,k) &= \left(\frac{3m-3}{3m-4} - \frac{i+j+k-2}{3m-4}\right) + \left(\frac{(i+1)+(j+1)+k-4}{3m-4}\right) \\ &= \frac{3m-3}{3m-4} \end{aligned}$$

fraction of its weight. Therefore, we can apply Lemma 3.3 with

$$\tau' = \frac{3m-3}{3m-4} ,$$

to conclude that $\kappa(B^{-1}A) = 3m-3 = 3\sqrt[3]{n} - 3$.

We now show that the splitting is feasible. Every edge in A contributes to one $\Pi_\wedge$ and one $\Pi_\vee$, except that edges on the faces where $i = 1$, $j = 1$, or $k = 1$ do not contribute to a $\Pi_\vee$. It is easy to verify that the total contribution of a negative edge that contributes to both a $\Pi_\wedge$ and a $\Pi_\vee$, say $(i,j,k) \leftrightarrow (i,j,k+1)$, is

$$\begin{aligned} W_\wedge(i,j,k) + W_\vee(i,j,k+1) &= \frac{3m-3}{3m-4} - \frac{i+j+k-2}{3m-4} + \frac{i+j+(k+1)-4}{3m-4} \\ &= 1 \ . \end{aligned}$$

The contribution of edges that contribute only to a $\Pi_\wedge$ is also at most 1:

$$\begin{aligned} W_\wedge(i,j,k) &= \frac{3m-3}{3m-4} - \frac{i+j+k-2}{3m-4} \\ &\le \frac{3m-3}{3m-4} - \frac{1}{3m-4} \\ &= 1 \end{aligned}$$

□

As in the 2D case, in an incomplete factorization of a problem with Dirichlet boundary conditions, the dropped edges are lighter than in the case we just analyzed, so the same $\Pi_\wedge$ and $\Pi_\vee$ submatrices can support them as well. Therefore, the following result holds:

THEOREM 6.4. *Let A represent a model Laplace equation discretized on a $\sqrt[3]{n}$-by-$\sqrt[3]{n}$-by-$\sqrt[3]{n}$ grid, with mixed boundary conditions as follows: A's row sums are zero everywhere except at grid vertices (i,j,k) with one index equal to $\sqrt[3]{n}$, where the row sums are 1. Let B be a modified incomplete-Cholesky factorization of A with no fill using the natural ordering. Then $\kappa(B^{-1}A) \le 3\sqrt[3]{n} - 3$.*

We have also analyzed the pure Neumann-boundary-conditions case, in which A and B have zero row sums everywhere. In this case, the dropped edges along the "back" faces of the grid (i, j, or $k = n^{1/3}$) have weight $-1/2$ and are, therefore, more difficult to support. Since their weights are the same as in the 2D case, it is possible to employ the 2D analysis to support them, but that leaves very little weight in the negative edges of the back faces for supporting the dropped edges in the interior of the grid. We have been able to prove a condition number bound of $O(n^{2/3})$, but not a $O(n^{1/3})$ bound. We omit the details from this paper.

7. Support-Graph Analysis of a Simple Multilevel Preconditioner. In this section we use support-graph theory to analyze a simple Multilevel Diagonal-Scaling (MDS) preconditioner for linear systems arising from finite-element discretizations of a Poisson's problem on a uniform grid. In one-dimension, we are able to show a condition number bound logarithmic in the size of the grid. We have also analyzed similar preconditioners in two dimensions, but we have been unable to prove polylogarithmic condition number bounds; we omit the details of the two dimensional analysis.

A multilevel diagonal-scaling preconditioner [12] applies diagonal-scaling at multiple discretization levels. Consider a matrix A whose underlying graph is G_A. The diagonal-scaling preconditioner B_{ds} for A is simply $B_{\mathrm{ds}} = \mathrm{diag}(A)$ or $B_{\mathrm{ds}}^{-1} = (\mathrm{diag}(A))^{-1}$. Now suppose that we construct a coarse grid representation A_{k-1} of A on a graph $G_{A_{k-1}}$ using a restriction operator R_{k-1} and an interpolation operator $I_{k-1} = R_{k-1}^T$, $A_{k-1} = R_{k-1} A I_{k-1} = R_{k-1} A R_{k-1}^T$. The restrition R_{k-1} takes a vector x defined on the vertices of G_A and returns a coarse representation $R_{k-1}x$ of the vector on the vertices of $G_{A_{k-1}}$. Given the restriction operator R_{k-1}, we can construct a 2-level diagonal-scaling preconditioner $B_{\mathrm{2ds}}^{-1} = (\mathrm{diag}(A))^{-1} + R_{k-1}^T (\mathrm{diag}(A_{k-1}))^{-1} R_{k-1}$. Now suppose that we coarsen A even further using a restriction operator $R_{k-2}R_{k-1}$ that first coarsens a vector using R_{k-1} and then again using a new operator R_{k-2}. If we construct a series $R_{k-1}, R_{k-2}, \ldots, R_0$ of restriction operators in this way, we can construct a multilevel diagonal-scaling preconditioner

$$
\begin{aligned}
B_{\mathrm{mds}}^{-1} = \quad & (\mathrm{diag}(A))^{-1} \\
& + R_{k-1}^T (\mathrm{diag}(A_{k-1}))^{-1} R_{k-1} \\
& + R_{k-1}^T R_{k-2}^T (\mathrm{diag}(A_{k-2}))^{-1} R_{k-2} R_{k-1} \\
& + \cdots \\
& + \prod_{i=k-1}^{0} R_i^T (\mathrm{diag}(A_0))^{-1} \prod_{i=0}^{k-1} R_i \; .
\end{aligned}
$$

Zhang [12] proposed this family of preconditioners and showed that for a certain class of matrices arising from finite-element discretizations and for certain restriction operators, $\kappa(B_{\mathrm{mds}}^{-1} A)$ is bounded by a constant independently of how fine the discretization. MDS preconditioners are a refinement of the so-called BPX preconditioners proposed by Bramble, et al. [2], in which $(\mathrm{diag}(A_j))^{-1}$ is replaced by the identity. (See also [10] for a discussion of MDS preconditioners).

MDS and BPX preconditioners are explicit, in the sense that the construction gives B^{-1} directly. But we can also view them as augmented preconditioners on a larger linear space that contains representations for all the coarse meshes, much like the augmented preconditioner B' of Gremban and Miller [5, 6]. When viewed as an augmented preconditioner, the construction for an MDS preconditioner actually gives its triangular decomposition. Suppose that we apply an MDS preconditioner by applying all the restriction operators in a sequence to obtain all the coarse representations of the vector that we apply B^{-1} to, then scaling by the inverses of the diagonal matrices, and then interpolating and adding contributions from the coarsest level to the finest. This process amounts to solving a triangular linear system (the bottom-up restriction process), scaling by a diagonal matrix, and solving another triangular linear system. In effect, we are solving a linear system whose coefficient matrix B' is already factored into LDL^T where L is lower triangular.

We have constructed Zhang's MDS preconditioners for model finite-element problems in one dimension. We also computed $B' = LDL^T$ explicitely and found that its graph $G_{B'}$ is a supergraph of G_L. That is, augmented MDS preconditioners have triangular factors that are sparser than the preconditioners themselves, much like the incomplete-Cholesky preconditioners. We have been unable, however, to analyze effectively these preconditioners using the theory presented in this paper because they have positive row sums on rows that correspond to coarse mesh vertices (These rows are zero in aumented matrix A' so the row sums are zero).

We have constructed instead MDS preconditioners that have the same row sums as A' and the same combinatorial structure as Zhang's preconditioners. More specifically, both our preconditioners have the same graph as Zhang's and their factors have the same graphs as the factors of Zhang's augmented preconditioners. We analyze these preconditioners in one dimension and show that $\lambda_f(A', B') \in [1, \log n]$.

7.1. Analysis of the One-Dimensional Case. Figure 7.1 shows the graph of our preconditioner together with the graph of the model problem. We obtained the matrix A from a finite-element discretization of a one-dimensional Laplace problem with Neumann boundary conditions. We constucted the precondioner level by level using the following algorithm:

1. Given a mesh with $n = 2^k + 1$ vertices (including the boundary), define $2^{k-1} + 1$ coarse-grid vertices, where evey other vertex in the fine grid is represented by a new vertex in the coarse grid.
2. Construct a restriction operator based on the stencil [1/4 1/2 1/4]. The restriction determines the edges between the finer and coarser grids. (The weight of the edges is determined so that the row sum on the coarse grid would be zero and the factorization of the preconditioner would give the restriction operator.) For our model problem, the weight of edges between the original grid and the first coarse grid have weights -1 and -2.
3. Compute the fill edges that are generated between coarse-grid vertices when we factor the preconditioner bottom up. Since these edges are not part of the graph of L, they must cancel out numerically. We therefore add edges between coarse-grid vertices that exactly cancel out the fill. These fill-canceling edges form a coarse-grid representation of A.
4. We now repeat the process, restricting to an even coarser grid. The restriction operator is the same operator, only scaled according to the scaling factor of the edges of A versus the fill-canceling edges. That is, if the edges of the original grid have weights -1 and the fill-canceling edges of a coarse mesh have weights α, then the weight of the edges to the next coarser mesh have weights $-\alpha$ and -2α. We shall see later that this scaling ensures that the spectum of the matrix pencil is bounded from below by 1.

We formalize and generalize this process in Section 7.2.

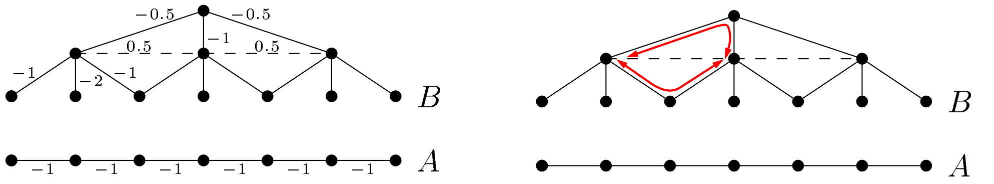

FIG. 7.1. *An MDS preconditioner for a one-dimensional Laplace problem with Neumann boundary conditions (left). In a larger problem, the positive edges in the next level would have weights* 0.25*. The paths that are used to support positive edges are shown on the right.*

We now show that for a size-n model problem A whose augmented matrix is

$$\tilde{A} = \begin{pmatrix} A_{11} & 0 \\ 0 & 0 \end{pmatrix} ,$$

the preconditioner B satisfies $\lambda_f(\tilde{A}, B) \in [1, \log(n+1)]$. Proving the upper bound $k = \log(n+1)$ on $\sigma B, \tilde{A}$ is intricate but not difficult. We simply show that $kB - \tilde{A}$ is nonnegative using the splitting lemma and the congestion-dilation lemma. The positive edges in $kB - \tilde{A}$, which are the fill-cancling edges in B and the original edges of A, are split and routed along length-2 paths of negative edges. Figure 7.1 shows how each positive edge is split and routed. Proving the lower bound on $\lambda_f(\tilde{A}, B)$ is more difficult. We prove the lower bound using a sequence of elimination and subtraction steps. The elimination steps always eliminate the current top-level vertices. According to Lemma 2.11, this does not change the bound on the support. The subtraction steps subtract from $\tilde{A} - B$ the fill edges from the last elimination step plus the fill-canceling edges in the next grid, as shown in Figure 7.2. We use Lemma 3.2 to show that the lower bound is maintained under these subtractions.

THEOREM 7.1. *Let A be a tridiagonal matrix of size $n = 2^k - 1$ with zero row sums and off-diagonal elements $A_{i,i+1} = A_{i+1,i} = -1$, and B be the corresponding MDS support graph, as show in Figure 7.1. Then $\lambda_f(\tilde{A}, B) \in [1, k]$.*

Proof. We begin by proving that $\sigma(B, \tilde{A}) \leq k$.

We split the vertices of B into k levels: the topmost vertex is at level 0, and the bottommost vertices are at level $k-1$.

We route each positive-weight edge through two paths of length 2, one using a node one level above the edge and one using a node one level below the edge, as shown in Figure 7.1. For a positive edge at level i, we route $1/(i+1)$ of its weight through level $i-1$ and $i/(i+1)$ of its weight through level $i+1$.

We prove that this is feasible by induction on i. In the proof, we use the facts that the weight of the edge at level i is 2^{1+i-k}, that the path through level $i-1$ consists of two edges of weight -2^{1+i-k} (the "vertical" edge has weight -2^{2+i-k}, which is split between the two length-2 paths that use that edge), and that the path through level $i+1$ consists of two edges of weight -2^{2+i-k}.

We now assume for the inductive step that the positive-weight edges of levels $1, \ldots, i-1$ can be routed in this manner, and that this routing leaves a $1/i$ fraction of the weight of the diagonal edges between level $i-1$ and i free for routing. This is obviously true for $i = 1$, since there are no positive-weight edges at level 0, and therefore $1/i = 1$ of the weight of the edges between level 0 and 1 remains free. The edges that we need to route at level i have weight 2^{1+i-k}, and we need to route them through a path of length 2 consisting of one edge of weight -2^{1+i-k} (half of the vertical

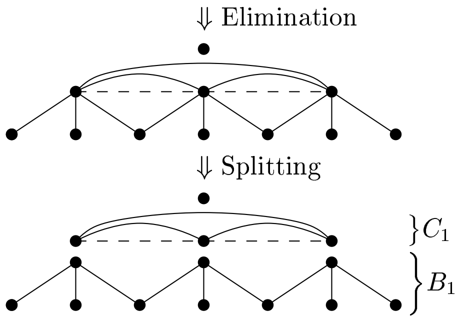

FIG. 7.2. *Alternating elimination and splitting steps in the analysis of the 1D MDS preconditioner. See the proof of Theorem 7.1 for details.*

edge, which supports two such routes), and one edge of weight $(1/i)\cdot(-2^{1+i-k})$ (which is what remains of the diagonal edge ofter it has supported its fraction of the positive edge at level $i-1$). This route can support a positive weight of

$$\frac{\left(-2^{1+i-k}\right)\times\frac{1}{i}\left(-2^{1+i-k}\right)}{\frac{1}{2}\left(-2^{1+i-k}\right)+\frac{1}{i}\left(-2^{1+i-k}\right)}=\frac{1}{i+1}\left(-2^{1+i-k}\right) \ .$$

Therefore, we need to route an $i/(i+1)$ fraction of the weight of the positive edge at level i using the path through level $i+1$. Since this path is also of length 2, we need to allocate weight

$$2\cdot\frac{i}{i+1}\left(-2^{1+i-k}\right)$$

for this path in each one of the edges that it uses, which is exactly $i/(i+1)$ of the weight of these edges. This concludes the proof of the induction.

The support graph B does not have positive weight edges at the bottommost level, $k-1$, but we need to route $(1/\tau)$ of the weight of the edges of A through B in order to prove that $\tau B-\tilde{A}$ is positive semidefinite. According to our inductive claim, the edges between level $i=k-1$ and $k-2$ can be used to route positive edges of weight

$$\frac{1}{i+1}\left(-2^{1+i-k}\right)=\frac{1}{k}\left(-2^{0}\right)=-\frac{1}{k}\ ,$$

so by setting $\tau=k$ we can route the edges of A (which have weight -1). This concludes the proof that $\sigma(B,\tilde{A})\leq k$, bounding $\lambda_f(\tilde{A},B)$ from above.

To establish the lower bound $\lambda_f(\tilde{A},B)\geq 1$ (or $\bar{\sigma}(B,\tilde{A})\leq 1$), we use the following strategy. We eliminate the topmost node of B at level 0, and denote the resulting matrix by B'. By Lemma 2.11, we have $\bar{\sigma}(B,\tilde{A})=\bar{\sigma}(B',\tilde{A})$. The elimination generates 3 fill edges, two of which are parallel to the positive edges of the next level, and one which is a true fill edge, as shown in Figure 7.2. We split B' into $B'=B_1+C_1$, where C_1 contains these these 3 edges, plus the two positive edges in the next level, plus diagonal weight that is designed to maintain zero row sums in C_1. We subtract C_1 from B', and we show below that C_1 is negative semidefinite and that $\mathrm{null}(B')\subseteq\mathrm{null}(C_1)$. Under these conditions, Lemma 3.2 guarantees that $\bar{\sigma}(B_1,\tilde{A})\geq\bar{\sigma}(B',\tilde{A})=\lambda_f(B,\tilde{A})$. Therefore, any upper bound we prove for $\lambda_f(B_1,\tilde{A})$ also upper bounds $\lambda_f(B',\tilde{A})=\lambda_f(B,\tilde{A})$.

We then continue to eliminate the nodes at level 1, subtract the fill that this elimination generates and the positive edges of level 2, and so on. Since all of these steps are identical except for scale, we only need to show once that C_1 is negative semidefinite and that $B' \subseteq C_1$.

The last four rows and columns of $2^{k-2}B$ are

$$\begin{bmatrix} 0 & 1 & 0 & -1 \\ 1 & 0 & 1 & -2 \\ 0 & 1 & 0 & -1 \\ -1 & -2 & -1 & 4 \end{bmatrix} .$$

Eliminating the last row and column (the top vertex) yields the corresponding submatrix of $2^{k-2}B'$,

$$\begin{bmatrix} -\frac{1}{4} & \frac{1}{2} & -\frac{1}{4} & 0 \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ -\frac{1}{4} & \frac{1}{2} & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} .$$

By Lemma 2.12, we can ignore the last row and column and focus on the leading 3-by-3 submatrix, which is the nonzero part of C_1. It can be easily checked that it is negative semidefinite, with eigenvalues 0, 0, and $-3/2$, and eigenvectors $(1,2,3)^T$, $(3,2,1)^T$, and $(1,-2,1)^T$.

We now show that $B' \subseteq C_1$. The preconditioner B has a one-dimensional null space spanned by the vector $(1,1,\ldots,1)^T$. The null space of B' is also one-dimensional and essentially the same, spanned by the vector $x = (1,1,\ldots,1,0)^T$ (the element corresponding to the top vertex of B is 0). It is easy to verify that $C_1 x = 0$ as well, proving that $B' \subseteq C_1$.

After $k-1$ alternating steps of elimination and subtraction, what remains of the support graph is B_{k-1}, with $\bar{\sigma}(B_{k-1}, \tilde{A}) \geq \bar{\sigma}(B', \tilde{A})$. All the edges of B_{k-1} are adjacent to vertices of A, so we can route these edges. The support graph B does not have edges between vertices at level $k-1$, so the Schur complement is a sum of submatrices of the form

$$\begin{bmatrix} \frac{3}{4} & -\frac{1}{2} & -\frac{1}{4} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{4} & -\frac{1}{2} & \frac{3}{4} \end{bmatrix} ,$$

which can be routed with congestion-dilation product $\tau = 1$ in the corresponding submatrices of A,

$$\begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} .$$

This concludes the entire proof. ◻

7.2. Generalizing the One-Dimensional Preconditioner. Although we have been unable to prove similar bounds for more realistic problems, we have been able to distill from this one-dimensional example a more general algorithm for constructing preconditioners. The algorithm leads to hierarchical preconditioners which parallelize well and that are easy and efficient to construct. When applied to the 1-dimensional model problem, this algorithm generates exactly the MDS precoditioner that we analyzed in the previous section.

We sketch here the algorithm that we use to construct these preconditioners.

1. Initialize G_B, the graph of the preconditioner, to have the same vertex set as G_A and no edges.
2. Decompose the vertices of G_A into overlapping subdomains. The only condition on the decomposition is that for every edge of G_A, some subdomain must include both of the edge's endpoints. (Clearly some decomposition will lead to better preconditioners than other decomposions, but this condition is the only one required to make the constrution itself work.) Edges can belong to more than one subdomain. In our model problem, each subdomain consists of exactly three adjacent vertices. Some vertices belong to one subdomain and some to two. Every edge of G_A belongs on exactly one subdomain in the model preconditioner.
3. Augment G_B with a new vertex v_s for each subdomain s.
4. Add an edge (v_s, v) for every subdomain s and for every vertex $v \in s$.
5. Decide how to route edges of G_A in G_B. Every edge $(u, w) \in G_A$ is routed along one or more length-2 paths in G_B. An edge that belongs to only one subdomain s is routed along in G_B using the path $u \leftrightarrow v_s \leftrightarrow w$. If an edge belongs to n subdomains, route $1/n$ of its weight in each subdomain.
6. Sum the total weight of G_A edges that are routed on each G_B edge. In our model preconditioner, the sums are -1 for some G_B edges and -2 for the rest.
7. Assign weights to the edges of G_B. The weights are the sums scaled by a factor α_s that may be different on each subdomain. The scale factor for a subdomain s are determined by eliminating v_s from G_B and ensuring that the edges of G_A in the subdomain support the fill-edges that the elimination generates with support at most 1. In the model preconditioner, the scale factors all happen to be 1.
8. Generate fill-canceling edges between subdomain vertices v_s in G_B. to do so, eliminate the edges of G_A from G_B (the non-subdomain vertices). This generates fill edges between the remaining subdomain vertices. Add to G_B edges to exactly cancel these fill edges.
9. Repeat the construction recursively, decomposing the subdomain vertices into coarser subdomains and adding edges that support the newly-added fill-caceling edges.

It is not hard to see that the construction ensures that such a preconditioner has two desirable properties. First, it can be factored bottom-up with no fill, since we add explicitly fill-canceling edges to counteract any fill. Furthermore, since there would be typically only $\Theta(\log n)$ levels of recursion in the construction, the triangular solves that apply the preconditioner have high levels of parallelism (e.g., they correspond to down-up traversals of a shallow directed acyclic graph). Second, the spectrum of the matrix pencil is bounded by 1 from below.

Unfortunately, we have not been able to prove strong upper bounds on the spectrum in more complex cases than the simple 1-dimensional model problem.

8. Conclusions. Support-graph theory has already motivated the design of two novel families of preconditioners. Vaidya's preconditioners, in particular, are more general and guarantee lower condition-numbers bounds than modified incomplete factorizations, a widely used class of preconditioners.

In this paper, we showed how the same theory can be used to prove tight condition-number bounds for MICC preconditioners on model problems, and a low condition-number bound for a simple MDS preconditioner. The bounds for MICC are new, since the result does not depend on a diagonal perturbation.

There has been some work on support-graph preconditioning besides the results presented in this paper. Guattery used support-graph theory to bound the condition number of incomplete factorizations without diagonal modification [7]. Howle and Vavasis generalized the preconditioners of Gremban and Miller to complex systems [9]. We are also aware that John Reif of Duke University is working on this subject, but to the best of our knowledge he has not yet published his results.

We believe that support-graph theory provides a new, largely unexploited tool for the analysis and design of preconditioners. We hope that this paper serves to make the techniques more accessible to the numerical analysis community and to stimulate further work in this promising area.

Acknowledgments. Thanks to Tony Chan for suggesting that we investigate MDS preconditioners, and for many constructive discussions. Thanks also to Steve Guattery and to Erik Boman for detailed comments on drafts of this paper. We are also indebted to Howard Elman and Henk van der Vorst for enlightening and helpful discussions.

REFERENCES

[1] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, 1994.

[2] J. H. Bramble, J. E. Pasciak, and J. Xu, *Parallel multilevel preconditioners*, Math. Comput., 55 (1990), pp. 1–22.

[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, MIT Press and McGraw-Hill, 1989.

[4] U. Elsner, *Graph partitioning: A survey*, Tech. Report SFB393/97-27, Technische Universität Chemnitz, Dec. 1997.

[5] K. Gremban, G. Miller, and M. Zagha, *Performance evaluation of a parallel preconditioner*, in 9th International Parallel Processing Symposium, Santa Barbara, April 1995, IEEE, pp. 65–69.

[6] K. D. Gremban, *Combinatorial Preconditioners for Sparse, Symmetric, Diagonally Dominant Linear Systems*, PhD thesis, School of Computer Science, Carnegie Mellon University, Oct. 1996. Technical Report CMU-CS-96-123.

[7] S. Guattery, *Graph embedding techniques for bounding condition numbers of incomplete factor preconditioners*, Tech. Report ICASE Report 97-47, NASA Langley Research Center, 1997.

[8] I. Gustafsson, *A class of first-order factorization methods*, BIT, 18 (1978), pp. 142–156.

[9] V. E. Howle and S. A. Vavasis, *Preconditioning complex-symmetric layered systems arising in electrical power modeling*, in Proceedings of the Copper Mountain Conference on Iterative Methods, Copper Mountain, Colorado, Mar. 1998. 7 unnumbered pages.

[10] B. F. Smith, P. E. Bjorstad, and W. D. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1996.

[11] P. M. Vaidya, *Solving linear equations with symmetric diagonally dominant matrices by constructing good preconditioners*. Unpublished manuscript. A talk based on the manuscript was presented at the IMA Workshop on Graph Theory and Sparse Matrix Computation, October 1991, Minneapolis.

[12] X. Zhang, *Multilevel Schwartz methods*, Numerische Mathematik, 63 (1992), pp. 521–539.