Electronic Transactions on Numerical Analysis.
Volume 54, pp. 51–67, 2021.

ISSN 1068–9613.
DOI: 10.1553/etna_vol54s51

ETNA
Kent State University and
Johann Radon Institute (RICAM)

HYPERGRAPH EDGE ELIMINATION—A SYMBOLIC PHASE FOR HERMITIAN EIGENSOLVERS BASED ON RANK-1 MODIFICATIONS*

KARSTEN KAHL† AND BRUNO LANG†

Abstract. It is customary to identify sparse matrices with the corresponding adjacency or incidence graphs. For the solution of a linear system of equations using Gaussian elimination, the representation by its adjacency graph allows a symbolic factorization that can be used to predict memory footprints and enables the determination of near-optimal elimination orderings based on heuristics. The Hermitian eigenvalue problem on the other hand seems to evade such treatment at first glance due to its inherent iterative nature. In this paper we prove this assertion wrong by revealing a tight connection of Hermitian eigensolvers based on rank-1 modifications with a symbolic edge elimination procedure. A symbolic calculation based on the incidence graph of the matrix can be used in analogy to the symbolic phase of Gaussian elimination to develop heuristics which reduce memory footprint and computations. Yet, we also show that the question of an optimal elimination strategy remains NP-complete, in analogy to the linear systems case.

Key words. symmetric eigenvalue problem, hypergraphs, Gaussian elimination

AMS subject classifications. 65F15, 05C50, 05C65, 68R10

1. Introduction. The divide-and-conquer algorithm is a well-known method for computing the eigensystem (eigenvalues and, optionally, associated eigenvectors) of a Hermitian tridiagonal matrix [5, 7, 8]. It can be parallelized efficiently [3, 8], and even serially it is among the fastest algorithms available [1, 7].

The method relies on the fact that if the eigensystem of a Hermitian matrix A_0 is known, then the eigenvalues of a "rank-1 modification" (or "rank-1 perturbation") of this matrix, $A_1 = A_0 + \rho zz^H$, can be determined efficiently by solving the so-called "secular equation" [4, 11], and A_1's eigenvectors can also be obtained stably from those of A_0 [12].

In the tridiagonal case this can be used to zero out a pair of off-diagonal entries $t_{k+1,k}$ and $t_{k,k+1} = \overline{t_{k+1,k}}$ near the middle of the tridiagonal matrix T such that T decomposes into two half-sized matrices and a rank-1 modification,

$$T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} \pm t_{k+1,k} zz^H,$$

with a vector z containing nonzeros at positions k and $k+1$ and zeros elsewhere. Having computed the eigensystems of T_1 and T_2 (possibly by a recursive application of the same scheme), the eigensystem of T is obtained from these using the rank-1 machinery.

In this work we extend this method to a more general setting. In Section 2 we show that the eigensystem of a Hermitian matrix can be computed via a sequence of rank-1 modifications, each of them removing entries of the matrix until a diagonal matrix is reached. Section 3 reviews some of the theory for rank-1 modifications, as far as it is essential for the subsequent discussion.

While this approach in principle also works for full matrices, it benefits heavily from sparsity. In Section 4 we show that the necessary work for a whole sequence of rank-1 modifications can be modelled in a graph setting, similarly to the fill-in arising in direct solvers for Hermitian positive definite linear systems; cf., e.g., [6, 10]. However, the removal of nodes

*Received January 15, 2020. Accepted September 28, 2020. Published online on November 12, 2020. Recommended by Thomas Mach. This work was partially funded by the DFG in the SFB TRR-55.
†University of Wuppertal, IMACM, Mathematics and Natural Sciences, D-42097 Wuppertal, Germany ({kkahl,lang}@math.uni-wuppertal.de).

ETNA
Kent State University and
Johann Radon Institute (RICAM)

from the graph associated with the matrix is not sufficient to fully describe the progress of the eigensolver; here, the removal of *edges* in hypergraphs [17] provides a natural description.

We present two ways to come back to node elimination. In Section 5 we consider the dual hypergraph, and in Section 6 we will see that the edge elimination is closely related to Gaussian elimination for the so-called edge-edge adjacency matrix (and thus to node elimination for the graph associated with that matrix). In particular, an NP-completeness result will be derived from this relation in Section 7. This result implies that, for a given sequence of rank-1 modifications, it will not be practical to determine an ordering of this sequence that is optimal in a certain sense.

Nevertheless, the hypergraph-based models allow us to devise heuristics for choosing among the possible sequences of rank-1 modifications one such that the overall consumption of resources is reduced. In Section 8 we discuss heuristics for the elimination orderings to reduce memory footprint and computations.

Throughout the paper we assume that $A \in \mathbb{C}^{n\times n}$ is Hermitian. The presentation is aimed at sparse matrices, but "sparsity" is to be understood in the widest sense, including full matrices.

2. Successive edge elimination. We first show that the Hermitian eigenvalue problem $AQ = Q\Lambda$ can be solved by a series of rank-1-modified eigenvalue problems. One way to do this is to have each rank-1 modification remove one pair of nonzero off-diagonal entries $a_{k,\ell}$ and $a_{\ell,k} = \overline{a_{k,\ell}}$, which in turn correspond to one edge of the undirected graph associated with A. Thus we first introduce the basic graph notation we require.

DEFINITION 2.1. *The undirected* adjacency graph $G_A = (V, E)$ *with vertex set* V *and edge set* E *that is associated with* $A \in \mathbb{C}^{n\times n}$ *is defined by*

$$V = \{1, \ldots, n\} \quad \textit{and} \quad E = \{\{k, \ell\} \subseteq V \mid k \neq \ell,\ a_{k,\ell} \neq 0\}.$$

As our method treats matrix entries by conjugate pairs and maintains hermiticity throughout, it is sufficient to consider only entries in the upper triangle of the matrix. We will therefore assume that $k < \ell$.

DEFINITION 2.2. *For each edge* $\{k, \ell\} \in E$ *with* $a_{k,\ell} = r_{k,\ell} \cdot e^{i\theta_{k,\ell}} \in \mathbb{C}$, *where* $r_{k,\ell} = |a_{k,\ell}|$ *and* $\theta_{k,\ell} \in [0, 2\pi)$, *we define a* vector representation $z_{\{k,\ell\}} \in \mathbb{C}^n$ of the edge *by*

$$\left(z_{\{k,\ell\}}\right)_j = \begin{cases} e^{i\theta_{k,\ell}} & \textit{if } j = k, \\ 1 & \textit{if } j = \ell, \\ 0 & \textit{otherwise}. \end{cases}$$

Using these vectors we can rewrite A as a sum of rank-1 modifications of a diagonal matrix.

LEMMA 2.3. *Let* $A \in \mathbb{C}^{n\times n}$ *be sparse and Hermitian and* $G_A = (V, E)$ *its associated graph. Then*

$$A = D + \sum_{\{k,\ell\}\in E} r_{k,\ell} \cdot z_{\{k,\ell\}} z_{\{k,\ell\}}^H, \tag{2.1}$$

where $D = \operatorname{diag}(d_1, \ldots, d_n)$ *with*

$$d_i \ = \ a_{i,i} - \sum_{\{k,\ell\}\in E,\, i\in\{k,\ell\}} r_{k,\ell} \ = \ a_{i,i} - \sum_{j=1, j\neq i}^{n} |a_{i,j}|. \tag{2.2}$$

Proof. For each edge $\{k,\ell\}$ with $k<\ell$, the rank-1 matrix $r_{k,\ell}\cdot z_{\{k,\ell\}}z_{\{k,\ell\}}^H$ is nonzero only at the four positions $(k,\ell)\times(k,\ell)$, where we find

$$\left(r_{k,\ell}\cdot z_{\{k,\ell\}}z_{\{k,\ell\}}^H\right)_{(k,\ell)\times(k,\ell)} = r_{k,\ell}\begin{bmatrix}1 & e^{i\theta_{k,\ell}}\\ e^{-i\theta_{k,\ell}} & 1\end{bmatrix} = \begin{bmatrix} r_{k,\ell} & a_{k,\ell}\\ \overline{a_{k,\ell}} & r_{k,\ell}\end{bmatrix}.$$

Thus the ith diagonal entry is modified only by those edges starting or ending at node i, which gives the first equality in equation (2.2). The second equality is a direct consequence of the definition of E and the hermiticity of A. ∎

REMARK 2.4. The entries of D in equation (2.1) correspond to the lower bounds of the Gershgorin intervals. By defining $z_{\{k,\ell\}}$ differently, $(z_{\{k,\ell\}})_k = -ie^{i\theta_{k,\ell}}$ and $(z_{\{k,\ell\}})_\ell = i$, the rank-1 modifications become $(-|a_{k,\ell}|)\cdot z_{\{k,\ell\}}z_{\{k,\ell\}}^H$, and thus one can also obtain a representation of A similar to equation (2.1) such that the entries of D correspond to the upper bounds of the Gershgorin intervals.

The solution of the Hermitian eigenvalue problem starting from equation (2.1) is now straightforward. Fixing an ordering of the edges, i.e., defining $E=\{e_1,\ldots,e_{|E|}\}$, we have

$$A = \left(D + r_{e_1}\cdot z_{e_1}z_{e_1}^H\right) + \sum_{j=2}^{|E|} r_{e_j}\cdot z_{e_j}z_{e_j}^H. \tag{2.3}$$

Assuming that the eigendecomposition of the Hermitian matrix $D + r_{e_1}\cdot z_{e_1}z_{e_1}^H$ has been computed,

$$D + r_{e_1}\cdot z_{e_1}z_{e_1}^H = Q_1D_1Q_1^H$$

with Q_1 unitary, we can rewrite equation (2.3) as

$$A = Q_1\left(D_1 + \sum_{j=2}^{|E|} r_{e_j}\cdot\left(Q_1^Hz_{e_j}\right)\left(Q_1^Hz_{e_j}\right)^H\right)Q_1^H,$$

i.e., we eliminated edge e_1 from equation (2.1). Successive elimination of the remaining $|E|-1$ edges involving the vector $Q_{j-1}^H\cdots Q_1^H\cdot z_{e_j} = (\prod_{i=1}^{j-1}Q_i)^H\cdot z_{e_j}$ in step j, finally yields the eigendecomposition of A,

$$A = \left(\prod_{j=1}^{|E|}Q_j\right)D_{|E|}\left(\prod_{j=1}^{|E|}Q_j\right)^H.$$

This approach is summarized in Algorithm 2.1.

In order to be able to compute the eigendecomposition in this way, we need an efficient way to solve eigenproblems of the kind “diagonal plus rank-1 matrix”. It is well known that these problems can be easily dealt with in terms of the secular function, as we review in Section 3. In order to come up with a symbolic representation of the elimination procedure, we have to analyze the effect of the elimination of a particular edge e_j on the remaining edges. This symbolic representation is developed in Section 4.

3. Computing eigenvalues of rank-1-modified diagonal matrices. In order to clarify the main tool needed throughout the remainder of this work, we review some classical results about the eigenvalues of rank-1 perturbed matrices. The results cited here date back to [9]

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Algorithm 2.1: Successive edge elimination.

Write $A = D_0 + \sum\limits_{e \in E} r_e \cdot z_e z_e^H$.
Choose an ordering of the edges $e_1, e_2, \ldots, e_{|E|}$.
Set $Q = I$.
for $j = 1$ **to** $|E|$
  Calculate the eigendecomposition of $D_{j-1} + r_{e_j} \cdot z_{e_j} z_{e_j}^H = Q_j D_j Q_j^H$.
  for $k = j+1$ **to** $|E|$
    $z_{e_k} = Q_j^H \cdot z_{e_k}$.
  $Q = QQ_j$.

and are also contained in [15, pp. 94–98]. They were later on used in [4, 5] to formulate the divide-and-conquer method for tridiagonal eigenproblems.

THEOREM 3.1 ([4, Theorem 1]). *Let $D + \rho z z^H = Q \Lambda Q^H$ be the eigendecomposition of the rank-1-modified matrix, where $D = \operatorname{diag}(d_1, \ldots, d_n) \in \mathbb{R}^{n \times n}$ with $d_1 \leq d_2 \leq \ldots \leq d_n$, $\|z\| = 1$, and $\rho > 0$. Then the diagonal entries of $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ are the roots of the "secular equation"*

$$f(\lambda) = 1 + \rho \sum_{j=1}^{n} \frac{|z_j|^2}{d_j - \lambda}. \tag{3.1}$$

More specifically, let the λ_j be ordered, $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Then it holds that

$$\lambda_j = d_j + \rho \mu_j \quad \textit{with} \quad 0 \leq \mu_j \leq 1 \quad \textit{for } j = 1, \ldots, n \quad \textit{and} \quad \sum_{j=1}^{n} \mu_j = 1.$$

Note that the requirement $\|z\| = 1$ can be dropped by replacing ρ with $\rho / \|z\|^2$. There are two important consequences of Theorem 3.1 found in [15, pp. 94–98].

LEMMA 3.2. *Using the same notation as in Theorem 3.1, we obtain the following:*

1. *In case the eigenvalues of D are pairwise distinct, we find that $\lambda_j = d_j$ if and only if $z_j = 0$.*
2. *In addition, if all $z_j \neq 0$, then we find that $d_j < \lambda_j < d_{j+1}$, $j = 1, \ldots, n$ ($d_{n+1} = \infty$).*
3. *Assume there exists a multiple eigenvalue d_j of D with multiplicity k; without loss of generality $d_{j-k+1} = \ldots = d_{j-1} = d_j$ and $\|z_{j-k+1,\ldots,j}\| \neq 0$. Then we find*

$$\lambda_i = d_i,\ i = j-k+1, \ldots, j-1, \quad \textit{and} \quad d_j < \lambda_j < d_{j+1} \quad (d_{n+1} = \infty).$$

Lemma 3.2 is one of the key algorithmic ingredients of the divide-and-conquer algorithm for tridiagonal eigenproblems and leads to a technique known as "deflation."

As described in [8] and exploited in the implementation of the divide-and-conquer method, the root-finding problem of equation (3.1) is highly parallel and can be efficiently solved by a modified Newton iteration using hyperbolae instead of linear ansatz functions.

Recall that in our context the vector for the jth rank-1 modification (elimination of e_j) is $(\prod_{i=1}^{j-1} Q_i)^H \cdot z_{e_j}$. Therefore, Lemma 3.2 implies that this elimination only requires the solution of the secular equation in at most

$$N_{e_j} = \operatorname{nnz}\left((\prod_{i<j} Q_i)^H \cdot z_{e_j} \right) \tag{3.2}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

intervals, where $\mathrm{nnz}(v)$ is the number of nonzero entries of a vector v. That is, at most N_{e_j} of the entries of D_{j-1} (i.e., eigenvalue approximations) change from D_{j-1} to D_j. Further, by Theorem 3.1, we obtain that all eigenvalues move in the same direction, and the total displacement of these eigenvalues is given by $r_{e_j} \cdot \|(\prod_{i<j} Q_i)^H \cdot z_{e_j}\|^2 = 2|a_{e_j}|$ because $r_{e_j} = |a_{e_j}|$ and the norm of the vector $\|z_{e_j}\| = \sqrt{2}$ does not change under the orthogonal transformation $(\prod_{i<j} Q_i)^H$.

Using the above reasoning, one would be able to estimate the cost of the overall elimination process for a given ordering of the edges *if* the number of nonzeros in the vectors $(\prod_{i<j} Q_i)^H z_{e_j}$ could be *predicted*. In the following section we show how to do this.

Being able to analyze the influence of the ordering of the edges on the complexity of the calculations (in terms of the number of roots of the secular equations that need to be calculated) also allows us to determine an ordering that leads to low overall cost. This topic is discussed in Sections 7 and 8.

4. Edge elimination, hypergraphs, and edge elimination in hypergraphs. In Section 2 we have seen that the eigendecomposition of a Hermitian (sparse) matrix A can be obtained by successively eliminating the edges $e_1, e_2, \dots, e_{|E|}$ of the graph G_A associated with the matrix A.

It is well known that in the context of Gaussian elimination for Hermitian positive definite matrices, the effect of eliminating one *node* (corresponding to selecting a pivot row and performing the row additions with this row) directly shows in the (undirected) graph G_A: removing the node and connecting all its former neighbors introduces exactly those edges that correspond to the new fill-in produced by the row operations [6, 10]. This allows one to determine the nonzero patterns of the matrices during the whole Gaussian elimination process before executing any floating-point operation.

Something similar can be done to determine the nonzero patterns of the vectors

$$(\prod_{i<j} Q_i)^H z_{e_j}$$

resulting from preceding eliminations. However, as we are eliminating *edges*, the graph G_A is not adequate for this purpose. We have to generalize the concept of a graph and use what is known in the literature as a hypergraph [17].

DEFINITION 4.1. *An undirected* hypergraph $G = (V, E)$ *is defined by a set of vertices* $V = \{v_1, \dots, v_n\}$ *and a set of hyperedges* $E = \{e_1, \dots, e_m\}$*, where* $\emptyset \neq e_j \subseteq V$*.*

EXAMPLE 4.2. The hypergraph with vertex set $V = \{1, 2, 3, 4, 5\}$ and set of hyperedges $E = \{e_1, e_2, e_3, e_4\} = \{\{1, 2, 5\}, \{2, 3\}, \{1, 3, 4, 5\}, \{3, 4\}\}$ is depicted in Figure 4.1, where each hyperedge is indicated by a closed line that contains all its vertices.

REMARK 4.3. The possibility to have edges with more or fewer vertices than two is the only difference to the usual definition of an undirected graph. In particular, the graph G_A of the matrix can be considered as a hypergraph.

In order to analyze the nonzero pattern of the vector $(\prod_{i<j} Q_i)^H z_{e_j}$ for the jth rank-1 modification, we first note that this vector can be obtained in two ways: "left-looking", when it is needed, by accumulating all previous transformations Q_i^H $(i < j)$, or "right-looking", by applying each transformation Q_i^H, once it has been computed, to all later z_{e_k}. In the following discussion, as well as in Algorithm 2.1, the right-looking approach is used.

We now consider the effect of one such operation from the matrix/vector point of view. Let us assume that the edges are ordered and consider the elimination of the first edge, e_1. Assume without loss of generality that $e_1 = \{1, 2\}$. By definition, z_{e_1} has only two nonzero

ETNA
Kent State University and
Johann Radon Institute (RICAM)

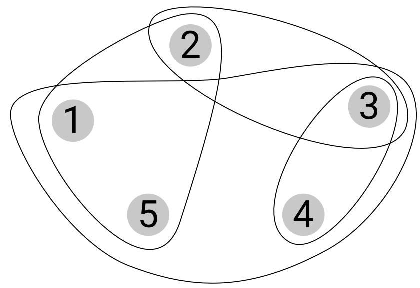

FIG. 4.1. *Drawing of the hypergraph defined in Example 4.2 with* $V = \{1, 2, 3, 4, 5\}$ *and hyperedges* $e_1 = \{1, 2, 5\}, e_2 = \{2, 3\}, e_3 = \{1, 3, 4, 5\}$*, and* $e_4 = \{3, 4\}$.

entries at the indices 1 and 2, and thus due to Theorem 3.1 and Lemma 3.2, we find

$$Q_1 = \begin{bmatrix} q_{11} & q_{12} & \\ q_{21} & q_{22} & \\ & & I_{(n-2)\times(n-2)} \end{bmatrix}.$$

Hence, for all edges e_j with $e_j \cap e_1 = \emptyset$, we have $Q_1^H \cdot z_{e_j} = z_{e_j}$. On the other hand, for all edges e_j with $e_j \cap e_1 \neq \emptyset$, we find that $Q_1^H \cdot z_{e_j}$ has entries at the indices $e_j \cup e_1$.

The situation for the ith elimination step is similar. Let the hyperedge e_j denote the nonzero pattern, i.e., the set of positions of the nonzeros of the current vector z_j (after the preceding transformations $Q_{i-1}^H \cdots Q_1^H \cdot z_j$). Then the transformed vector $Q_i^H \cdot z_j$ will have nonzeros at the same positions e_j if $e_j \cap e_i = \emptyset$ and at positions $e_j \cup e_i$ if the two hyperedges overlap.

REMARK 4.4. Strictly speaking this holds only if the transformation $Q_i^H \cdot z_j$ does not introduce new (“cancellation”) zeros in the vector. In the symbolic processing for sparse linear systems it is commonly assumed that this does not happen; we will do so as well.

We summarize the above observation in the following theorem.

THEOREM 4.5. *Let* $G = (V, E)$ *be an undirected hypergraph with* $E \neq \emptyset$*. Let* $x \in E$ *be the edge to be eliminated, and let*

$$E = E_x \cup E_{\not x}, \quad \text{where} \quad \begin{cases} E_x = \{e \in E \mid e \cap x \neq \emptyset\} & \text{and} \\ E_{\not x} = \{e \in E \mid e \cap x = \emptyset\}. \end{cases}$$

Then the hypergraph after elimination of x *is given by* $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ *with* $\widetilde{V} = V$ *and* $\widetilde{E} = \{e \cup x, e \in E_x \setminus \{x\}\} \cup E_{\not x}$.

Now it is easy to show that the subsequent elimination of all edges to compute the eigendecomposition as described in Section 2 is equivalent to the elimination of all edges in the same ordering from the hypergraph G_A as defined here (starting with the graph G_A). Thus it is natural to discuss questions such as complexity and optimal edge orderings in the “geometrical” context of these graphs as it has been successfully done for the solution of linear systems; e.g., optimal node orderings to reduce fill-in.

REMARK 4.6. In the above discussion we have assumed that each step of the algorithm eliminates a “true edge” $e = \{k, \ell\}$, zeroing a pair of matrix entries $a_{k,\ell}$ and $a_{\ell,k}$. However, this is not mandatory. Note that Theorem 4.5 describes the evolution of the nonzero patterns also if the eliminated edge x is a hyperedge as well with more than just 2×2 matrix entries being touched by the corresponding rank-1 modification. In addition, the (off-diagonal) matrix

ETNA
Kent State University and
Johann Radon Institute (RICAM)

entries at the positions $x \times x$ need not be zeroed out completely with the elimination. This allows for more general elimination strategies, including the extremes

- each rank-1 modification zeroes one off-diagonal pair of matrix entries (cf. Section 2), and
- the ith rank-1 modification zeroes the whole ith column and row of the matrix; this typically leads to the minimum *number* of rank-1 modifications, but according to the above, the operations $Q_i^H \cdot e_j$ will make the vectors dense very quickly,

as well as many intermediate variants. For example, if the underlying model leads to low-rank off-diagonal blocks in the matrix, then these can be removed with a reduced number of steps: for a size-($r \times s$) block of rank ρ, ρ rank-1 modifications (with identical hyperedges) are sufficient instead of $r \cdot s$. We will come back to this generalization in Section 7.

5. Duality between edge elimination and node elimination. In this section we will show that edge elimination can also be expressed as *node* elimination in a suitable hypergraph. This requires some preparation.

DEFINITION 5.1. *Let $G = (V, E)$ be a hypergraph with nodes $V = \{v_1, \ldots, v_n\}$ and hyperedges $E = \{e_1, \ldots, e_m\}$. The* (node-edge) incidence matrix $I_{VE} \in \mathbb{R}^{|V|\times|E|}$ *of G is then defined by*

$$(I_{VE})_{ij} = \begin{cases} 1, & \text{if } v_i \in e_j, \\ 0, & \text{otherwise}, \end{cases}$$

and the adjacency matrices of the hypergraph *are given by*

$$\begin{aligned} A_V &= I_{VE} \cdot I_{VE}^T \in \mathbb{R}^{|V|\times|V|} \quad &(\text{vertex-vertex adjacency matrix}), \\ A_E &= I_{VE}^T \cdot I_{VE} \in \mathbb{R}^{|E|\times|E|} \quad &(\text{edge-edge adjacency matrix}). \end{aligned}$$

The latter two names are explained by the following lemma:

LEMMA 5.2. *Given a hypergraph $G = (V, E)$, its adjacency matrices have the properties*

$$(A_V)_{ij} \neq 0 \quad \text{iff} \quad \text{there exists } e \in E \text{ such that } v_i, v_j \in e,$$

i.e., nodes v_i and v_j are connected by at least one hyperedge, and

$$(A_E)_{ij} \neq 0 \quad \text{iff} \quad \text{there exists } v \in V \text{ such that } v \in e_i \cap e_j,$$

i.e., the hyperedges e_i and e_j share at least one node v.

Proof. This follows immediately from Definition 5.1 and the calculation of matrix-matrix products due to

$$\left(I_{VE} \cdot I_{VE}^T\right)_{ij} = \sum_{k=1}^{|E|} (I_{VE})_{i,k} \cdot \left(I_{VE}^T\right)_{k,j} = \sum_{k=1}^{|E|} (I_{VE})_{i,k} \cdot (I_{VE})_{j,k}$$

and similarly for A_E. ☐

We also note that the transpose of the incidence matrix of G, I_{VE}^T, is also the incidence matrix I_{VE}^* of the dual of the hypergraph, which is defined as follows.

DEFINITION 5.3. *Let $G = (V, E)$ be a hypergraph with nodes $V = \{v_1, \ldots, v_n\}$ and hyperedges $E = \{e_1, \ldots, e_m\}$. Then the* dual of G *is a hypergraph $G^* = (V^*, E^*)$ with nodes $V^* = \{v_1^*, \ldots, v_m^*\}$ and hyperedges $E^* = \{e_1^*, \ldots, e_n^*\}$ such that*

$$e_i^* = \{v_j^* \in V^* \,:\, v_i \in e_j\}.$$

By construction, edge elimination in a hypergraph is equivalent to node elimination in its dual, as can be seen as well in the following small example.

EXAMPLE 5.4. Consider the hypergraph $G = (V, E)$ of Example 4.2 (Figure 4.1) and its dual $G^* = (V^*, E^*)$ given by their incidence matrices I_{VE} and I^*_{VE}, respectively:

$$I_{VE} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}, \qquad I^*_{VE} = I^T_{VE} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

If we eliminate edge e_1 in G or, equivalently, node v^*_1 in G^*, then the resulting hypergraphs $\widetilde{G}$ and $\widetilde{G^*}$ are given by

$$\widetilde{I_{VE}} = \begin{bmatrix} & \mathbf{1} & 1 & 0 \\ & 1 & \mathbf{1} & 0 \\ & 1 & 1 & 1 \\ & 0 & 1 & 1 \\ & \mathbf{1} & 1 & 0 \end{bmatrix}, \qquad \widetilde{I^*_{VE}} = \begin{bmatrix} & & & & \\ \mathbf{1} & 1 & 1 & 0 & \mathbf{1} \\ 1 & \mathbf{1} & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix},$$

with boldface entries representing the growth of the hyperedges and their duals through the elimination. Note that "node elimination" in a hypergraph is not the same as standard node elimination in a graph; it corresponds to merging the top row into all non-disjoint rows of the matrix I^*_{VE}.

There is another way to describe edge elimination in G_A as node elimination in a suitable *graph*, and since this corresponds to a square matrix with symmetric nonzero pattern, it allows one to draw on the results available for the solution of sparse symmetric positive definite linear systems [6, 10]. To this end we take a closer look at the edge-edge adjacency matrix A_E, more specifically, at the process of executing Gaussian elimination for that matrix.

6. Gaussian elimination for the edge-edge adjacency matrix. Let $G = (V, E)$ denote a hypergraph. We now investigate how eliminating one of G's edges changes the nonzero pattern in the edge-edge adjacency matrix.

Let us first consider the symbolic elimination of an edge x, as defined in Section 4. This elimination amounts to the following changes:

$$e \in E \setminus \{x\} \to \begin{cases} e \cup x, & \text{if } e \cap x \neq \emptyset, \\ e, & \text{otherwise.} \end{cases}$$

In particular this implies that all edges $e \in E \setminus \{x\}$ with $e \cap x \neq \emptyset$ share all vertices of x after its elimination. Thus, in terms of the edge-edge adjacency matrix A_E, the elimination results in a full block of nonzero entries covering all $e \in E \setminus \{x\}$ with $e \cap x \neq \emptyset$.

On the other hand let us consider one step of symbolic Gaussian elimination applied to the edge-edge adjacency matrix and note that A_E is symmetric. Without loss of generality let us assume that A_E is permuted such that the edge x is listed first. Nonzero entries in the first column of A_E then correspond to edges e that share at least one vertex with x, i.e., for which $e \cap x \neq \emptyset$. Thus, in the symbolic elimination step we now have to merge the nonzero pattern of the first matrix row into the nonzero pattern of each row corresponding to an edge e with $e \cap x \neq \emptyset$. Due to symmetry this again results in a full block of nonzeros covering these edges (a clique in the graph G_{A_E} associated with the matrix) and corresponds exactly to the nonzero pattern generated by the symbolic edge elimination.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Thus, in terms of the edge-edge connectivity structure, the symbolic edge elimination process is equivalent to a symbolic Gaussian elimination applied to the edge-edge adjacency matrix. Therefore this source of complexity, caused by increasing connectivity among the remaining edges, can be approached in the same way as it is done in Gaussian elimination applied to sparse linear systems of equations.

Unfortunately, this does not cover all of the complexities of the process. If a fill-in element appears in A_E during Gaussian elimination, then this merely signals that all nodes from the hyperedge e_j will be joined to those of e_i. Therefore, the overall fill-in reflects the number of times when some hyperedge will grow. It does, however, not convey information about the current number of nodes in the hyperedges, which would be necessary for assessing the cost for the corresponding rank-1 modification; see equation (3.2).

7. NP-completeness results. In this section we will show that even the problem of minimizing the "number of growths" is NP-complete.

This follows directly from a well-known result stating the NP-completeness of fill-in minimization [16] together with the following lemma. (Recall that a symmetric matrix is irreducible if its graph is connected.)

LEMMA 7.1. *The nonzero pattern of any symmetric positive definite irreducible n-by-n matrix can be interpreted as the edge-edge adjacency matrix of a suitable hypergraph $G = (V, E)$ with $|E| = n$ edges.*

Proof. Define

$$V = \{v_{i,j} \mid i > j, a_{i,j} \neq 0\},$$

that is, we have one node for each nonzero in the strict lower triangle of A. Let the set of hyperedges be $E = \{e_1, \dots, e_n\}$, where

$$e_j = \{v_{i,j} \mid i > j, a_{i,j} \neq 0\} \cup \{v_{j,i} \mid j > i, a_{j,i} \neq 0\},$$

i.e., e_j contains just those nodes corresponding to nonzeros in column j or row j of A's strict lower triangle. Note that $e_j \neq \emptyset$ because otherwise the jth row and column of A would contain just the diagonal entry, i.e., A were reducible. Then, for $k > j$ we have

$$e_k \cap e_j = \{v_{i,j} \mid i > j, a_{i,j} \neq 0\} \cap \{v_{k,i} \mid k > i, a_{k,i} \neq 0\}$$

(the other three intersections being empty), and this is nonempty iff there is a node $v_{i,j} \equiv v_{k,i}$ in both column j and row k, i.e., $a_{k,j} \neq 0$. Using Lemma 5.2, this implies that A and $A_E = I_{VE}^H I_{VE}$ have the same nonzero pattern. □

REMARK 7.2. In most cases, the same nonzero pattern may also be obtained with hypergraphs containing fewer nodes. It is therefore tempting to take I_{VE} to be the nonzero pattern of the Cholesky factor U from $A = U^H U$ in order to obtain the sparsity pattern of A with a hypergraph containing just n nodes. Unfortunately, cancellation in the product $U^H U$ may introduce zeros in A that are not present in the product $I_{VE}^H I_{VE}$ obtained this way, and this cancellation can be structural. In fact, exhaustive search reveals that, for $n = 5$, the pattern

$$A = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

cannot be obtained as $I_{VE}^H I_{VE}$ with any hypergraph containing fewer than six nodes, and six nodes are sufficient according to the proof of Lemma 7.1 because the strict lower triangle of A contains six nonzeros.

Note that for Lemma 7.1 we have assumed that we may start with a hypergraph; cf. Remark 4.6. If this is not allowed and we restrict ourselves to eliminating "true edges," thus zeroing one pair of matrix entries $a_{k,\ell}$ and $a_{\ell,k}$ per step, then a simple combinatorial argument shows that there must be symmetric positive definite matrices whose nonzero pattern cannot be interpreted as that of an edge-edge adjacency matrix $A_E = I_{VE}^T \cdot I_{VE}$ to *any* graph $G = (V, E)$.

To see this, we note that the number of nonzero patterns for a symmetric n-by-n matrix A is $\nu_A = 2^{n(n-1)/2} = \left(2^{(n-1)/2}\right)^n$, because each of the $n(n-1)/2$ entries in the strict lower triangle may be zero or not. Now assume that the matrix has the same nonzero pattern as $I_{VE}^T \cdot I_{VE}$ for some graph $G = (V, E)$ with n edges and some number of nodes v. Then $I_{VE} \in \mathbb{R}^{v \times n}$ contains exactly two nonzeros in each of its columns, and we may assume without loss of generality that $v \le 2n$, because at most $2n$ rows of I_{VE} can contain a nonzero, and rows with all zeros can be removed without affecting the product $I_{VE}^T \cdot I_{VE}$ (this corresponds to removing isolated nodes from G). Then there are at most $\binom{2n}{2} = 2n(2n-1)/2$ possible combinations for the positions of the two nonzeros in each column of I_{VE}, leading to the overall number of possible matrices I_{VE} being bounded by $\nu_{I_{VE}} \le \left(\frac{2n(2n-1)}{2}\right)^n$. Since $2^{(n-1)/2} > \frac{2n(2n-1)}{2}$ for large n, we also have $\nu_A > \nu_{I_{VE}}$, and therefore not all symmetric matrices can be interpreted as edge-edge adjacency matrices.

In this situation the proof of NP-completeness for fill-in minimization does not carry over, and it is currently not known whether this restricted problem is indeed NP-complete.

In the light of these results one still may try to find orderings that lead to reduced (arithmetic or memory) complexity without being optimal in the above sense. This will be discussed in the following.

8. Heuristics for choosing edge elimination orderings.

Based on the findings in Sections 2 and 4, it is natural to analyze the complexity of Algorithm 2.1 in terms of the overall number of roots of the secular equation that have to be calculated during all edge eliminations. Combining this analysis with the cost for the calculation of a single root of the secular equation gives us direct access to the complexity of the Hermitian (sparse) eigenvalue problem.

LEMMA 8.1. *Let $G_A = (V, E)$ be an undirected graph of a matrix A, interpreted as a hypergraph. Further define an ordering of the edges $e_1, \ldots, e_{|E|}$. Then the total number N of secular equation roots that have to be calculated in Algorithm 2.1 is given by*

$$N = \sum_{j=1}^{|E|} N_{e_j},$$

using the definition of N_{e_j} from equation (3.2).

Minimum incidence (MI) ordering. In analogy to the minimum-degree ordering in Gaussian elimination, the first heuristic that comes to mind accounts for the number of incident edges. In the hypergraph setting, two edges e and x are incident iff $e \cap x \neq \emptyset$, i.e., when by eliminating x, the edge e changes and vice versa. Introducing the quantities

$$\mu_{\mathrm{i}}(x) = |\{e \in E \mid e \cap x \neq \emptyset\}|,$$

the strategy thus chooses in every step the edge with the fewest incident edges. Once an edge x is eliminated, the number of incident edges needs to be updated only for all edges e that have been incident with x.

Minimal root number (MR) ordering. Another heuristic is to account for the number of roots of the secular equation that need to be calculated when eliminating a hyperedge. That is,

TABLE 8.1
Symbolic elimination for the chain graph with $N = 256$ nodes. Reported are the accumulated number of roots that need to be calculated over the whole elimination process. For comparison, the number of root calculations in the divide-and-conquer algorithm for this problem is $256 \times \log_2(256) = 2048$.

Heuristic	μ_{i}	μ_{r}	$\mu_{\mathrm{c}}^{(1)}$	$\mu_{\mathrm{c}}^{(2)}$
$\sum \lvert x\rvert$	16766	2048	2048	2152

we define the quantities

$$\mu_{\mathrm{r}}(x) = |x|,$$

and the MR strategy chooses in every step the edge with the smallest number of contained vertices. After the elimination of an edge, μ_{r} needs to be updated for all edges incident with the eliminated edge.

Minimal roots/costs with look-ahead (MC). The last heuristic under consideration modifies the MR heuristic by adding a look-ahead component. The elimination of an edge x incurs a growth of all edges e with $x \cap e \neq \emptyset$ by $|x \cup e| - |e|$ vertices. This in turn relates to the number of roots that need to be calculated in a future elimination. Due to the fact that the cost of eliminating an edge x with $|x|$ nodes is proportional to $|x|^2$, we consider the two measures

$$\mu_{\mathrm{c}}^{(k)}(x) = |x|^k + \sum_{e \cap x \neq \emptyset} |x \cup e|^k - |e|^k,$$

for $k = 1, 2$, and choose to eliminate the edge with the current smallest value of $\mu_{\mathrm{c}}^{(k)}$. Due to the look-ahead nature of the measure, updating it now involves not only the edges incident with x but also the next-neighbors as well.

In order to assess the efficiency of these heuristics, they have been applied to matrices with different sparsity patterns, i.e., different structures of the associated graph G_A.

I. The chain graph. In order to enable a comparison of our approach to the tridiagonal divide-and-conquer algorithm, we first apply the symbolic process to a chain of N nodes, which is the graph corresponding to a tridiagonal matrix. The divide-and-conquer strategy for this graph results in the calculation of N roots on each level of the recursion for a total of $N \log_2(N)$ roots.

As can be seen from the results in Table 8.1, both the strategy that chooses the edge with the currently smallest number of contained vertices, based on μ_{r}, as well as the strategy that accounts for the current and future cost of eliminating an edge, based on $\mu_{\mathrm{c}}^{(1)}$, result in elimination orderings which are equivalent to the divide-and-conquer strategy. While the strategy based on the measure $\mu_{\mathrm{c}}^{(2)}$ comes close to the optimal total number of roots, the strategy based on choosing to eliminate the edge with the least number of incident edges fails spectacularly and eliminates the edges in lexicographic ordering.

The progress of the elimination for a chain graph with $N = 8$ nodes is shown in Figure 8.1. Again, μ_{r} and $\mu_{\mathrm{c}}^{(1)}$ achieve the same $\sum |x|$ value as the tridiagonal divide-and-conquer approach, $N \log_2(N) = 24$, $\mu_{\mathrm{c}}^{(2)}$ is slightly worse ($\sum |x| = 25$), and μ_{i} leads to the lexicographic ordering ($\sum |x| = 35$).

II. Banded matrices. Banded matrices with (semi-)bandwidth b, i.e., $a_{k,\ell} = 0$ whenever $|\ell - k| > b$, can be handled in different ways.

Analogously to the tridiagonal case, we can start the elimination process with the graph G_A that is associated with the matrix and now contains $|E| = n - b(b + 1)/2$ edges $\{k, \ell\}$, where $k, \ell \in \{1, \dots, n\}$ and $k < \ell \leq k + b$. Thus, the elimination of each edge corresponds to zeroing one pair of entries within the band.

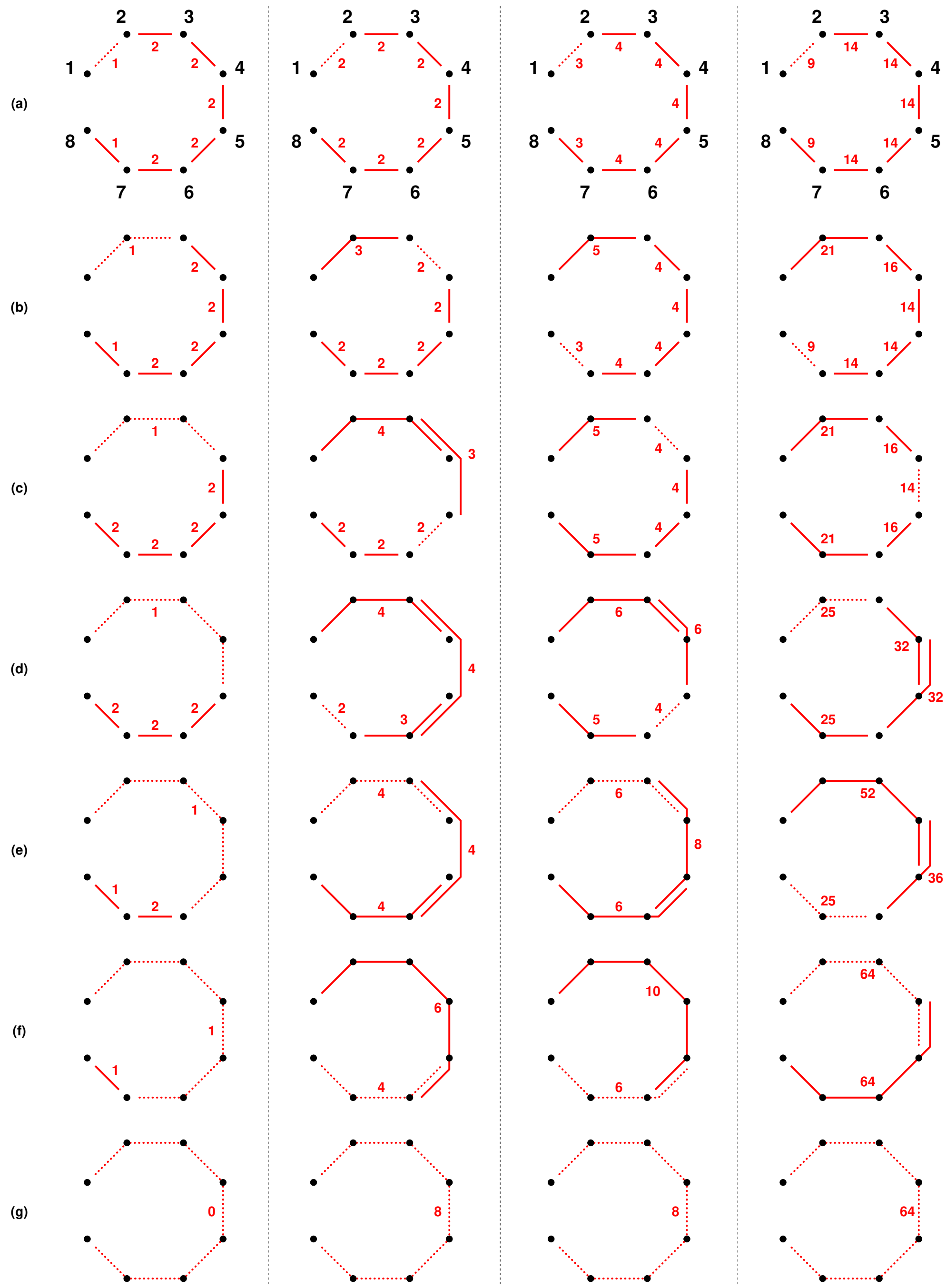

FIG. 8.1. *Order of (hyper)edge elimination for a size-8 tridiagonal matrix with the strategies (from left to right) Minimum incidence (MI, minimize $\mu_{\mathrm{i}}(x)$), Minimal root number (MR, minimize $\mu_{\mathrm{r}}(x)$), Minimal roots with look-ahead (MC1, minimize $\mu_{\mathrm{c}}^{(1)}(x)$), and Minimal costs with look-ahead (MC2, minimize $\mu_{\mathrm{c}}^{(2)}(x)$). For each of the seven elimination steps (a) to (g), the remaining (hyper)edges are shown together with their μ values, and the (hyper)edge selected for elimination is highlighted as a dotted line. If the minimum is not unique, then the "first" minimizing hyperedge (clockwise) is chosen for elimination.*

ETNA
Kent State University and
Johann Radon Institute (RICAM)

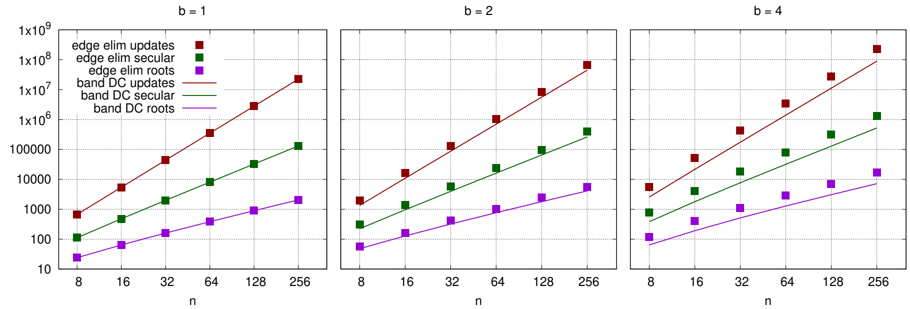

FIG. 8.2. *Overall number of roots (magenta), effort for the secular equations (green), and for the eigenvector updates (red) for bandwidths $b = 1$ (left), $b = 2$ (center), and $b = 4$ (right). Squares correspond to the μ_r heuristic, starting with the graph G_A, and solid lines correspond to the same heuristic, starting with the hypergraph H_A described in the main text.*

Divide-and-conquer algorithms for banded eigenvalue problems typically follow a different strategy; cf., e.g., [2]. Here, the matrix is considered to be block tridiagonal with some block size $m \geq b$, and the super-diagonal blocks $A_{j,j+1}$ are nonzero only in a $b \times b$ triangle at their lower left corner. These off-diagonal blocks are zeroed out by a sequence of b rank-1 modifications, where the length-$2b$ vectors $z = (u^T, v^T)^T$ can be determined from an SVD $A_{j,j+1} = U\Sigma V^*$ of that triangle. To allow for this procedure, we start with a "true" hypergraph H_A containing, for each j, a number of m identical size-m hyperedges $\{(j-1)m+1, \ldots, jm\}$ with the nodes of block $A_{j,j}$ (this covers the computation of $A_{j,j}$'s eigenvalues and -vectors) and b identical size-$2b$ hyperedges $\{jm - b + 1, \ldots, jm, jm + 1, \ldots, jm + b\}$ (covering the rank-1 modifications for zeroing $A_{j,j+1}$).

Figure 8.2 displays the results for both approaches (solid lines: starting with G_A; squares: starting with H_A) for bandwidths $b = 1$ (tridiagonal—here both approaches coincide), $b = 2$, and $b = 4$, and block size $m = b$. We report the statistics for the overall number of roots to compute, $\mathsf{roots} = \sum |x|$, the overall cost of the secular equations, $\mathsf{secular} = \sum |x|^2$, and the overall cost of the eigenvector updates, $\mathsf{updates} = \sum |x|^2 c_x$, where c_x is the number of columns in Q that are affected by eliminating x. All numbers are for the heuristic μ_r, which performed almost identically to $\mu_\mathrm{c}^{(1)}$ and slightly better than $\mu_\mathrm{c}^{(2)}$, whereas μ_i was much worse. The data clearly indicate that $\mathsf{roots} \sim n$, $\mathsf{secular} \sim n^2$, and $\mathsf{updates} \sim n^3$. A closer look reveals that μ_r, applied to H_A, indeed essentially leads to the banded divide-and-conquer scheme as described above. Since the overall work is dominated by the final length-n rank-1 modifications, this explains why roots, secular, and updates increase linearly with b when starting with H_A (the band divide-and-conquer does b full-length rank-1 modifications), whereas they are of order b^2 when starting with G_A (then the final $b(b + 1)/2$ rank-1 modifications are full-length). In particular, the complexity of the band divide-and-conquer scheme is lower by a factor $\approx (b + 1)/2$.

III. Structured graphs. Structured graphs are often encountered in discretizations of partial differential equations. The resulting graphs are planar and usually possess a large diameter. In Figure 8.3 we report results in terms of the accumulated number of roots $\sum |x|$ and the cost of root elimination $\sum |x|^2$ of the hypergraph edge elimination approach for a uniform 16×16 lattice. We compare the results for the four heuristics with a statistical baseline of 20 random elimination orderings. (Random orderings roughly correspond to processing the nonzeros just in the order in which they may have been inserted into a sparse data structure.) As can be seen from the figure, all four heuristics yield largely reduced cost measures compared to the baseline. Notably, the ordering of the heuristics in terms of the two

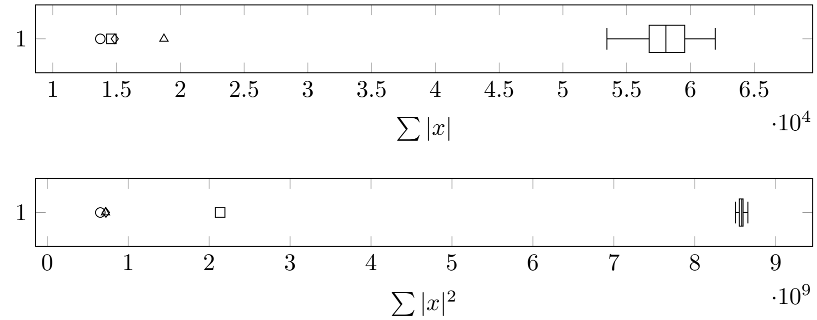

FIG. 8.3. *Accumulated number of roots* $\sum |x|$ *(top) and root calculation costs* $\sum |x|^2$ *(bottom) for a regular* 16×16 *lattice graph with* $N = 256$ *nodes. Results for the heuristics are plotted as* $(\mu_{\mathrm{i}}, \triangle)$, $(\mu_{\mathrm{r}}, \square)$, $(\mu_{\mathrm{c}}^{(1)}, \circ)$, *and* $(\mu_{\mathrm{c}}^{(2)}, \diamond)$ *(towards the left), and the boxplots close to the right summarize the results for 20 random elimination orderings.*

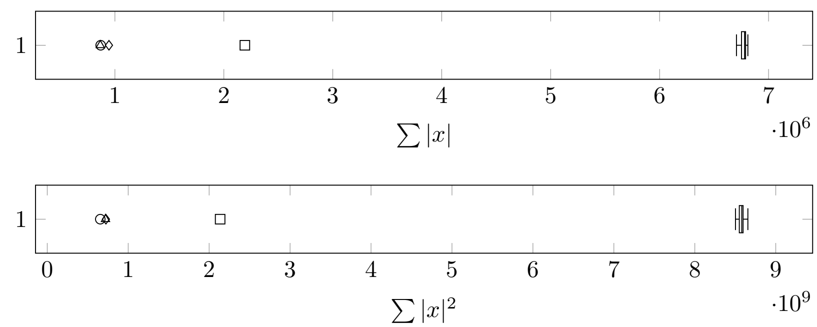

FIG. 8.4. *Accumulated number of roots* $\sum |x|$ *(top) and root calculation costs* $\sum |x|^2$ *(bottom) for a triangulation of the unit disc with* $N = 1313$ *nodes. Results for the heuristics are plotted as* $(\mu_{\mathrm{i}}, \triangle)$, $(\mu_{\mathrm{r}}, \square)$, $(\mu_{\mathrm{c}}^{(1)}, \circ)$, *and* $(\mu_{\mathrm{c}}^{(2)}, \diamond)$. *The boxplots represent results of* 20 *random elimination orderings.*

cost measures is not identical, i.e., an overall minimal number of accumulated root calculations does not immediately lead to a minimal accumulated root elimination cost.

Next we apply the same test setup to a graph that is a triangulation of the unit disc with 1313 nodes. In Figure 8.4 we report the accumulated number of roots $\sum |x|$ and the root elimination costs $\sum |x|^2$ for the four heuristics and report the statistical baseline of 20 random orderings. Again we see that all four heuristics are clearly better than using a random elimination ordering.

IV. Sparse random graphs. Finally we compare the heuristics for randomly generated graphs. We use the Matlab built-in function `sprandsym` to generate an undirected graph with N nodes with a nonzero density of $\frac{8}{N}$. The average degree of the resulting graphs is thus approximately 8. We now test the heuristics for 20 such graphs of size $N = 128$. In Figure 8.5 we report the number of edges of the matrices used in the tests.

In Figure 8.6 we report the results of the heuristics applied to these randomly generated sparse graphs. We report both the accumulated number of roots $\sum |x|$ as well as the accumulated cost of root calculations $\sum |x|^2$. In order to gauge the potential gains realized by the heuristics, we include boxplots of 20 random elimination orderings as well.

Overall, our experiments suggest that, while none of the proposed strategies is consistently superior, choosing the hyperedge with minimum $\mu_{\mathrm{c}}^{(1)}$-value for elimination seems to be a

ETNA
Kent State University and
Johann Radon Institute (RICAM)

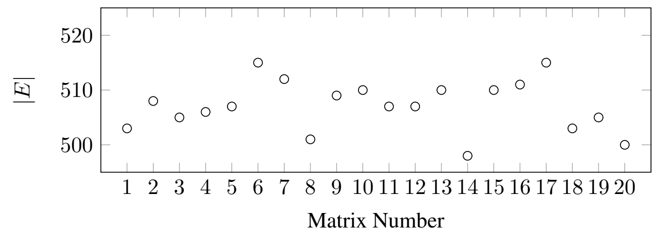

FIG. 8.5. *Number of edges $|E|$ of 20 randomly generated sparse graphs.*

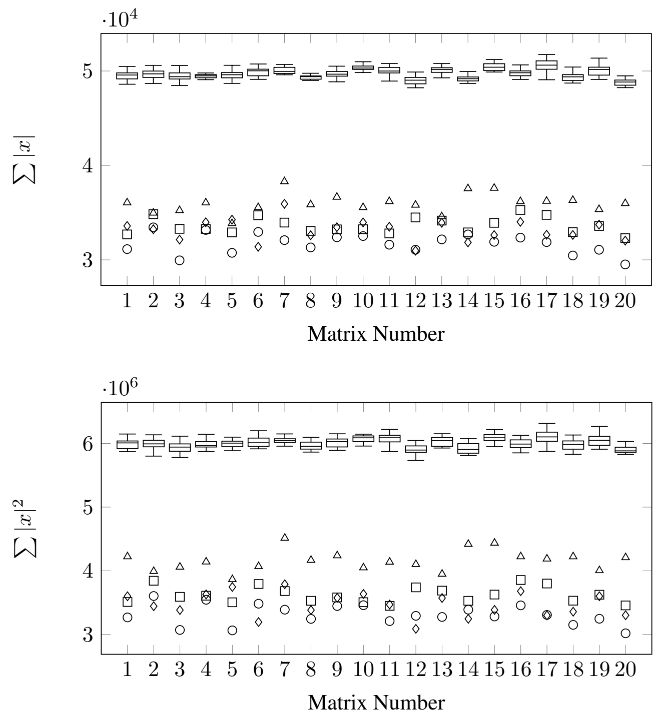

FIG. 8.6. *Accumulated number of roots $\sum |x|$ (top) and root calculation costs $\sum |x|^2$ (bottom) for the 20 randomly generated sparse graphs. Results for the heuristics are plotted as $(\mu_{\mathrm{i}}, \triangle)$, $(\mu_{\mathrm{r}}, \square)$, $(\mu_{\mathrm{c}}^{(1)}, \circ)$, and $(\mu_{\mathrm{c}}^{(2)}, \diamond)$. Each boxplot represents results of 20 random elimination orderings.*

reasonable way to reduce both cost measures, the total number of roots to compute, $\sum |x|$, and the operations to do this, $\sum |x|^2$.

9. Concluding remarks.

We have shown in this paper that symmetric eigensolvers based on rank-1 modifications can be interpreted as an elimination process, where all edges of the corresponding graph need to be eliminated. This symbolic equivalence is facilitated by a hypergraph point of view and in complete analogy to the vertex elimination that characterizes the symbolic solution of linear systems by means of Gaussian elimination.

Furthermore, we showed that the hypergraph information in every stage of the elimination process is captured by the symbolic Gaussian elimination applied to the edge-edge adjacency matrix—a formal dual to the regular vertex-vertex adjacency matrix. Exploiting this connection, we were able to transfer the result of NP-hardness for the calculation of an optimal elimination ordering from the linear systems case to the symmetric eigenvalue problem.

While optimality cannot be achieved, we proposed different heuristics to determine good elimination orderings and numerically explored their use. In particular, we compared them to a baseline of random elimination orderings, where they proved to be vastly superior to this baseline. We also explored if the chosen heuristics are able to reproduce the optimal ordering in case that the graph of the matrix is a chain graph, i.e., the matrix is tridiagonal. In this case, the proposed edge elimination algorithm with optimal elimination ordering is equivalent to an iterative (rather than recursive) formulation of the divide-and-conquer approach to tridiagonal symmetric eigenvalue problems.

Considered from the point of view of this paper, the usual approach of an initial reduction to tridiagonal form and the subsequent solution of the tridiagonal eigenvalue problem can be viewed as the reduction to a chain graph with subsequent edge elimination, for which an optimal elimination strategy is known.

Note that several other algorithms for solving Hermitian tridiagonal eigenvalue problems are known, such as QR or MRRR, and some of these may be faster than divide-and-conquer in certain situations. However, they cannot be related to elimination in hypergraphs or graphs in the way proposed in this work.

The equivalence of the Hermitian eigenvalue problem and symbolic hypergraph edge elimination can be easily transferred to the calculation of the singular value decomposition based on the observation that the singular value decomposition $AV = U\Sigma$ of $A \in \mathbb{C}^{m\times n}$ can be computed by considering the Hermitian eigenvalue problem

$$\begin{bmatrix} 0 & A^H \\ A & 0 \end{bmatrix} \begin{bmatrix} V & V \\ U & -U \end{bmatrix} = \begin{bmatrix} V & V \\ U & -U \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & -\Sigma \end{bmatrix}.$$

More efficient techniques avoiding the increase in problem size and the issues in “tearing apart” U and V may be possible by considering non-Hermitian rank-1 updates and directed graphs based on the theory in [13, 14]; this will be the subject of future investigations.

Acknowledgements. The authors would like to thank Kathrin Klamroth and Michael Stiglmayr for many fruitful discussions and the unknown referees and the editor for their valuable comments that helped to improve the presentation.

REFERENCES

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, 3rd. ed., SIAM, Philadelphia, 1999.

[2] P. Arbenz, *Divide and conquer algorithms for the bandsymmetric eigenvalue problem*, Parallel Comput., 18 (1992), pp. 1105–1128.

[3] T. Auckenthaler, V. Blum, H.-J. Bungartz, T. Huckle, R. Johanni, L. Krämer, B. Lang, H. Lederer, and P. R. Willems, *Parallel solution of partial symmetric eigenvalue problems from electronic structure calculations*, Parallel Comput., 37 (2011), pp. 783–794.

[4] J. R. Bunch, C. P. Nielsen, and D. C. Sorensen, *Rank-one modification of the symmetric eigenproblem*, Numer. Math., 31 (1978/79), pp. 31–48.

[5] J. J. M. Cuppen, *A divide and conquer method for the symmetric tridiagonal eigenproblem*, Numer. Math., 36 (1980/81), pp. 177–195.

[6] T. A. Davis, *Direct Methods for Sparse Linear Systems*, SIAM, Philadelphia, 2006.

[7] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[8] J. J. Dongarra and D. C. Sorensen, *A fully parallel algorithm for the symmetric eigenvalue problem*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. S139–S154.

[9] F. R. Gantmacher and M. G. Kreĭn, *Oszillationsmatrizen, Oszillationskerne und kleine Schwingungen mechanischer Systeme*, Akademie-Verlag, Berlin, 1960.

[10] A. George and J. W. H. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Englewood Cliffs, 1981.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[11] G. H. GOLUB, *Some modified matrix eigenvalue problems*, SIAM Rev., 15 (1973), pp. 318–334.
[12] M. GU AND S. C. EISENSTAT, *A stable and efficient algorithm for the rank-one modification of the symmetric eigenproblem*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 1266–1276.
[13] R. C. THOMPSON, *Principal submatrices. IX. Interlacing inequalities for singular values of submatrices*, Linear Algebra Appl., 5 (1972), pp. 1–12.
[14] ———, *The behavior of eigenvalues and singular values under perturbations of restricted rank*, Linear Algebra Appl., 13 (1976), pp. 69–78.
[15] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.
[16] M. YANNAKAKIS, *Computing the minimum fill-in is NP-complete*, SIAM J. Algebraic Discrete Methods, 2 (1981), pp. 77–79.
[17] A. A. ZYKOV, *Hypergraphs*, Uspehi Mat. Nauk, 29 (1974), pp. 89–154.