Journal of Computational and Applied Mathematics 434 (2023) 115332

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

# Fast computation of error bounds for all eigenpairs of a Hermitian and all singular pairs of a rectangular matrix with emphasis on eigen- and singular value clusters

Siegfried M. Rump [a,b,*], Marko Lange [c]

[a] Institute for Reliable Computing, Hamburg University of Technology, Am Schwarzenberg-Campus 3, 21073 Hamburg, Germany
[b] Waseda University, Faculty of Science and Engineering, 3–4–1 Okubo, Shinjuku-ku, Tokyo 169–8555, Japan
[c] Dermalog Identification Systems GmbH, Mittelweg 120, 20148 Hamburg, Germany

ARTICLE INFO

*Article history:*
Received 11 May 2022
Received in revised form 12 March 2023

*MSC:*
65G20
65F15

*Keywords:*
Verification method
All eigenpairs
All singular pairs
Hermitian matrix
Unitary matrix
INTLAB

ABSTRACT

We present verification methods to compute error bounds for all eigenvectors of a Hermitian matrix as well as for all singular vectors of a rectangular real or complex matrix. In case of clusters these are bounds for an orthonormal basis of the invariant subspace or singular vector space, respectively. Individual error bounds for all eigenvalues and singular values including clustered and/or multiple ones are computed as well. The computed bounds do contain the true result with mathematical certainty, and the algorithms apply to interval data as well. In that case the computed bounds are true for every real/complex matrix within the tolerances. The computational complexity to compute inclusions of all eigen/singular pairs of an $n\times n$ matrix or $m\times n$ matrix is $\mathcal{O}(n^3)$ or $\mathcal{O}(mn^2)$ operations, respectively.

## 1. Introduction and notation

In this note we derive a verification method to compute bounds for all eigenvalues and -vectors of a Hermitian matrix. The principles of that method are then used to compute error bounds for all singular values and vectors of a rectangular matrix. The total computing time for an $n \times n$ or $m \times n$ matrix is $\mathcal{O}(n^3)$ or $\mathcal{O}(mn^2)$ operations, respectively.

Readily applicable bounds for all eigenvalues of a Hermitian $n\times n$ matrix follow directly from perturbation theory [1,2]. There are many aspects of perturbation bounds for the spectrum of self-adjoint operators, in particular based on the Rayleigh quotient and Rayleigh/Ritz bounds. Indeed, some of the most well known perturbation results for eigenvalues of Hermitian matrices can be traced back to Temple's famous inequality on Rayleigh quotients [3,4]. A short review over a priori, a posteriori and mixed type bounds on eigenvalues of self-adjoint operators is given, for instance, in [5].

The famous works by Davis and Kahan yield generalized bounds for multiple eigenvalues and their eigenvectors [6,7]. Naturally, many subsequent related works introduced further improvements. In particular we want to mention the generalization of quadratic residual bounds for multiple eigenvalues [8,9] similar to Kato-Temple's inequality [4].

In this note we are concerned with verification methods [10–12], i.e., methods to compute completely rigorous error bounds for the solution of a problem in floating-point arithmetic. The correctness of the bounds includes all procedural

[*] Corresponding author at: Institute for Reliable Computing, Hamburg University of Technology, Am Schwarzenberg-Campus 3, 21073 Hamburg, Germany.
*E-mail addresses:* rump@tuhh.de (S.M. Rump), markonoon@gmail.com (M. Lange).

https://doi.org/10.1016/j.cam.2023.115332
0377-0427/© 2023 Elsevier B.V. All rights reserved.

and in particular all rounding errors due to the use of finite precision floating-point arithmetic. In regard to the problem of computing bounds for all eigenpairs of a Hermitian matrix, this leads to two major limitations.

One limitation is that, although eigenvalues of self-adjoint operators are always perfectly well conditioned, eigenvectors may be ill-posed. For instance, a double eigenvalue corresponds to a 2-dimensional invariant subspace $X$, but for any $x \in X$ there is an arbitrary small perturbation of the input matrix such that $x$ is a unique eigenvector. As a consequence, in the presence of rounding errors, it is therefore not possible to compute tight verified bounds for eigenvectors to numerically inseparable eigenvalues. Instead, in that case our algorithm will compute verified bounds for the invariant subspace corresponding to a cluster of eigenvalues. A similar statement is true for singular vectors.

The second limitation is about the applicability of the respective perturbation bounds. Some majorization bounds for the given eigenvalue bounds have to be evaluated in floating-point arithmetic. As a consequence, a mathematically provably tighter bound might not lead to a better inclusion when used in the context of verification methods. We will elaborate on this further in the respective sections.

This note is organized as follows. We first give some historical remarks and a very brief overview of verification methods for computing rigorous bounds for eigenpairs of Hermitian matrices. Subsequently we discuss some auxiliary routines for computing an upper and lower bound of the singular values of a Hermitian or a rectangular matrix, and for a given subspace we estimate the distance to a matrix with orthonormal columns spanning that space. In Section 4 our method to compute error bounds for all eigenvalues is given. The next section presents a method to improve eigenvalue approximations beforehand and to refine the computed bounds. Both improvements are based on Rayleigh quotients and will also be applied to singular values. The usage of other known bounds is discussed as well. Next, our method to compute error bounds for all eigenvectors and/or invariant subspaces of a Hermitian matrix is presented. At the end of Section 6 we give some comparisons between the presented method for symmetric/Hermitian matrices with the methods in [13] for general matrices.

Finally, we present our fast method for computing error bounds for all singular values and vectors of a general rectangular matrix. The note is closed by an appendix showing how to accelerate the Matlab code.

## 2. Short history and notation

The first verification method for the algebraic eigenproblem is presented by Krawczyk [14] who applies his method for nonlinear systems [15] to $Ax - \lambda x = 0$ with some normalization of $x$. Krawczyk's method, however, is a refinement of initially provided bounds. Moore [16] proposed to use Brouwer's fixed point theorem and proof of nonsingularity of some matrix to derive an existence test. Krawczyk's operator and Moore's ansatz are already contained in [17, p.12].

One might apply that method $n$ times, but besides the complexity $\mathcal{O}(n^4)$ it fails for multiple eigenvalues and cannot guarantee that all eigenvalues are covered. Several publications concentrate on verified error bounds on one eigenpair, for example [18,19]; in [20] a method is introduced for double eigenvalues.

Historically, the next step are verification methods for multiple eigenvalues and corresponding invariant subspaces introduced in [21]. Bounds are computed regardless of the Jordan structure, but for only one cluster. Based on that a verification method for computing bounds for all eigenpairs including multiple eigenvalues and clusters of a general real or complex matrix is introduced in [13]. It uses a simultaneous preconditioning technique for all eigenvalues reducing the total computational effort to $\mathcal{O}(n^3)$ operations.

An efficient algorithm to compute tight error bounds for all simple eigenpairs of a symmetric positive definite matrix is given in [22], extending the work in [23]. However, the method works only for simple, not for clusters and/or multiple eigenvalues. Moreover, bounding techniques for the respective invariant subspaces are missing. In this note we fill these gaps. In addition we prove similar bounds for singular values and vectors of rectangular matrices as well.

Recently another method has been published in [24] for computing inclusions of a few eigenvalues in some region together with their eigenvectors of the generalized Hermitian eigenproblem. The method works for clusters and uses complex moments and the Rayleigh–Ritz procedure. However, according to the authors it is not suitable to compute inclusions of all eigenvalues and -vectors.

This note presents fast verification methods for the computation of error bounds for all eigenvalues and eigenvectors of a general symmetric or Hermitian matrix, and error bounds for all singular values and vectors of a general real or complex rectangular matrix, both with special emphasis on clustered eigenvalues and/or singular values. The presented methods are stable, the bounds are tight. They are based on a general estimation of the distance of a nearly orthogonal/unitary basis of a subspace to a truly orthogonal/unitary basis together with perturbation bounds for invariant and/or singular subspaces.

The methods apply to real or complex interval matrices as well. In that case the bounds are valid for each individual symmetric or Hermitian matrix within the given tolerances. The method in [13] covers the first case, i.e., computes bounds for all eigenpairs of a general real or complex matrix. However, the methods to be presented take advantage of the orthogonality/unitarity of the eigenvectors and outperform the general algorithm in [13] in case of not so well separated clusters. Moreover, they are faster and apply to singular pairs as well.

Denote by $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$ the field of real or complex numbers. We use the short notation $M_{n,k}$ for the set of (real or complex) $n \times k$ matrices, and use $M_n$ if $k = n$. The $n \times n$ identity matrix is denoted by $I_n$, where the subindex is omitted if clear from the context. The singular values of a matrix $A \in M_{m,n}$ with $m \geq n$ are denoted by $\sigma_1(A) \geq \cdots \geq \sigma_n(A)$, and throughout this note $\|\cdot\|$ denotes the spectral norm, i.e., the largest singular value. For brevity, we use $[n] := \{1,\ldots,n\}$ for $n \in \mathbb{N}$.

An introduction to verification methods can be found in [10–12]. Error bounds are computed using interval arithmetic, and we will use boldface letters for interval quantities. Not much knowledge about verification methods and/or interval arithmetic is necessary to follow the exposition, only familiarity with Matlab notation. Also, the representation of intervals, for example infimum-supremum or midpoint-radius, is not important: throughout this note we only use the *inclusion property*, namely, that interval operations $op \in \{+,-,\cdot,/\}$ are defined such that for compatible interval quantities $\mathbf{A}, \mathbf{B}$

$$\forall A \in \mathbf{A}\ \forall B \in \mathbf{B}: \qquad A\ op\ B \in \mathbf{A}\ op\ \mathbf{B} \tag{2.1}$$

is satisfied. For details see [10–12]. We will use Matlab notation [25] and INTLAB [26], the Matlab/Octave toolbox for reliable computing. For $M \in M_n(\mathbb{K})$ and nonnegative $R \in M_n$ the command `midrad(M,R)` is a superset of $\{A \in M_n(\mathbb{K})\colon |A-M| \le R\}$ with entrywise comparison and absolute value. Moreover, $\mathbf{X} = f(\mathbf{A})$ for an interval quantity $\mathbf{A}$ and the induced function $f$ implies that $f(A) \in \mathbf{X}$ for all $A \in \mathbf{A}$.

For a scalar interval $\mathbf{X}$, the magnitude is defined by $\max\{|x| : x \in \mathbf{X}\} \ge 0$. The definition applies entrywise to vectors and matrices so that $B = \text{mag}(\mathbf{A})$ satisfies $|A_{ij}| \le B_{ij}$, and similarly, the mignitude $B = \text{mig}(\mathbf{A})$ satisfies $0 \le B_{ij} \le |A_{ij}|$ for all $A \in \mathbf{A}$ and all $i,j$, cf. [10]. In both cases $B$ is a nonnegative vector/matrix.

## 3. Routines for verified singular value bounds

In the sequel we need upper bounds for the spectral norm of Hermitian $H \in M_n$ and of general $B \in M_n$, as well as a lower bound on the smallest singular value of $X \in M_{n,k}$. In the latter case the matrix $X$ has usually nearly orthonormal columns.

For general $B \in M_n$ we may use $\|B\| \le \sqrt{\|B\|_1 \|B\|_\infty}$. A slightly better bound is obtained as follows. For Hermitian $H \in M_n$ and every positive vector $\tilde{x}$, Perron–Frobenius Theory [2] and a theorem by Collatz [27] give

$$\|H\| \le \| |H| \| = \mathrm{r}(|H|) \le \max_{1\le i\le n} \frac{(|H|\tilde{x})_i}{\tilde{x}_i}, \tag{3.1}$$

where $|H|$ denotes the matrix of absolute values and r denotes the spectral radius. A good choice for $\tilde{x}$ is obtained by few power iterations for $|H|$. For general $B \in M_n$ we use $\|B\|^2 = \mathrm{r}(B^*B)$. Executable Matlab/INTLAB code for general and Hermitian, point or interval matrix is as follows.

```
function N = NormBnd(A,herm)
  % ||A|| <= N for point or interval matrix A
  % if herm=true, then A is Hermitian
  x = ones(size(A,1),1);
  M = [1 2];
  iter = 0;
  A = mag(A);
  while ( abs(diff(M)/sum(M))>.1 ) && ( iter<10 )
    iter = iter+1;
    y = A*x;
    if ~herm, y = A'*y; end
    x = y./x;
    M = [min(x) max(x)];
    scale = max(y);
    x = max( y/scale , 1e-12 );
  end
  if herm
    N = mag((intval(A)*x)./x);
  else
    N = mag(sqrt( (A'*(intval(A)*x)) ./ x ));
  end
```

Basically, the code is self-explanatory. The upper bound of $\|A\|$ is computed based on an approximation `x` of the Perron root of mag$(A) \in M_n$. An operation is executed as an interval operation if at least one operand is of type `intval`. Therefore, the type cast `intval(A)` in the final computation of `N` ensures that interval operations are used before taking the magnitude of the interval result.[1] As a consequence, `N` is a true upper bound for $\|A\|_2$ for all $A \in \mathbf{A}$. The last statement in the while-loop ensures that the components of `x` do not become too small. We mention that the performance can be improved by using directed roundings rather than interval operations, see the appendix. That is because the interpretation

[1] The type cast `intval(A)` does not change `A` if already of type `intval`.

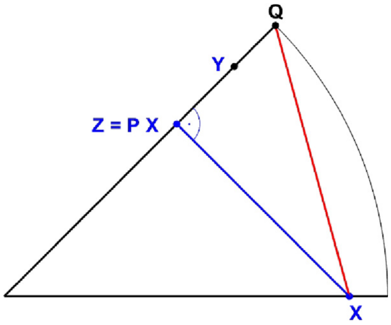

**Fig. 1.** Distance to orthonormal subspace.

overhead, in particular for user-defined data types, is significant. For better readability we refrain from doing this here and leave it at giving comments in the Appendix.

For smaller dimension the bound computed by Algorithm NormBnd is some 20%, for larger dimensions some 8% better than $\sqrt{\|B\|_1\|B\|_\infty}$. The latter bound is usually faster to compute, so if not critical we recommend this simple bound. However, Algorithm NormBnd may be useful to separate clustered eigenvalues.

Let $X \in M_{m,n}$ with $m \geq n$ be given. Then $\|X^*X - I\| \leq \alpha < 1$ implies [28] the singular value bounds

$$\sqrt{1-\alpha} \leq \sigma_i(X) \leq \sqrt{1+\alpha} \quad \text{and} \quad \frac{1}{\sqrt{1+\alpha}} \leq \sigma_i(X^+) \leq \frac{1}{\sqrt{1-\alpha}} \tag{3.2}$$

for $1 \leq i \leq n$ where $X^+$ denotes the pseudoinverse. Note that $\sqrt{1-\alpha}$ is a very good lower bound for the smallest singular value if $X$ has nearly orthonormal columns. Executable Matlab/INTLAB code for a point or interval input matrix is as follows.

```
function s = singmin(X)
 % s <= sigma_min(X) for rectangular X
 alpha = min( 1 , NormBnd( eye(size(X,2))-X'*intval(X) , true ) );
 s = mig( sqrt(1-intval(alpha)) );
```

Using the minimum in the computation of alpha implies the trivial lower bound s = 0 in case $\alpha \geq 1$. In our applications the input matrix has always nearly orthogonal columns so that likely the computed lower bound by Algorithm singmin is very close to 1. For general matrices, also methods in [11, Section 10.8] may be used.

To estimate the distance to an orthonormal subspace of eigenvectors and singular vectors we use the following lemma.

**Lemma 3.1.** *Let $X, Y \in M_{m,n}$ with $m \geq n$ be given. Define $\alpha := \|I - X^*X\|$ and $\delta := \|X - Y\|$. Let $\mathcal{V}$ be an n-dimensional subspace of the $\mathbb{R}^m$ that contains all columns of Y. Then there exists $Q \in M_{m,n}$ with $Q^*Q = I$ whose columns span $\mathcal{V}$ and*

$$\|Q - X\| \leq \alpha + \sqrt{2}\delta.$$

**Proof.** Let $P$ be the projection onto $\mathcal{V}$, and denote $Z := PX$. The columns of $Z$ span a subspace of $\mathcal{V}$ (see Fig. 1). Let $Z = U\Sigma V^*$ with $U \in M_{m,n}$, $\Sigma, V \in M_n$ be an economy size singular value decomposition of $Z$. As usual, we assume the singular value ordered decreasingly. If $Z$ has full rank, then $U$ spans $\mathcal{V}$ and we choose $Q = UV^*$. Otherwise, if some singular values of $Z$ are zero, still suitable columns of $U$ span $\mathcal{V}$, and we choose again $Q = UV^*$.

The columns of $Z - X = PX - X$ lie in the orthogonal complement to $\mathcal{V}$ such that $Q^*(Z - X) = 0 = Z^*(Z - X)$. Together with $\|Z - X\| \leq \|Y - X\| = \delta$ and

$$\|C + D\|^2 = \|(C + D)^*(C + D)\| \leq \|C\|^2 + \|D\|^2 + \|C^*D + D^*C\|,$$

this implies

$$\|Q - X\|^2 \leq \|Q - Z\|^2 + \|Z - X\|^2 \leq \|U(I - \Sigma)V^*\|^2 + \delta^2 = \|I - \Sigma\|^2 + \delta^2 \,.$$

Denote the diagonal matrix of singular values of $X$ by $S$. Then $\|S - \Sigma\| \leq \|X - Z\|$ by [1, Corollary 8.1.6]. Hence

$$\begin{aligned} \|I - \Sigma\| &\leq \|I - S\| + \|S - \Sigma\| \leq \|I - S\| + \|X - Z\| \\ &\leq \|I - S\| + \delta = \|(I + S)^{-1}(I - S^2)\| + \delta \\ &\leq \|I - S^2\| + \delta = \|I - X^*X\| + \delta = \alpha + \delta \end{aligned}$$

and a computation finishes the proof.

**Table 1**
Factor of improvement of Wilkinson's bound over Gershgorin circles.

| | Min | Mean | Median | Max |
|---|---|---|---|---|
| 10 | 2.4 | 3.3 | 3.0 | 4.8 |
| 100 | 5.6 | 8.9 | 8.7 | 14.9 |
| 1000 | 24.2 | 27.7 | 27.2 | 33.8 |

The bound remains true if $\alpha \geq 1$ but may not be useful. In our practical applications, $\alpha$ is of the order of the relative rounding error unit and thus negligible, so that the bound is essentially $\sqrt{2}\delta$. For $\alpha = 0$, the bound is sharp as by $X = (\sqrt{1-\varepsilon^2}, z, \varepsilon)^T$ and $Y = (0, z, \varepsilon)^T$ with $z$ depicting arbitrarily many zeros, for which $Q = (0, z, 1)^T$ and $\|Q - X\| = \sqrt{2(1-\varepsilon)} = \dfrac{\sqrt{2}}{\sqrt{1+\varepsilon}}\|X - Y\|$.

## 4. Eigenvalue bounds

Throughout this section let $A \in M_n$ be a Hermitian matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$. Let $A\tilde{X} \approx \tilde{X}\tilde{\Lambda}$ be an approximate eigendecomposition of $A$, for example, computed by the Matlab command `[Ls,Xs] = eig(A)`. Numerical experience suggests that we can expect $\tilde{X}$ to be nearly unitary, and the eigenvalue approximations to be accurate of the order $\mathbf{u}\|A\|$ for $\mathbf{u}$ denoting the relative rounding error unit.

Assume $\|\tilde{X}^*\tilde{X} - I\| \leq \alpha < 1$. Then $\tilde{X}$ has full rank and the spectra of $\tilde{X}^{-1}A\tilde{X}$ and $A$ coincide. In order to avoid the computation of $\tilde{X}^{-1}$ we use (3.2) to see

$$\|\tilde{X}^*A\tilde{X} - \tilde{X}^{-1}A\tilde{X}\| = \|(\tilde{X}^*\tilde{X} - I)\tilde{X}^{-1}A\tilde{X}\| \leq \frac{\alpha\sqrt{1+\alpha}}{\sqrt{1-\alpha}}\|A\| =: \beta. \tag{4.1}$$

By using Gershgorin circles we then conclude that the spectrum of $A$, which is real, is in the union of $\tilde{G}_i$ with

$$\tilde{G}_i := \{x : |x - B_{ii}| \leq \sum_{j \neq i} |B_{ij}| + \beta\} \qquad \text{for } B := \tilde{X}^*A\tilde{X}.$$

The radii of the Gershgorin circles depend on the 1-norm of the off-diagonal elements of $B = \tilde{X}^*A\tilde{X}$. Wilkinson showed [8] that the spectrum of $A$ differs from the diagonal elements of $\tilde{\Lambda}$ by not more than

$$\frac{\|A\tilde{X} - \tilde{X}\tilde{\Lambda}\|}{\sigma_{\min}(\tilde{X})} \leq \frac{\|A\tilde{X} - \tilde{X}\tilde{\Lambda}\|}{\sqrt{1-\alpha}} =: \delta. \tag{4.2}$$

The factor of the radii by Gershgorin circles over 100 random matrices compared to Wilkinson's bound is displayed in Table 1.

The union of the Gershgorin circles contains the spectrum of $A$, but not every circle needs to contain an eigenvalue. In contrast, each interval $[B_{ii} - \delta, B_{ii} + \delta]$ contains an eigenvalue of $A$, and the spectrum is contained in the union of the intervals.

Define $E := A\tilde{X} - \tilde{X}\tilde{\Lambda}$. The bound in (4.2) needs an upper bound on the norm $\|E\|$, adding another $\mathcal{O}(n^3)$ operations. That can be improved for subsets of eigenvalues using [29], which removed the factor $\sqrt{2}$ in Kahan's well known result [1, Theorem 8.1.8].

**Theorem 4.1.** *Let the Hermitian matrix $A \in M_n$ have eigenvalues $\lambda_1, \ldots, \lambda_n$, and let the Hermitian matrix $H \in M_k$ have eigenvalues $\xi_1, \ldots, \xi_k$. Let $X \in M_{n,k}$ have full column rank. Then there exist $k$ eigenvalues $\lambda_{i_1}, \ldots, \lambda_{i_k}$ of $A$ such that*

$$\max_{1 \leq j \leq k} |\xi_j - \lambda_{i_j}| \leq \frac{\|AX - XH\|}{\sigma_{\min}(X)}.$$

This covers Wilkinson's bound (4.2) for $k = n$. For $k = 1$ it means that each interval

$$\mathbf{L}_j := [\tilde{\Lambda}_{jj} - \delta_j, \tilde{\Lambda}_{jj} + \delta_j] \quad \text{with} \quad \delta_j := \frac{\|Ee_j\|}{\|\tilde{X}e_j\|} \tag{4.3}$$

contains an eigenvalue of $A$. Here only the norms of the columns of $E$ and $\tilde{X}$ are necessary avoiding the extra $\mathcal{O}(n^3)$ effort to bound the matrix norm $\|E\|$.

The union of Gershgorin circles contains the spectrum of $A$, however, the union of the $\mathbf{L}_j$ may not. Consider

$$A = \begin{pmatrix} 5 & 0 \\ 0 & 2 \end{pmatrix}, \quad \tilde{X} = \begin{pmatrix} 0 & -1 \\ 3 & -2 \end{pmatrix} \quad \text{and} \quad \tilde{\Lambda} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}.$$

Then $E := A\tilde{X} - \tilde{X}\tilde{\Lambda} = \begin{pmatrix} 0 & -4 \\ 0 & -2 \end{pmatrix}$, $\delta_1 = 0$ and $\delta_2 = \sqrt{20}/\sqrt{5} = 2$, such that

$$\mathbf{L}_1 = [2, 2] \quad \text{and} \quad \mathbf{L}_2 = [-1, 3].$$

The eigenvalue 2 is contained in both intervals, the eigenvalue 5 in none.

A remedy might be to collect potential clusters of eigenvalues. The connected components of Gershgorin circles contain exactly as many eigenvalues of $A$ as circles form the component, but that is not true for the $\mathbf{L}_j$. Consider

$$A = \begin{pmatrix} -3 & -10 & 2 \\ -10 & -8 & 6 \\ 2 & 6 & 13 \end{pmatrix}, \ \tilde{X} = \begin{pmatrix} -2 & 11 & 1 \\ 2 & -8 & -3 \\ -3 & 1 & -11 \end{pmatrix} \text{ and } \tilde{\Lambda} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 23 \end{pmatrix}.$$

Then $E := A\tilde{X} - \tilde{X}\tilde{\Lambda} = \begin{pmatrix} -20 & -17 & -18 \\ -14 & 8 & 17 \\ -31 & -19 & 94 \end{pmatrix}$, and $\delta_1 \approx 9.57$, $\delta_2 \approx 1.96$ and $\delta_3 \approx 8.49$, such that

$$\mathbf{L}_1 \subset [-9.58, 9.58], \quad \mathbf{L}_2 \subset [4.04, 7.96] \quad \text{and} \quad \mathbf{L}_3 = [14.50, 31.50].$$

The spectrum of $A$ is $\lambda = (-17\ ,\ 4.38\ ,\ 14.62)$, so that $\lambda_2 \in \mathbf{L}_1 \cap \mathbf{L}_2$ and $\lambda_3 \in \mathbf{L}_3$. The eigenvalue $\lambda_1 = -17$ is not contained in any $\mathbf{L}_i$. Collecting the intervals $\mathbf{L}_1$ and $\mathbf{L}_2$ to a cluster $\mu := \{1, 2\}$ gives $\delta_\mu := \frac{\|E(:,\mu)\|}{\sigma_{\min}(\tilde{X}(:,\mu))} \approx 16.47$ and new inclusion intervals $\mathbf{L}'_1 = 0 \pm \delta_\mu \subset [-16.47, 16.47]$ and $\mathbf{L}'_2 = 6 \pm \delta_\mu \subset [-10.47, 22.47]$. Again the eigenvalue $\lambda_1 = -17$ is not included in any interval $\mathbf{L}'_1, \mathbf{L}'_2$ or $\mathbf{L}_3$.

Another choice is to adapt the eigenvalue approximations of the cluster, namely replacing the diagonal of $\tilde{\Lambda}$ using the mean of the clustered eigenvalues, in our example by $(3\ ,\ 3\ ,\ 23)$. That changes $\delta_\mu$ into 13.58 and the cluster inclusions into $\mathbf{L}''_1 = \mathbf{L}''_2 = 3 \pm \delta_\mu \subset [-10.59, 16.59]$, so that again $\lambda_1 = -17$ is not contained in any interval.

Our remedy is to collect eigenvalue clusters recursively. This is done by the following executable Matlab/INTLAB code for a given real or complex, point or interval input matrix. The function `NormBnd(Y)` is applied only to matrices with nearly orthonormal columns, therefore we use the simple bound $\sqrt{\|Y\|_1 \|Y\|_\infty}$.

```
function [L,mu] = verifyeigall(A)
 n = size(A,1);
 mA = mid(A);
 [Xs,Ls] = eig(mA);
 E = A*intval(Xs) - Xs*intval(Ls);
 lambdas = diag(Ls);
 singXsmin = mig( vecnorm(intval(Xs)) );
 normE = vecnorm(E);
 delta = mag( ( normE ./ singXsmin )' );
 num_mu = n;
 while 1
  L = midrad(lambdas,delta);
  Linf = repmat(L.inf,1,n); Lsup = repmat(L.sup,1,n);
  dist = ( Linf<=Linf' ) & ( Lsup>=Linf' );
  dist = dist | dist';
  [mu,binsizes] = conncomp(graph(dist),'OutputForm','cell');
  J = find(binsizes>1);
  if any(J) && ( numels(mu)~=num_mu )
   num_mu = numels(mu);
   for j=J
    v = mu{j};
    singXsmin(v) = singmin(Xs(:,v));
    normE(v) = NormBnd(E(:,v));
    delta(v) = mag( normE(v)/singXsmin(v(1)) );
   end
  else
   break
  end
 end
```

We add a few comments. The code works for interval input $\mathbf{A}$, in which case the results are true for every symmetric or Hermitian $A \in \mathbf{A}$. Therefore approximate numerical computations use `mA`, a matrix close[2] to the midpoint of $\mathbf{A}$. For

[2] We cannot expect `mA` to be the exact midpoint because that needs not be representable.

non-interval input, `A` and `mA` coincide. For the eigenapproximations $(\tilde{X},\tilde{\Lambda}) :=$ `(Xs,Ls)` the matrix $E$ is an inclusion of $A\tilde{X}-\tilde{X}\tilde{\Lambda}$, so that before the while-loop

$$\frac{\|(A\tilde{X}-\tilde{X}\tilde{\Lambda})e_j\|}{\|\tilde{X}e_j\|} \le \delta_j \qquad \text{for all } \ j \in [n].$$

According to (4.3) each interval $\mathbf{L}_j$ contains an eigenvalue. In the while-loop *dist* is computed such that

$$dist_{ij} = true \qquad \Leftrightarrow \qquad \mathbf{L}_i \cap \mathbf{L}_j \neq \emptyset,$$

and $\mu$ are the connected components of the distance matrix *dist*. If all $\mathbf{L}_j$ are initially mutually disjoint, i.e., all connected components contain only one element, then each $\mathbf{L}_j$ contains a unique eigenvalue of $A$ and Algorithm `verifyeigall` stops. That situation corresponds to an empty index set J. Note that this statement is true for each symmetric or Hermitian $A \in \mathbf{A}$ in case of an interval input matrix $\mathbf{A}$.

Otherwise, some eigenvalues of $A$ may not be contained in any $\mathbf{L}_j$ and the while-loop continues until the maximal connected components $\mu$ are determined. The corresponding elements of $\delta$ are recomputed such that at the end of the while-loop

$$\frac{\|E(:,\mu_j)\|}{\sigma_{\min}(\tilde{X}(:,\mu_j))} \le \delta_j \tag{4.4}$$

holds true for all $\mu_j$ and $j \in$ J. The algorithm stops when all connected components $\mathfrak{L}_j := \cup_{i\in\mu_j}\mathbf{L}_i$ are mutually disjoint. Thus, Theorem 4.1 implies that each $\mathfrak{L}_j$ contains exactly $|\mu_j|$ eigenvalues of $A$, and, setting $k := |\mu|$ and because $\sum_{j\in[k]}|\mu_j| = n$, the spectrum is included in $\cup_{j\in[k]}\mathfrak{L}_j$. Note that in the extremely unlikely event that the lower bound[3] `singXsmin(v(1))` of $\sigma_{\min}(\tilde{X}(:,v))$ is zero, all eigenvalue inclusions $\mathbf{L}_j$ become $\pm\infty$.

If there are no clusters, then $k = n$ and the result of Algorithm `verifyeigall` are mutually disjoint intervals containing exactly one eigenvalue of $A$. In case of clusters and for point matrix $A$ we did not encounter cases where the while-loop was executed more than once — unless we searched for that. In contrast, for an interval matrix $\mathbf{A}$ it may happen that the final cluster size is determined by several executions of the while-loop.

The main computing time of Algorithm `verifyeigall` goes into the computation of $E = A\tilde{X}-\tilde{X}\tilde{\Lambda}$ requiring $\mathcal{O}(n^3)$ operations. If there are no or few clusters, the additional time for the while-loop is limited by $\mathcal{O}(np^2)$ operations for the lower bounds on $\sigma_{\min}(\tilde{X}(:,\mu_j))$ with $p := |\mu_j|$. In the worst case, that is one big cluster, that may cost another $\mathcal{O}(n^3)$ operations.

For simplicity we presented Algorithm `verifyeigall` in a way that the total computing time might be $\mathcal{O}(n^4)$. Indeed, the while-loop might start with a single cluster of two eigenvalues and increase that one by one until one big cluster of $n$ eigenvalues. The way the algorithm is presented the computing time of `singmin(Xs(:,v))` is $np^2$ for a cluster `v` of $p$ eigenvalues, which means in total $n\sum_{p=1}^{n} p^2 = \mathcal{O}(n^4)$ operations. In an efficient implementation one would compute the diagonal of $\tilde{X}^*\tilde{X}$ and then, if necessary, step by step the missing entries of the diagonal blocks.

Another way to treat this problem is as follows. We guess a lower bound $\beta$ for the smallest singular value of `E`. Based on that we compute the clusters in one step, and it remains to certify $\beta \le \sigma_{\min}(E_\mu)$ for all $E_\mu =$ `E(:,mu{j})` with $j \in$ J, i.e., for clusters with at least two elements. For $k$ clusters of sizes $p_j$ denote $w := (p_1,\ldots,p_k) \in \mathbb{R}^k$. Then $\|w\|_1 \le n$ and the total effort is at most $n\sum_{j=1}^{k} p_j^2 = n\|w\|_2^2 \le n^3$. Numerical evidence suggests that in double precision the guess $1-10n\cdot 2^{-53}$ never fails, but the guess for $\beta$ may be much more generous such as, for example, $\beta := 1-10^{-5}$ without changing the final result significantly.

**Theorem 4.2.** *Let a real or complex, point or interval matrix* $\mathbf{A}$ *be given, and let* $(\mathbf{L},\mu)$ *be the result of Algorithm* `verifyeigall` *applied to* $\mathbf{A}$*. Then for each symmetric or Hermitian* $A \in \mathbf{A}$ *the following is true. There is a numbering of the eigenvalues* $\lambda_1,\ldots,\lambda_n$ *of* $A$ *such that* $\lambda_j \in \mathbf{L}_j$ *for all* $j \in [n]$*. Moreover,* $\mu = (\mu_1,\ldots,\mu_k)$ *is a partition of* $[n]$ *into* $k$ *sets* $\mu_j$ *such that* $\mathfrak{L}_j := \cup_{i\in\mu_j}\mathbf{L}_j$ *is a set of* $k$ *mutually disjoint intervals, and for all* $j \in [k]$ *each* $\mathfrak{L}_j$ *contains exactly* $|\mu_j|$ *eigenvalues of* $A$*.*

In Algorithm `verifyeigall` the midpoints $\tilde{\Lambda}_{ii}$ do not change, also if clusters are determined in the while-loop. As has been mentioned, another strategy is to use the mean $(\sum_{i\in\mu_j}\tilde{\Lambda}_{ii})/|\mu_j|$ as new midpoint if a cluster $\mu_j$ is discovered. As a drawback, for each newly formed cluster $\mu_j$ the submatrix $E(:,\mu_j) = A\tilde{X}(:,\mu_j) - \tilde{X}(:,\mu_j)\tilde{\Lambda}(\mu_j,\mu_j)$ has to be recomputed.

Numerical experience suggests that the computed eigenvalue inclusions of the presented Algorithm `verifyeigall` are better for clustered eigenvalues when not changing the midpoints $\tilde{\Lambda}_{ii}$ of the $\mathbf{L}_i$. By numerical evidence, the eigenvector inclusions are generally better for simple eigenvalues, but worse for clusters.

[3] Note that `singXsmin(v)` is a vector with identical elements being a lower bound to $\sigma_{\min}(\tilde{X}(:,v))$.

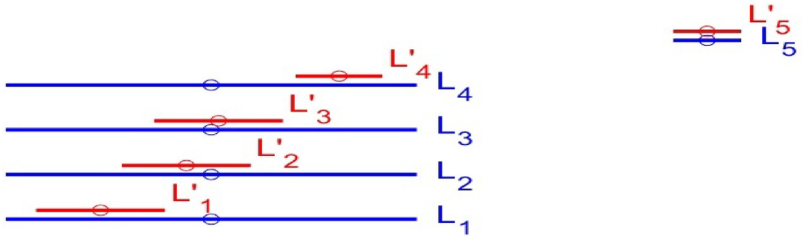

**Fig. 2.** Eigenvalue inclusions by Algorithm `verifyeigall` [red] compared to taking the mean of eigenvalue approximations of a cluster [blue].

Moreover, sometimes the strategy not to change the original approximations $\tilde{\Lambda}_{ii}$ leads to fewer clusters. As an example consider

$$\mathbf{A} = \begin{pmatrix} 16 & 7 & 0 & 3 & 7 \\ 7 & -4 & -1 & -2 & 1 \\ 0 & -1 & -6 & 5 & 1 \\ 3 & -2 & 5 & -6 & 3 \\ 7 & 1 & 1 & 3 & -2 \end{pmatrix} \pm 0.5\,. \tag{4.5}$$

The eigenvalue inclusions $\mathbf{L}_j$ computed in INTLAB are as follows, left the result of Algorithm `verifyeigall` as presented, and right when taking the mean of eigenvalue approximations of a cluster in the while-loop, see Fig. 2.

```
intval ans =
  [  -15.0571,    -8.1235] [  -21.1455,     9.7633]
  [  -10.4787,    -3.5451] [  -21.1455,     9.7633]
  [   -8.7632,    -1.8295] [  -21.1455,     9.7633]
  [   -1.1711,     3.4393] [  -21.1455,     9.7633]
  [   18.9542,    22.5746] [   18.9542,    22.5746]
```

As can be seen, for `verifyeigall` the first three eigenvalues are collected into one cluster with different midpoints and constant radius 3.47, where the alternative computes a cluster of the first four eigenvalues with identical inclusions, each of radius 15.45. We come to that example again in the section for eigenvector inclusions.

## 5. Improvement of eigenvalue approximations and inclusions

We will improve our eigenvalue inclusions in two ways. First, the initial eigenvalue approximations in $\tilde{\Lambda}$ will be corrected to obtain smaller residuals. Second, after inclusions of all eigenvalues are known, those can be sharpened. Both improvements are based on Rayleigh quotients. The same principle is used to improve singular value inclusions.

The quality of the verified bounds for simple eigenvalues depend on the spectral norm of the columns of $A\tilde{X} - \tilde{X}\tilde{\Lambda}$, i.e., on residuals $\|A\tilde{x} - \tilde{\lambda}\tilde{x}\|$. So first we improve a given approximation $\tilde{\lambda}$ into $\tilde{\lambda} + \varepsilon$ by minimizing $f(\varepsilon) := \|A\tilde{x} - (\tilde{\lambda} + \varepsilon)\tilde{x}\|^2$. Setting $y := A\tilde{x} - \tilde{\lambda}\tilde{x}$ we obtain

$$f(\varepsilon) = \|y\|^2 - 2\varepsilon\tilde{x}^*y + \varepsilon^2\|\tilde{x}\|^2 \quad \text{which is mimimal for } \tilde{\varepsilon} = \frac{\tilde{x}^*y}{\|\tilde{x}\|^2},$$

so that $\tilde{\lambda} + \tilde{\varepsilon}$ becomes the Rayleigh quotient. For singular values the quality of the inclusions depend on residuals $y := A\tilde{v} - \tilde{\sigma}\tilde{u}$ and lead to the corrected singular value approximation

$$\tilde{\sigma} \quad \to \quad \tilde{\sigma} + \frac{\tilde{u}^*y}{\|\tilde{u}\|^2}\,.$$

For condition numbers up to $10^{16}$, Fig. 3 shows the minimum (solid line), mean (dotted line) and median (dashed line) ratio of the improvement of all residual bounds for the singular values for a set of 100 random 100 × 100 real matrices (left) and 100 random 100 × 100 complex matrices (right). As can be seen the best ratio is up to 0.3 for larger condition number. That means that the relative error of the improved bounds is up to a factor 3 smaller than the original bounds.

In Fig. 4 the same ratios are shown for the improvement of the eigenvalue bounds. Here the improvement is up to a factor 1.5 for all condition numbers. The median is close to 1 which means that only a few corrections are significant. Mainly the singular values and eigenvalues of largest absolute value enjoy the best improvement because a correction of a small eigen- or singular value has small impact on the residual, and in order to save computing time the correction may be restricted to the former.

Second, once inclusions of all eigenvalues are at hand, we sharpen those of simple eigenvalues. Once the gap of $\varrho$ to the remaining spectrum is known, the distance to its nearest eigenvalue can be estimated by [30, Theorem 5]:

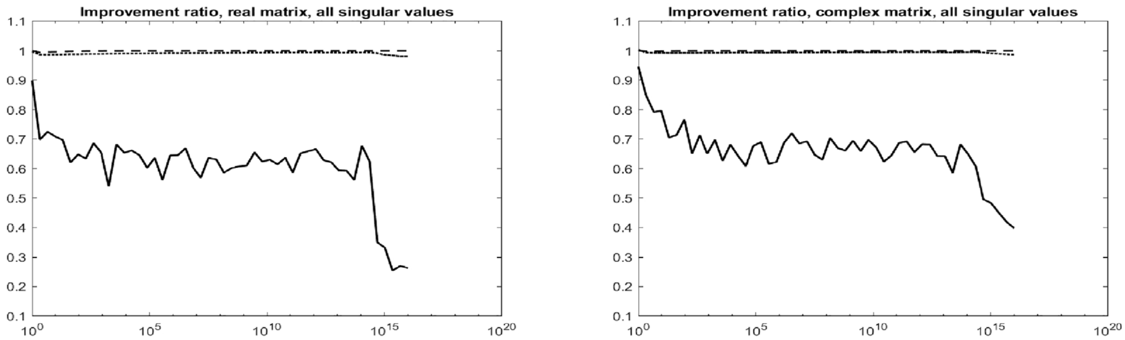

**Fig. 3.** Improvement of the bounds by correction of the singular value approximations.

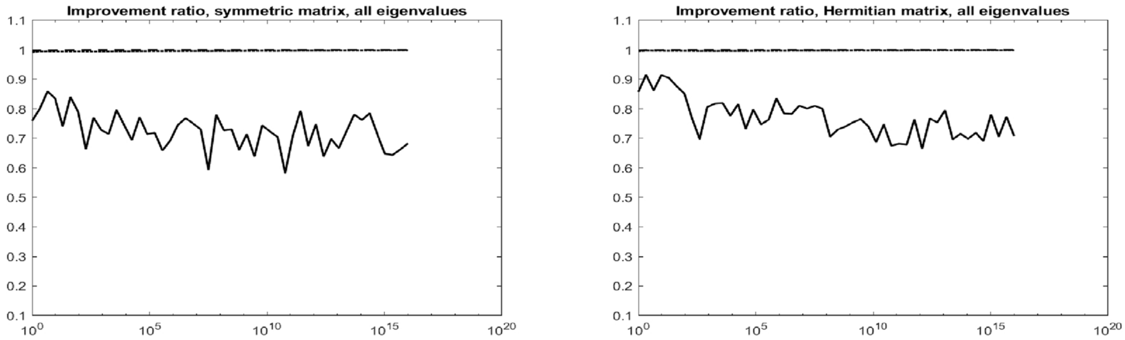

**Fig. 4.** Improvement of the bounds by correction of the eigenvalue approximations.

**Theorem 5.1.** *For Hermitian $A \in M_n$ and nontrivial $x \in \mathbb{K}^n$ let $\lambda$ be the closest eigenvalue to $\varrho(x) := \dfrac{x^*Ax}{x^*x}$, and $\varepsilon$ be the separation of $\varrho$ to the next closest eigenvalue. Then*

$$|\lambda - \varrho| \le \frac{\|Ax - \varrho x\|^2}{\varepsilon \|x\|^2} \,. \tag{5.1}$$

The authors note that this residual bound, which also follows by Kato-Temple’s inequalities [4], gives considerable insight but is not readily computable because $\varepsilon$ is not known. Fortunately, we have bounds for the eigenvalue gaps and can apply Theorem 5.1.

For a given eigenvector approximation $\tilde{x}$ of a Hermitian matrix, the Rayleigh quotient $\varrho$ is the best eigenvalue approximation. Both the improvement of the initial approximations in $\tilde{\Lambda}$ and the eigenvalue bounds by Theorem 5.1 are based on $\varrho$, where we first need an approximation and second an inclusion of the Rayleigh quotient $\varrho$.

Therefore, we compute inclusions `rho` of the Rayleigh quotients at the beginning of Algorithm `verifyeigall` and replace $\tilde{\Lambda}$ by the diagonal matrix of midpoints of `rho`. After having inclusions of all eigenvalues at hand we improve the inclusions of the simple eigenvalues by Theorem 5.1 based on the inclusion of `rho`.

For an interval input matrix **A** the assertions of Theorem 4.2 are true for all matrices $A \in \mathbf{A}$. The computation of the Rayleigh quotient $\varrho$ is based on approximate eigenvector approximations, and to that end it seems suitable to use the midpoint matrix `mA = mid(A)` of **A** as in Algorithm `verifyeigall`. To achieve tight bounds for the eigenvectors and/or invariant subspaces it is important that the computed eigenvector approximation matrix is close to unitary. But that may not be true if the input matrix is numerically but not mathematically symmetric or Hermitian.

To describe the problem, we generate a numerically symmetric matrix `A` with a double eigenvalue and calculate the residual `I-X’*X` of the computed eigenvector approximation `X`.

```
n = 5;
d = randn(n,1); d(1) = d(2);
Q = orth(randn(n));
A = Q’*diag(d)*Q;
normA = norm(A’-A)
[X,L] = eig(A); res1 = norm(eye(n)-X’*X)
B = (A’+A)/2;
```

```
normB = norm(B'-B)
[X,L] = eig(B); res2 = norm(eye(n)-X'*X)
```

Typical results are

```
normA =
  2.1999e-16
res1 =
  0.1414
normB =
  0
res2 =
  1.7760e-15
```

In other words, the eigenvector approximation without symmetrization is far from being orthogonal. The initial matrix `A` is numerically symmetric, but not mathematically. Therefore Matlab uses an algorithm for general matrices, and this leads typically to numerically linear dependent eigenvector approximations for the clustered eigenvalue.

For an interval input matrix **A**, the midpoint matrix computed by `mA = mid(A)` need not be symmetric or Hermitian, even if the bounds are. Therefore we compute eigenvector approximations of the symmetrized matrix `B = (A'+A)/2` which must be symmetric or Hermitian, also in the presence of rounding errors because floating-point addition is commutative, symmetric to zero and division by 2 is exact.

Another strategy is to use the Schur decomposition $A = UTU^*$. Since the input matrix is expected to be numerically symmetric or Hermitian, i.e., very close to normal, the Schur matrix $T$ is close to diagonal. The problem described before is solved because the transformation matrix `U` is intended to be unitary.

In order to fully use the remarkable quadratic approximation property (5.1) of the Rayleigh quotient it is important to compute it using some increased precision as, for example, described in [31–34]. This should be used to implement routines

$$\texttt{norm\_X2(x)},\quad \texttt{norm\_xAx(A, x)}\quad \text{and}\quad \texttt{norm\_Axrhomid2(A, x, rho.mid)} \tag{5.2}$$

which give vectors of inclusions of $\|x\|^2$, $x^*Ax$ and $\|Ax - rx\|^2$ for each column $x$ of `Xs`, respectively, where $r :=$ `rho.mid` is the midpoint of the Rayleigh quotient inclusion `rho`. Then, for $\tilde{x}$ denoting the $i$th column of $\tilde{X}$, the $i$th entry of `rho` is an inclusion of the Rayleigh quotient of $\tilde{x}$. In order to use the Rayleigh quotient to compute the eigenvalue approximations we then replace the line

```
[Xs,Ls] = eig(mA);
```

in Algorithm `verifyeigall` by the lines

```
[Xs,~] = eig((mA'+mA)/2);
norm2x = norm_X2(Xs);
rho = norm_xAx(A,Xs)./norm2x; % inclusion of Rayleigh quotient
Ls = diag(rho.mid);
```

Note that only the eigenvector matrix `Xs` of `eig` is needed, the eigenvalue approximations are computed using the Rayleigh quotients based on `Xs`. To improve the already computed eigenvalue inclusions `L` we append the call

```
L = refineeig(A,Xs,L,rho,norm2x)
```

after Algorithm `verifyeigall` using the following code:[4]

```
function L = refineeig(A,Xs,L,rho,norm2x)
  n = size(A,2);
  LL = repmat(L,1,n);
  e = mig(LL-LL');
  e(1:n+1:n^2) = inf;
  e = min(e);               % gaps between eigenvalues
  index_s = find(e>0);      % indices of simple eigenvalues
  res = norm_Axrhomid2(A,Xs,rho.mid)./norm2x;
  res = sqrt(max(0,res)) + rho.rad;
  Lnew = ( rho + midrad(0,mag(sqr(res)./e)) )';
  L(index_s) = intersect(L(index_s),Lnew(index_s));
```

[4] For brevity we use the Matlab notation `rho.mid` and `rho.rad`, where `rho.mid` is an approximation of the midpoint of the interval `rho` and `rho.rad` an upper bound for its radius rad(`rho`). The quantities are computed such that `midrad(rho.mid,rho.rad)` contains `rho`.

**Table 2**
Improvement of eigenvalue inclusions by Algorithm `refineeig`.

| Relerr | Symmetric matrix | | | Hermitian matrix | | |
|---|---|---|---|---|---|---|
| | Minimum | Median | Max | Minimum | Median | Max |
| Initial | $2.0 \cdot 10^{-14}$ | $7.7 \cdot 10^{-14}$ | $7.8 \cdot 10^{-11}$ | $2.8 \cdot 10^{-14}$ | $9.5 \cdot 10^{-14}$ | $7.1 \cdot 10^{-11}$ |
| Refined | $9.5 \cdot 10^{-15}$ | $1.1 \cdot 10^{-14}$ | $1.2 \cdot 10^{-14}$ | $9.5 \cdot 10^{-15}$ | $1.1 \cdot 10^{-14}$ | $4.0 \cdot 10^{-13}$ |
| Ratio | 2.2 | 7.3 | 6612 | 2.9 | 8.8 | 178 |

**Table 3**
Improvement of eigenvalue inclusions in the presence of clusters by Algorithm `refineeig`.

| Relerr | Symmetric matrix | | | Hermitian matrix | | |
|---|---|---|---|---|---|---|
| | Minimum | Median | Max | Minimum | Median | Max |
| Initial | $1.8 \cdot 10^{-14}$ | $5.8 \cdot 10^{-14}$ | $1.6 \cdot 10^{-13}$ | $6.9 \cdot 10^{-14}$ | $1.3 \cdot 10^{-13}$ | $4.1 \cdot 10^{-13}$ |
| Refined | $2.0 \cdot 10^{-15}$ | $1.0 \cdot 10^{-14}$ | $2.4 \cdot 10^{-14}$ | $9.5 \cdot 10^{-15}$ | $1.1 \cdot 10^{-14}$ | $1.2 \cdot 10^{-14}$ |
| Ratio | 9.2 | 5.6 | 6.6 | 7.2 | 12.6 | 34 |

Let $j \in$ `index_s` be fixed but arbitrary. The set `index_s` comprises only of clusters of size 1, so that the eigenvalue $\lambda_j$ of $A$ in $\mathbf{L}_j$ is unique and simple. In order to avoid extensive index computations, the bounds in `Lnew` are computed for all indices $1 \ldots n$ but are valid only for the indices in `index_s`. That is taken into account in the last statement computing the refined `L`.

Denote $\mathbf{L}_j = [\tilde{\lambda}_j - \delta_j, \tilde{\lambda}_j + \delta_j]$ based on the approximate eigenpair $\tilde{\lambda}_j, \tilde{x}$ and $\delta_j := \|A\tilde{x} - \tilde{\lambda}_j\tilde{x}\| / \|\tilde{x}\|$. The Rayleigh quotient of $\tilde{x}$ is an element of $\mathbf{L}_j$ as by

$$|\varrho - \tilde{\lambda}_j| = \frac{|\tilde{x}^*(A - \tilde{\lambda}_j I)\tilde{x}|}{\tilde{x}^*\tilde{x}} \leq \frac{\|A\tilde{x} - \tilde{\lambda}_j\tilde{x}\|}{\|\tilde{x}\|} = \delta_j \,. \tag{5.3}$$

The distance vector $\mathtt{e}$ satisfies $\min\{|\xi_i - \xi_j| : \xi_i \in \mathbf{L}_i, \xi_j \in \mathbf{L}_j, i \neq j, \} \leq \mathtt{e}_\mathtt{j}$, so that for the eigenvalues $\lambda_i$ of $A$ in particular

$$\min_{i \neq j} |\lambda_i - \varrho| \leq \mathtt{e}_\mathtt{j}$$

because $\lambda_i \notin \mathbf{L}_j$ for $i \neq j$. Thus Theorem 5.1 is applicable and yields

$$|\lambda - \varrho| \leq \frac{\|Ax - \varrho x\|^2}{\mathtt{e}_\mathtt{j}\, \|\mathtt{x}\|^2} \,. \tag{5.4}$$

Then `Lnew` is computed according to (5.4) using

$$\frac{\|Ax - \varrho x\|}{\|x\|} \in \frac{\|Ax - rx\|}{\|x\|} \pm \mathrm{rad}(\mathtt{rho}) \,.$$

The final value of `res` uses `sqrt(max(0,res))` to cure possible interval overestimation, and the intersection of the entries of `L` and `Lnew` is only necessary for interval input.

For Hermitian $A = A_1 + iA_2$ and $z = x + iy$ it is advisable to use $z^*Az = x^*A_1x + y^*A_2y - 2x^*A_2y$ and $z^*z = x^*x + y^*y$ because the imaginary part vanishes. Moreover, if complex midpoint-radius arithmetic is used as in INTLAB, that is superior to taking the real part of the interval products. As has been mentioned it is better to calculate $Ax$ in the computation of the Rayleigh quotient and the residual $Ax - rx$ with some extra precision.

We next provide some computational results. For dimension $n = 1000$ we generate symmetric and Hermitian matrices randomly and calculate the eigenvalue inclusions by Algorithm `verifyeigall` and using `refineeig`. Then we calculate the minimum $m_1$, median $m_2$ and maximum $m_3$ of the relative errors of both inclusions. This is done for 100 sample matrices and the median of the $m_i$ is displayed together with their ratio in Table 2. As can be seen there is often a considerable refinement of the eigenvalue bounds, and there seems not too much difference between the symmetric and Hermitian case.

Next, for $e := 10^{-11}$, we generate a symmetric matrix with 10 random eigenvalues in a circle of radius $e$ around 0.1, another 10 random eigenvalues in a circle of radius $e$ around 0.2, and another 980 random eigenvalues in $[-1, -0.3] \cup [0.3, 1]$, and similarly for a Hermitian matrix. Since there is no improvement for the clustered eigenvalues, we consider only the relative errors of the simple eigenvalues. Again, the median of the results of 100 samples is taken and shown in Table 3.

Although the comparison is for the simple eigenvalues, there seems some influence of clusters to the eigenapproximations. Therefore the improvement is more moderate, and maybe slightly better for Hermitian matrices.

One might think about the application of mathematically provable tighter bounds for even better accuracy. For example, in [5] the authors proved a bound similar to that in (5.1). They showed that the residual can be replaced with its projection onto a smaller subspace $\mathcal{V}$ provided that $\mathcal{V}$ contains $x$ as well as the eigenvector to $\lambda$. However, $\mathcal{V}$ is not given since we do not know the respective eigenvector. By applying the method described in Section 6 it is possible to derive an inclusion of $\mathcal{V}$ from which then a verified inclusion of the respective projection matrix can be computed. Nevertheless, the introduction of additional rounding errors may outweigh the benefit from the tighter bound.

This issue is even more present when applying the quadratic residual bounds from [8] or [9] to a cluster of eigenvalues. All computations would need to be done in higher precision to compensate the additionally introduced floating-point rounding errors. However, the additional computational effort might be better spent on more accurate eigenpair approximations. A further investigation of these possibilities is surely of interest but lies outside of the scope of this note.

## 6. Eigenvector bounds

Let $(\mathbf{L}, \mu)$ be the results of Algorithm `verifyeigall` applied to Hermitian $A \in M_n$ with eigendecomposition $AX = X\Lambda$. The eigenvalue inclusions are $\mathbf{L}_j = [\tilde{\lambda}_j - \delta_j, \tilde{\lambda}_j + \delta_j]$ for $j \in [n]$, and $\mu = (\mu_1, \ldots, \mu_k)$ is a partition of $[n]$. Moreover, the cluster inclusions $\mathfrak{L}_\ell := \cup_{j\in\mu_\ell}\mathbf{L}_j$ are mutually disjoint for $\ell \in [k]$. Applying Theorem 4.1 to each cluster separately shows that there is a suitable numbering $\lambda_1, \ldots, \lambda_n$ of the eigenvalues of $A$ such that $\lambda_j \in \mathbf{L}_j$ for all $j \in [n]$. To obtain bounds for the respective eigenvectors or invariant subspaces, we exploit the following lemma which is closely related to Davis' and Kahan's celebrated $\sin(\Theta)$ theorem [7]. Indeed, for its short proof we borrowed from the original work.

**Lemma 6.1.** *Let Hermitian $A \in M_n$ be given, denote its eigendecomposition by $AX = X\Lambda$, and let $\tilde{X} \in M_{n,p}$ and Hermitian $\tilde{\Lambda} \in M_p$ with $p \le n$ be given. Let $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ and denote the eigenvalues of $\tilde{\Lambda}$ by $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_p$. Let $\mu \subseteq [n]$ with $|\mu| = p$ be given and assume that there exists positive $\varepsilon$ with*

$$\varepsilon \le \min_{i\notin\mu, j\in\mu} |\lambda_i - \tilde{\lambda}_j|. \tag{6.1}$$

*Then there exists $Y \in M_{n,p}$ whose columns lie in the invariant subspace $\mathcal{V}$ of $A$ to the eigenvalues $\{\lambda_j : j \in \mu\}$ with*

$$\|\tilde{X} - Y\| \le \frac{\|A\tilde{X} - \tilde{X}\tilde{\Lambda}\|}{\varepsilon}. \tag{6.2}$$

**Proof.** For $B \in M_n$ denote by $B_\mu \in M_{n,p}$ its submatrix with columns in $\mu$, and set $\overline{\mu} := [n]\backslash\mu$. Then $X_\mu$ spans the invariant subspace of $A$ to the eigenvalues $\{\lambda_j : j \in \mu\}$, and $X_{\overline{\mu}}$ is its orthogonal complement. Denote by $L := [\min_{1\le j\le p} \tilde{\lambda}_j, \max_{1\le j\le p} \tilde{\lambda}_j]$ the convex hull of the eigenvalues of $\tilde{\Lambda}$, by $\hat{\lambda}$ its midpoint and by $r$ its radius, so that $L = [\hat{\lambda} - r, \hat{\lambda} + r]$. The assumption (6.1) and $\varepsilon > 0$ give $|\lambda_i - \hat{\lambda}| \ge \varepsilon + r$ for all $i \notin \mu$. Then, borrowing from the proofs of [35, Lemma 3.1] and the celebrated $\sin(\Theta)$ theorem [7] and using $r = \|\tilde{\Lambda} - \hat{\lambda}I_p\|$ it follows

$$\begin{aligned}
\|A\tilde{X} - \tilde{X}\tilde{\Lambda}\| &= \sigma_1(X_{\overline{\mu}}^*)\|X\Lambda X^*\tilde{X} - \tilde{X}\tilde{\Lambda}\| \ge \|\Lambda(\overline{\mu},\overline{\mu})X_{\overline{\mu}}^*\tilde{X} - X_{\overline{\mu}}^*\tilde{X}\tilde{\Lambda}\| \\
&= \| \left(\Lambda(\overline{\mu},\overline{\mu}) - \hat{\lambda}I_{n-p}\right) X_{\overline{\mu}}^*\tilde{X} - X_{\overline{\mu}}^*\tilde{X}\left(\tilde{\Lambda} - \hat{\lambda}I_p\right) \| \\
&\ge (\varepsilon + r)\|X_{\overline{\mu}}^*\tilde{X}\| - r\|X_{\overline{\mu}}^*\tilde{X}\| = \varepsilon\|X_{\overline{\mu}}^*\tilde{X}\| \\
&= \varepsilon\|X_{\overline{\mu}}X_{\overline{\mu}}^*\tilde{X}\| = \varepsilon\| \left(I - X_\mu X_\mu^*\right)\tilde{X} \|.
\end{aligned}$$

Since $Y := X_\mu X_\mu^*\tilde{X}$ is a subspace of $X_\mu$, that finishes the proof.

We use the notation of the previous section. Let $\ell \in [k]$ be fixed but arbitrary, and set $p := |\mu_\ell|$. Define $S := \tilde{X}(:, \mu_\ell) \in M_{n,p}$ and $\tilde{L} := \tilde{\Lambda}(\mu_\ell, \mu_\ell) \in M_p$. Then $AS - S\tilde{L} = E(:, \mu_\ell)$ and Theorem 4.1 yield

$$|\tilde{\lambda}_j - \lambda_j| \le \frac{\|E(:, \mu_\ell)\|}{\sigma_{\min}(S)} \le \delta_j \qquad \text{for all } j \in \mu_\ell.$$

The quantity

$$\varepsilon_\ell := \min_{i\notin\mu_\ell, j\in\mu_\ell} |\lambda_i - \tilde{\lambda}_j| \ge \min_{i\notin\mu_\ell, j\in\mu_\ell} |\tilde{\lambda}_i - \tilde{\lambda}_j| - \delta_i \tag{6.3}$$

can be computed in $\mathcal{O}(n^2)$ operations, and $\varepsilon_\ell$ is positive because the cluster inclusions are mutually disjoint. Thus, Lemma 6.1 is applicable and proves that there exists $Y \in M_{n,p}$ whose columns lie in the invariant subspace $\mathcal{V}$ of $A$ to its eigenvalues $\{\lambda_j : j \in \mu_\ell\}$ with

$$\|S - Y\| \le \frac{\|E(:, \mu_\ell)\|}{\varepsilon_\ell} =: \tau_\ell. \tag{6.4}$$

Note that $Y$ may be rank-deficient, it might even be the zero matrix if $\tilde{X}$ is orthogonal to $X_\mu$. Although that seems hardly possible in practice, the result (6.2) remains true and allows to compute an inclusion of a matrix with orthonormal columns spanning $\mathcal{V}$ using Lemma 3.1. For $q, r$ in the same partition $\mu_\ell$ the radii $\tau_q = \tau_r$ are the same.

In order to compute mathematically correct error bounds for an invariant subspace to corresponding eigenvalue clusters belonging to the partition $\mu$ the first line

```
function [L,mu] = verifyeigall(A)
```

is changed into

```
function [L,mu,X] = verifyeigall(A)
```

where the corresponding columns in X contain an orthonormal basis of the invariant subspaces. The following executable Matlab/INTLAB code is appended to Algorithm `verifyeigall`. It includes the refinement of the eigenvalues as described in the previous section.

```
L = refineeig(A,Xs,L,rho,norm2x);
if numels(mu)==1      % only one cluster
  if isinf(delta(1)) || isnan(delta(1))
    rX = inf(1,n);
  else
    rX = zeros(1,n);
  end
else
  lam = intval( repmat(L.mid,1,n)’ );
  e = mig(min( abs(lam-Linf) , abs(lam-Lsup) ));
  e(1:n+1:n^2) = inf;
  for j=J
    v = mu{j};
    e(v,v) = inf;
  end
  tau = normE ./ min(e);
  alpha = mag( 1 - sum(intval(Xs).*conj(Xs)) );
  rX = mag( alpha + intval(’sqrt2’)*tau );
  for j=J
    v = mu{j};
    Ip = eye(length(v));
    alpha = NormBnd(Ip - Xs(:,v)’*intval(Xs(:,v)));
    rX(v) = mag( alpha + intval(’sqrt2’)*tau(v) );
  end
end
X = midrad( Xs , repmat(rX,n,1) );
```

If `numels(mu)==1` that means that there is only one cluster collecting all eigenvalues and that exceptional case is handled first. Otherwise, the distance matrix of the eigenvalue inclusions is computed such that the entries of the vector `min(e)` are the quantities as in (6.3). Note that the eigenvalue inclusions `L` have been refined and `lam` uses the new midpoints `L.mid`. If the index set `J` is empty, then there are no clusters, exclusively simple eigenvalues and the elements of `rX` bound the norm distance of the columns of `Xs` to a true eigenvector normed to 1. Otherwise, the radii are corrected according to (6.4) such that, for all `j∈J` and `v=mu{j}`, `X(:,v)` contains an orthonormal matrix spanning the invariant subspace of $A$ to its eigenvalues.

**Theorem 6.2.** *Let a real or complex, point or interval matrix* $\mathbf{A}$ *be given, and let* $(\mathbf{L}, \mu, \mathbf{X})$ *be the result of Algorithm* `verifyeigall` *applied to* $\mathbf{A}$*. Then for each symmetric or Hermitian* $A \in \mathbf{A}$ *the following is true. For the partition* $\mu = (\mu_1, \ldots, \mu_k)$ *of* $[n]$ *into k sets* $\mu_\ell$*, each* $\mathbf{X}(:, \mu_\ell)$ *contains an orthonormal basis of the invariant subspace to the eigenvalues in* $\bigcup_{j\in\mu_\ell}\mathbf{L}_j$*.*

Denote by $A$ the matrix (4.5) and consider $\mathbf{A} := A \pm r$ for different radii $r$. The following Table 4 shows the clusters for taking the mean of eigenvalue clusters as midpoint of the inclusions as described before, and the original Algorithm `verifyeigall`. As can be seen the cluster sizes are the same except for radius 0.45 where eigenvalues $1 \ldots 4$ form a single cluster when using the mean of eigenvalue clusters as midpoint. Next we show in the rows of Table 5 the median of the radii of the inclusions of the invariant subspaces of $\mathbf{A} := A \pm r$ for different radii $r$, left taking the mean of eigenvalues for clusters and right Algorithm `verifyeigall`.

For increasing values of $r$ the radii of the original Algorithm `verifyeigall` become slightly superior though there is not too much difference, but become worse for the cluster $\{2, 3\}$. The radii in the last column are not really comparable because only the original Algorithm `verifyeigall` is able to separate the eigenvalues.

A typical computational result for larger dimension is displayed in Table 6. First, we take a $1000 \times 1000$ random symmetric matrix and show the minimum, mean, median and maximum of the relative error of the inclusions of eigenvalues and eigenvectors, where for eigenvectors we first take the median of relative errors so that outlayers of

**Table 4**
Cluster sizes for mean of eigenvalue clusters and original Algorithm `verifyeigall`.

| Radius | Mean of eigenvalues | Algorithm `verifyeigall` |
|---|---|---|
| 0.10 | 1, 2, 3, 4, 5 | 1, 2, 3, 4, 5 |
| 0.20 | 1, 2, 3, 4, 5 | 1, 2, 3, 4, 5 |
| 0.25 | 1, {2,3}, 4, 5 | 1, {2,3}, 4, 5 |
| 0.30 | 1, {2,3}, 4, 5 | 1, {2,3}, 4, 5 |
| 0.35 | 1, {2,3}, 4, 5 | 1, {2,3}, 4, 5 |
| 0.40 | 1, {2,3}, 4, 5 | 1, {2,3}, 4, 5 |
| 0.45 | {1,2,3,4}, 5 | 1, {2,3}, 4, 5 |

**Table 5**
Radii of the inclusions of the invariant subspaces of $\mathbf{A} := A \pm r$ for different $r$.

| 0.10 | | 0.20 | | 0.25 | | 0.30 | | 0.35 | | 0.40 | | 0.45 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.10 | 0.10 | 0.21 | 0.21 | 0.35 | 0.32 | 0.46 | 0.42 | 0.60 | 0.54 | 0.78 | 0.69 | 0.17 | 0.88 |
| 0.30 | 0.30 | 0.84 | 0.84 | 0.32 | 0.39 | 0.40 | 0.50 | 0.49 | 0.62 | 0.58 | 0.75 | 0.17 | 0.91 |
| 0.30 | 0.30 | 0.84 | 0.84 | 0.32 | 0.39 | 0.40 | 0.50 | 0.49 | 0.62 | 0.58 | 0.75 | 0.17 | 0.91 |
| 0.08 | 0.08 | 0.16 | 0.16 | 0.24 | 0.23 | 0.31 | 0.29 | 0.38 | 0.36 | 0.47 | 0.44 | 0.17 | 0.53 |
| 0.02 | 0.02 | 0.04 | 0.04 | 0.05 | 0.05 | 0.06 | 0.06 | 0.07 | 0.07 | 0.08 | 0.08 | 0.14 | 0.09 |

**Table 6**
Relative error of eigenvalue and eigenvector inclusions for a random symmetric matrix.

| Rel. error | Minimum | Mean | Median | Maximum |
|---|---|---|---|---|
| Eigenvalues | $1.9 \cdot 10^{-14}$ | $1.5 \cdot 10^{-12}$ | $7.9 \cdot 10^{-14}$ | $1.3 \cdot 10^{-9}$ |
| Eigenvectors | $2.3 \cdot 10^{-12}$ | $3.8 \cdot 10^{-11}$ | $2.7 \cdot 10^{-11}$ | $4.0 \cdot 10^{-10}$ |

**Table 7**
Relative error of eigenvalue and eigenvector inclusions for two 10-fold clusters.

| Rel. error | Minimum | Mean | Median | Maximum |
|---|---|---|---|---|
| Eigenvalues | $1.7 \cdot 10^{-14}$ | $6.9 \cdot 10^{-14}$ | $6.3 \cdot 10^{-14}$ | $8.9 \cdot 10^{-13}$ |
| Eigenvectors | $6.7 \cdot 10^{-12}$ | $3.8 \cdot 10^{-3}$ | $7.8 \cdot 10^{-11}$ | $7.3 \cdot 10^{-1}$ |

**Table 8**
Relative error of eigenvalue and eigenvector inclusions for two 10-fold clusters with threshold on the cluster size.

| Rel. error | Minimum | Mean | Median | Maximum |
|---|---|---|---|---|
| Eigenvalues | $1.7 \cdot 10^{-14}$ | $6.9 \cdot 10^{-14}$ | $6.3 \cdot 10^{-14}$ | $8.9 \cdot 10^{-13}$ |
| Eigenvectors | $1.1 \cdot 10^{-12}$ | $9.2 \cdot 10^{-10}$ | $7.5 \cdot 10^{-11}$ | $2.2 \cdot 10^{-7}$ |

small eigenvector components do not dominate the result. The minimum distance, i.e., gap between the eigenvalues is about $4.6 \cdot 10^{-3}$ corresponding to the accuracy of the eigenvector inclusions.

The picture changes for clusters. For $e := 10^{-11}$ we generate a symmetric matrix with 10 random eigenvalues in a circle of radius $e$ around 0.1, another 10 random eigenvalues in a circle of radius $e$ around 0.2, and another 980 random eigenvalues in $[-1, -0.3] \cup [0.3, 1]$. The results are shown in Table 7. There is not much difference in the eigenvalue inclusions, and generally the eigenvector inclusions are of similar quality. However, the mean and maximum relative error is much worse. The reason is that the clusters could be separated by Algorithm `verifyeigall` so that only one cluster of size 2 remained rather than 2 clusters of size 10, all other eigenvalues have unique intervals and the bounds become poor due to the small gap. As an advantage, individual bounds for almost all eigenvectors are computed, as a disadvantage the bounds of the clustered eigenvectors are of less quality.

If it is sufficient to collect the clustered eigenvectors into invariant subspaces, a cure is to define a threshold `kappa` so that eigenvalues with distance below `kappa` are considered as a cluster. To that end the computation of `dist` in Algorithm `verifyeigall` is changed into

```
dist = ( Linf-kappa*abs(Linf)<=Linf’ ) & ...
    ( Lsup+kappa*abs(Lsup)>=Linf’ );
```

Using $\texttt{kappa} = 10^{-10}$ the results are shown in Table 8. Now inclusions of two clusters of 10 eigenvalues each with corresponding 10-dimensional invariant subspace are computed. The remaining inclusions cover the simple eigenvalues and eigenvectors, and all inclusions are of reasonable quality. These results are typical for other dimensions and cluster sizes, so that additional test results do not give much more information.

We close this section with some comparison between our presented methods for symmetric/Hermitian matrices and those in [13] for general real or complex matrices. First we look at the necessary separation of clusters. To that end we

**Table 9**
Median of relative errors by Algorithm `verifyeigall` and [13] for large tolerances.

| R | Eigenvalues | | Eigenvectors/Orthogonal subspaces | | Clusters |
|---|---|---|---|---|---|
| | `verifyeigall` | [13] | `verifyeigall` | [13] | |
| 0.006 | $3.4 \cdot 10^{-3}$ | $2.7 \cdot 10^{-3}$ | $2.4 \cdot 10^{-2}$ | $3.2 \cdot 10^{-2}$ | 1,2,3,4,5 |
| 0.007 | $3.9 \cdot 10^{-3}$ | – | $2.8 \cdot 10^{-2}$ | – | 1,2,3,4,5 |
| 0.1 | $5.6 \cdot 10^{-2}$ | – | $2.8 \cdot 10^{-1}$ | – | 1,2,3,4,5 |
| 0.3 | $2.4 \cdot 10^{-1}$ | – | $5.9 \cdot 10^{-1}$ | – | 1,{2,3},4,5 |
| 0.5 | $4.9 \cdot 10^{-1}$ | – | $8.4 \cdot 10^{-1}$ | – | {1,2,3},4,5 |
| 0.6 | $7.5 \cdot 10^{-1}$ | – | $3.4 \cdot 10^{-1}$ | – | {1,2,3,4},5 |

**Table 10**
Median of relative errors by Algorithm `verifyeigall` and [13] for large dimension.

| n | Eigenvalues | | Eigenvectors/Orthogonal subspaces | | Time ratio |
|---|---|---|---|---|---|
| | `verifyeigall` | [13] | `verifyeigall` | [13] | |
| 10 | $1.0 \cdot 10^{-15}$ | $2.4 \cdot 10^{-16}$ | $5.1 \cdot 10^{-15}$ | $5.6 \cdot 10^{-17}$ | 1.2 |
| 30 | $1.4 \cdot 10^{-15}$ | $2.4 \cdot 10^{-16}$ | $2.0 \cdot 10^{-14}$ | $5.6 \cdot 10^{-17}$ | 1.3 |
| 100 | $1.9 \cdot 10^{-15}$ | $2.4 \cdot 10^{-16}$ | $9.0 \cdot 10^{-14}$ | $4.2 \cdot 10^{-17}$ | 2.0 |
| 300 | $2.3 \cdot 10^{-15}$ | $2.3 \cdot 10^{-16}$ | $3.3 \cdot 10^{-13}$ | $4.2 \cdot 10^{-17}$ | 3.4 |
| 1000 | $2.4 \cdot 10^{-15}$ | $2.3 \cdot 10^{-16}$ | $1.3 \cdot 10^{-12}$ | $3.5 \cdot 10^{-17}$ | 4.1 |
| 3000 | $3.5 \cdot 10^{-15}$ | $2.4 \cdot 10^{-16}$ | $5.0 \cdot 10^{-12}$ | $3.5 \cdot 10^{-17}$ | 4.9 |
| 10,000 | $1.4 \cdot 10^{-14}$ | $2.4 \cdot 10^{-16}*$ | $7.2 \cdot 10^{-11}$ | $3.5 \cdot 10^{-17}*$ | 6.6 |

use the matrix in (4.5) with different radii $R$, i.e., apply the algorithms to `midrad(A,R)`. For each value of $R$, the median relative error of all eigenvalue inclusions computed by Algorithm `verifyeigall` and the general algorithm in [13] is displayed in Table 9, followed by the median relative error of all eigenvector/orthogonal subspace inclusions. The last column gives the size of the eigenvalue clusters detected by Algorithm `verifyeigall`.

For a radius $R = 0.006$ both algorithm can separate the eigenvalues, the cluster size for both is always 1. The quality of the inclusions is comparable, however, it is weak due to the large radius of all matrix components. The dimension of the matrix is 5, so the spectral norm of the radius matrix is $5R$. Hence, the maximum relative perturbation of the eigenvalues is of the order $5R/\|A\|$ and, for example in the first row, we cannot expect an error much better than $5R/\|A\| \approx 1.4 \cdot 10^{-3}$. That means the inclusions computed by Algorithm `verifyeigall` are wide, but without much room for improvement. From radius $R \geq 0.007$, the clusters are too close and the algorithm in [13] cannot compute any inclusion at all. One reason is the simultaneous preconditioning technique. In contrast, Algorithm `verifyeigall` treats the eigenvalues individually and can separate all of them until $R \leq 0.1$. For even larger radii, clusters appear, but still inclusions are computed.

Finally we present some accuracy and timing comparisons between Algorithm `verifyeigall` and the algorithm in [13] for random symmetric matrices of dimension $n$. As before the median relative error of all eigenvalue inclusions computed by Algorithm `verifyeigall` and the general algorithm in [13] is displayed in Table 10, followed by the median relative error of all eigenvector/orthogonal subspace inclusions. The last column displays the time ratio of the algorithm in [13] divided by that for Algorithm `verifyeigall`.

As can be seen the algorithm in [13] produces inclusions with smaller relative errors than Algorithm `verifyeigall`, in particular for the eigenvector/invariant subspace inclusions. However, the $*$ indicates that in 20% of the test cases the algorithm in [13] failed to compute an inclusion because the eigenvalues were not sufficiently separated.

There are two reasons for the better inclusions of the algorithm in [13]. First, the algorithm in [13] is based on [21] and computes inclusions of the error with respect to approximations of the eigenvalues and -vectors. In turn those approximations are improved by one Newton step. Generally, an inclusion of the error with respect to a good approximation is superior to a direct inclusion of the solution. That principle is the basis of many verification algorithms, cf. [11]. Second, the algorithm in [13] provides componentwise error bounds rather than the normwise bounds by Algorithm `verifyeigall`. The inclusions computed by [13] converted into normwise error estimates are still better by almost an order of magnitude, however, the discrepancy is not that large. Together, that explains the better accuracy, in particular for the eigenspaces. The drawback is that with increasing dimension the algorithm in [13] for general real or complex matrices becomes significantly slower than the new Algorithm `verifyeigall` and, as we saw before, the necessity of well separated clusters.

## 7. Singular value and vector bounds

For $A \in M_{m,n}$ let $A = X\Sigma Y^*$ be the economy size singular decomposition with $X \in M_{m,n}$ and $\Sigma, Y \in M_n$. Denote the singular values of $A$ by $\sigma_1 \geq \cdots \geq \sigma_n$, the orthogonal complement of $X$ by $X^\perp$, and set

$$B := \begin{pmatrix} 0 & A^* \\ A & 0 \end{pmatrix} \quad \text{and} \quad Q := \frac{1}{\sqrt{2}} \begin{pmatrix} Y & 0 & Y \\ X & \sqrt{2}X^\perp & -X \end{pmatrix}. \tag{7.1}$$

Then $Q^*BQ = \mathrm{diag}(\sigma_1, \ldots, \sigma_n, -\sigma_1, \ldots, -\sigma_n, 0, \ldots, 0)$. Error bounds for the singular values of a matrix based on residuals are very similar to those given in Theorem 6.2 for eigenvalues of a Hermitian matrix. As in the case of eigenvalues, better bounds can be derived using Lange's result [36]:

**Theorem 7.1.** *Let $A \in M_{m,n}$ and $H \in M_p$ with $m \geq n \geq p$. Denote the singular values of $H$ by $\theta_1 \leq \ldots \leq \theta_p$. For $X \in M_{m,p}$ and $Y \in M_{n,p}$ define the residuals*

$$E := AY - XH \qquad \text{and} \qquad F := A^*X - YH^*.$$

*Then there is a subset of singular values $\sigma_{i_1}, \ldots, \sigma_{i_p}$ of $A$ such that*

$$\max_{1\leq j\leq p} |\sigma_{i_j} - \theta_j| \leq \frac{\sqrt{\|E\|^2 + \|F\|^2}}{\sigma_{\min}(Y)}. \tag{7.2}$$

We note that in [36] the sharper bound

$$\max_{1\leq j\leq p} |\sigma_{i_j} - \theta_j| \leq \max\left\{ \sqrt{\frac{\|E\|^2 + \|F\|^2}{\sigma_{\min}(X)^2 + \sigma_{\min}(Y)^2}}, \frac{\|E\|}{\sigma_{\min}(Y)}, \frac{\|F\|}{\sigma_{\min}(Y)} \right\}$$

is proved. However, singular value and vector approximations are generally of high accuracy, so that (7.2) is very tight and more than sufficient for our purposes.

Theorems 4.1 and 7.1 are of similar nature, so Algorithm `verifyeigall` can be adapted directly. Now we added, as for eigenvalue clusters, the extra parameter `kappa` to collect singular values with distance below into one cluster. If not specified, `kappa` is set to zero.

```
function [S,mu,X,Y] = verifysvdall(A,kappa)
 if nargin==1, kappa = 0; end
 [m,n] = size(A);
 [Xs,Ls,Ys] = svd(mid(A),0);
 norm2z = norm_X2([Xs;Ys]);
 rho = norm_zBz(A,Xs,Ys)./norm2z; % inclusion of Rayleigh quotient
 Ls = diag(rho.mid);
 E = A*intval(Ys) - Xs*intval(Ls);
 F = A’*intval(Xs) - Ys*intval(Ls);
 sings = diag(Ls);
 singYsmin = mig( vecnorm(intval(Ys)) );
 normG = vecnorm(intval([E;F]));
 delta = mag( normG ./ singYsmin )’;
 num_mu = n;
 while 1
  S = max( 0 , midrad(sings,delta) );
  Sinf = repmat(S.inf,1,n); Ssup = repmat(S.sup,1,n);
  dist = ( Sinf-kappa*abs(Sinf)<=Sinf’ ) & ...
      ( Ssup+kappa*abs(Ssup)>=Sinf’ );
  dist = dist | dist’;
  [mu,binsizes] = conncomp(graph(dist),’OutputForm’,’cell’);
  J = find(binsizes>1);
  if any(J) && ( numels(mu)~=num_mu )
   num_mu = numels(mu);
   for j=J
    v = mu{j};
    singYsmin(v) = singmin(Ys(:,v));
    normG(v) = sqrt(NormBnd(E(:,v))^2+NormBnd(F(:,v))^2);
    delta(v) = mag( normG(v)/singYsmin(v(1)) );
   end
  else
   break
  end
 end
```

We apply the principle of eigenvalue improvements presented in Section 5 to improve the singular value approximations and inclusions. Here `norm_zBz` computes an inclusion of $z^*Bz$ for `z := [x;y]` similar to `norm_xAx`. Note that the first line of the while-loop assures that the inclusions of the singular values are nonnegative. As for eigenvalues, the singular values are refined applying Theorem 5.1 to the matrix $B$ in (7.1) by appending

**Table 11**
Improvement of singular value inclusions by Algorithm `refineeig`.

| Relerr | Symmetric matrix | | | Hermitian matrix | | |
|---|---|---|---|---|---|---|
| | Minimum | Median | Max | Minimum | Median | Max |
| Initial | $3.7 \cdot 10^{-14}$ | $5.5 \cdot 10^{-14}$ | $9.1 \cdot 10^{-14}$ | $4.3 \cdot 10^{-14}$ | $6.1 \cdot 10^{-14}$ | $9.6 \cdot 10^{-14}$ |
| Refined | $6.0 \cdot 10^{-15}$ | $6.6 \cdot 10^{-15}$ | $7.2 \cdot 10^{-15}$ | $9.6 \cdot 10^{-15}$ | $1.1 \cdot 10^{-14}$ | $1.1 \cdot 10^{-14}$ |
| Ratio | 6.1 | 8.3 | 12.6 | 4.5 | 5.8 | 8.4 |

```
L = refinesvd(A,Xs,Ys,S,rho,norm2z);
```

to the code above. The code is very similar to `refineeig` except that

```
if m>n, e(1:n+1:n^2) = S.inf; else e(1:n+1:n^2) = 2*S.inf; end
```

is used to cover the distance to the negative eigenvalues of $B$. Otherwise, for given left and right singular vector $x$ and $y$, the routines in (5.2) are adapted to compute inclusions of

$$D := \| \begin{pmatrix} x \\ y \end{pmatrix} \| \quad \text{and} \quad N := y^*A^*x + x^*Ay \quad \text{and} \quad \| \begin{pmatrix} Ay - rx \\ A^*x - ry \end{pmatrix} \|$$

for the inclusion `rho` of the Rayleigh quotient $N/D$ and $r :=$ `rho.mid` as before. Similar to eigenvalues, for complex input $A = A_1 + iA_2$, $x = x_1 + ix_2$ and $y = y_1 + iy_2$ it is better to use

$$y^*A^*x + x^*Ay = 2(\, x_1(A_1^*y_1 - A_2y_2) + x_2(A_1y_2 + A_2y_1)\,)\,.$$

As for eigenvalues we compare the singular value inclusion without and with refinement. The first test set are real or complex randomly generated 1000 × 200 matrices, where the median of the minimum, median and maximum of the relative errors of 100 samples is displayed. There is still considerable improvement, although not as large as for eigenvalues. In the presence of clustered singular values the results do not differ too much (see Table 11).

Next we discuss the computation of verified error bounds for the left and right singular vectors. Let $(\mathbf{S}, \mu)$ be results of Algorithm `verifysvdall` applied to rectangular $A \in M_{m,n}$ with economy-size singular decomposition $A = X\Sigma Y^*$. The singular value inclusions are $\mathbf{S}_j = [\tilde\sigma_j - \delta_j, \tilde\sigma_j + \delta_j]$ for $j \in [n]$, and $\mu = (\mu_1, \ldots, \mu_k)$ is a partition of $[n]$. As for eigenvalues, the cluster inclusions $\mathfrak{S}_\ell := \cup_{j\in\mu_\ell}\mathbf{S}_j$ are mutually disjoint for all $\ell \in [k]$. Applying Theorem 7.1 to each cluster separately shows that there is a suitable numbering $\sigma_1, \ldots, \sigma_n$ of the singular values of $A$ such that $\sigma_j \in \mathbf{S}_j$ for all $j \in [n]$.

For inclusions of singular vectors, there is the additional problem of zero or close to zero singular values. We first discuss square, then rectangular input matrices.

We first assume that $A$ is square so that $B$ as in (7.1) has no extra nullspace to deal with. Let $\tilde X$, $\tilde\Sigma$, $\tilde Y$ with $\tilde\Sigma :=$ `Ls` be the approximate singular value decomposition of $A$ computed by `[Xs,Ls,Ys] = svd(A)`. An approximation of the eigenspace of $B$ is

$$\tilde Z = \begin{pmatrix} \tilde Y & \tilde Y \\ \tilde X & -\tilde X \end{pmatrix}\,,$$

where the normalization is omitted. Following our inclusion approach for eigenvectors of Hermitian matrices, we define

$$G := \begin{pmatrix} A\tilde Y - \tilde X\tilde\Sigma \\ A^*\tilde X - \tilde Y\tilde\Sigma \end{pmatrix} \in M_{m+n,n}\,. \tag{7.3}$$

Let $\ell \in [k]$ be fixed but arbitrary and set $p := |\mu_\ell|$. Then

$$G(:, \mu_\ell) = B\tilde Z(:, \mu_\ell) - \tilde Z(:, \mu_\ell)\tilde\Sigma(\mu_\ell, \mu_\ell)\,. \tag{7.4}$$

The left and right singular subspaces to the singular values $\{\sigma_j : j \in \mu_\ell\}$ are spanned by $\hat X := X(:, \mu_\ell)$ and $\hat Y := Y(:, \mu_\ell)$, respectively. As for eigenvalues, we may assume $\sigma_{\min}(\tilde Y) > 0$, otherwise the bounds become meaningless. Note that this implies that $\tilde Z$ has full rank as well.

Following our approach for eigenvector inclusion, we define

$$\varepsilon_\ell := \min_{i\notin\mu_\ell, j\in\mu_\ell} |\sigma_i - \tilde\sigma_j| \geq \min_{i\notin\mu_\ell, j\in\mu_\ell} |\tilde\sigma_i - \tilde\sigma_j| - \delta_i. \tag{7.5}$$

Suppose $\ell < k$. Then the definitions (7.5) for the singular value clusters of $A$ and (6.3) for the eigenvalue clusters of $B$ coincide, and we apply Lemma 6.1 to $B$ to show that there exists $V \in M_{m,p}$ whose columns lie in the invariant subspace of $B$ to $\{\lambda_j : j \in \mu_\ell\}$ with

$$\|\tilde Z(:, \mu_\ell) - V\| \leq \frac{\|G(: \mu_\ell)\|}{\varepsilon_\ell} =: \tau_\ell\,. \tag{7.6}$$

The matrix $\tilde{Z}(:,\mu_\ell)$ approximates an eigenspace of $B$ to the eigenvalues $\{\lambda_j : j \in \mu_\ell\}$, which are positive because $\ell < k$. Since $V$ is composed of the left and right singular subspaces of $A$, it follows that both the distance of $\tilde{X}(:,\mu_\ell)$ to a subspace of the left singular vectors and the distance of $\tilde{Y}(:,\mu_\ell)$ to a subspace of the corresponding right singular vectors of $\{\sigma_j : j \in \mu_\ell\}$ is bounded by $\tau_\ell$. Then an inclusion of matrices with orthonormal columns spanning the corresponding space of left and right singular vectors is computed by Lemma 3.1.

For $\ell = k$, the singular value cluster $\mu_k$ is a special case because the distance to its negative counterpart may be smaller than the distance to the next larger singular value inclusion. Hence Lemma 6.1 is not directly applicable. Luckily this does not require a redefinition of (7.5) because we may use [35, Lemma 3.1] to handle this case:

**Lemma 7.2.** *Let $A \in M_{m,n}$, $H \in M_q$, $Q \in M_{n,q}$, $P \in M_{m,q}$ with $m \geq n$ be given. Define the residuals $E := AQ - PH$, $F := A^*P - QH^*$ and let $A = X\Sigma Y^*$ be an economy-size singular value decomposition of $A$ with $X \in M_{m,n}$, $\Sigma \in \mathbb{R}^{n\times n}$, $Y \in M_n$ and non-increasing order of singular values (with possible ambiguities in the choice of singular vectors). Furthermore, for some $s \in \{1,\ldots,n\}$, denote by $X_s$ the matrix consisting of the first $s$ columns of $X$ and let $Y_s$ be accordingly. If there is a $\zeta$ such that $\sigma_s(A) \geq \sigma_1(H) + \zeta$, then*

$$\zeta \cdot \max\{\|Y_s^*Q\|, \|X_s^*P\|\} \leq \max\{\|E\|, \|F\|\} \tag{7.7}$$

*is satisfied for any unitarily invariant norm $\|\cdot\|$.*

We set $\zeta = \varepsilon_k$ according to (7.5), $s = n - |\mu_k|$, $H = \tilde{\Sigma}(\mu_k,\mu_k)$, $Q = \tilde{Y}(:,\mu_k)$ and $P = \tilde{X}(:,\mu_k)$. Then

$$\sigma_s = \sigma_{n-|\mu_k|} \geq \max_{j\in\mu_k} \tilde{\sigma}_j + \zeta = \|H\| + \zeta$$

shows that Lemma 7.2 is applicable. The matrices $X_s$ and $Y_s$ span the orthogonal complements of $\hat{X} = X(:,\mu_k)$ and $\hat{Y} = Y(:,\mu_k)$, respectively. Thus

$$d := \max\{\|\left(I - \hat{Y}\hat{Y}^*\right)\tilde{Y}(:,\mu_k)\|, \|\left(I - \hat{X}\hat{X}^*\right)\tilde{X}(:,\mu_k)\|\} \tag{7.8}$$

is the maximal distance of $P = \tilde{X}(:,\mu_k)$ and $Q = \tilde{Y}(:,\mu_k)$ to their orthogonal projection onto $\hat{X}$ and $\hat{Y}$, respectively, and Lemma 7.2 yields, as in (7.6),

$$\begin{aligned} d &= \max\{\|Y_sY_s^*Q\|, \|X_sX_s^*P\|\} = \max\{\|Y_s^*Q\|, \|X_s^*P\|\} \\ &\leq \frac{\max\{\|AQ - PH\|, \|A^*P - QH^*\|\}}{\varepsilon_k} \leq \frac{\|G(:,\mu_k)\|}{\varepsilon_k} =: \tau_k. \end{aligned} \tag{7.9}$$

Let us now consider the rectangular case $m > n$. Now $B$ has additional $m - n$ zero eigenvalues requiring special attention. The singular vector spaces to a cluster $\mu_\ell$ for $\ell < k$ can be handled using Lemma 6.1 as before. Moreover, since the orthogonal complement $Y_s$ to $Y(:,\mu_k)$ is the same as in the quadratic case, Lemma 7.2 is still applicable as before, and the inclusion for the right singular vectors are derived in the same way as for quadratic $A$. Only the left singular vectors corresponding to the nullspace need some extra consideration.

To that end, we extend the set of singular values by the $m - n$ trivial zeros $\sigma_{n+1},\ldots,\sigma_m$. Naturally, we set $\tilde{\sigma}_{n+1} = \cdots = \tilde{\sigma}_m = 0$. The clustering has now to be done over the extended set, and we use a new index set $\overline{\mu}$. For the definition of $\overline{\mu}$, which coincides with $\mu$ in the first $k - 1$ indices, we distinguish two cases.

First, if $0 \in \mathfrak{S}_k$, then numerically the singular values $\{\sigma_j : j \in \mu_k\}$ cannot be distinguished from zero and we set $\overline{\mu}_k := \mu_k \cup \mathcal{N}$ with $\mathcal{N} := \{n+1,\ldots,m\}$. Second, if $0 \notin \mathfrak{S}_k$, we set extend $\overline{\mu}$ by $\overline{\mu}_k := \mu_k$ and $\overline{\mu}_{k+1} := \mathcal{N}$.

The index $\bar{k}$ shall be $k$ or $k+1$ depending on the separation of the nullspace. The quantities $\bar{\varepsilon}_\ell$ are then defined as in (7.5) but for the new index sets and including the trivial singular value entries. Using these modified definitions the inclusion of the left singular vector subspaces can be realized as in the quadratic case.

The inclusion of the left singular vector space corresponding to $\bar{\mu}_{\bar{k}}$ requires the full singular value decomposition of $A$. However, if we are not interested in the nullspace and/or the smallest non-trivial singular value could be separated from zero i.e., $\bar{k} = k + 1$, then we may skip $\bar{\mu}_{\bar{k}}$ from consideration. The advantage is that it allows us to use an economy-size approximate singular value decomposition of $A$. On the other hand, if we use an economy-size decomposition and cannot separate $\sigma_n$ from zero, the subspace inclusion derived for $\mu_k$ can be any subset of the space spanned by $[X(:,\mu_k)\ \ X^\perp]$.

Alternatively, the singular subspace of the additional $m - n$ zero singular values may be computed as the kernel of $A^*$ by one of the methods given in [37]. With a residual iteration as described in [37] it may be that the additional zero singular values can be separated from the cluster $\mu_\ell$.

In order to compute singular vector inclusions, we append the following code to Algorithm `verifysvdall`. This includes the refinement by Theorem 5.1, but for simplicity the extra treatment of the nullspace, which is straightforward to add, is omitted.

```
S = refinesvd(A,Xs,Ys,S,rho,norm2z);
if numels(mu)==1
 if isinf(delta(1)) || isnan(delta(1))
  r_left = inf(1,n);
```

**Table 12**
Relative error of singular value and singular vector inclusions for a random 1000 × 200 matrix.

| Rel. error | Minimum | Mean | Median | Maximum |
|---|---|---|---|---|
| Singular values | $4.0 \cdot 10^{-14}$ | $5.8 \cdot 10^{-14}$ | $5.6 \cdot 10^{-14}$ | $8.9 \cdot 10^{-14}$ |
| Left singular vectors | $1.1 \cdot 10^{-10}$ | $1.5 \cdot 10^{-9}$ | $9.3 \cdot 10^{-10}$ | $1.4 \cdot 10^{-8}$ |
| Right singular vectors | $5.4 \cdot 10^{-11}$ | $6.8 \cdot 10^{-10}$ | $4.3 \cdot 10^{-10}$ | $6.7 \cdot 10^{-9}$ |

```
   else
    r_left = zeros(1,n);
   end
   r_right = r_left;
  else
   sing = intval( repmat(S.mid,1,n)' );
   e = mig( min( abs(sing-Sinf) , abs(sing-Ssup) ));
   e(1:n+1:n^2) = inf;
   for j=J
    v = mu{j};
    e(v,v) = inf;
   end
   min_e = min(e);
   r_right = mag( normG ./ min_e );
   r_left = r_right;
   if m>n  % take care of additional zero singular values
    r_left(end) = mag( normG(end)/min(min_e(end),S.inf(end)) );
   end
  end
  alphaX = mag(1 - sum(intval(Xs).*conj(Xs)));
  alphaY = mag(1 - sum(intval(Ys).*conj(Ys)));
  rX = mag( alphaX + intval('sqrt2')*r_left );
  rY = mag( alphaY + intval('sqrt2')*r_right );
  for j=J
   v = mu{j};
   Ip = eye(length(v));
   alphaX = NormBnd(Ip - Xs(:,v)'*intval(Xs(:,v)));
   rX(v) = mag( alphaX + intval('sqrt2')*r_left(v) );
   alphaY = NormBnd(Ip - Ys(:,v)'*intval(Ys(:,v)));
   rY(v) = mag( alphaY + intval('sqrt2')*r_right(v) );
  end
  X = midrad( Xs , repmat(rX,m,1) );
  Y = midrad( Ys , repmat(rY,n,1) );
```

Here `sing` uses the midpoint of `S` because the singular value inclusions `S` have been refined. Note that for rectangular $A$ the last entry of the radius `r_left` for the left singular vectors is adapted. If $0 \in \mathfrak{S}_k$, then `S.inf(k)=0` and `r_left` becomes infinity.

**Theorem 7.3.** *Let an $m \times n$ real or complex, point or interval matrix $\mathbf{A}$ be given, and let $(\mathbf{S}, \mu, \mathbf{P}, \mathbf{Q})$ be the results of Algorithm* `verifysvdall` *applied to $\mathbf{A}$. Then for each $A \in \mathbf{A}$ the following is true. There is a numbering of the singular values $\sigma_1, \ldots, \sigma_n$ of $A$ such that $\sigma_j \in \mathbf{S}_j$ for all $j \in [n]$. For the partition $\mu = (\mu_1, \ldots, \mu_k)$ of $[n]$ into $k$ sets $\mu_\ell$ the $\mathfrak{S}_\ell := \cup_{j \in \mu_\ell} \mathbf{S}_j$ are a set of $k$ mutually disjoint intervals, each containing exactly $|\mu_\ell|$ singular values of $A$, and $\mathbf{P}(:, \mu_\ell)$ and $\mathbf{Q}(:, \mu_\ell)$ contain an orthonormal basis of the corresponding left and right singular value subspaces.*

In (7.6) we use $\|G\|$ as in (7.3), whereas in (7.7) the maximum of $\|E\|$ and $\|F\|$ is used. For the correctness of the code note that

$$\max(\|E\|, \|F\|) \leq \|G\| \,.$$

Otherwise the proof of correctness is similar to that for the eigenproblem.

As for eigenvalue and eigenvector inclusions we show some typical computational results for larger dimension. We take a 1000 × 200 random matrix and show the minimum, mean, median and maximum of the relative errors of the inclusions of singular values and the left and right singular vectors. The results are displayed in Table 12. The minimum distance, i.e., gap between the singular values is about $5.9 \cdot 10^{-3}$ which corresponds roughly to the accuracy of the singular vector inclusions.

**Table 13**
Relative error of singular value and singular vector inclusions for two 10-fold clusters.

| Rel. error | Minimum | Mean | Median | Maximum |
|---|---|---|---|---|
| Singular values | $4.1 \cdot 10^{-14}$ | $7.6 \cdot 10^{-14}$ | $6.2 \cdot 10^{-14}$ | $2.8 \cdot 10^{-13}$ |
| Left singular vectors | $3.5 \cdot 10^{-10}$ | $3.2 \cdot 10^{-2}$ | $4.2 \cdot 10^{-9}$ | $9.2 \cdot 10^{-1}$ |
| Right singular vectors | $5.6 \cdot 10^{-11}$ | $3.4 \cdot 10^{-2}$ | $7.7 \cdot 10^{-10}$ | $9.2 \cdot 10^{-1}$ |

**Table 14**
Relative error of singular value and singular vector inclusions for two 10-fold clusters with threshold on the cluster size.

| Rel. error | Minimum | Mean | Median | Maximum |
|---|---|---|---|---|
| Singular values | $1.4 \cdot 10^{-16}$ | $5.7 \cdot 10^{-14}$ | $5.7 \cdot 10^{-14}$ | $1.3 \cdot 10^{-13}$ |
| Left singular vectors | $8.9 \cdot 10^{-11}$ | $1.1 \cdot 10^{-8}$ | $3.1 \cdot 10^{-9}$ | $2.0 \cdot 10^{-8}$ |
| Right singular vectors | $1.7 \cdot 10^{-11}$ | $2.1 \cdot 10^{-9}$ | $5.8 \cdot 10^{-10}$ | $4.2 \cdot 10^{-8}$ |

The picture changes for clusters. For $e := 10^{-11}$ we generate a $1000 \times 200$ matrix with 10 random singular values in a circle of radius $e$ around 0.1, another 10 random singular values in a circle of radius $e$ around 0.2, and the other 180 randomly in $[0.3, 1]$. The results are shown in Table 13. There is not much difference in the singular value inclusions, and generally the singular vector inclusions are of similar quality as before. However, the mean and maximum relative error become much worse. The reason, similar to eigenvalues, is that Algorithm `verifysvdall` could separate the clusters into 200 individual intervals for the singular values. Therefore, the gap between the inclusions becomes small resulting in a poor quality of the inclusions. As before we use the threshold `kappa`$= 10^{-10}$ and obtain the results shown in Table 14. Now inclusions for two clusters of 10 singular values each with corresponding 10-dimensional invariant subspace are computed, together with the remaining inclusions for simple singular values and singular vectors. As a result, the singular vector inclusions are now of reasonable quality. As for eigenvalues, these results are typical for other dimensions and cluster sizes, and additional test results do not give much more information.

## Data availability

No data was used for the research described in the article.

## Acknowledgment

The authors wish to thank Florian Bünger for valuable remarks, and the anonymous referees for their fruitful comments.

## Appendix

As has been mentioned, Matlab introduces quite some interpretation overhead, in particular if user-defined data types such as `intval` are used. That can be improved significantly by using function calls and/or calculating left and right bounds individually using directed rounding as Florian Bünger did for the Taylor model and AWA toolbox in INTLAB [38]. For our applications we give a few examples.

Directed roundings are used as follows. The INTLAB command `setround(−1)` implies that all numerical operations including vector and matrix operations are executed using rounding downwards, i.e., the computed result is less than or equal to the exact result. That remains true until the next change of the rounding mode. Similarly, `setround(1)` changes the rounding to upwards, and `setround(0)` to nearest. The command

```
singXsmin = mig( vecnorm(intval(Xs)) );
```

in Algorithm `verifyeigall` computes a vector of lower bound of the Euclidean norm of the columns of `Xs` and can be replaced by

```
setround(-1)
singXsmin = sqrt(sum(sqr(Xs)));
```

After changing the rounding mode to downwards, `sqr(Xs)` yields lower bounds for the entrywise squares of the elements of `Xs`, followed by lower bounds for the column sums and their square roots. That code works for real input `Xs` and is easily adapted to complex input. Another example is

```
E = A*intval(Xs) - Xs*intval(Ls);
```

which may be replaced by

**Table 15**
Ratio of computing time using Matlab's operator concept vs. directed roundings.

| | n = 10 | n = 30 | n = 100 | n = 300 | n = 1000 |
|---|---|---|---|---|---|
| singXsmin | 57.6 | 55.0 | 27.3 | 35.3 | 22.5 |
| E | 11.6 | 6.8 | 5.6 | 3.2 | 2.4 |
| alpha | 16.1 | 12.4 | 4.9 | 4.7 | 3.4 |

```
D = repmat(diag(Ls)',n,1);
setround(-1)
Einf = A*Xs + Xs.*D;
setround(1)
Esup = A*Xs + Xs.*D;
E = intval(Einf,Esup,'infsup');
```

Finally we mention

```
alpha = mag( 1 - sum(intval(Xs).*conj(Xs)) );
```

which was used for eigenvector and singular vector bounds. It gives a vector of upper bounds of $|1 - x^*x|$ for the columns $x$ of Xs. For real input it may be replaced by

```
setround(-1)
alpha = abs(sum(X.*conj(X)) - 1);
setround(1)
alpha = max( alpha , abs(sum(X.*conj(X)) - 1) );
```

Here lower and upper bounds of sum(X.*conj(X)) - 1 are computed, and since the absolute value and maximum does not cause additional rounding errors, the final alpha is correct.

These are just a few examples. Following we give some computational results. For each dimension we execute the code for 100 samples and show the ratio of computing times (see Table 15).

This means quite some improvement. The ratio is larger for small dimensions and/or if $\mathcal{O}(n^2)$ operations are to be interpreted.

## References

[1] G. Golub, C. Van Loan, Matrix Computations, fourth ed., Johns Hopkins University Press, Baltimore, 2013.
[2] R. Horn, C. Johnson, Matrix Analysis, Cambridge University Press, second ed., 2012.
[3] G. Temple, The theory of Rayleigh's principle as applied to continuous systems, Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. 119 (1928) 276–293.
[4] T. Kato, On the upper and lower bounds of eigenvalues, J. Phys. Soc. Japan 4 (1949) 334–339.
[5] P. Zhu, M. Argentati, A. Knyazev, Bounds for the Rayleigh quotient and the spectrum of self-adjoint operators, SIAM J. Matrix Anal. Appl. 34 (2013) 244–256.
[6] W. Kahan, Inclusion Theorems for Clusters of Eigenvalues of Hermitian Matrices, Technical Report, Computer Science Dept. Univ. of Toronto, 1967.
[7] C. Davis, W. Kahan, The rotation of eigenvectors by a perturbation III, SIAM J. Numer. Anal. 7 (1970) 1–46.
[8] G. Stewart, J. Sun, Matrix Perturbation Theory, Academic Press, 1990.
[9] P. Zhu, A.V. Knyazev, Rayleigh–Ritz majorization error bounds of mixed type, SIAM J. Matrix Anal. Appl. 38 (2017) 30–49.
[10] A. Neumaier, Interval methods for systems of equations, in: Encyclopedia of Mathematics and Its Applications, Cambridge University Press, 1990.
[11] S. Rump, Verification methods: Rigorous results using floating-point arithmetic, Acta Numer. 19 (2010) 287–449.
[12] S. Oishi, K. Ichihara, M. Kashiwagi, K. Kimura, X. Liu, H. Masai, Y. Morikura, T. Ogita, K. Ozaki, S. Rump, K. Sekine, A. Takayasu, N. Yamanaka, Principle of Verified Numerical Computations, Corona publisher, Tokyo, Japan, 2018 (in Japanese).
[13] S. Rump, Computational error bounds for multiple or nearly multiple eigenvalues, SIAM J. Matrix Anal. Appl. 43 (2022) 1736–1754.
[14] R. Krawczyk, Fehlerabschätzung reeller Eigenwerte und Eigenvektoren von Matrizen, Computing 4 (1969) 281–293.
[15] R. Krawczyk, Newton-Algorithmen zur Bestimmung von Nullstellen mit Fehlerschranken, Computing 4 (1969) 187–201.
[16] R. Moore, A test for existence of solutions for non-linear systems, SIAM J. Numer. Anal. 4 (1977) 611–615.
[17] W. Kahan, A more Complete Interval Arithmetic, in: Lecture Notes for a Summer Course at the University of Michigan, 1968.
[18] A. Deif, Rigorous perturbation bounds for eigenvalues and eigenvectors of a matrix, J. Comput. Appl. Math. 57 (1995) 403–412.
[19] T. Yamamoto, Error bounds for computed eigenvalues and eigenvectors, Numer. Math. 34 (1980) 189–199.
[20] G. Alefeld, H. Spreuer, Iterative improvement of componentwise errorbounds for invariant subspaces belonging to a double or nearly double eigenvalue, Computing 36 (1986) 321–334.
[21] S. Rump, Computational error bounds for multiple or nearly multiple eigenvalues, Linear Algebra Appl. 324 (2001) 209–226.
[22] S. Miyajima, T. Ogita, S. Rump, S. Oishi, Fast verification for all eigenpairs in symmetric positive definite generalized eigenvalue problems, Reliab. Comput. 14 (2010) 24–45.
[23] S. Miyajima, T. Ogita, S. Oishi, Numerical verification for each eigenpair of symmetric matrix, Trans. JSIAM 16 (2006) 535–552 (in Japanese).
[24] A. Imakura, K. Morikuni, A. Takayasu, Verified eigenvalue and eigenvector computations using complex moments and the Rayleigh–Ritz procedure for generalized Hermitian eigenvalue problems, J. Comput. Appl. Math. 424 (2023) 114994.

[25] MATLAB User's Guide, The MathWorks Inc., 1987.

[26] S. Rump, INTLAB - INTerval LABoratory, in: T. Csendes (Ed.), Developments in Reliable Computing, Kluwer Academic Publishers, Dordrecht, 1999, pp. 77–104, http://www.ti3.tuhh.de/intlab.

[27] L. Collatz, Einschließungssatz für die charakteristischen Zahlen von Matrizen, Math. Z. 48 (1942) 221–226.

[28] S. Rump, Verified bounds for singular values, in particular for the spectral norm of a matrix and its inverse, BIT Numer. Math. 51 (2011) 367–384.

[29] Z. Cao, J. Xie, R. Li, A sharp version of Kahan's theorem on clustered eigenvalues, Linear Algebra Appl. 245 (1996) 147–156.

[30] W. Kahan, B.N. Parlett, E. Jiang, Residual bounds on approximate eigensystems of nonnormal matrices, SIAM J. Numer. Anal. 19 (1982) 470–484.

[31] O. Møller, Quasi double precision in floating-point arithmetic, BIT Numer. Math. 5 (1965) 37–50.

[32] S. Linnainmaa, Software for doubled-precision floating point computations, ACM Trans. Math. Softw. 7 (1981) 272–283.

[33] D. Bailey, A Fortran-90 based multiprecision system, ACM Trans. Math. Softw. 21 (1995) 379–387.

[34] K. Ozaki, T. Ogita, S. Rump, S. Oishi, Accurate matrix multiplication with multiple floating-point numbers, in: Proceedings of 2007 International Symposium on Nonlinear Theory and Its Applications, Vancouver, Canada, September 16–19, 2007.

[35] M. Lange, S. Rump, Verified inclusions for a nearest matrix of specified rank deficiency via a generalization of Wedin's sin(theta) theorem, BIT 61 (2021) 361–380.

[36] M. Lange, Residual bounds for some or all singular values, Linear Algebra Appl. 464 (2015) 28–37.

[37] R. Kobayashi, M. Lange, A. Minamihata, S. Rump, Verified inclusion of a basis of the null space, Reliab. Comput. 27 (2020) 26–41.

[38] F. Bünger, A Taylor model toolbox for solving ODEs implemented in MATLAB/INTLAB, J. Comput. Appl. Math. 368 (2020) 112511.