SIAM J. MATRIX ANAL. APPL.
Vol. 27, No. 4, pp. 930–951

SUPPORT-GRAPH PRECONDITIONERS*

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

Abstract. We present a 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 a class of existing preconditioners, modified incomplete Cholesky. The paper also contains a presentation of Vaidya's preconditioners, which was previously missing from the literature.

Key words. support preconditioners, combinatorial preconditioners, support theory, support-tree preconditioners, modified incomplete factorizations

AMS subject classifications. 65F10, 65F05, 65F50, 65Y20, 05C85

DOI. 10.1137/S0895479801384019

1. Introduction.

This paper presents new applications of a technique for constructing and analyzing preconditioners called *support-graph preconditioning*. Predecessors of support-graph methods can be found in the work from the late 80s by Notay, Beauwens, and collaborators in which graph-theoretic notions (principally paths) are used in the analysis of preconditioners [2, 3, 15, 16, 17]. These insights were extended by Vaidya [20], who described his work in a talk in 1991 but did not publish a paper. Vaidya used support-graph techniques to design a family of novel preconditioners based on spanning trees in graphs. Later, Gremban, Miller, and Zagha [9, 10] 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 a class of *known* preconditioners for model problems. Specifically, we use the extended technique to analyze certain modified incomplete Cholesky (MICC) preconditioners (see [12]).

The principal 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, it shows that the techniques are more widely applicable than previously appreciated. Second, we feel that the new proofs provide useful insights into these preconditioners;

*Received by the editors January 23, 2001; accepted for publication (in revised form) by S. A. Vavasis July 18, 2005; published electronically February 16, 2006. This research was supported in part by DARPA contract DABT63-95-C-0087 and by NSF contract ASC-96-26298. The third and fourth authors were supported by the Applied Mathematics Research program, U.S. DOE, Office of Science, and performed this work at Sandia National Labs, operated for the U.S. DOE under contract DE-AC04-94AL85000. A preliminary version of this paper was presented at the Copper Mountain Conference on Iterative Methods, Copper Mountain, CO, 1998.

http://www.siam.org/journals/simax/27-4/38401.html

†Xerox Palo Alto Research Center, Palo Alto, CA 94304 (bern@parc.edu).

‡Department of Computer Science, University of California, Santa Barbara, CA 93106-5110 (gilbert@cs.ucsb.edu).

§Sandia National Laboratories, Albuquerque, NM 87185-1110 (bah@cs.sandia.gov).

¶Raytheon Missile Systems, 1151 E. Hermans Road, Tucson, AZ 85706 (Nhat_X_Nguyen@raytheon.com).

‖School of Computer Science, Tel-Aviv University, Tel Aviv 69978, Israel (stoledo@tau.ac.il).

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 Ph.D. thesis [10], Vaidya's preconditioners have not been described in any published form. We seek to rectify this situation. Our complete presentation 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 pencil (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, sections 2 and 3, describes support-graph theory. The second part of the paper, sections 4 and 5, describes the preconditioners of Vaidya and of Gremban and Miller. The third part of the paper, section 6, describes support-graph analysis of MICC. Our conclusions from this research are presented in section 7.

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.

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 matrix. 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 diagonally dominant Stieltjes matrix, and by $O(n^{1.2})$ for diagonally dominant Stieltjes matrices whose underlying graphs are planar.

The strengths of Vaidya's preconditioners, especially of his second class, are 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 two dimensions 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 factorization. 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 a diagonally dominant Stieltjes 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 results of the condition-number bounds for modified incomplete factorizations are well known, but our proof provides a new perspective on why these methods work.

2. Basic support graph theory. This section describes the basic linear-algebra tools that Vaidya and Gremban and Miller have developed to analyze their preconditioners. These preconditioners are for *diagonally dominant Stieltjes matrices*, symmetric, diagonally dominant matrices with nonpositive off-diagonals. Vaidya and Gremban and 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 sections 4 and 5. Our own extensions are presented in section 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 (in floating-point arithmetic, it helps to orthogonalize the search directions against the null space if A is singular). In such cases, the number of iterations is proportional to the 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 square root of the ratio of the extreme finite generalized eigenvalues of the pencil (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 $x \neq 0$ 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 $\max \lambda_f(A, B)$, 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 $\max \lambda_f(A, B)$ 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 and Miller state the lemma and give a proof [10, Lemma 4.4]. We state it under a weaker hypothesis than Gremban and Miller, which is nevertheless strong enough for Gremban and Miller's proof. A more general version of this lemma can be found in Axelsson [1, Theorem 10.1].

LEMMA 2.2 (support lemma [10]). *If $\lambda \in \lambda_f(A, B)$ where B is positive semidefinite and* $\text{null}(A) \subseteq \text{null}(B)$*, then $\lambda \leq \sigma(A, B)$.*

Some matrix pencils 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 $\lambda_f(A, B) = \{1\}$, 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 matrix pencil 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 nonzero vector x such that

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

or

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

We use the support lemma to prove an upper bound τ on $\max \lambda_f(A, B)$ 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 by Gremban and Miller [10, 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 and 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 and 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 path in G_B (by path we always refer to a simple path). 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 shown in Figure 2.1. Both Vaidya and Gremban and Miller use

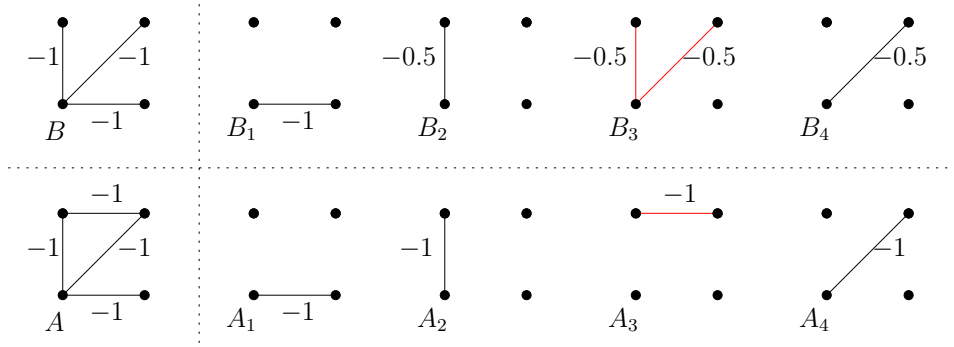

FIG. 2.1. *A graph representation of a splitting of* $A = A_1 + \cdots + A_4$ *and* $B = B_1 + \cdots + B_4$ *such that each* A_i *represents a single edge and each* B_i *is a path that supports the edge* A_i*. This splitting proves that* $\max \lambda_f(A, B) \le \sigma(A, B) \le 4$ *since the worst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

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

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

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_{11}, $C_{i+2,i+2}$, 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 that consists of edges of weight a. 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 a diagonally dominant Stieltjes 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 unit congestion, so the smallest 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 section 4.

The support, splitting, and congestion-dilation lemmas are analytical tools that are used to bound the condition number of a preconditioned system. Hence, they can also be used when one can prove that a low congestion-dilation embedding exists but cannot construct it explicitly. Gremban and Miller show how to construct and analyze preconditioners for some problems; they use congestion-dilation condition-number bounds using implicit embeddings [10, Theorem 5.1].

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

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

$$= \begin{pmatrix} k^2 & -k & -k & \cdots & -k \\ -k & 1 & 1 & \cdots & 1 \\ -k & 1 & 1 & \cdots & 1 \\ \vdots & & \ddots & & \vdots \\ -k & 1 & 1 & \cdots & 1 \end{pmatrix}.$$

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

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

which is clearly positive semidefinite. Hence, C is positive semidefinite. ∎

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 action 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 generalization.

The next lemma, stated under a stronger hypothesis by Gremban and Miller [10, Lemma 4.9], shows that $\lambda_f(A, B)$ is invariant under nonsingular transformations that are applied to both A and B. Their proof holds for this more general statement.

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 *action 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 and Miller do not use this lemma but use a similar one that is tailored more precisely to their 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 $Ay = \lambda By$, and thus $y^T(A - \lambda B)y = 0$. 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. ∎

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.

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

We cannot use congestion-dilation arguments directly to bound $\sigma(B, A)$; 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,

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

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 $\max \lambda_f(B, A)$ 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 section 5 how Gremban and Miller use this technique to analyze the condition number of their 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 and 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 section 3.2 were also derived independently by Guattery [11].

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 action support rather than eigenvalues.

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

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

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

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

$$x^T(\tau G^T BG - G^T AG)x \geq 0$$

for all x such that $G^T AGx \neq 0$ and $G^T BGx \neq 0$, so $\bar{\sigma}(G^T AG, G^T BG) \leq \tau = \bar{\sigma}(A, B)$. 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* $\text{null}(A) \subseteq \text{null}(C)$. *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

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

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 $x^T Cx \geq 0$ for all x, we must have

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

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

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

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 action 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 action support. Thus, the bound that we prove on the matrices after this series of transformations is also a bound on the original matrix and preconditioner.

3.2. Positive off-diagonal 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 preconditioner 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 section 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 [20]. Vaidya's first family applies to all symmetric, diagonally dominant matrices; the second family applies only to diagonally dominant Stieltjes 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-by-n diagonally dominant Stieltjes 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

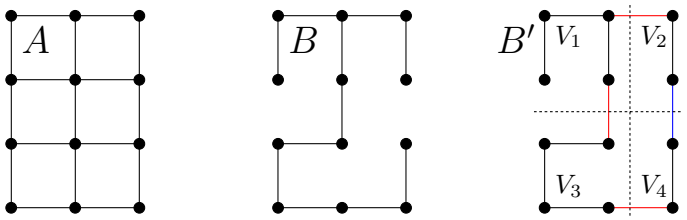

FIG. 4.1. *A graph G_A, a spanning tree G_B of G_A (middle), and a spanning tree augmented with extra edges ($G_{B'}$, right). The augmentation is performed by cutting the tree into subgraphs $V_1, \ldots, V_4$ of roughly equal size, and adding the heaviest edge between each pair of subgraphs.*

Lemma 2.5 in section 2, this transformation does not change the condition-number bounds (although it may change the condition number itself).

Both families use preconditioners B whose underlying graphs G_B are subgraphs of G_A (using the 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 of G_A. As illustrated in Figure 4.1, the preconditioner B is the diagonally dominant Stieltjes 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. Furthermore, all the edges along the path have edges at least as heavy as 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 $m/2$ 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 $\lambda_f(A, B) \subseteq [1, O(mn)]$.

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 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 [4]. A maximum-weight basis is a generalization of the maximum spanning tree; see [7, 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; note that if V_i and V_j are connected by an edge in T, they remain connected by the same edge in G_B, since G_B is formed by simply adding edges to 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 $m \leq dn$. The details of the partitioning of T into V_i's is given in [6, 4]. A more sophisticated partitioning algorithm, in which the quality of the decomposition is given by the average row density rather than the maximum row density, is given by Spielman and Teng [19]. 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 at least as heavy as 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 or A). It follows that the number of nonleaf 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; for example, if it is planar, 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 V_i and V_j, these edges do not disappear and are not merged into other edges in the contraction process. This supercontracted 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 supercontracted 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 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 diagonally dominant Stieltjes 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 three dimensions 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.

Chen and Toledo present experimental results with Vaidya's preconditioners in [6].

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 and diagonally dominant, and that its off-diagonal entries are all nonpositive. (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 nonpositive 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 than 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 and Miller use 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 and Miller, therefore, develop and use Lemmas 2.11 and 2.12 to eliminate the extra vertices in $G_{B'}$. Once these nodes are eliminated, they use the congestion-dilation lemma to bound $\sigma(B_{11} - B_{12}B_{22}^{-1}B_{12}^T, A)$, 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

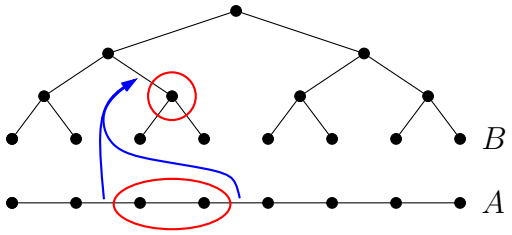

FIG. 5.1. *An illustration of the preconditioners of Gremban and Miller. A is partitioned hierarchically, and the vertices of B represent subgraphs in that partition. The circled vertex of B represents the subgraph of A consisting of the two circled vertices. The weight of an edge of B is the sum of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

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, [8]). 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 subgraph G_i in the decomposition is connected by edges to the subgraphs of G_i in the decomposition $G_{i_1}, G_{i_2}, \ldots, G_{i_\ell}$, 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 $\max \lambda_f(A, B)$. 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 provides an upper bound on $\max \lambda_f(A, B)$.

Proving a lower bound on $\min \lambda_f(A, B)$ is more difficult. As explained above, the bound results from applying the congestion-dilation lemma to the Schur complement $S = B_{11} - B_{12} B_{22}^{-1} B_{12}^T$ 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 $n^{1/d} \times \cdots \times n^{1/d}$ 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 subsume, 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 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 two-dimensional meshes, Gremban and Miller's preconditioner performs fewer iterations and solves systems faster the other preconditioners. On a three-dimensional 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. The convergence rates of various incomplete factorization preconditioners have been studied extensively, starting with the work of Gustafsson [12]. For MICC (the method we will analyze below), Gustafsson's analysis required a perturbation of the diagonal of the matrix. This limitation was removed in later work by Beauwens [2, 3], Notay [16], and others. The somewhat more challenging rank deficient case was also addressed by Notay [15, 17].

In this section, we present another analysis of the performance of modified incomplete factorization preconditioners on model problems. Our results are not new, and in fact are special cases of some of the results alluded to above. But we hope that the simplicity of the analysis (particularly as it easily handles rank deficiency) serves to illustrate the utility of the tools we have developed. We should also note that Guattery [11] has performed a somewhat similar analysis of unmodified incomplete factorizations.

Let $B = LL^T$ be a level-0 modified incomplete factorization of a diagonally dominant Stieltjes 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 a diagonally dominant Stieltjes matrix with zero row sums. A modified incomplete factorization of a model problem is shown in Figure 6.1.

We first analyze the MICC factorization for a two-dimensional model problem, a Laplace equation with Neumann boundary conditions on a regular grid.

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 MICC preconditioner, then $B = A - R$, where R is the matrix

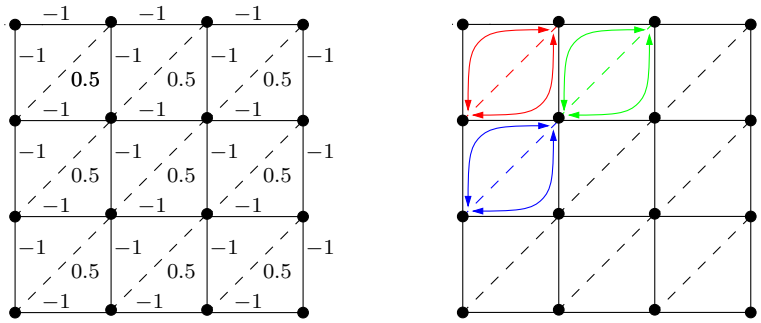

FIG. 6.1. *An incomplete-factorization preconditioner for a model problem with zero row sums (left). The model problem has the same underlying graph but without the positive dashed edges. The (complete) Cholesky factor of this matrix is the incomplete Cholesky factor of the model problem. The figure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

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 in Figure 6.1 shows how each entry of R can be supported by two paths of length 2 within A. If we were to divide the weight of each A edge evenly, using half to support the R edge above it or to the left and half to support the R edge below it or to the right, 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 can 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 MICC 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 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. Also, the construction of a modified incomplete factorization ensures that B has the same row sums as A, namely zero.

Using Lemma 3.3, $\kappa(B^{-1}A) \leq (1/(1-\bar{\sigma}(R, A)))$, 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

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

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 π 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

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

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 nonnegative row sums discretized on a $\sqrt{n}$-by-$\sqrt{n}$ grid, as shown in Figure* 6.1. *Let B be a MICC factorization of A with no fill. Then $\kappa(B^{-1}A) \le 2\sqrt{n}-2$.*

Similar analyses can be performed for three-dimensional model problems, although the bookkeeping is somewhat more tedious. The bounds we have obtained this way are $O(n^{1/3})$ for Dirichlet boundary conditions and $O(n^{2/3})$ for Neumann boundary conditions. We omit the details from this paper.

7. Conclusions. Support-graph theory has already motivated the design of several novel families of preconditioners. Vaidya's preconditioners, in particular, are more general and guarantee lower condition-number bounds than modified incomplete factorizations, a widely used class of preconditioners. We show in this paper that the same theory can be used to prove tight condition-number bounds for MICC preconditioners on model problems.

Significant work has been done 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 [11]. Howle and Vavasis generalized the preconditioners of Gremban and Miller to complex systems [13]. Reif proposed and analyzed Vaidya-like preconditioners, including recursive variants [18]. Chen and Toledo conducted an experimental evaluation of Vaidya's preconditioners [6]. Boman and Hendrickson formulated a general theory of support for symmetric positive definite matrices; the conjection-dilation arguments that we used in this paper are a special case of their theory [5]. Boman et al. generalized the Vaidya preconditioners presented in this paper to all symmetric diagonally dominant matrices [4]. Spielman and Teng [19] improve Vaidya's preconditioners for diagonally dominant Stieltjes matrices by using sharper spectral bounds, by improving the tree-partitioning algorithm, and by exploiting a sophisticated spanning-tree construction algorithm. Maggs et al. show how to construct a Gremban and Miller-like preconditioner for any symmetric diagonally dominant matrix [14].

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 the two anonymous referees for many helpful suggestions, corrections, and comments. Thanks to Tony Chan for many constructive discussions. Thanks also to Steve Guattery and 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, Cambridge, UK, 1994.
[2] R. Beauwens, *Upper eigenvalue bounds for pencils of matrices*, Linear Algebra Appl., 62 (1984), pp. 87–104.
[3] R. Beauwens, *Approximate factorizations with modified S/P consistently ordered M-factors*, Numer. Linear Algebra Appl., 1 (1994), pp. 3–17.
[4] E. G. Boman, D. Chen, B. Hendrickson, and S. Toledo, *Maximum-weight-basis preconditioners*, Numer. Linear Algebra Appl., 11 (2004), pp. 695–721.
[5] E. G. Boman and B. Hendrickson, *Support theory for preconditioning*, SIAM J. Matrix Anal. Appl., 25 (2003), pp. 694–717.
[6] D. Chen and S. Toledo, *Vaidya's preconditioners: Implementation and experimental study*, Electron. Trans. Numer. Anal., 16 (2003), pp. 30–49.
[7] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 1989.
[8] U. Elsner, *Graph Partitioning: A Survey*, Technical report SFB393/97-27, Technische Universität Chemnitz, Chemnitz, Germany, 1997.
[9] K. D. Gremban, G. L. Miller, and M. Zagha, *Performance evaluation of a new parallel preconditioner*, in Proceedings of the 9th International Parallel Processing Symposium, Santa Barbara, CA, 1995, IEEE, Computer Society Press, Los Alamitos, CA, 1995, pp. 65–69.

[10] K. D. Gremban, *Combinatorial Preconditioners for Sparse, Symmetric, Diagonally Dominant Linear Systems*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA., 1996.
[11] S. Guattery, *Graph Embedding Techniques for Bounding Condition Numbers of Incomplete Factor Preconditioners*, Technical report ICASE 97-47, NASA Langley Research Center, Hampton, VA, 1997.
[12] I. Gustafsson, *A class of first-order factorization methods*, BIT, 18 (1978), pp. 142–156.
[13] V. E. Howle and S. A. Vavasis, *An iterative method for solving complex-symmetric systems arising in electrical power modeling*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 1150–1178.
[14] B. M. Maggs, G. L. Miller, O. Parekh, R. Ravi, and S. Leung Maverick Woo, *Finding effective support-tree preconditioners*, in Proceedings of the 17th ACM Symposium on Parallel Algorithms and Architectures, Las Vegas, NV, 2005.
[15] Y. Notay, *Solving positive (semi)definite linear systems by preconditioned iterative methods*, in Preconditioned Conjugate Gradient Methods, Lecture Notes in Math. 1457, Springer-Verlag, New York, 1990, pp. 105–125.
[16] Y. Notay, *Conditioning analysis of modified block incomplete factorizations*, Linear Algebra Appl., 154–156 (1991), pp. 711–722.
[17] Y. Notay, *Conditioning of Stieltjes matrices by S/P consistently ordered approximate factorizations*, Appl. Numer. Math., 10 (1992), pp. 381–396.
[18] J. H. Reif, *Efficient approximate solution of sparse linear systems*, Comput. Math. Appl., 36 (1998), pp. 37–58 (see also an errata in 38 (1999), p. 141).
[19] D. A. Spielman and S.-H. Teng, *Solving sparse, symmetric, diagonally-dominant linear systems in time* $O(m^{1.31})$, in Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, Washington, DC, 2003, pp. 416–427.
[20] P. M. Vaidya, *Solving Linear Equations with Symmetric Diagonally Dominant Matrices by Constructing Good Preconditioners*, manuscript, 1991.