SIAM J. MATRIX ANAL. APPL.
Vol. 31, No. 5, pp. 2650–2664

DIRECTION-PRESERVING AND SCHUR-MONOTONIC SEMISEPARABLE APPROXIMATIONS OF SYMMETRIC POSITIVE DEFINITE MATRICES*

MING GU[†], XIAOYE S. LI[‡], AND PANAYOT S. VASSILEVSKI[§]

Abstract. For a given symmetric positive definite matrix $A \in \mathbf{R}^{N\times N}$, we develop a fast and backward stable algorithm to approximate A by a symmetric positive definite semiseparable matrix, accurate to a constant multiple of any prescribed tolerance. In addition, this algorithm preserves the product, AZ, for a given matrix $Z \in \mathbf{R}^{N\times d}$, where $d \ll N$. Our algorithm guarantees the positive-definiteness of the semiseparable matrix by embedding an approximation strategy inside a Cholesky factorization procedure to ensure that the Schur complements during the Cholesky factorization all remain positive definite after approximation. It uses a robust direction-preserving approximation scheme to ensure the preservation of AZ. We present numerical experiments and discuss the potential implications of our work.

Key words. semiseparable matrix factorization, positivity-preserving, direction-preserving

AMS subject classifications. 65F05, 65F50

DOI. 10.1137/090774331

1. Introduction.

1.1. Motivation and background. Given any symmetric positive definite (SPD) matrix A and any tolerance $\tau > 0$, in this paper we present a fast backward stable algorithm to construct an SPD semiseparable matrix that approximates A while preserving the product, AZ, for a given matrix $Z \in \mathbf{R}^{N\times d}$ for $d \ll N$. The idea of preserving the actions of A on certain vectors (directions) goes back to the early pointwise approximate factorization methods by Dupont, Kendall, and Rachford [11], Gustafsson [15], and Notay [23]. The motivation there was that by imposing certain row-sum criteria to the incomplete factorization of matrices coming from finite difference approximation of second order elliptic equations, it can lead to improving the condition number of the preconditioned matrix by an order of magnitude better than the one of the original finite difference matrix (i.e., from $\emptyset(h^{-2})$ to $\emptyset(h^{-1})$). One of our motivations here is that an approximate factorization of a discretization matrix can lead to Schur complement matrices that can be viewed as coarse discretization matrices if they preserve the near null-space of the original fine-grid matrix. Our goal is to have a general procedure that can ensure this property for any given number of directions d. For example, in the application of two-dimensional elasticity

*Received by the editors October 20, 2009; accepted for publication (in revised form) by D. B. Szyld July 28, 2010; published electronically September 16, 2010.
http://www.siam.org/journals/simax/31-5/77433.html

[†]Department of Mathematics, University of California, 861 Evans Hall, Berkeley, CA 94720 (mgu@math.berkeley.edu). This author's work was supported in part by NSF Career Award CCR-9702866.

[‡]Computational Research Division, Lawrence Berkeley National Laboratory, MS 50F-1650, One Cyclotron Road, Berkeley, CA 94720 (xsli@lbl.gov). This author's work was supported in part by the Director, Office of Science, Office of Advanced Scientific Computing Research, of the U.S. Department of Energy under contract DE-AC02-05CH11231.

[§]Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, P.O. Box 808, L-560, Livermore, CA 94551 (panayot@llnl.gov). This author's work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344.

equations it is important to preserve the so-called rigid-body modes, in which case we have $d = 3$. For other applications, such as the "adaptive algebraic multigrid" (AMG; cf., e.g., [2]), it is important that the coarse space contains several "algebraically smooth" directions. Although in the present paper we do not pursue the application of our direction-preserving approximate factorization method to AMG, this is one of our main motivations to develop and study the proposed approximate factorization technique.

In what follows we adopt the so-called semiseparable matrix structure which in certain applications by using high enough rank in the approximation can lead to virtually exact factorization of the matrix. Thus, by choosing the rank we have a whole spectrum of approximate block factorization methods that can vary in accuracy from simple preconditioners (comparable to symmetric Gauss–Seidel) to highly accurate (but potentially expensive for large rank) and virtually exact factorizations.

The semiseparable structure is a matrix analogue of semiseparable integral kernels as described by Kailath in [19]. This matrix analogue was most likely first described by Koltracht, Gohberg, and Kailath in [21]. In that paper it is shown that, under further technical restrictions, an LDU factorization is possible with a complexity n^2N, where n is the complexity of the semiseparable description and N is the dimension of the matrix—in effect an algorithm linear in the size of the matrix when n is small. In their papers Dewilde and Alpay [9] and Dym, Alpay, and Dewilde [12] introduce a new formalism for time-varying systems which provides for a framework closely analogous to the classical time invariant state space description and which allows for the generalization of many time invariant methods to the time-varying case. When applied to matrices, this formalism generalizes the formalism used in [21] and allows for more general types of efficient operations (by "efficient" we mean operations that are linear in the size of the matrix). In the book *Time-Varying Systems and Computations* [10], Dewilde and van der Veen describe the various operations that are possible on time-varying systems in great detail, including the efficient application of orthogonal transformations. In particular, they show how a URV-type transformation on a general (possibly infinite-dimensional), semiseparable system can be done with an efficient recursive procedure. This procedure is based on the ideas by van der Veen and Dewilde in [25] and by Gohberg and Eidelman in [14]. In the former paper the connection with Kalman filtering as a special case of the procedures is also discussed.

In the literature, various efficient representations for rank structured matrices have been proposed, and efficient and accurate algorithms have been developed using these representations [1, 3, 7, 8, 9, 12, 21, 16, 17, 18, 19, 26, 27]. In particular, several efficient algorithms have been developed for approximating a symmetric matrix A by a symmetric semiseparable matrix, accurate to a constant multiple of any given tolerance $\tau > 0$ [9, 12, 21]. Fast backward stable algorithms have also been constructed to approximate A with an SPD semiseparable matrix (see [24]).

This current work was also motivated by such work as well as work on construction of monotonic preconditioners for sparse SPD matrices. Recent work on superfast direct methods for discretized matrices from elliptic operators uses the semiseparable matrix structure as a basic tool in solving discretized elliptic PDE problems (see [4, 3, 6, 22]). In the process of generalizing these methods to construct robust and effective preconditioners, we are led to the problem of constructing semiseparable SPD matrices to approximate a given dense SPD matrix A for a very large given tolerance $\tau > 0$. Additionally, as mentioned earlier (e.g., in the AMG application), it is often unnecessary (potentially expensive) to maintain a high order of approxima-

Table 1.1
Dimensions of matrices in (1.1). k_i *and* l_i *are column dimensions of* U_i *and* P_i*, respectively.*

Matrix	U_i	V_i	W_i	P_i	Q_i	R_i
Dimensions	$m_i \times k_i$	$m_i \times k_{i-1}$	$k_{i-1} \times k_i$	$m_i \times l_i$	$m_i \times l_{i+1}$	$l_{i+1} \times l_i$

tions along a small number of known directions defined by a given matrix $Z \in \mathbf{R}^{N\times d}$. Values such as $1, 2,$ or 3 are typical for d in these cases.

1.2. The paper outline. In this paper, we present an efficient and backward stable algorithm for solving such problem for any given tolerance $\tau > 0$. This work will form the basis of our efficient construction of effective preconditioners for sparse matrices arising from discretized PDEs. As in [24], we embed the semiseparable matrix construction scheme of [3] inside the Cholesky factorization procedure for A to ensure that each approximate Schur complement during the Cholesky factorization remains positive definite. In addition, we ensure that the matrix-matrix product AZ remains unchanged throughout the entire procedure, up to rounding errors.

To be more specific, let B be a semiseparable $N \times N$ matrix. Then there exist n positive integers $m_1, \dots, m_n$, with $N = m_1 + \cdots + m_n$, to block-partition A as

(1.1)
$$B=(B_{i,j}), \quad \text{where } B_{i,j} \in \mathbf{R}^{m_i\times m_j} \text{ satisfies } B_{i,j} = \begin{cases} D_i & \text{if } i=j, \\ U_i W_{i+1} \cdots W_{j-1} V_j^T & \text{if } j > i, \\ P_i R_{i-1} \cdots R_{j+1} Q_j^T & \text{if } j < i. \end{cases}$$

The sequences $\{U_i\}_{i=1}^{n-1}$, $\{V_i\}_{i=2}^{n}$, $\{W_i\}_{i=2}^{n-1}$, $\{P_i\}_{i=2}^{n}$, $\{Q_i\}_{i=1}^{n-1}$, $\{R_i\}_{i=2}^{n-1}$, and $\{D_i\}_{i=1}^{n}$ are all matrices whose dimensions are defined in Table 1.1. While any matrix can be represented in this form for large enough k_i's and l_i's, our main focus will be on matrices of this special form that have relatively small values for the k_i's and l_i's (see section 3). In the above equation, empty products are defined to be the identity matrix. For $n = 4$, the matrix B has the form

$$B = \begin{pmatrix} D_1 & U_1V_2^T & U_1W_2V_3^T & U_1W_2W_3V_4^T \\ P_2Q_1^T & D_2 & U_2V_3^T & U_2W_3V_4^T \\ P_3R_2Q_1^T & P_3Q_2^T & D_3 & U_3V_4^T \\ P_4R_3R_2Q_1^T & P_4R_3Q_2^T & P_4Q_3^T & D_4. \end{pmatrix}.$$

Throughout this paper we will assume that the D_i's are square matrices. It is shown in [10] that this class of matrices is closed under inversion and includes banded matrices and semiseparable matrices as well as their inverses as special cases.

The semiseparable structure of a given matrix B depends on the sequence m_i. Different sequences will lead to different representations.

If D_k is symmetric and $P_k = V_k$, $R_k = W_k^T$, and $Q_k = U_k$ for all possible values of k, then B is a symmetric matrix. On the other hand, if D_k is upper triangular and $P_k = 0$ for all possible values of k, then B is an upper triangular semiseparable matrix.

As is well known, the Cholesky factor of an SPD semiseparable matrix is upper triangular semiseparable. Conversely, let R be a nonsingular upper triangular semiseparable matrix. Then R^TR is an SPD semiseparable matrix.

In section 2 we present the construction algorithm. In section 3, we discuss numerical experiment results with this construction algorithm. In section 4 we discuss potential applications of this work and draw some conclusions.

2. The construction algorithm. The main goal of this section is to present our semiseparable matrix construction algorithm. To this end, we need to establish some notation.

2.1. Notation. Let $A \in \mathbf{R}^{N\times N}$ be an SPD matrix with block partitioning

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{1,2}^T & A_{2,2} & \cdots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1,n}^T & A_{2,n}^T & \cdots & A_{n,n} \end{pmatrix}, \tag{2.1}$$

where $A_{k,k} \in \mathbf{R}^{m_k\times m_k}$ so that $N = \sum_{k=1}^n m_k$. With a slight abuse of notation, we will denote

$$A_{k,s:t} = \begin{pmatrix} A_{k,s} & \cdots & A_{k,t} \end{pmatrix} \quad \text{and} \quad A_{i:j,s:t} = \begin{pmatrix} A_{i,s} & \cdots & A_{i,t} \\ \vdots & \ddots & \vdots \\ A_{j,s} & \cdots & A_{j,t} \end{pmatrix}.$$

For any given matrix H and a given tolerance τ, we consider an orthogonal decomposition of the form

$$H = \begin{pmatrix} U & \widehat{U} \end{pmatrix} \begin{pmatrix} V & \widehat{V} \end{pmatrix}^T, \tag{2.2}$$

where the matrix $(U \quad \widehat{U})$ is column orthogonal so that $U^T\widehat{U} = 0$. Throughout this paper, we will decompose various matrices in the form of (2.2) such that U has as few columns as possible and such that $\|\widehat{V}\|_2 = O(\tau)$. Equation (2.2) will be our main tool for performing low-rank numerical approximations.

2.2. Direction-preserving approximations. We start by considering direction-preserving low-rank approximations. Let $H \in R^{m\times n}$, $F \in R^{n\times d}$, and $G \in R^{m\times d}$; we seek approximations of the form (2.2) that further preserve the matrix-matrix products HF and G^TH for $d \ll \min(m, n)$. That is, we would like to preserve the following equalities when H is approximated by UV^T:

$$HF = UV^TF \quad \text{and} \quad G^TH = G^TUV^T. \tag{2.3}$$

To this end, we first use QR factorization of F to get

$$F = Q_F \begin{pmatrix} R_F \\ 0 \end{pmatrix} = \begin{pmatrix} Q_F^1 & Q_F^2 \end{pmatrix} \begin{pmatrix} R_F \\ 0 \end{pmatrix}, \tag{2.4}$$

where $Q_F^1 \in \mathbf{R}^{n\times d}$ and $Q_F^2 \in \mathbf{R}^{n\times(n-d)}$. It is immediate that

$$HF = HQ_F^1R_F \tag{2.5}$$

Next, we use QR factorization of the $m \times (2d)$ matrix $\begin{pmatrix} G & HQ_F^1 \end{pmatrix}$ to get

$$\begin{pmatrix} G & HQ_F^1 \end{pmatrix} = Q_G \begin{pmatrix} R_G \\ 0 \end{pmatrix} \equiv Q_G \begin{pmatrix} R_G^1 & R_G^2 \\ 0 & 0 \end{pmatrix}, \tag{2.6}$$

where $Q_G \in \mathbf{R}^{m\times m}$ and $R_G \in \mathbf{R}^{(2d)\times(2d)}$. Let $R_G \equiv (R_G^1 \;\; R_G^2)$; we then have

$$G = Q_G \begin{pmatrix} R_G^1 \\ 0 \end{pmatrix}, \tag{2.7}$$

where $R_G^1 \in \mathbf{R}^{(2d)\times d}$.

Finally, we compute the matrix

$$\widehat{H} = Q_G^T H Q_F \equiv \begin{pmatrix} Q_G^T H Q_F^1 & Q_G^T H Q_F^2 \end{pmatrix} \overset{(2.6)}{\equiv} \left(\begin{pmatrix} R_G^2 \\ 0 \end{pmatrix} \quad Q_G^T H Q_F^2 \right).$$

Our goal is to approximate H by approximating $\widehat{H}$ instead. Note that $H = Q_G \widehat{H} Q_F^T$.

By construction, it is sufficient to preserve the first d columns and rows of $\widehat{H}$ in order to preserve HF and $G^T H$. Furthermore, our choices of the QR factorizations result in the lower left corner of $\widehat{H}$ being 0 as below:

$$\widehat{H} = \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \\ 0 & \widehat{H}_{2,2} \end{pmatrix}, \tag{2.8}$$

with $\widehat{H}_{1,1} \equiv R_G^2 \in \mathbf{R}^{(2d)\times d}$.

We now compute an orthogonal decomposition in the style of (2.2) for $\widehat{H}_{2,2}$,

$$\widehat{H}_{2,2} = (U_1\ U_2) \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix},$$

with columns of $(U_1\ U_2)$ orthonormal and $\|V_2\|_2 = O(\tau)$. This leads to an orthogonal decomposition of the form (2.2) for $\widehat{H}$ with

$$\widehat{H} = \left[\begin{pmatrix} I & \\ & U_1 \end{pmatrix} \qquad \begin{pmatrix} 0 \\ U_2 \end{pmatrix} \right] \left[\begin{matrix} \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \\ 0 & V_1^T \end{pmatrix} \\ \begin{pmatrix} 0 & V_2^T \end{pmatrix} \end{matrix} \right]. \tag{2.9}$$

Since $H = Q_G \widehat{H} Q_F^T$, we can define

$$U = Q_G \begin{pmatrix} I & \\ & U_1 \end{pmatrix}, \quad \widehat{U} = Q_G \begin{pmatrix} 0 \\ U_2 \end{pmatrix},$$

and

$$V = Q_F \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \\ 0 & V_1^T \end{pmatrix}^T, \quad \text{and} \quad \widehat{V} = Q_F \begin{pmatrix} 0 & V_2^T \end{pmatrix}^T,$$

which leads to an orthogonal decomposition of the form (2.2) for H with

$$H = \begin{pmatrix} U & \widehat{U} \end{pmatrix} \begin{pmatrix} V & \widehat{V} \end{pmatrix}^T. \tag{2.10}$$

We now show that (2.3) is true under this approximation. To verify the first part, we have

$$\begin{aligned} U V^T F &= Q_G \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \\ 0 & U_1 V_1^T \end{pmatrix} Q_F^T F = Q_G \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \\ 0 & U_1 V_1^T \end{pmatrix} \begin{pmatrix} R_F \\ 0 \end{pmatrix} \\ &= Q_G \begin{pmatrix} R_G^2 \\ 0 \end{pmatrix} R_F \overset{(2.6)}{=} H Q_F^1 R_F \overset{(2.5)}{=} HF. \end{aligned}$$

To verify the second part, we have

$$G^T H \overset{(2.7)}{=} \begin{pmatrix} {R_G^1}^T & 0 \end{pmatrix} Q_G^T H = \begin{pmatrix} {R_G^1}^T & 0 \end{pmatrix} \widehat{H} Q_F^T = {R_G^1}^T \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \end{pmatrix} Q_F^T,$$

and

$$G^T U V^T \overset{(2.7)}{=} \begin{pmatrix} {R_G^1}^T & 0 \end{pmatrix} Q_G^T \, Q_G \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \\ 0 & U_1 V_1^T \end{pmatrix} Q_F^T = {R_G^1}^T \begin{pmatrix} \widehat{H}_{1,1} & \widehat{H}_{1,2} \end{pmatrix} Q_F^T \,.$$

Thus, the above two quantities are equal.

It costs $O((m+n)d^2)$ flops to compute both QR factorizations; it costs $O(mnd)$ flops to compute $\widehat{H}$; and it costs $O(mnr)$ flops to compute an orthogonal decomposition for $\widehat{H}_{2,2}$, where r is the column dimension of V_1; and it costs about $O((m+n)(d+r)^2)$ flops to compute the representation (2.10). So the total cost of this compression scheme is about $O(mn(r+d))$ flops.

This compression scheme is numerically stable. We have, in fact, presented the scheme in such a way that every step of the computation is known to be numerically stable. We leave out the details of the proof for numerical stability as they are tedious and do not provide much new insight into the scheme.

2.3. Construction of approximate Cholesky factorization. To begin our procedure, we first recall the following standard block Cholesky factorization procedure:

for $k = 1, 2, \ldots, n$:
  Cholesky factorize $R_{k,k}^T R_{k,k} := A_{k,k}$;
  Compute $R_{k,k+1:n} := R_{k,k}^{-T} A_{k,k+1:n}$;
  Schur complement $A_{k+1:n,k+1:n} := A_{k+1:n,k+1:n} - R_{k,k+1:n}^T \cdot R_{k,k+1:n}$;
end for.

For each k, the first step in this procedure computes the Cholesky factorization of the kth diagonal block; the second step computes the rest of kth block row; and the last step computes the Schur complement of the kth block. The output of this procedure is the upper triangular matrix

$$R = \begin{pmatrix} R_{1,1} & R_{1,2} & \cdots & R_{1,n} \\ & R_{2,2} & \cdots & R_{2,n} \\ & & \ddots & \vdots \\ & & & R_{n,n} \end{pmatrix} \quad \text{such that} \quad A = R^T R.$$

In the following, we will modify the above procedure to find an approximate Cholesky factorization satisfying

$$S^T S = A + O\left(\sqrt{\|A\|_2 \tau}\right) \quad \text{and} \quad S^T S Z = A Z, \tag{2.11}$$

where

$$Z = \begin{pmatrix} Z_1 & & \\ & \ddots & \\ & & Z_n \end{pmatrix},$$

and where S is an upper triangular semiseparable matrix of the form (cf. (1.1))

$$S = \begin{pmatrix} D_1 & S_{1,2} & \cdots & S_{1,n} \\ & D_2 & \cdots & S_{2,n} \\ & & \ddots & \vdots \\ & & & D_n \end{pmatrix}, \tag{2.12}$$

with the D_k's being upper triangular and $S_{k,t} = U_k W_{k+1} \cdots W_{t-1} V_t^T$.

In light of the block Cholesky factorization procedure above, we begin by computing

$$D_1^T D_1 = A_{11} \quad \text{and} \quad H_1 = D_1^{-T} A_{1,2:n}.$$

Our next step is to compute a low-rank approximation to H_1 without changing AZ. Note that

$$AZ = \begin{pmatrix} D_1^T D_1 Z_1 + D_1^T H_1 Z_{2:n} \\ H_1^T G_1 + A_{2:n,2:n} Z_{2:n} \end{pmatrix},$$

where $G_1 = D_1 Z_1$. To preserve AZ, we only need to find a low-rank approximation to H_1 while preserving both $H_1 Z_{2:n}$ and $G_1^T H_1$. Here we compute an orthogonal decomposition of H_1 in the style of (2.10) as follows:

$$H_1 = \begin{pmatrix} U_1 & \widehat{U}_1 \end{pmatrix} \begin{pmatrix} \mathcal{Q}_1 & \widehat{\mathcal{Q}}_1 \end{pmatrix}^T,$$

where the matrix $(U_1 \;\; \widehat{U}_1)$ is column orthogonal and $\|\widehat{\mathcal{Q}}_1\|_2 \leq \tau$. It follows that

$$H_1^T H_1 = \mathcal{Q}_1 \mathcal{Q}_1^T + \widehat{\mathcal{Q}}_1 \widehat{\mathcal{Q}}_1^T.$$

According to the block Cholesky factorization procedure, $(D_1 \quad H_1)$ is actually the first block row of R. Hence the Schur complement of the first block becomes

$$\mathcal{A}_1 = A_{2:n,2:n} - H_1^T H_1 = A_{2:n,2:n} - \mathcal{Q}_1 \mathcal{Q}_1^T - \widehat{\mathcal{Q}}_1 \widehat{\mathcal{Q}}_1^T.$$

We now approximate $\mathcal{A}_1$ by

$$\widetilde{\mathcal{A}}_1 = A_{2:n,2:n} - \mathcal{Q}_1 \mathcal{Q}_1^T = \mathcal{A}_1 + \widehat{\mathcal{Q}}_1 \widehat{\mathcal{Q}}_1^T = \mathcal{A}_1 + O\left(\tau^2\right).$$

Since A is SPD, both the Schur complement $\mathcal{A}_1$ and its approximation $\widetilde{\mathcal{A}}_1$ must also be SPD. We note that this approximation amounts to adding a symmetric positive semidefinite matrix of norm at most τ^2 to the original matrix.

We further approximate $H_1 = R_{1,2:n}$ by $U_1 \mathcal{Q}_1^T$. Since this approximation is done on the Cholesky factor of A, the amount of perturbation to A is only $O(\|D_1\|_2 \tau) = O(\sqrt{\|A\|_2} \tau)$.

After these two approximations, we obtain the first block row in the Cholesky factor as

$$\begin{pmatrix} D_1 & U_1 \mathcal{Q}_1^T \end{pmatrix},$$

and the Schur complement is now $\widetilde{\mathcal{A}}_1$. We will only store the current $A_{2:n,2:n}$ and $\mathcal{Q}_1$ instead of computing $\widetilde{\mathcal{A}}_1$ explicitly.

To continue, partition

$$\mathcal{Q}_1^T = \begin{pmatrix} V_2^T & \widehat{H}_1 \end{pmatrix}.$$

The Schur complement becomes

$$\widetilde{\mathcal{A}}_1 = \begin{pmatrix} A_{2,2} - V_2 V_2^T & A_{2,3:n} - V_2 \widehat{H}_1 \\ \left(A_{2,3:n} - V_2^T \widehat{H}_1\right)^T & A_{3:n,3:n} - \widehat{H}_1^T \widehat{H}_1 \end{pmatrix}.$$

For approximations on the second block, we first compute

$$A_{2,2} := A_{2,2} - V_2 V_2^T \quad \text{and} \quad A_{2,3:n} := A_{2,3:n} - V_2 \widehat{H}_1.$$

We then use Cholesky factorization of $A_{2,2} := D_2^T D_2$, compute $H_2 := D_2^{-T} A_{2,3:n}$, and define

$$\mathcal{D}_2 = \begin{pmatrix} D_1 & U_1 V_2^T \\ & D_2 \end{pmatrix} \quad \text{and} \quad \mathcal{H}_2 = \begin{pmatrix} U_1 & \\ & I \end{pmatrix}.$$

With this notation and the approximation to H_1, we can rewrite the matrix A as

$$A \approx \begin{pmatrix} \mathcal{D}_2^T \mathcal{D}_2 & \mathcal{D}_2^T \mathcal{H}_2 \begin{pmatrix} \widehat{H}_1 \\ H_2 \end{pmatrix} \\ \begin{pmatrix} \widehat{H}_1 \\ H_2 \end{pmatrix}^T \mathcal{H}_2^T \mathcal{D}_2 & A_{3:n,3:n} \end{pmatrix}. \tag{2.13}$$

The product AZ was preserved in the approximation to H_1. In approximating $\binom{\widehat{H}_1}{H_2}$, we only need to preserve the product of matrix on the right-hand side of (2.13) and Z. But this can be done by preserving $\binom{\widehat{H}_1}{H_2} Z_{3:n}$ and $G_2^T \binom{\widehat{H}_1}{H_2}$, where $G_2 = \mathcal{H}_2^T \mathcal{D}_2 Z_{1:2}$.

Preserving these directions, we compute an orthogonal decomposition in the style of (2.10) as follows:

$$\begin{pmatrix} \widehat{H}_1 \\ H_2 \end{pmatrix} = \begin{pmatrix} \mathcal{U}_2 & \widehat{\mathcal{U}}_2 \end{pmatrix} \begin{pmatrix} \mathcal{Q}_2 & \widehat{\mathcal{Q}}_2 \end{pmatrix}^T,$$

where the matrix $(\mathcal{U}_2 \;\; \widehat{\mathcal{U}}_2)$ is column orthogonal and $\|\widehat{\mathcal{Q}}_2\|_2 \le \tau$. As before, approximating $\binom{\widehat{H}_1}{H_2}$ by $\mathcal{U}_2 \mathcal{Q}_2^T$ will not change the original matrix-matrix product AZ.

We write the Schur complement of $A_{2,2}$ as

$$\begin{aligned} \mathcal{A}_2 &= A_{3:n,3:n} - \widehat{H}_1^T \widehat{H}_1 - H_2^T H_2 = A_{3:n,3:n} - \begin{pmatrix} \widehat{H}_1 \\ H_2 \end{pmatrix}^T \begin{pmatrix} \widehat{H}_1 \\ H_2 \end{pmatrix} \\ &= A_{3:n,3:n} - \mathcal{Q}_2 \mathcal{Q}_2^T - \widehat{\mathcal{Q}}_2 \widehat{\mathcal{Q}}_2^T. \end{aligned}$$

We now approximate $\mathcal{A}_2$ by

$$\widetilde{\mathcal{A}}_2 = A_{3:n,3:n} - \mathcal{Q}_2 \mathcal{Q}_2^T$$

and the first two blocks of the Cholesky factor by

$$\begin{pmatrix} D_1 & U_1 V_2^T & U_1 W_2 \mathcal{Q}_2^T \\ & D_2 & U_2 \mathcal{Q}_2^T \end{pmatrix},$$

where we have used the partition

$$\mathcal{U}_2 = \begin{pmatrix} W_2 \\ U_2 \end{pmatrix}.$$

Again, this approximation ensures that the matrix-matrix product AZ remains unchanged and the Schur complement $\widetilde{\mathcal{A}}_2$ remains SPD.

To continue this procedure by induction, we assume that at the kth step for $k < n-1$, the approximate Cholesky factor with the first k blocks has the form

$$\begin{pmatrix} D_1 & U_1V_2^T & \cdots & U_1W_2\cdots W_{k-1}V_k^T & U_1W_2\cdots W_k\mathcal{Q}_k^T \\ & D_2 & \cdots & U_2W_3\cdots W_{k-1}V_k^T & U_2W_3\cdots W_k\mathcal{Q}_k^T \\ & & \ddots & \vdots & \vdots \\ & & & D_k & U_k\mathcal{Q}_k^T \end{pmatrix},$$

and the approximate Schur complement has the form

$$\widetilde{\mathcal{A}}_k = A_{k+1:n,k+1:n} - \mathcal{Q}_k\mathcal{Q}_k^T.$$

As before, partition

$$\mathcal{Q}_k^T = \begin{pmatrix} V_{k+1}^T & \widehat{H}_k \end{pmatrix}$$

so that

$$\widetilde{\mathcal{A}}_k = \begin{pmatrix} A_{k+1,k+1} - V_{k+1}V_{k+1}^T & A_{k+1,k+2:n} - V_{k+1}\widehat{H}_k \\ \left(A_{k+1,k+2:n} - V_{k+1}\widehat{H}_k\right)^T & A_{k+2:n,k+2:n} - \widehat{H}_k^T\widehat{H}_k \end{pmatrix}.$$

We explicitly compute

$$A_{k+1,k+1} := A_{k+1,k+1} - V_{k+1}V_{k+1}^T \quad \text{and} \quad A_{k+1,k+2:n} := A_{k+1,k+2:n} - V_{k+1}\widehat{H}_k.$$

We then use Cholesky factorization of $A_{k+1,k+1} := D_{k+1}^T D_{k+1}$ and compute

$$H_{k+1} := D_{k+1}^{-T} A_{k+1,k+2:n}.$$

Define

$$\mathcal{D}_{k+1} = \begin{pmatrix} D_1 & U_1V_2^T & \cdots & U_1W_2\cdots W_{k-1}V_k^T & U_1W_2\cdots W_kV_{k+1}^T \\ & D_2 & \cdots & U_2W_3\cdots W_{k-1}V_k^T & U_2W_3\cdots W_kV_{k+1}^T \\ & & \ddots & \vdots & \vdots \\ & & & D_k & U_kV_{k+1}^T \\ & & & & D_{k+1} \end{pmatrix}$$

and

$$\mathcal{H}_{k+1} = \begin{pmatrix} U_1W_2\cdots W_k & \\ U_2W_3\cdots W_k & \\ \vdots & \\ U_k & \\ & I \end{pmatrix}.$$

We can write the matrix approximation as follows:

$$A \approx \begin{pmatrix} \mathcal{D}_{k+1}^T\mathcal{D}_{k+1} & \mathcal{D}_{k+1}^T\mathcal{H}_{k+1}\begin{pmatrix} \widehat{H}_k \\ H_{k+1} \end{pmatrix} \\ \begin{pmatrix} \widehat{H}_k \\ H_{k+1} \end{pmatrix}^T \mathcal{H}_{k+1}^T\mathcal{D}_{k+1} & A_{k+2:n,k+2:n} \end{pmatrix}.$$

In order to keep the matrix-matrix product AZ unchanged, we only need to preserve $\binom{\widehat{H}_k}{H_{k+1}}Z_{k+2:n}$ and $G_{k+1}^T\binom{\widehat{H}_k}{H_{k+1}}$ for $G_{k+1} = \mathcal{H}_{k+1}^T\mathcal{D}_{k+1}Z_{1:k+1}$. As before, we compute an approximation of $\binom{\widehat{H}_k}{H_{k+1}}$ in the style of (2.10) as follows:

$$\begin{pmatrix} \widehat{H}_k \\ H_{k+1} \end{pmatrix} = \begin{pmatrix} \mathcal{U}_{k+1} & \widehat{\mathcal{U}}_{k+1} \end{pmatrix} \begin{pmatrix} \mathcal{Q}_{k+1} & \widehat{\mathcal{Q}}_{k+1} \end{pmatrix}^T, \tag{2.14}$$

where the matrix $(\mathcal{U}_{k+1} \;\; \widehat{\mathcal{U}}_{k+1})$ is column orthogonal and $\|\widehat{\mathcal{Q}}_{k+1}\|_2 \le \tau$.

It follows that the Schur complement for block $k+1$ is

$$\mathcal{A}_{k+1} = A_{k+2:n,k+2:n} - \widehat{H}_k\widehat{H}_k^T - H_{k+1}^T H_{k+1} = A_{k+2:n,k+2:n} - \begin{pmatrix} \widehat{H}_k \\ H_{k+1} \end{pmatrix} \begin{pmatrix} \widehat{H}_k \\ H_{k+1} \end{pmatrix}^T.$$

Again, this allows us to write

$$\mathcal{A}_{k+1} = A_{k+2:n,k+2:n} - \mathcal{Q}_{k+1}\mathcal{Q}_{k+1}^T - \widehat{\mathcal{Q}}_{k+1}\widehat{\mathcal{Q}}_{k+1}^T,$$

which is then approximated by

$$\widetilde{\mathcal{A}}_{k+1} = A_{k+2:n,k+2:n} - \mathcal{Q}_{k+1}\mathcal{Q}_{k+1}^T.$$

Since the difference between $\widetilde{\mathcal{A}}_{k+1}$ and $\mathcal{A}_{k+1}$ is a symmetric positive semidefinite matrix, $\widetilde{\mathcal{A}}_{k+1}$ must itself be an SPD matrix.

After these computational steps, the approximate Cholesky factor becomes

$$\begin{pmatrix} D_1 & U_1V_2^T & \cdots & U_1W_2\cdots W_{k-1}V_k^T & U_1W_2\cdots W_kV_{k+1}^T & U_1W_2\cdots W_k\widehat{H}_k \\ & D_2 & \cdots & U_2W_3\cdots W_{k-1}V_k^T & U_2W_3\cdots W_kV_{k+1}^T & U_2W_3\cdots W_k\widehat{H}_k \\ & & \ddots & \vdots & \vdots & \vdots \\ & & & D_k & U_kV_{k+1} & U_k\widehat{H}_k \\ & & & & D_{k+1} & H_{k+1} \end{pmatrix}.$$

Partitioning

$$\mathcal{U}_{k+1} = \begin{pmatrix} W_{k+1} \\ U_{k+1} \end{pmatrix}$$

in the numerical low-rank approximation of $\binom{\widehat{H}_k}{H_{k+1}}$ (see (2.14)) leads to $\widehat{H}_k \approx W_{k+1}\mathcal{Q}_{k+1}^T$ and $H_{k+1} \approx U_{k+1}\mathcal{Q}_{k+1}^T$, thus ending up with a new approximate Cholesky factor of the form

$$\begin{pmatrix} D_1 & U_1V_2^T & \cdots & U_1W_2\cdots W_{k-1}V_k^T & U_1W_2\cdots W_kV_{k+1}^T & U_1W_2\cdots W_kW_{k+1}\mathcal{Q}_{k+1}^T \\ & D_2 & \cdots & U_2W_3\cdots W_{k-1}V_k^T & U_2W_3\cdots W_kV_{k+1}^T & U_2W_3\cdots W_kW_{k+1}\mathcal{Q}_{k+1}^T \\ & & \ddots & \vdots & \vdots & \vdots \\ & & & D_k & U_kV_{k+1} & U_kW_{k+1}\mathcal{Q}_{k+1}^T \\ & & & & D_{k+1} & U_{k+1}\mathcal{Q}_{k+1}^T \end{pmatrix}.$$

Throughout the steps, the matrix-matrix product AZ is always kept unchanged.

This completes the induction for $k < n-1$. For $k = n-1$, the new approximate Cholesky factor still has the form similar to above, without the last column. This is exactly the form of the matrix S defined in (2.12). This ends the proof. ◻

It can easily be seen from the algorithm description that every approximate Schur complement during the Cholesky factorization is obtained by adding symmetric positive semidefinite matrices of norm at most τ^2 to the true one. We also perform an approximation of the order $O(\sqrt{\|A\|_2}\tau)$ for low-rank approximation at every step of the algorithm. Hence the total truncation error $O(\sqrt{\|A\|_2}\tau)$ in (2.11) could be $O(n)$ times larger than $\sqrt{\|A\|_2}\tau$.

Assume that each diagonal block in A has roughly the same number of columns. Let p be the maximum dimension in all the diagonal blocks, and assume that p is bigger than the column dimension of every matrix U_k. Then the cost for each step is $O(Np^2)$ flops, leading to a total cost of $O(n^2p^3) = O(N^2p)$ flops for the whole construction algorithm.

As is shown in [3], the column dimensions of U_k in S turn out to be precisely the rank of $S_{1:k,k+1:n}$ for $k = 1, \ldots, n-1$. If $A_{1:k,k+1:n}$ has small numerical rank for the given tolerance for $k = 1, \ldots, n-1$, the matrix S constructed above will also have small rank in each of its upper off-diagonal blocks. Otherwise, some U_k's would need to have large numbers of columns.

We have presented our construction algorithm using SVDs. However, any rank-revealing decomposition satisfying (2.2) will also work. Good examples are rank-revealing QR factorizations and rank-revealing modified Gram–Schmidt procedures. It is likely that this will lead to considerable speedup for a small loss in compression.

As before, this construction algorithm is numerically stable, and we also leave out the details of the tedious and yet not very insightful proof.

For the purpose of computing a preconditioner, we can further require that the number of columns in U_k not exceed a certain preset number, like *MaxRank*. This is equivalent to restricting the number of columns in U in (2.2) to never exceed a certain preset number, like *MaxRank*. In our numerical experiments, we simply set the submatrix $\widehat{H}_{2,2} = 0$ in (2.8). This simple strategy has still led to very effective preconditioners; see section 3.

3. Numerical results. We have written a C code implementing our construction algorithm. In the following we report the numerical results with this code. Here we concentrate on demonstrating the effectiveness of our semiseparable matrix approximations as preconditioners.

First, we consider finite-element discretizations on uniform triangular mesh of size h, with piecewise linear functions of the following diffusion equation defined on the unit square $\Omega = [0,1] \times [0,1]$:

$$-\text{div}\,(k(x,\ y)\nabla u) = f(x,y), \tag{3.1}$$

where the coefficient $k(x,\ y)$ is a two-by-two matrix of the form $\epsilon I + \mathbf{b}\mathbf{b}^T$ for a given $\epsilon > 0$ and variable direction vector

$$\mathbf{b} = \begin{bmatrix} \cos\alpha(1 - x\cos\alpha) \\ \sin\alpha(1 - y\sin\alpha) \end{bmatrix}.$$

In the test we chose $\epsilon = 0.01$ and $\alpha = \frac{\pi}{3}$. We assume a mixture of Dirichlet and Neumann boundary conditions.

We use standard lexicographic ordering of the unknowns (or mesh-points). The block structure of the matrix is obtained by putting together all p consecutive nodes in a block. In the test we varied the block size p; the number of direction vectors, d, between zero and three; and the maximum rank r. Note that our algorithm requires

$r \geq 2d$ and the block size p to be at least the rank r. We present results of two settings of block size and maximum rank: the smaller one with $p = 8$ and $r = 2d + 2$, and the larger one with $p = 20$ and $r = 2d + 10$. The direction vectors correspond to the constant vector for $d = 1$, and additionally, $d = 2$ and $d = 3$ correspond to the vectors coming from the linear functions x and y evaluated at the nodes of the mesh. We use the thus constructed block factorization matrix as a preconditioner in the preconditioned conjugate gradient (PCG) method. We list in Table 3.1 the number of iterations m for which the respective residuals satisfy $\sqrt{\mathbf{r}_m^T \mathbf{r}_m} \leq 10^{-6}\sqrt{\mathbf{r}_0^T \mathbf{r}_0}$. We do not use the preconditioned residual norm since we want to compare the different preconditioners corresponding to different d (the number of directions) using fixed norm. We also include the time to construct the approximate factor preconditioner. The tests were run on a 1.9 GHz IBM Power5 machine at the National Energy Research Scientific Computing Center.

The results in Table 3.1 show the improvement of the number of iterations using increasing number of directions. There are only two cases where the iteration count for $d = 2$ is larger than that for $d = 0$ or $d = 2$. Nevertheless, $d = 3$ always achieves the lowest iteration count. It is clear that the preconditioner for larger d's is more expensive to construct and apply. Also, as expected, larger rank results in better approximate factorization. It is good that the extra construction cost is acceptable—with more than doubling the block size and rank, the construction time is not more than doubled, and the increase is smaller as the problem size increases.

TABLE 3.1

Number of PCG iterations for anisotropic diffusion equation: $\epsilon = 0.01$, $\alpha = \frac{\pi}{3}$. The times (in seconds) for constructing the preconditioner are shown for $d = 0$ and $d = 3$.

	$p = 8$, $r = 2d + 2$						$p = 20$, $r = 2d + 10$						CG
h^{-1}	$d = 0$	time	$d = 1$	$d = 2$	$d = 3$	time	$d = 0$	time	$d = 1$	$d = 2$	$d = 3$	time	iters
12	28	0.00	24	21	20	0.01	7	0.00	1	1	1	0.00	51
24	61	0.05	55	51	51	0.07	28	0.05	24	23	20	0.13	116
48	115	0.57	113	121	110	0.91	77	1.00	65	65	53	1.14	240
96	233	8.52	221	216	210	13.74	158	15.48	139	185	118	18.49	479

The purpose of the second test that we performed is to achieve high tolerance in the approximation when we factorize a dense SPD matrix. We consider the model anisotropic diffusion problem (3.1) for a set of diffusion direction vectors $\mathbf{b}$. The dense matrix under consideration is obtained as follows. We order the nodes using the nested-dissection ordering [13, 20]. In this ordering, the last $n \times n$ ($n = 1/h$)-dimensional Schur complement, S, is a dense SPD matrix, costing traditional direct solvers $O(n^3)$ operations to factorize. We approximate this matrix by $R^T R$, where R is an upper triangular semiseparable matrix with maximum off-diagonal rank at most 2. We require that a single direction $Z = (1, \ldots, 1)^T$ be preserved under our compression scheme. Z in this case is a well-known rigid-body mode of our model problem under our discretization. This implies that we must set $\widehat{H}_{2,2} = 0$ in (2.10) at every step of compression, even though the matrix S in consideration can be very ill conditioned. Let $\widehat{S} = R^{-T} S R^{-1}$. Obviously, $R^T R$ is a good preconditioner if $\kappa(\widehat{S})$, the condition number of $\widehat{S}$, is much smaller than $\kappa(S)$, the condition number of S. Table 3.2 summarizes our results for this problem. We observe that $\kappa(\widehat{S})$ always hovers around 1, indicating high effectiveness of $R^T R$ as a preconditioner for S. In other words, the last $n \times n$-dimensional dense Schur complement in the traditional

Table 3.2
Approximation on the Schur complements for model problem (3.1).

	$n = 200$, $\mathbf{b}$ is unit random				$n = 400$, $\mathbf{b}$ is unit random			
ϵ	1	10^{-4}	10^{-8}	10^{-12}	1	10^{-4}	10^{-8}	10^{-12}
$\kappa(S)$	4.7×10^2	5.1×10^3	1.3×10^2	5.6×10^2	4.9×10^2	2.9×10^2	6.4×10^5	4.7×10^2
$\kappa(\widehat{S})$	2.9	1.3	1.5	1.9	3.2	1.7	2.4×10^1	2.0
	$n = 200$, $\mathbf{b} = (1, 0)^T$				$n = 400$, $\mathbf{b} = (1, 0)^T$			
ϵ	1	10^{-4}	10^{-8}	10^{-12}	1	10^{-4}	10^{-8}	10^{-12}
$\kappa(S)$	2.8×10^2	2.0×10^5	2.0×10^9	2.0×10^{13}	5.7×10^2	4.0×10^5	4.0×10^9	4.2×10^{13}
$\kappa(\widehat{S})$	2.8	1.6	1.5	1.0	3.2	2.2	1.0	1.0

Cholesky factorization can be well represented by a semiseparable representation with off-diagonal rank 2.

Finally, we considered the two-dimensional linear elasticity equation

$$-\left(\mu\overrightarrow{\triangle u} + \lambda \bigtriangledown \overrightarrow{\bigtriangledown \bullet u}\right) = \overrightarrow{f} \quad \text{in} \quad \Omega = (0,1) \times (0,1), \tag{3.2}$$

$$\overrightarrow{u} = \overrightarrow{0} \quad \text{on} \quad \partial\Omega, \tag{3.3}$$

where $\overrightarrow{u} \in R^2$ is the displacement vector field and λ and μ are the Lamé constants. This PDE is very ill conditioned when the ratio λ/μ is very large; this limit is known as the incompressible limit and is associated with the mechanical behavior of elastometric materials and plastic flow in metals, for example. Iterative methods, including standard geometric multigrid, converge very slowly or even diverge for very large λ/μ. However, such situations are important as they are ubiquitous in nature; one of our chosen example problems, in fact, possesses this behavior in its linearized form. The two direction vectors correspond to the two well-known rigid-body modes. Let $u = (u_1 \quad u_2)$. One of the rigid-body modes is such that all the discretized u_1 nodes are 1 and all the discretized u_2 nodes are 0; the other is such that all the discretized u_1 nodes are 0 and all the discretized u_2 nodes are 1. Table 3.3 shows the PCG convergence history and the condition number of $\widehat{A} = R^{-T}AR^{-1}$, where R is the approximate semiseparable Cholesky factor. It is clear that with higher ratio λ/μ, the system is much more ill conditioned and requires many more PCG iterations. When $\lambda/\mu = 1$, preserving directions and increasing block/rank size are beneficial. When $\lambda/\mu = 10^4$, preserving directions is generally beneficial, but larger block/rank size is not helpful.

Table 3.3
Number of PCG iterations and the condition number $\kappa(\widehat{A} = R^{-T}AR^{-1})$ for the elasticity equations. The last two columns show the number of CG iterations and the condition number of the initial matrix A.

(λ, μ)	h^{-1}	$p = 8$, $r = 2d + 2$				$p = 20$, $r = 2d + 10$				CG	
		$d = 0$	$\kappa(\widehat{A})$	$d = 2$	$\kappa(\widehat{A})$	$d = 0$	$\kappa(\widehat{A})$	$d = 2$	$\kappa(\widehat{A})$	iters	$\kappa(A)$
$(1.0, 1.0)$	8	32	1.5×10^1	25	9.7×10^1	16	2.9×10^1	11	1.9×10^1	68	2.9×10^2
	16	62	6.4×10^2	48	4.7×10^2	64	8.6×10^2	31	2.0×10^2	119	1.2×10^3
	32	123	2.5×10^3	92	1.7×10^3	83	3.0×10^3	62	1.2×10^3	228	4.7×10^3
$(1.0, 10^{-4})$	8	243	3.1×10^5	236	3.5×10^5	12	1.3×10^1	9	1.3×10^1	405	5.7×10^5
	16	549	1.1×10^6	440	9.7×10^5	1230	1.7×10^6	1203	2.0×10^6	1214	2.1×10^6
	32	1216	4.5×10^6	1258	4.3×10^6	1867	7.0×10^6	1996	8.6×10^6	3149	8.3×10^6

Table 3.4
Preconditioner effectiveness on the Schur complements for the elasticity equations.

	$n = 200$				$n = 400$			
λ/μ	1	10^4	10^8	10^{12}	1	10^4	10^8	10^{12}
$\kappa(S)$	1.7×10^2	2.2×10^5	2.2×10^9	1.5×10^{13}	3.3×10^2	4.5×10^5	4.5×10^9	1.8×10^{13}
$\kappa(\widehat{S})$	1.6	2.1	2.1	2.0	2.0	2.4	2.4	2.2

For the elasticity problem, we also examined the last Schur complement matrix arising from direct Cholesky factorization with nested-dissection ordering. This time, we allow the maximum off-diagonal rank to be at most 4 in the semiseparable representation and still require our compression scheme to preserve the two rigid-body modes. The results are shown in Table 3.4. This time, $\kappa(\widehat{S})$ hovers around 1, even when S is ill conditioned.

To summarize, our results show that for both diffusion and elasticity problems, our direction-preserving factorization method is very efficient and achieves very good approximation for the Schur complement matrices corresponding to the top level separator. Our future main goal is to use this factorization algorithm to construct reduced (Schur complement) matrices that have prescribed actions on certain direction vectors and not as much as stand-alone preconditioners (as explained in the introduction of this paper).

4. Conclusions. We presented an efficient and backward stable algorithm for constructing SPD semiseparable matrices that approximate a given dense SPD matrix A with a guaranteed a priori given tolerance $\tau > 0$. In the literature, there are several different classes of semiseparable matrices that have similar low-rank structures [6, 5, 16, 17, 18]. Work has begun to extend our algorithm to such low-rank structures. Ultimately, such algorithms will be used to form the basis of efficient algorithms to construct effective preconditioners for sparse matrices arising from discretized PDEs.

Alternatively, giving up on the guaranteed tolerance property, the proposed algorithm provides an SPD factorized matrix that has the same actions as the original SPD matrix on a given set of direction vectors. More generally, the proposed algorithm has the property that it provides approximate Schur complement (reduced) matrices that have the same actions as the corresponding exact Schur complements on a given set of direction vectors. The latter property offers the potential to construct coarse matrices for AMGs, which is a topic of future research.

REFERENCES

[1] M. Bebendorf, *Hierarchical Matrices*, Lect. Notes Comput. Sci. Eng. 63, Springer, Berlin, 2008.

[2] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, *Adaptive algebraic multigrid*, SIAM J. Sci. Comput., 27 (2006), pp. 1261–1286.

[3] S. Chandrasekaran, P. Dewilde, M. Gu, T. Pals, X. Sun, A. J. van der Veen, and D. White, *Some fast algorithms for sequentially semiseparable representations*, SIAM J. Matrix Anal. Appl., 27 (2005), pp. 341–364.

[4] S. Chandrasekaran and M. Gu, *A fast and stable solver for recursively semi-separable systems of equations*, in Structured Matrices in Mathematics, Computer Science and Engineering II, Contemporary Mathematics, V. Oslshevsky, ed., AMS, Providence, RI, 2001.

[5] S. Chandrasekaran, M. Gu, and W. Lyons, *A fast and stable adaptive solver for hierarchically semi-separable representations*, Calcolo, 42 (2005), pp. 171–185.

[6] S. Chandrasekaran, M. Gu, and T. Pals, *A fast ULV decomposition solver for hierarchically semiseparable representations*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 603–622.

[7] S. Delvaux and M. Van Barel, *Structures preserved by Schur complementation*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 229–252.

[8] S. Delvaux and M. Van Barel, *A Givens-weight representation for rank structured matrices*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 1147–1170.

[9] P. Dewilde and D. Alpay, *Time-varying signal approximation and estimation*, in Proceedings of the International Symposium MTNS-89, Vol. III, Birkhäuser Verlag, Berlin, 1990, pp. 1–22.

[10] P. Dewilde and A. van der Veen, *Time-Varying Systems and Computations*, Kluwer Academic Publishers, Boston, 1998.

[11] T. Dupont, R. P. Kendall, and H. H. Rachford, Jr., *An approximate factorization procedure for solving self-adjoint elliptic difference equations*, SIAM J. Numer. Anal., 5 (1968), pp. 559–573.

[12] H. Dym, D. Alpay, and P. Dewilde, *Lossless inverse scattering and reproducing kernels for upper triangular matrices*, in Extension and Interpolation of Linear Operators and Matrix Functions, Oper. Theory Adv. Appl., I. Gohberg, ed., Birkhäuser, Basel, 1990, pp. 61–135.

[13] A. George, *Nested dissection of a regular finite element mesh*, SIAM J. Numer. Anal., 10 (1973), pp. 345–363.

[14] I. Gohberg and Y. Eidelman, *A modification of the Dewilde van der Veen method for inversion of finite structured matrices*, Linear Algebra Appl., 343 (2001), pp. 419–450.

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

[16] W. Hackbusch, *A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part* I*: Introduction to $\mathcal{H}$-matrices*, Computing, 62 (1999), pp. 89–108.

[17] W. Hackbusch and B. N. Khoromskij, *A sparse arithmetic based on $\mathcal{H}$-matrices. Part* II*: Application to multi-dimensional problems*, Computing, 64 (2000), pp. 21–47.

[18] W. Hackbusch and B. N. Khoromskij, *A sparse $\mathcal{H}$-matrix arithmetic: General complexity estimates*, J. Comput. Appl. Math., 125 (2000), pp. 479–501.

[19] T. Kailath, *Fredholm resolvents, Wiener-Hopf equations, and Riccati differential equations*, IEEE Trans. Inform. Theory, 15 (1969), pp. 665–672.

[20] M. S. Khaira, G. L. Miller, and T. J. Sheffler, *Nested Dissection: A Survey and Comparison of Various Nested Dissection Algorithms*, Technical report CMU-CS-106R, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1992.

[21] I. Koltracht, I. Gohberg, and T. Kailath, *Linear complexity algorithms for semiseparable matrices*, Integral Equations Operator Theory, 8 (1985), pp. 780–804.

[22] W. Lyons, H. D. Cecineros, S. Chandrasekaran, and M. Gu, *Fast algorithms for spectral collocation with non-periodic boundary conditions*, J. Comput. Phys., 207 (2005), pp. 173–191.

[23] Y. Notay, *DRIC: A dynamic version of the RIC method*, Numerical Linear Algebra Appl., 1 (1994), pp. 511–532.

[24] A. J. van der Veen, *Approximate inversion of a large semiseparable positive matrix*, in Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS-04), Brussels, July 2004.

[25] A. J. van der Veen and P. Dewilde, *Inner-outer factorization and the inversion of locally finite systems of equations*, Linear Algebra Appl., 313 (2000), pp. 53–100.

[26] R. Vandebril, M. Van Barel, and N. Mastronardi, *Matrix Computations and Semiseparable Matrices, Volume* I*: Linear Systems*, Johns Hopkins University Press, Baltimore, MD, 2008.

[27] R. Vandebril, M. Van Barel, and N. Mastronardi, *Matrix Computations and Semiseparable Matrices, Volume* II*: Eigenvalue and Singular Value Methods*, Johns Hopkins University Press, Baltimore, MD, 2008.