Japan Journal of Industrial and Applied Mathematics (2019) 36:435–459
https://doi.org/10.1007/s13160-019-00348-4

ORIGINAL PAPER

Iterative refinement for symmetric eigenvalue decomposition II: clustered eigenvalues

Takeshi Ogita[1] · **Kensuke Aishima**[2]

Received: 22 October 2018 / Revised: 5 February 2019 / Published online: 22 February 2019

Abstract
We are concerned with accurate eigenvalue decomposition of a real symmetric matrix A. In the previous paper (Ogita and Aishima in Jpn J Ind Appl Math 35(3): 1007–1035, 2018), we proposed an efficient refinement algorithm for improving the accuracy of all eigenvectors, which converges quadratically if a sufficiently accurate initial guess is given. However, since the accuracy of eigenvectors depends on the eigenvalue gap, it is difficult to provide such an initial guess to the algorithm in the case where A has clustered eigenvalues. To overcome this problem, we propose a novel algorithm that can refine approximate eigenvectors corresponding to clustered eigenvalues on the basis of the algorithm proposed in the previous paper. Numerical results are presented showing excellent performance of the proposed algorithm in terms of convergence rate and overall computational cost and illustrating an application to a quantum materials simulation.

Keywords Accurate numerical algorithm · Iterative refinement · Symmetric eigenvalue decomposition · Clustered eigenvalues

Mathematics Subject Classification 65F15 · 15A18 · 15A23

This study was partially supported by CREST, JST and JSPS KAKENHI Grant numbers 16H03917, 25790096.

✉ Takeshi Ogita
ogita@lab.twcu.ac.jp

Kensuke Aishima
aishima@hosei.ac.jp

[1] Division of Mathematical Sciences, School of Arts and Sciences, Tokyo Woman's Christian University, 2-6-1 Zempukuji, Suginami-ku, Tokyo 167-8585, Japan

[2] Faculty of Computer and Information Sciences, Hosei University, 3-7-2 Kajino-cho, Koganei-shi, Tokyo 184-8584, Japan

1 Introduction

Let A be a real symmetric $n \times n$ matrix. Since solving a standard symmetric eigenvalue problem $Ax = \lambda x$, where $\lambda \in \mathbb{R}$ is an eigenvalue of A and $x \in \mathbb{R}^n$ is an eigenvector of A associated with λ, is ubiquitous in scientific computing, it is important to develop reliable numerical algorithms for calculating eigenvalues and eigenvectors accurately. Excellent overviews on the symmetric eigenvalue problem can be found in references [20,23].

We are concerned with the eigenvalue decomposition of A such that

$$A = XDX^{\mathrm{T}}, \tag{1}$$

where X is an $n \times n$ orthogonal matrix whose ith columns are eigenvectors $x_{(i)}$ of A (called an eigenvector matrix) and $D = (d_{ij})$ is an $n \times n$ diagonal matrix whose diagonal elements are the corresponding eigenvalues $\lambda_i \in \mathbb{R}$, i.e., $d_{ii} = \lambda_i$ for $i = 1, \ldots, n$. Throughout the paper, we assume that

$$\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n,$$

and the columns of X are ordered correspondingly.

We here collect notation used in this paper. Let I and O denote the identity matrix and the zero matrix of appropriate size, respectively. Unless otherwise specified, $\|\cdot\|$ means $\|\cdot\|_2$, which denotes the Euclidean norm for vectors and the spectral norm for matrices. For legibility, if necessary, we distinguish between the approximate quantities and the computed results, e.g., for some quantity α we write $\widetilde{\alpha}$ and $\widehat{\alpha}$ as an approximation of α and a computed result for α, respectively.

The accuracy of an approximate eigenvector depends on the gap between the corresponding eigenvalue and its nearest neighbor eigenvalue (cf., e.g., [20, Theorem 11.7.1]). For simplicity, suppose all eigenvalues of A are simple. Let $\widehat{X} \in \mathbb{R}^{n\times n}$ be an approximation of X. Let $z_{(i)} := \widehat{x}_{(i)}/\|\widehat{x}_{(i)}\|$ for $i = 1, 2, \ldots, n$, where $\widehat{x}_{(i)}$ are the i-th columns of $\widehat{X}$. Moreover, for each i, suppose that the Ritz value $\mu_i := z_{(i)}^{\mathrm{T}} A z_{(i)}$ is closer to λ_i than to any other eigenvalues. Let $gap(\mu_i)$ denote the smallest difference between μ_i and any other eigenvalue, i.e., $gap(\mu_i) := \min_{j\neq i} |\mu_i - \lambda_j|$. Then, it holds for all i that

$$|\sin\theta(x_{(i)}, z_{(i)})| \le \frac{\|Az_{(i)} - \mu_i z_{(i)}\|}{gap(\mu_i)}, \quad \theta(x_{(i)}, z_{(i)}) := \arccos(|x_{(i)}^{\mathrm{T}} z_{(i)}|).$$

Suppose $\widehat{X}$ is obtained by some backward stable algorithm with the relative rounding error unit $\mathbf{u}$ in floating-point arithmetic. For example, $\mathbf{u} = 2^{-53}$ for IEEE 754 binary64. Then, there exists $\Delta_A^{(i)}$ such that

$$(A + \Delta_A^{(i)})z_{(i)} = \mu_i z_{(i)}, \quad \|\Delta_A^{(i)}\| = \mathcal{O}(\|A\|\mathbf{u}),$$

which implies $\|Az_{(i)} - \mu_i z_{(i)}\| = \mathcal{O}(\|A\|\mathbf{u})$, and hence, for all i,

$$|\sin\theta(x_{(i)}, z_{(i)})| \le \alpha_i, \quad \alpha_i = \mathcal{O}\left(\frac{\|A\|\mathbf{u}}{gap(\mu_i)}\right). \tag{2}$$

The smaller the eigenvalue gap, the worse the accuracy of a computed eigenvector. Therefore, refinement algorithms for eigenvectors are useful for obtaining highly accurate results. For example, highly accurate computations of a few or all eigenvectors are crucial for large-scale electronic structure calculations in material physics [24,25], in which specific interior eigenvalues with associated eigenvectors need to be computed. On related work on refinement algorithms for symmetric eigenvalue decomposition, see the previous paper [17] for details.

In [17], we proposed a refinement algorithm for the eigenvalue decomposition of A, which works not for an individual eigenvector but for all eigenvectors. Since the algorithm is based on Newton's method, it converges quadratically, provided that an initial guess is sufficiently accurate. In practice, although the algorithm refines computed eigenvectors corresponding to sufficiently separated simple eigenvalues, it cannot refine computed eigenvectors corresponding to "nearly" multiple eigenvalues. This is because it is difficult for standard numerical algorithms in floating-point arithmetic to provide sufficiently accurate initial approximate eigenvectors corresponding to nearly multiple eigenvalues as shown in (2). The purpose of this paper is to remedy this problem, i.e., we aim to develop a refinement algorithm for the eigenvalue decomposition of a symmetric matrix with clustered eigenvalues.

We briefly explain the idea of our proposed algorithm. We focus on the so-called $\sin\theta$ theorem by Davis–Kahan [5, Section 2] as follows. For an index set $\mathcal{J}$ with $|\mathcal{J}| = \ell < n$, let $X_{\mathcal{J}} \in \mathbb{R}^{n\times\ell}$ denote the eigenvector matrix comprising $x_{(j)}$ for all $j \in \mathcal{J}$. For $1 \le k \le \ell$, let μ_k denote the Ritz values for the subspace spanned by some given vectors with $\mu_1 \le \cdots \le \mu_\ell$, and let z_k be the corresponding normalized Ritz vectors. Assume that the eigenvalues λ_i for all $i \notin \mathcal{J}$ are entirely outside of $[\mu_1, \mu_\ell]$. Let *Gap* denote the smallest difference between the Ritz values μ_k for all k, $1 \le k \le \ell$, and the eigenvalues λ_i for all $i \notin \mathcal{J}$, i.e., $Gap := \min\{|\mu_k - \lambda_i| \,:\, 1 \le k \le \ell,\ i \notin \mathcal{J}\}$. Moreover, let $Z_{\mathcal{J}} := [z_1, \ldots, z_\ell] \in \mathbb{R}^{n\times\ell}$. Then, we obtain

$$|\sin\Theta(X_{\mathcal{J}}, Z_{\mathcal{J}})| \le \frac{\|AZ_{\mathcal{J}} - Z_{\mathcal{J}}(Z_{\mathcal{J}}^{\mathrm{T}} A Z_{\mathcal{J}})\|}{Gap},$$
$$\Theta(X_{\mathcal{J}}, Z_{\mathcal{J}}) := \arccos(\|X_{\mathcal{J}}^{\mathrm{T}} Z_{\mathcal{J}}\|).$$

This indicates that the subspace spanned by eigenvectors associated with the clustered eigenvalues is not very sensitive to perturbations, provided that the gap between the clustered eigenvalues and the others is sufficiently large. That means backward stable algorithms can provide a sufficiently accurate initial guess of the "subspace" corresponding to the clustered eigenvalues. To extract eigenvectors from the subspace correctly, relatively larger gaps are necessary between the clustered eigenvalues as can be seen from (2). Thus, we first apply the algorithm (Algorithm 1: RefSyEv) in the previous paper [17] to the initial approximate eigenvector matrix for improving

the subspace corresponding to the clustered eigenvalues. Then, we divide the entire problem into subproblems, each of which corresponds to each cluster of eigenvalues. Finally, we expand eigenvalue gaps in each subproblem by using a diagonal shift and compute eigenvectors of each subproblem, which can be used for refining approximate eigenvectors corresponding to clustered eigenvalues in the entire problem.

One might notice that the above procedure is similar to the classical shift-invert technique to transform eigenvalue distributions. In addition, the MRRR algorithm [6] also employs a shift strategy to increase relative gaps between clustered eigenvalues for computing the associated eigenvectors. In other words, it is well known that the diagonal shift is useful for solving eigenvalue problems accurately. Our contribution is to show its effectiveness on the basis of appropriate error analysis with the adaptive use of higher precision arithmetic, which leads to the derivation of the proposed algorithm.

In the same spirit of the previous paper [17], our proposed algorithm primarily comprises matrix multiplication, which accounts for the majority of the computational cost. Therefore, we can utilize higher precision matrix multiplication efficiently. For example, XBLAS [13] and other efficient algorithms [16,19,22] based on so-called error-free transformations for accurate matrix multiplication are available for practical implementation.

The remainder of the paper is organized as follows. In Sect. 2, we recall the refinement algorithm (Algorithm 1) proposed in the previous paper [17] together with its convergence theory. For practical use, we present a rounding error analysis of Algorithm 1 in finite precision arithmetic in Sect. 3, which is useful for setting working precision and shows achievable accuracy of approximate eigenvectors obtained by using Algorithm 1. In Sect. 4, we show the behavior of Algorithm 1 for clustered eigenvalues, which explains the effect of nearly multiple eigenvalues on computed results and leads to the derivation of the proposed algorithm. On the basis of Algorithm 1, we propose a refinement algorithm (Algorithm 2: RefSyEvCL) that can also be applied to matrices with clustered eigenvalues in Sect. 5. In Sect. 6, we present some numerical results showing the behavior and performance of the proposed algorithm together with an application to a quantum materials simulation as a real-world problem.

For simplicity, we basically handle only real matrices. As mentioned in the previous paper [17], the discussions in this paper can also be extended to generalized symmetric (Hermitian) definite eigenvalue problems.

2 Basic algorithm and its convergence theory

In this section, we introduce the refinement algorithm proposed in the previous paper [17], which is the basis of the algorithm proposed in this paper.

Let $A = A^{\mathrm{T}} \in \mathbb{R}^{n\times n}$. The eigenvalues of A are denoted by $\lambda_i \in \mathbb{R}$, $i = 1, \ldots, n$. Then $\|A\| = \max_{1\le i\le n} |\lambda_i| = \max(|\lambda_1|, |\lambda_n|)$. Let $X \in \mathbb{R}^{n\times n}$ denote an orthogonal eigenvector matrix comprising normalized eigenvectors of A, and let $\widehat{X}$ denote an approximation of X with $\widehat{X}$ being nonsingular. In addition, define $E \in \mathbb{R}^{n\times n}$ such that

$$X = \widehat{X}(I + E).$$

In the previous paper, we presented the following algorithm for the eigenvalue decomposition of A, which is designed to be applied iteratively. For later use in Sect. 5, the algorithm also allows the case where an input $\widehat{X}$ is rectangular, i.e., $\widehat{X} \in \mathbb{R}^{n\times\ell}$, $\ell < n$.

Algorithm 1 RefSyEv: Refinement of approximate eigenvectors of a real symmetric matrix. Higher-precision arithmetic is required for all the computations except line 6.

Input: $A = A^{\mathrm{T}} \in \mathbb{R}^{n\times n}$, $\widehat{X} \in \mathbb{R}^{n\times\ell}$, $1 \le \ell \le n$
Output: $X' \in \mathbb{R}^{n\times\ell}$, $\widetilde{D} = \mathrm{diag}(\widetilde{\lambda}_i) \in \mathbb{R}^{\ell\times\ell}$, $\widetilde{E} \in \mathbb{R}^{\ell\times\ell}$, $\omega \in \mathbb{R}$
1: **function** $[X', \widetilde{D}, \widetilde{E}, \omega] \leftarrow$ RefSyEv$(A, \widehat{X})$
2: $\quad R \leftarrow I - \widehat{X}^{\mathrm{T}}\widehat{X}$
3: $\quad S \leftarrow \widehat{X}^{\mathrm{T}}A\widehat{X}$
4: $\quad \widetilde{\lambda}_i \leftarrow s_{ii}/(1 - r_{ii})$ for $i = 1, \ldots, \ell$ ▷ Compute approximate eigenvalues.
5: $\quad \widetilde{D} \leftarrow \mathrm{diag}(\widetilde{\lambda}_i)$
6: $\quad \omega \leftarrow 2(\|S - \widetilde{D}\|_2 + \|A\|_2\|R\|_2)$
7: $\quad \widetilde{e}_{ij} \leftarrow \begin{cases} \dfrac{s_{ij} + \widetilde{\lambda}_j r_{ij}}{\widetilde{\lambda}_j - \widetilde{\lambda}_i} & \text{if } |\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega \\ r_{ij}/2 & \text{otherwise} \end{cases}$ for $1 \le i, j \le \ell$ ▷ Compute $\widetilde{E}$.
8: $\quad X' \leftarrow \widehat{X} + \widehat{X}\widetilde{E}$ ▷ Update $\widehat{X}$ by $\widehat{X}(I + \widetilde{E})$.
9: **end function**

In [17, Theorem 1], we presented the following theorem that states the quadratic convergence of Algorithm 1 if all eigenvalues are simple and a given $\widehat{X}$ is sufficiently close to X.

Theorem 1 (Ogita–Aishima [17]) *Let A be a real symmetric $n \times n$ matrix with simple eigenvalues λ_i, $i = 1, 2, \ldots, n$, and a corresponding orthogonal eigenvector matrix $X \in \mathbb{R}^{n\times n}$. For a given nonsingular $\widehat{X} \in \mathbb{R}^{n\times n}$, suppose that Algorithm 1 is applied to A and $\widehat{X}$ in real arithmetic, and X' is the quantity calculated in Algorithm 1. Define E and E' such that $X = \widehat{X}(I + E)$ and $X = X'(I + E')$, respectively. If*

$$\|E\| < \min\left(\frac{\min_{i\neq j}|\lambda_i - \lambda_j|}{10n\|A\|}, \frac{1}{100}\right), \tag{3}$$

then we have

$$\|E'\| < \frac{5}{7}\|E\|, \tag{4}$$

$$\limsup_{\|E\|\to 0} \frac{\|E'\|}{\|E\|^2} \le \frac{6n\|A\|}{\min_{i\neq j}|\lambda_i - \lambda_j|}. \tag{5}$$

In the following, we review the discussion in [17, §3.2] for exactly multiple eigenvalues. If $\widetilde{\lambda}_i \approx \widetilde{\lambda}_j$ corresponding to multiple eigenvalues $\lambda_i = \lambda_j$, we compute $\widetilde{e}_{ij} = \widetilde{e}_{ji} = r_{ij}/2$ for (i, j) such that $|\widetilde{\lambda}_i - \widetilde{\lambda}_j| \le \omega$.

To investigate the above exceptional process, define the index sets $\mathcal{M}_k$, $k = 1, 2, \ldots, n_{\mathcal{M}}$, for multiple eigenvalues $\{\lambda_i\}_{i\in\mathcal{M}_k}$ satisfying the following conditions:

$$\begin{cases} \text{(a) } \mathcal{M}_k \subseteq \{1, 2, \ldots, n\} \text{ with } n_k := |\mathcal{M}_k| \geq 2 \\ \text{(b) } \lambda_i = \lambda_j, \ \forall i, j \in \mathcal{M}_k \\ \text{(c) } \lambda_i \neq \lambda_j, \ \forall i \in \mathcal{M}_k, \ \forall j \in \{1, 2, \ldots, n\} \setminus \mathcal{M}_k \end{cases}. \tag{6}$$

Note that the eigenvectors corresponding to multiple eigenvalues are not unique. Hence, using the above index sets, let Y be an eigenvector matrix defined such that, for all k, the $n_k \times n_k$ submatrices of $\widehat{X}^{-1}Y$ corresponding to $\{\lambda_i\}_{i\in\mathcal{M}_k}$ are symmetric and positive definite. Since then Y is unique, define F such that $Y = \widehat{X}(I + F)$. Define $R := I - \widehat{X}^{\mathrm{T}}\widehat{X}$ and $S := \widehat{X}^{\mathrm{T}}A\widehat{X}$. Then, using the orthogonality $Y^{\mathrm{T}}Y = I$ and the diagonality $Y^{\mathrm{T}}AY = D$, we have

$$F + F^{\mathrm{T}} = R + \Delta_1, \quad \|\Delta_1\| \leq \chi(\epsilon)\epsilon^2, \tag{7}$$

$$D - DF - F^{\mathrm{T}}D = S + \Delta_2, \quad \|\Delta_2\| \leq \chi(\epsilon)\|A\|\epsilon^2, \tag{8}$$

where $\epsilon := \|F\|$ and

$$\chi(\epsilon) := \frac{3 - 2\epsilon}{(1 - \epsilon)^2}.$$

The above equations can be obtained in the same manner as in our previous paper [17, Eqs. (7) and (11)] by replacing E with F in the equations.

In a similar way to Newton's method (cf. e.g., [3, p. 236]), dropping the second order terms in (7) and (8) yields Algorithm 1, and the next convergence theorem is provided [17, Theorem 2].

Theorem 2 (Ogita–Aishima [17]) *Let A be a real symmetric $n \times n$ matrix with the eigenvalues λ_i, $i = 1, 2, \ldots, n$. Suppose A has multiple eigenvalues with index sets $\mathcal{M}_k$, $k = 1, 2, \ldots, n_{\mathcal{M}}$, satisfying (6). Let $\mathcal{V}$ be the set of $n \times n$ orthogonal eigenvector matrices of A. For a given nonsingular $\widehat{X} \in \mathbb{R}^{n\times n}$, suppose that Algorithm 1 is applied to A and $\widehat{X}$ in real arithmetic, and X' and ω are the quantities calculated in Algorithm 1. Let $Y, Y' \in \mathcal{V}$ be defined such that, for all k, the $n_k \times n_k$ submatrices of $\widehat{X}^{-1}Y$ and $(X')^{-1}Y'$ corresponding to $\{\lambda_i\}_{i\in\mathcal{M}_k}$ are symmetric and positive definite. Define F and F' such that $Y = \widehat{X}(I + F)$ and $Y' = X'(I + F')$, respectively. Furthermore, suppose that*

$$\|F\| < \frac{1}{3}\min\left(\frac{\min_{\lambda_i\neq\lambda_j}|\lambda_i - \lambda_j|}{10n\|A\|}, \frac{1}{100}\right).$$

Then, we obtain

$$\|F'\| < \frac{5}{7}\|F\|,$$

$$\limsup_{\|F\|\to 0} \frac{\|F'\|}{\|F\|^2} \le 3\left(\frac{6n\|A\|}{\min_{\lambda_i\neq\lambda_j}|\lambda_i-\lambda_j|}\right).$$

On the basis of the above convergence theorems, let us consider the iterative refinement using Algorithm 1:

$$X^{(0)} \leftarrow \widehat{X} \in \mathbb{R}^{n\times n}, \quad X^{(\nu+1)} \leftarrow \mathsf{RefSyEv}(A, X^{(\nu)}) \quad \text{for } \nu = 0, 1, \ldots$$

Then, $X^{(\nu+1)} = X^{(\nu)}(I + \widetilde{E}^{(\nu)})$ for $\nu = 0, 1, \ldots$, where $\widetilde{E}^{(\nu)} = (\widetilde{e}_{ij}^{(\nu)})$ are the quantities calculated in line 7 of Algorithm 1. In practice, it is likely that ordinary precision floating-point arithmetic, such as IEEE 754 binary32 or binary64, is used for calculating an approximation $\widehat{X}$ to an eigenvector matrix X of a given symmetric matrix A by some backward stable algorithm. It is natural to use such $\widehat{X}$ as an initial guess $X^{(0)}$ in Algorithm 1. However, if A has nearly multiple eigenvalues, it is difficult to obtain a sufficiently accurate $X^{(0)}$ in ordinary precision floating-point arithmetic such that Algorithm 1 works well. To overcome this problem, we develop a practical algorithm for clustered eigenvalues, which is proposed as Algorithm 2 in Sect. 5.

3 Rounding error analysis for basic algorithm

If Algorithm 1 is performed in finite precision arithmetic with the relative rounding error unit $\mathbf{u}_h$, the accuracy of a refined eigenvector matrix X' is restricted by $\mathbf{u}_h$. Since $\widehat{X}$ is improved quadratically when using real arithmetic, $\mathbf{u}_h$ must correspond to $\|E\|^2$ to preserve the convergence property of Algorithm 1. We explain the details in the following. For simplicity, we consider the real case. The extension to the complex case is obvious.

Let $\mathbb{F}_h$ be a set of floating-point numbers with the relative rounding error unit $\mathbf{u}_h$. We define the rounding operator fl_h such that $fl_h : \mathbb{R} \to \mathbb{F}_h$ and assume the use of the following standard floating-point arithmetic model [11]. For $a, b \in \mathbb{F}_h$ and $\circ \in \{+, -, \times, /\}$, it holds that

$$fl_h(a \circ b) = (a \circ b)(1 + \delta_h), \quad |\delta_h| \le \mathbf{u}_h. \tag{9}$$

For example, it is satisfied in IEEE 754 floating-point arithmetic barring overflow and underflow.

Suppose all elements of A and $\widehat{X}$ are exactly representable in $\mathbb{F}_h$, i.e., $A, \widehat{X} \in \mathbb{F}_h^{n\times n}$, and $\|\widehat{x}_{(i)}\| \approx 1$ for all i. Let $\widehat{R}$, $\widehat{S}$, and $\widehat{E}$ denote the computed results of R, S, and $\widetilde{E}$ in Algorithm 1, respectively. Define Δ_R, Δ_S, and Δ_E such that

$$\widehat{R} = R + \Delta_R, \quad \widehat{S} = S + \Delta_S, \quad \widehat{E} = \widetilde{E} + \Delta_E.$$

From a standard rounding error analysis as in [11], we obtain

$$(\Delta_R)_{ij} = \mathcal{O}(\mathbf{u}_h), \quad (\Delta_S)_{ij} = \mathcal{O}(\|A\|\mathbf{u}_h) \quad \text{for } 1 \le i, j \le n.$$

For the computed results $\widehat{\lambda}_i$ of $\widetilde{\lambda}_i$, $i = 1, 2, \ldots, n$, in Algorithm 1,

$$\begin{aligned}\widehat{\lambda}_i &= \frac{\widehat{s}_{ii}}{(1-\widehat{r}_{ii})(1+\delta_1)}(1+\delta_2), \quad |\delta_k| \le \mathbf{u}_h,\ k = 1, 2\\ &= \frac{\widehat{s}_{ii}}{1-\widehat{r}_{ii}}(1+\phi) = \frac{s_{ii}+(\Delta_S)_{ii}}{1-r_{ii}-(\Delta_R)_{ii}}(1+\phi), \quad |\phi| = \mathcal{O}(\mathbf{u}_h)\\ &= \widetilde{\lambda}_i + \varepsilon_i, \quad |\varepsilon_i| = \mathcal{O}(\|A\|\mathbf{u}_h).\end{aligned}$$

For all (i, j) satisfying $|\widehat{\lambda}_i - \widehat{\lambda}_j| > \widehat{\omega}$, where $\widehat{\omega}$ is an approximation of ω computed in floating-point arithmetic in Algorithm 1,

$$\begin{aligned}\widehat{e}_{ij} &= \frac{(\widehat{s}_{ij} + \widehat{\lambda}_j\widehat{r}_{ij}(1+\delta_3))(1+\delta_4)}{(\widehat{\lambda}_j - \widehat{\lambda}_i)(1+\delta_5)}(1+\delta_6), \quad |\delta_k| \le \mathbf{u}_h,\ k = 3, 4, 5, 6\\ &= \frac{\widehat{s}_{ij} + \widehat{\lambda}_j\widehat{r}_{ij}}{\widehat{\lambda}_j - \widehat{\lambda}_i} + \tau_{ij}, \quad |\tau_{ij}| = \mathcal{O}(\beta_{ij}\mathbf{u}_h), \quad \beta_{ij} := \frac{\|A\|}{|\lambda_i - \lambda_j|}\\ &= \frac{s_{ij} + (\Delta_S)_{ij} + (\widetilde{\lambda}_j + \varepsilon_j)(r_{ij} + (\Delta_R)_{ij})}{\widetilde{\lambda}_j - \widetilde{\lambda}_i + (\varepsilon_j - \varepsilon_i)} + \tau_{ij}\\ &= \widetilde{e}_{ij} + \gamma_{ij}^{(1)}, \quad |\gamma_{ij}^{(1)}| = \mathcal{O}(\beta_{ij}\mathbf{u}_h).\end{aligned}$$

Then

$$|(\Delta_E)_{ij}| = |\widehat{e}_{ij} - \widetilde{e}_{ij}| \le |\gamma_{ij}^{(1)}| = \mathcal{O}(\beta_{ij}\mathbf{u}_h).$$

For other (i, j), we have

$$\begin{aligned}\widehat{e}_{ij} &= \frac{\widehat{r}_{ij}}{2}(1+\delta_7) = \frac{r_{ij} + (\Delta_R)_{ij}}{2}(1+\delta_7), \quad |\delta_7| \le \mathbf{u}_h\\ &= \widetilde{e}_{ij} + \gamma_{ij}^{(2)}, \quad |\gamma_{ij}^{(2)}| = \mathcal{O}(\mathbf{u}_h).\end{aligned}$$

Then

$$|(\Delta_E)_{ij}| = |\widehat{e}_{ij} - \widetilde{e}_{ij}| \le |\gamma_{ij}^{(2)}| = \mathcal{O}(\mathbf{u}_h).$$

In summary, we obtain

$$\|\Delta_E\| \le \sqrt{\sum_{1\le i,j\le n} |(\Delta_E)_{ij}|^2} = \mathcal{O}(\beta\mathbf{u}_h), \quad \beta := \frac{\|A\|}{\min_{\lambda_i\ne\lambda_j}|\lambda_i - \lambda_j|}, \tag{10}$$

where β is the reciprocal of the minimum gap between the eigenvalues normalized by $\|A\|$. For the computed result $\widehat{X}'$ of $X' = \widehat{X}(I + \widetilde{E})$ in Algorithm 1,

$$\begin{aligned}\widehat{X}' &= \widehat{X} + \widehat{X}\widehat{E} + \widehat{\Delta}, \quad \|\widehat{\Delta}\| = \mathcal{O}(\mathbf{u}_h)\\ &= \widehat{X}(I + \widetilde{E}) + \widehat{X}(\widehat{E} - \widetilde{E}) + \widehat{\Delta}\end{aligned}$$

$$= X' + \widehat{X}\Delta_E + \widehat{\Delta},$$

and, using (10),

$$\|\widehat{X}' - X'\| \leq \|\widehat{X}\|\|\Delta_E\| + \|\widehat{\Delta}\| = \mathcal{O}(\beta \mathbf{u}_h). \tag{11}$$

Thus, if a given $\widehat{X}$ is sufficiently close to X in such a way that the assumption (3) holds, combining (5) and (11) yields

$$\|\widehat{X}' - X\| \leq \|\widehat{X}' - X'\| + \|X' - X\| = \mathcal{O}(\beta \cdot \max(\mathbf{u}_h, \|E\|^2)). \tag{12}$$

If A has nearly multiple eigenvalues and (3) does not hold, then the convergence of Algorithm 1 to an eigenvector matrix of A is guaranteed neither in real arithmetic nor in finite precision arithmetic regardless of the value of $\mathbf{u}_h$. We will deal with such an ill-conditioned case in Sect. 5.

Remark 1 As can be seen from (12), with a fixed $\mathbf{u}_h$, iterative use of Algorithm 1 eventually computes an approximate eigenvector matrix that is accurate to $\mathcal{O}(\beta \mathbf{u}_h)$, provided that the assumption (3) in Theorem 1 holds in each iteration. This will be confirmed numerically in Sect. 6. □

Let us consider the most likely scenario where $\widehat{X}$ is computed by some backward stable algorithm in ordinary precision floating-point arithmetic with the relative rounding error unit $\mathbf{u}$. From (2), we have

$$\max_{1 \leq i \leq n} |\sin \theta(x_{(i)}, \widehat{x}_{(i)})| \leq \alpha, \quad \alpha = \mathcal{O}(\beta \mathbf{u})$$

under the assumption that $\beta \approx \|A\| / \min_{1 \leq i \leq n} gap(\mu_i)$. Thus,

$$\|E\| \approx \|\widehat{X} - X\| = \mathcal{O}(\beta \mathbf{u}).$$

From (12), we obtain

$$\|\widehat{X}' - X\| = \mathcal{O}(\beta \cdot \max(\mathbf{u}_h, \beta^2 \mathbf{u}^2)). \tag{13}$$

Therefore, $\mathbf{u}_h$ should be less than $\beta^2 \mathbf{u}^2$ in order to preserve convergence speed for the first iteration by Algorithm 1.

Suppose that $\|\widehat{X} - X\| = c\beta\mathbf{u}$ and $\|\widehat{X}' - X\| = c'\beta^3\mathbf{u}^2$ where c and c' are some constants. If $c''\beta^2\mathbf{u} < 1$ for $c'' := c'/c$, then an approximation of X is improved in the sense that $\|\widehat{X}' - X\| < \|\widehat{X} - X\|$. In other words, if β is too large such that $c''\beta^2\mathbf{u} \geq 1$, Algorithm 1 may not work well.

In general, define $E^{(\nu)} \in \mathbb{R}^{n \times n}$ such that $X = \widehat{X}^{(\nu)}(I + E^{(\nu)})$ for $\nu = 0, 1, \ldots$, where $\widehat{X}^{(0)}$ is an initial guess and $\widehat{X}^{(\nu)}$ is a result of the νth iteration of Algorithm 1 with working precision $\mathbf{u}_h^{(\nu)}$ for $\nu = 1, 2, \ldots$. To preserve the convergence speed, we need to set $\mathbf{u}_h^{(\nu)}$ satisfying $\mathbf{u}_h^{(\nu)} < \|E^{(\nu-1)}\|^2$ as can be seen from (12). Although we do not know $\|E^{(\nu-1)}\|$, we can estimate $\|E^{(\nu-1)}\|$ by $\|\widetilde{E}^{(\nu-1)}\|$ where $\widetilde{E}^{(\nu-1)}$ is computed at the $(\nu - 1)$-st iteration of Algorithm 1.

4 Effect of nearly multiple eigenvalues in basic algorithm

In general, a given matrix A in floating-point format does not have exactly multiple eigenvalues. It is necessary to discuss the behavior of Algorithm 1 for A with some nearly multiple eigenvalues $\lambda_i \approx \lambda_j$ such that $|\widetilde{\lambda}_i - \widetilde{\lambda}_j| \le \omega$ in line 7. We basically discuss the behavior in real arithmetic. The effect of the rounding error is briefly explained in Remark 2 at the end of this section.

For simplicity, we assume $\widetilde{\lambda}_1 \le \widetilde{\lambda}_2 \le \cdots \le \widetilde{\lambda}_n$. In the following analysis, define $A_\omega := X D_\omega X^{\mathrm{T}}$ where $D_\omega = \mathrm{diag}(\lambda_i^{(\omega)})$ with

$$\lambda_1^{(\omega)} = \lambda_1, \quad \lambda_i^{(\omega)} = \begin{cases} \lambda_{i-1}^{(\omega)} & \text{if } \widetilde{\lambda}_i - \widetilde{\lambda}_{i-1} \le \omega \\ \lambda_i & \text{otherwise} \end{cases} \quad \text{for } 2 \le i \le n, \tag{14}$$

which means that the clustered eigenvalues of A_ω are all multiple in each cluster. Then, A_ω is a perturbed matrix such that

$$A_\omega = A + \Delta_\omega, \quad \|\Delta_\omega\| = \|D - D_\omega\| = \max_{1 \le i \le n} |\lambda_i - \lambda_i^{(\omega)}|.$$

Throughout this section, we assume that

$$|\widetilde{\lambda}_i - \widetilde{\lambda}_j| \le \omega \quad \text{for } (i, j) \text{ such that } \lambda_i^{(\omega)} = \lambda_j^{(\omega)}. \tag{15}$$

Importantly, although each individual eigenvector associated with the nearly multiple eigenvalues is very sensitive to perturbations, the subspace spanned by such eigenvectors is not sensitive. Thus, $\widehat{X}$ computed by a backward stable algorithm is sufficiently close to an eigenvector matrix of A_ω. Below, we show that Algorithm 1 computes $\widetilde{E}$ that approximates an exact eigenvector matrix Y_ω defined in the same manner as Y in Sect. 2. Note that Y_ω is the eigenvector matrix of the above A_ω close to A, where A_ω has exactly multiple eigenvalues.

Recall that the submatrices of $\widehat{X}^{-1} Y_\omega$ corresponding to the multiple eigenvalues of A_ω are symmetric and positive definite. Then, we see that Algorithm 1 computes an approximation of Y_ω as follows. Define R and S_ω as

$$R := I - \widehat{X}^{\mathrm{T}} \widehat{X}, \quad S_\omega := \widehat{X}^{\mathrm{T}} A_\omega \widehat{X},$$

corresponding to A_ω. We see S_ω is considered a perturbed matrix of $S := \widehat{X}^{\mathrm{T}} A \widehat{X}$. Note that, in Algorithm 1, $\widetilde{E}$ is computed with R and S. Here, we introduce an ideal matrix $\widetilde{E}_\omega$ computed with R and S_ω, where $\widetilde{E}_\omega$ is quadratically convergent to F_ω. In the following, we estimate $\widetilde{E}_\omega - \widetilde{E}$ due to the above perturbation. To this end, we estimate each element of $S_\omega - S$ as in the following lemma.

Lemma 1 *Let A be a real symmetric $n \times n$ matrix with eigenvalues λ_i, $i = 1, 2, \ldots, n$, and a corresponding orthogonal eigenvector matrix X. In Algorithm 1, for a given nonsingular $\widehat{X} \in \mathbb{R}^{n \times n}$, define $A_\omega := X D_\omega X^{\mathrm{T}}$ where $D_\omega = \mathrm{diag}(\lambda_i^{(\omega)})$ as in (14),*

and

$$\delta_\omega := \max_{1\le i\le n} |\lambda_i - \lambda_i^{(\omega)}|. \tag{16}$$

In addition, define $S_\omega := \widehat{X}^{\mathrm{T}} A_\omega \widehat{X}$. *Then, we have*

$$|s_{ij} - s_{ij}^{(\omega)}| \le \begin{cases} \delta_\omega(1 + 2\epsilon_\omega + \chi(\epsilon_\omega)\epsilon_\omega^2) & \text{if } \lambda_i^{(\omega)} = \lambda_j^{(\omega)} \quad (17) \\ \epsilon_\omega \delta_\omega(2 + \chi(\epsilon_\omega)\epsilon_\omega) & \text{otherwise} \quad (18) \end{cases}$$

for all (i, j), *where*

$$\epsilon_\omega := \|F_\omega\|, \quad \chi(\epsilon_\omega) := \frac{(3 - 2\epsilon_\omega)}{(1 - \epsilon_\omega)^2}.$$

Proof Define Q such that $X = Y_\omega Q$. Then, Q is a block diagonal matrix. More precisely, for $Q = (q_{ij})$, we have

$$q_{ij} = 0 \text{ for } (i, j) \text{ such that } \lambda_i^{(\omega)} \ne \lambda_j^{(\omega)}.$$

It is easy to see that

$$S - S_\omega = \widehat{X}^{\mathrm{T}}(A - A_\omega)\widehat{X} = \widehat{X}^{\mathrm{T}} Y_\omega Q(D - D_\omega)Q^{\mathrm{T}} Y_\omega^{\mathrm{T}} \widehat{X}.$$

Let $D_Q := Q(D - D_\omega)Q^{\mathrm{T}}$. In a similar way to (8), we have

$$D_Q - D_Q F_\omega - F_\omega^{\mathrm{T}} D_Q = (S - S_\omega) + \Delta(\delta_\omega),$$

where

$$\|D_Q\| = \delta_\omega, \quad \|\Delta(\delta_\omega)\| \le \chi(\epsilon_\omega)\epsilon_\omega^2 \delta_\omega.$$

Then (17) follows. Moreover, noting D_Q is a block diagonal matrix, we obtain (18). □

For the perturbation analysis of $\widetilde{E}$, the next lemma is crucial.

Lemma 2 *Let* A *be a real symmetric* $n \times n$ *matrix with eigenvalues* λ_i, $i = 1, 2, \ldots, n$, *and a corresponding orthogonal eigenvector matrix* X. *In Algorithm 1, for a given nonsingular* $\widehat{X} \in \mathbb{R}^{n\times n}$, *define* $A_\omega := X D_\omega X^{\mathrm{T}}$ *where* $D_\omega = \mathrm{diag}(\lambda_i^{(\omega)})$ *as in* (14). *Assume that* (15) *is satisfied. Define* $R = (r_{ij})$ *and* $S_\omega = (s_{ij}^{(\omega)})$ *such that* $R := I - \widehat{X}^{\mathrm{T}}\widehat{X}$ *and* $S_\omega := \widehat{X}^{\mathrm{T}} A_\omega \widehat{X}$. *Suppose positive numbers* ω_1 *and* ω_2 *satisfy*

$$|s_{ij} - s_{ij}^{(\omega)}| \le \begin{cases} \omega_1 & \text{if } \lambda_i^{(\omega)} = \lambda_j^{(\omega)} \\ \omega_2 & \text{otherwise} \end{cases} \quad \text{for all } (i, j).$$

We assume that, for all (i, j) *in line 7 of Algorithm 1, the formulas of* $\widetilde{e}_{ij}^{(\omega)}$ *are the same as those of* $\widetilde{e}_{ij}$*, i.e.,*

$$\widetilde{e}_{ij}^{(\omega)} = \begin{cases} \dfrac{s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}}{\widetilde{\lambda}_j^{(\omega)} - \widetilde{\lambda}_i^{(\omega)}} & \text{if } |\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega \\ r_{ij}/2 & \text{otherwise} \end{cases} \quad \text{for all } (i, j),$$

where $\widetilde{\lambda}_i^{(\omega)} = s_{ii}^{(\omega)}/(1 - r_{ii})$ *for* $i = 1, 2, \ldots, n$*, as in line 4. Moreover, let*

$$c := \max_{1 \le i \le n} \frac{1}{1 - r_{ii}}. \tag{19}$$

Then, for (i, j) *such that* $|\widetilde{\lambda}_i - \widetilde{\lambda}_j| \le \omega$*, we have*

$$\widetilde{e}_{ij}^{(\omega)} = \widetilde{e}_{ij}. \tag{20}$$

Moreover, for (i, j) *such that* $|\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega$*, we have*

$$|\widetilde{e}_{ij}^{(\omega)} - \widetilde{e}_{ij}| \le \frac{2c\,\omega_1}{|\widetilde{\lambda}_j - \widetilde{\lambda}_i| - 2c\,\omega_1} \frac{|s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}|}{|\widetilde{\lambda}_j - \widetilde{\lambda}_i|} + \frac{\omega_2 + c\,\omega_1 |r_{ij}|}{|\widetilde{\lambda}_j - \widetilde{\lambda}_i|}. \tag{21}$$

Proof For (i, j) such that $|\widetilde{\lambda}_i - \widetilde{\lambda}_j| \le \omega$, since we see

$$\widetilde{e}_{ij}^{(\omega)} = \frac{r_{ij}}{2} = \widetilde{e}_{ij},$$

we have (20). Next, for $\widetilde{\lambda}_i^{(\omega)}$, $i = 1, \ldots, n$, we have

$$\widetilde{\lambda}_i^{(\omega)} = \frac{s_{ii}^{(\omega)}}{1 - r_{ii}}, \quad |\widetilde{\lambda}_i^{(\omega)} - \widetilde{\lambda}_i| \le \frac{\omega_1}{1 - r_{ii}} \quad \text{for } i = 1, \ldots, n. \tag{22}$$

Thus, from (19), we have

$$|\widetilde{\lambda}_i^{(\omega)} - \widetilde{\lambda}_i| \le c\,\omega_1 \quad \text{for } i = 1, \ldots, n. \tag{23}$$

For (i, j) such that $|\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega$, since we see

$$\widetilde{e}_{ij}^{(\omega)} = \frac{s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}}{\widetilde{\lambda}_j^{(\omega)} - \widetilde{\lambda}_i^{(\omega)}},$$

we evaluate the errors based on the following inequalities:

$$|\widetilde{e}_{ij}^{(\omega)} - \widetilde{e}_{ij}| \le \left| \widetilde{e}_{ij}^{(\omega)} - \frac{s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}}{\widetilde{\lambda}_j - \widetilde{\lambda}_i} \right| + \left| \widetilde{e}_{ij} - \frac{s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}}{\widetilde{\lambda}_j - \widetilde{\lambda}_i} \right|.$$

In the right-hand side, using (22) and (23), we see

$$\left| \widetilde{e}_{ij}^{(\omega)} - \frac{s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}}{\widetilde{\lambda}_j - \widetilde{\lambda}_i} \right| \le \frac{2c\,\omega_1}{|\widetilde{\lambda}_j - \widetilde{\lambda}_i| - 2c\,\omega_1} \frac{|s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}|}{|\widetilde{\lambda}_j - \widetilde{\lambda}_i|},$$
$$\left| \widetilde{e}_{ij} - \frac{s_{ij}^{(\omega)} + \widetilde{\lambda}_j^{(\omega)} r_{ij}}{\widetilde{\lambda}_j - \widetilde{\lambda}_i} \right| \le \frac{\omega_2 + c\,\omega_1 |r_{ij}|}{|\widetilde{\lambda}_j - \widetilde{\lambda}_i|}.$$

Therefore, we obtain (21). □

In the following, we estimate ω_1 and ω_2 in Lemma 2. In the right-hand sides of (17) and (18) in Lemma 1, we see

$$\begin{cases} \delta_\omega(1 + 2\epsilon_\omega + \chi(\epsilon_\omega)\epsilon_\omega^2) \to \delta_\omega \\ \epsilon_\omega \delta_\omega(2 + \chi(\epsilon_\omega)\epsilon_\omega) \to 2\epsilon_\omega \delta_\omega \end{cases} \quad \text{as } \epsilon_\omega \to 0.$$

If D, F, S in (8) are replaced with $D_\omega, F_\omega, S_\omega$, respectively, we see $s_{ij}^{(\omega)} = \mathcal{O}(\|A_\omega\|\epsilon_\omega)$ $(i \ne j)$ as $\epsilon_\omega \to 0$. In addition, $r_{ij} = \mathcal{O}(\epsilon_\omega)$ $(i \ne j)$ as $\epsilon_\omega \to 0$ from (7). Hence, letting $\omega_1 = \delta_\omega$ and $\omega_2 = 2\epsilon_\omega \delta_\omega$ in Lemma 2, we obtain

$$\|\widetilde{E}_\omega - \widetilde{E}\| = \mathcal{O}\left(\frac{\epsilon_\omega \delta_\omega}{\min_{|\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega} |\lambda_i - \lambda_j|} \right) \quad \text{as } \epsilon_\omega \to 0.$$

Since we suppose $\delta_\omega = \mathcal{O}(\|A\|\epsilon_\omega)$ in the situation where $\widehat{X}$ is computed by a backward stable algorithm, we have

$$\|\widetilde{E}_\omega - \widetilde{E}\| = \mathcal{O}\left(\frac{\|A\|\epsilon_\omega^2}{\min_{|\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega} |\lambda_i - \lambda_j|} \right) \quad \text{as } \epsilon_\omega \to 0.$$

Therefore, $\widetilde{E}$ is sufficiently close to $\widetilde{E}_\omega$ and F_ω under the above mild assumptions. Although Y_ω can be very far from any eigenvector matrix of A, the subspace spanned by the columns of Y_ω corresponding to the clustered eigenvalues adequately approximates that by the exact eigenvectors whenever $\|A_\omega - A\|$ is sufficiently small. In the following, we derive an algorithm for clustered eigenvalues using such an important feature.

Remark 2 In this section, we proved that $\widetilde{E}$ is sufficiently close to F_ω under the mild assumptions. In Sect. 3, the effect of rounding errors on $\widetilde{E}$ is evaluated as in (10), i.e., $\Delta_E := \widehat{E} - \widetilde{E}$ is sufficiently small, where $\widehat{E}$ is computed in finite precision arithmetic.

The rounding error analysis is not caused by the perturbation analysis to F_ω in this section. Thus, it is easy to see that $\|\widehat{E} - F_\omega\| \le \|\widehat{E} - F_\omega\| + \|\widehat{E} - F_\omega\|$ simply holds for the individual estimation for each error $\|\widehat{E} - F_\omega\|$ and $\|\widehat{E} - F_\omega\|$ respectively, and hence, the computed $\widehat{E}$ is sufficiently close to F_ω corresponding to A_ω. □

5 Proposed algorithm for nearly multiple eigenvalues

On the basis of the basic algorithm (Algorithm 1), we propose a practical version of an algorithm for improving the accuracy of computed eigenvectors of symmetric matrices that can also deal with nearly multiple eigenvalues.

Recall that, in Algorithm 1, we choose $\widetilde{e}_{ij}$ for all (i, j) as

$$\widetilde{e}_{ij} = \begin{cases} \dfrac{s_{ij} + \widetilde{\lambda}_j r_{ij}}{\widetilde{\lambda}_j - \widetilde{\lambda}_i} & \text{if } |\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega \quad (24) \\ r_{ij}/2 & \text{otherwise} \quad (25) \end{cases}$$

where ω is defined in line 6 of the algorithm.

5.1 Observation

First, we show the drawback of Algorithm 1 concerning clustered eigenvalues. For this purpose, we take

$$A = \begin{bmatrix} 1+\varepsilon & 1 & 1+\varepsilon \\ 1 & 1 & -1 \\ 1+\varepsilon & -1 & 1+\varepsilon \end{bmatrix}, \quad \begin{cases} \lambda_1 = -1 \\ \lambda_2 = 2 \\ \lambda_3 = 2 + 2\varepsilon \end{cases} \text{ for any } \varepsilon > 0 \quad (26)$$

as an example, where λ_2 and λ_3 are nearly double eigenvalues for small ε. We set $\varepsilon = 2^{-50} \approx 10^{-15}$ and adopt the MATLAB built-in function eig in IEEE 754 binary64 arithmetic to obtain $X^{(0)} := \widehat{X}$. Then, we apply Algorithm 1 iteratively to A and $X^{(\nu)}$ beginning from $\nu = 0$. To check the accuracy of $X^{(\nu)}$ with respect to orthogonality and diagonality, we display $R^{(\nu)} := I - (X^{(\nu)})^{\mathrm{T}} X^{(\nu)}$ and $S^{(\nu)} := (X^{(\nu)})^{\mathrm{T}} A X^{(\nu)}$.

For $X^{(0)}$ obtained by eig, we obtain the following results.

$$R^{(0)} \approx \begin{bmatrix} \texttt{-2.7e-16} & \texttt{-1.3e-16} & \texttt{-6.8e-17} \\ \texttt{-1.3e-16} & \texttt{1.4e-16} & \texttt{-5.0e-17} \\ \texttt{-6.8e-17} & \texttt{-5.0e-17} & \texttt{-2.2e-16} \end{bmatrix}, \quad S^{(0)} \approx \begin{bmatrix} \texttt{-1.0e+00} & \texttt{-1.3e-16} & \texttt{-6.8e-17} \\ \texttt{-1.3e-16} & \texttt{2.0e+00} & \texttt{1.7e-17} \\ \texttt{-6.8e-17} & \texttt{1.7e-17} & \texttt{2.0e+00} \end{bmatrix}$$

The following shows the results of two iterations of Algorithm 1 in real arithmetic.

$$R^{(1)} \approx \begin{bmatrix} \texttt{5.4e-32} & \texttt{9.1e-33} & \texttt{2.6e-32} \\ \texttt{9.1e-33} & \texttt{1.2e-33} & \texttt{4.4e-33} \\ \texttt{2.6e-32} & \texttt{4.4e-33} & \texttt{4.0e-32} \end{bmatrix}, \quad S^{(1)} \approx \left[\begin{array}{c|cc} \texttt{-1.0e+00} & \texttt{9.1e-33} & \texttt{2.6e-32} \\ \hline \texttt{9.1e-33} & \texttt{2.0e+00} & \texttt{-8.3e-17} \\ \texttt{2.6e-32} & \texttt{-8.3e-17} & \texttt{2.0e+00} \end{array}\right]$$

$$R^{(2)} \approx \left[\begin{array}{c|cc} \texttt{2.2e-63} & \texttt{1.2e-33} & \texttt{-4.3e-34} \\ \hline \texttt{1.2e-33} & \texttt{-2.2e-03} & \texttt{9.1e-34} \\ \texttt{-4.3e-34} & \texttt{9.1e-34} & \texttt{-2.2e-03} \end{array}\right], \quad S^{(2)} \approx \left[\begin{array}{c|cc} \texttt{-1.0e+00} & \texttt{1.2e-33} & \texttt{-4.3e-34} \\ \hline \texttt{1.2e-33} & \texttt{2.0e+00} & \texttt{1.8e-19} \\ \texttt{-4.3e-34} & \texttt{1.8e-19} & \texttt{2.0e+00} \end{array}\right]$$

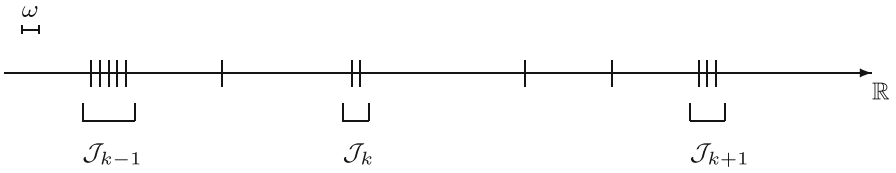

Fig. 1 Relationship between $\widetilde{\lambda}_i$ and $\mathcal{J}_k$ (short vertical lines denote $\widetilde{\lambda}_i$)

In the first iteration, $|\widetilde{\lambda}_2 - \widetilde{\lambda}_3| \approx 1.77 \cdot 10^{-15}$ and $\omega \approx 2.17 \cdot 10^{-15}$, so that $|\widetilde{\lambda}_2 - \widetilde{\lambda}_3| < \omega$ and Algorithm 1 regards $\widetilde{\lambda}_2$ and $\widetilde{\lambda}_3$ as clustered eigenvalues. Then, the diagonality corresponding to λ_2 and λ_3 is not improved due to the choice (24), while the orthogonality of $X^{(1)}$ is refined due to the choice (25). In the second iteration, $|\widetilde{\lambda}_2 - \widetilde{\lambda}_3| \approx 1.77 \cdot 10^{-15}$ and $\omega \approx 1.66 \cdot 10^{-16}$, so that $|\widetilde{\lambda}_2 - \widetilde{\lambda}_3| > \omega$ and Algorithm 1 regards $\widetilde{\lambda}_2$ and $\widetilde{\lambda}_3$ as separated eigenvalues. However, $\|E\| \approx 4.69 \cdot 10^{-2} > 1/100$, i.e., the assumption (3) in Theorem 1 is not satisfied. As a result, the orthogonality of $X^{(2)}$ corresponding to λ_2 and λ_3 is badly broken, and the refinement of the diagonality stagnates with respect to the nearly double eigenvalues λ_2 and λ_3.

In the following, we overcome such a problem for general symmetric matrices.

5.2 Outline of the proposed algorithm

As mentioned in Sect. 1, the $\sin\theta$ theorem by Davis–Kahan suggests that backward stable algorithms can provide a sufficiently accurate initial guess of a subspace spanned by eigenvectors associated with clustered eigenvalues for each cluster. We explain how to refine approximate eigenvectors by extracting them from the subspace correctly.

Suppose that Algorithm 1 is applied to $A = A^{\mathrm{T}} \in \mathbb{R}^{n \times n}$ and its approximate eigenvector matrix $\widehat{X} \in \mathbb{R}^{n \times n}$. Then, we obtain X', $\widetilde{\lambda}$, and ω where $X' \in \mathbb{R}^{n \times n}$ is a refined approximate eigenvector matrix, $\widetilde{\lambda}_i$, $i = 1, 2, \ldots, n$, are approximate eigenvalues, and $\omega \in \mathbb{R}$ is the criterion that determines whether $\widetilde{\lambda}_i$ are clustered. Using $\widetilde{\lambda}$ and ω, we can easily obtain the index sets $\mathcal{J}_k$, $k = 1, 2, \ldots, n_{\mathcal{J}}$, for the clusters $\{\widetilde{\lambda}_i\}_{i \in \mathcal{J}_k}$ of the approximate eigenvalues satisfying all the following conditions (see also Fig. 1).

$$\begin{cases} \text{(a) } \mathcal{J}_k \subseteq \{1, 2, \ldots, n\} \text{ with } n_k := |\mathcal{J}_k| \geq 2 \\ \text{(b) } \min\limits_{j \in \mathcal{J}_k \setminus \{i\}} |\widetilde{\lambda}_i - \widetilde{\lambda}_j| \leq \omega, \ \forall i \in \mathcal{J}_k \\ \text{(c) } |\widetilde{\lambda}_i - \widetilde{\lambda}_j| > \omega, \ \forall i \in \mathcal{J}_k, \ \forall j \in \{1, 2, \ldots, n\} \setminus \mathcal{J}_k \end{cases}. \tag{27}$$

Now the problem is how to refine $X'(:, \mathcal{J}_k) \in \mathbb{R}^{n \times n_k}$, which denotes the matrix comprising approximate eigenvectors corresponding to the clustered approximate eigenvalues $\{\widetilde{\lambda}_i\}_{i \in \mathcal{J}_k}$.

From the observation about the numerical results in the previous section, we develop the following procedure for the refinement.

1. Find clusters of approximate eigenvalues of A and obtain the index sets $\mathcal{J}_k$, $k = 1, 2, \ldots, n_{\mathcal{J}}$ for those clusters.
2. Define $V_k := X'(:, \mathcal{J}_k) \in \mathbb{R}^{n \times n_k}$ where $n_k := |\mathcal{J}_k|$.
3. Compute $T_k = V_k^{\mathrm{T}}(A - \mu_k I)V_k$ where $\mu_k := (\min_{i \in \mathcal{J}_k} \widetilde{\lambda}_i + \max_{i \in \mathcal{J}_k} \widetilde{\lambda}_i)/2$.
4. Perform the following procedure for each $T_k \in \mathbb{R}^{n_k \times n_k}$.

   (i) Compute an eigenvector matrix W_k of T_k.
   (ii) Update $X'(:, \mathcal{J}_k) \in \mathbb{R}^{n \times n_k}$ by $V_k W_k$.

This procedure is interpreted as follows. We first apply an approximate similarity transformation to A using a refined eigenvector matrix X', such as $S' := (X')^{\mathrm{T}} A X'$. Then, we divide the problem for $S' \in \mathbb{R}^{n \times n}$ into subproblems for $S'_k \in \mathbb{R}^{n_k \times n_k}$, $k = 1, 2, \ldots, n_{\mathcal{J}}$, corresponding to the clusters. We then apply a diagonal shift to S'_k, such as $T_k := S'_k - \mu_k I$ to relatively separate the clustered eigenvalues around μ_k. Rather than using these to obtain T_k, we perform steps 2 and 3 in view of computational efficiency and accuracy. Finally, we update the columns of X' corresponding to $\mathcal{J}_k$ using an eigenvector matrix W_k of T_k by $V_k W_k$.

5.3 Proposed algorithm

Here, we present a practical version of a refinement algorithm for eigenvalue decomposition of a real symmetric matrix A, which can also be applied to the case where A has clustered eigenvalues.

In Algorithm 2, the function $\mathsf{fl}(C)$ rounds an input matrix $C \in \mathbb{R}^{n \times n}$ to a matrix $T \in \mathbb{F}^{n \times n}$, where $\mathbb{F}$ is a set of floating-point numbers in ordinary precision, such as the IEEE 754 binary64 format. Here, "round-to-nearest" rounding is not required; however, some faithful rounding, such as chopping, is desirable. Moreover, the function $\mathsf{eig}(T)$ is similar to the MATLAB function, which computes all approximate eigenvectors of an input matrix $T \in \mathbb{F}^{n \times n}$ in working precision arithmetic. This is expected to adopt some backward stable algorithm as implemented in the LAPACK routine xSYEV [2]. From lines 13–17 in Algorithm 2, we aim to obtain sufficiently accurate approximate eigenvectors $X'(:, \mathcal{J}_k)$ of A, where the columns of $X'(:, \mathcal{J}_k)$ correspond to $\mathcal{J}_k$. For this purpose, we iteratively apply Algorithm 1 (RefSyEv) to $A - \mu_k I$ and $V_k^{(\nu)}$ until $V_k^{(\nu)}$ for some ν becomes as accurate as other eigenvectors associated with well-separated eigenvalues. Note that the spectral norms $\|\widetilde{E}\|_2$ and $\|\widetilde{E}_k\|_2$ can be replaced by the Frobenius norms $\|\widetilde{E}\|_{\mathrm{F}}$ and $\|\widetilde{E}_k\|_{\mathrm{F}}$.

For the example (26), we apply Algorithm 2 (RefSyEvCL) to A and the same initial guess $X^{(0)}$ as before. The results of two iterations are as follows.

$$R^{(1)} \approx \begin{bmatrix} 5.4e\text{-}32 & -1.0e\text{-}32 & 2.5e\text{-}32 \\ -1.0e\text{-}32 & 1.4e\text{-}33 & 2.9e\text{-}49 \\ 2.5e\text{-}32 & 2.9e\text{-}49 & 1.4e\text{-}33 \end{bmatrix}, \quad S^{(1)} \approx \begin{bmatrix} -1.0e{+}00 & -1.0e\text{-}32 & 2.5e\text{-}32 \\ -1.0e\text{-}32 & 2.0e{+}00 & -1.3e\text{-}48 \\ 2.5e\text{-}32 & -1.3e\text{-}48 & 2.0e{+}00 \end{bmatrix}$$

$$R^{(2)} \approx \begin{bmatrix} 2.2e\text{-}63 & -5.5e\text{-}64 & 1.4e\text{-}63 \\ -5.5e\text{-}64 & 1.1e\text{-}64 & -2.6e\text{-}64 \\ 1.4e\text{-}63 & -2.6e\text{-}64 & 6.5e\text{-}64 \end{bmatrix}, \quad S^{(2)} \approx \begin{bmatrix} -1.0e{+}00 & -5.5e\text{-}64 & 1.4e\text{-}63 \\ -5.5e\text{-}64 & 2.0e{+}00 & -2.6e\text{-}64 \\ 1.4e\text{-}63 & -2.6e\text{-}64 & 2.0e{+}00 \end{bmatrix}$$

Algorithm 2 RefSyEvCL: Refinement of approximate eigenvectors of a real symmetric matrix with clustered eigenvalues. Higher-precision arithmetic is required for all the computations except line 11 and the computations of $\|\cdot\|_2$.

Input: $A, \widehat{X} \in \mathbb{R}^{n\times n}$ with $A = A^{\mathrm{T}}$
Output: $X' \in \mathbb{R}^{n\times n}$
1: **function** $X' \leftarrow$ RefSyEvCL$(A, \widehat{X})$
2: $[X', \widetilde{D}, \widetilde{E}, \omega] \leftarrow$ RefSyEv$(A, \widehat{X})$ ▷ Apply Algorithm 1 to A and $\widehat{X}$.
3: **if** $\|\widetilde{E}\|_2 \geq 1$, **return**, **end if** ▷ Improvement cannot be expected.
4: Determine the index sets $\mathcal{J}_k, k = 1, \ldots, n_{\mathcal{J}}$, as in (27) for eigenvalue clusters using $\widetilde{D} = \mathrm{diag}(\widetilde{\lambda}_i)$ and ω. ▷ $n_{\mathcal{J}}$: The number of clusters.
5: **for** $k \leftarrow 1, 2, \ldots, n_{\mathcal{J}}$ **do** ▷ Refine eigenvectors for each cluster.
6: $V_k \leftarrow X'(:, \mathcal{J}_k)$ ▷ Pick out $V_k \in \mathbb{R}^{n\times n_k}$ where $n_k := |\mathcal{J}_k|$.
7: $\mu_k \leftarrow (\min_{i\in\mathcal{J}_k} \widetilde{\lambda}_i + \max_{i\in\mathcal{J}_k} \widetilde{\lambda}_i)/2$ ▷ Determine the shift constant μ_k.
8: $A_k \leftarrow A - \mu_k I$ ▷ Shift A for separating clustered eigenvalues.
9: $C_k \leftarrow V_k^{\mathrm{T}} A_k V_k$ ▷ Compute $C_k \in \mathbb{R}^{n_k\times n_k}$.
10: $T_k \leftarrow \mathrm{fl}(C_k)$ ▷ Conversion from higher-precision to ordinary precision.
11: $[W_k, \sim] \leftarrow \mathrm{eig}(T_k)$ ▷ Compute eigenvectors of C_k in ordinary precision.
12: $\nu \leftarrow 1$; $V_k^{(1)} \leftarrow V_k \cdot W_k$
13: **repeat**
14: $[V_k^{(\nu+1)}, \sim, \widetilde{E}_k] \leftarrow$ RefSyEv$(A_k, V_k^{(\nu)})$ ▷ Apply Alg. 1 to A_k and $V_k^{(\nu)}$.
15: **if** $\|\widetilde{E}_k\|_2 \geq 1$, **return**, **end if** ▷ Improvement cannot be expected.
16: $\nu \leftarrow \nu + 1$
17: **until** $\|\widetilde{E}_k\|_2 \leq \|\widetilde{E}\|_2$
18: $X'(:, \mathcal{J}_k) \leftarrow V_k^{(\nu)}$ ▷ Update the columns of X' corresponding to $\mathcal{J}_k$.
19: **end for**
20: **end function**

Thus, Algorithm 2 works well for this example, i.e., the approximate eigenvectors corresponding to the nearly double eigenvalues λ_2 and λ_3 are improved in terms of both orthogonality and diagonality.

Remark 3 For a generalized symmetric definite eigenvalue problem $Ax = \lambda Bx$ where A and B are real symmetric with B being positive definite, we can modify the algorithms as follows.

- In Algorithm 1 called at line 2 in Algorithm 2, replace $R \leftarrow I - \widehat{X}^{\mathrm{T}}\widehat{X}$ with $R \leftarrow I - \widehat{X}^{\mathrm{T}}B\widehat{X}$.
- Replace $A_k \leftarrow A - \mu_k I$ with $A_k \leftarrow A - \mu_k B$ in line 8 of Algorithm 2.

Note that B does not appear in Algorithm 1 called at line 14 in Algorithm 2. □

6 Numerical results

We present numerical results to demonstrate the effectiveness of the proposed algorithm (Algorithm 2: RefSyEvCL). All numerical experiments discussed in this section were conducted using MATLAB R2016b on our workstation with two CPUs (3.0 GHz Intel Xeon E5-2687W v4 (12 cores)) and 1 TB of main memory, unless otherwise specified. Let $\mathbf{u}$ denote the relative rounding error unit ($\mathbf{u} = 2^{-24}$ for IEEE

binary32 and $\mathbf{u} = 2^{-53}$ for binary64). To realize multiple-precision arithmetic, we adopt Advanpix Multiprecision Computing Toolbox version 4.2.3 [1], which utilizes well-known, fast, and reliable multiple-precision arithmetic libraries including GMP and MPFR. We also use the multiple-precision arithmetic with sufficiently long precision to simulate real arithmetic. In all cases, we use the MATLAB function norm for computing the spectral norms $\|R\|$ and $\|S-\widetilde{D}\|$ in Algorithm 1 in binary64 arithmetic, and we approximate $\|A\|$ by $\max(|\widetilde{\lambda}_1|, |\widetilde{\lambda}_n|)$. We discuss numerical experiments for some dozens of seeds for the random number generator, and all results are similar to those provided in this section. Therefore, we adopt the default seed as a typical example using the MATLAB command `rng('default')` to ensure reproducibility of problems.

6.1 Convergence property

Here, we confirm the convergence property of the proposed algorithm for various eigenvalue distributions.

6.1.1 Various eigenvalue distributions

In the same way as the previous paper [17], we again generate real symmetric and positive definite matrices using the MATLAB function randsvd from Higham's test matrices [11] by the following MATLAB command.

```
>> A = gallery('randsvd',n,-cnd,mode);
```

The eigenvalue distribution and condition number of A can be controlled by the input arguments $\texttt{mode} \in \{1, 2, 3, 4, 5\}$ and $\texttt{cnd} =: \alpha \geq 1$, as follows:

1. one large: $\lambda_1 \approx 1, \lambda_i \approx \alpha^{-1}, i = 2, \ldots, n$
2. one small: $\lambda_n \approx \alpha^{-1}, \lambda_i \approx 1, i = 1, \ldots, n-1$
3. geometrically distributed: $\lambda_i \approx \alpha^{-(i-1)/(n-1)}, i = 1, \ldots, n$
4. arithmetically distributed: $\lambda_i \approx 1 - (1-\alpha^{-1})(i-1)/(n-1), i = 1, \ldots, n$
5. random with uniformly distributed logarithm: $\lambda_i \approx \alpha^{-r(i)}, i = 1, \ldots, n$, where $r(i)$ are pseudo-random values drawn from the standard uniform distribution on $(0, 1)$.

Here, $\kappa(A) \approx \texttt{cnd}$ for $\texttt{cnd} < \mathbf{u}^{-1} \approx 10^{16}$. As shown in [17], for $\texttt{mode} \in \{1, 2\}$, there is a cluster of nearly multiple eigenvalues, so that Algorithm 1 (RefSyEv) does not work effectively.

As in [17], we set $n = 10$ and $\texttt{cnd} = 10^8$ to generate moderately ill-conditioned problems in binary64 and consider the computed results obtained using multiple-precision arithmetic with sufficiently long precision as the exact eigenvalues $\lambda_i, i = 1, 2, \ldots, n$. We compute $X^{(0)}$ as an initial approximate eigenvector matrix using the MATLAB function eig in binary64 arithmetic.

In the previous paper [17], we observed the quadratic convergence of Algorithm 1 in the case of $\texttt{mode} \in \{3, 4, 5\}$, while Algorithm 1 failed to improve the accuracy of the initial approximate eigenvectors in the case of $\texttt{mode} \in \{1, 2\}$, since the test matrices for $\texttt{mode} \in \{1, 2\}$ have nearly multiple eigenvalues. To confirm the behavior

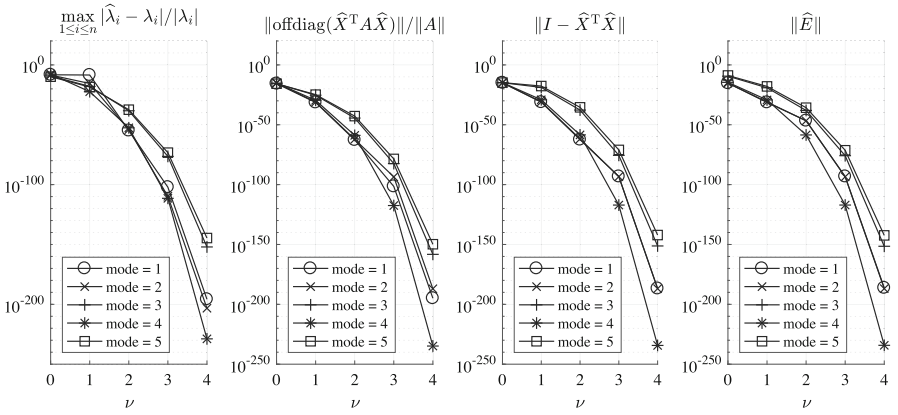

Fig. 2 Results of iterative refinement by Algorithm 2 (RefSyEvCL) in real arithmetic for symmetric and positive definite matrices generated by randsvd with $n = 10$ and $\kappa(A) \approx 10^8$

of Algorithm 2, we apply Algorithm 2 to the same examples. The results are shown in Fig. 2, which provides $\max_{1\le i\le n} |\widehat{\lambda}_i - \lambda_i|/|\lambda_i|$ as the maximum relative error of the computed eigenvalues $\widehat{\lambda}_i$, $\|\mathrm{offdiag}(\widehat{X}^{\mathrm{T}} A \widehat{X})\|/\|A\|$ as the diagonality of $\widehat{X}^{\mathrm{T}} A \widehat{X}$, $\|I - \widehat{X}^{\mathrm{T}}\widehat{X}\|$ as the orthogonality of a computed eigenvector matrix $\widehat{X}$, and $\|\widehat{E}\|$ where $\widehat{E}$ is a computed result of E in Algorithm 1. Here, $\mathrm{offdiag}(\cdot)$ denotes the off-diagonal part of a given matrix. The horizontal axis shows the number of iterations ν of Algorithm 2. As can be seen from the results, Algorithm 2 works very well even in the case of `mode` $\in \{1, 2\}$.

6.1.2 Clustered eigenvalues

As an example of clustered eigenvalues, we show the results for the Wilkinson matrix [23], which is symmetric and tridiagonal with pairs of nearly equal eigenvalues. The Wilkinson matrix $W_n = (w_{ij}) \in \mathbb{R}^{n\times n}$ consists of diagonal entries $w_{ii} := \frac{|n-2i+1|}{2}$, $i = 1, 2, \ldots, n$, and super- and sub-diagonal entries being all ones. We apply Algorithm 2 to the Wilkinson matrix with $n = 21$. The results are displayed in Fig. 3. As can be seen, Algorithm 2 works well.

Next, we show the convergence behavior of Algorithm 2 with limited computational precision for larger matrices with various β, which denotes the reciprocal of the minimum gap between the eigenvalues normalized by $\|A\|$ as defined in (10). If β is too large such as $\beta^2\mathbf{u} \ge 1$ as mentioned at the end of Sect. 3, we cannot expect to improve approximate eigenvectors by Algorithm 1. We generate test matrices as follows. Set $k \in \mathbb{N}$ with $k \le n-2$. Let $A = QDQ^{\mathrm{T}}$, where Q is an orthogonal matrix and D is a diagonal matrix where

$$d_{ii} = \begin{cases} 1 - (i-1)\beta^{-1} & \text{for } i = 1, 2, \ldots, k \\ -1 + \dfrac{n-i}{n-k-1} 2^{-1} & \text{for } i = k+1, \ldots, n \end{cases}.$$

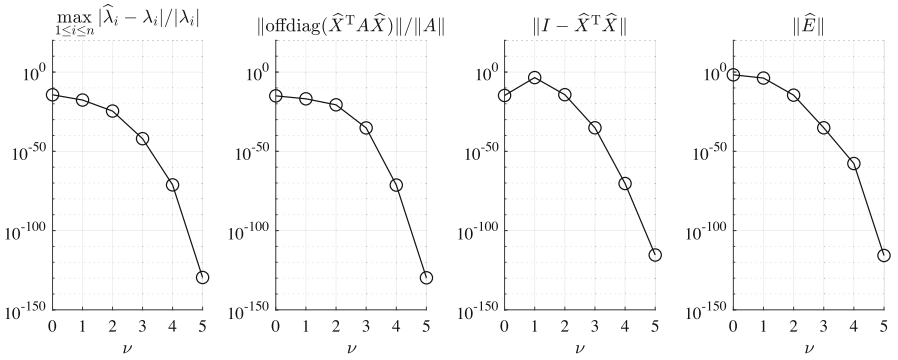

Fig. 3 Results of iterative refinement by Algorithm 2 (RefSyEvCL) in real arithmetic for the Wilkinson matrix with $n = 21$

Then, k eigenvalues are clustered close to 1 with the gap β^{-1}, and $n - k$ eigenvalues are distributed equally in $[-1, -\frac{1}{2}]$. We compute $A \approx \widetilde{Q}\widetilde{D}\widetilde{Q}^{\mathrm{T}}$ in IEEE 754 binary64 arithmetic, where $\widetilde{Q}$ is a pseudo-random approximate orthogonal matrix and $\widetilde{D}$ is a floating-point approximation of D. We fix $n = 100$ and $k = 10$ and vary β between 10^2 and 10^{14}. To make the results more illustrative, we provide a less accurate initial guess $X^{(0)}$ using binary32 arithmetic. In the algorithm, we adopt binary128 (so-called quadruple precision) for high-precision arithmetic. Then, the maximum relative accuracy of the computed results is limited to $\mathbf{u}_h = 2^{-113} \approx 10^{-34}$. For binary128 arithmetic, we use the multiple-precision toolbox, in which binary128 arithmetic is supported as a special case by the command `mp.Digits(34)`. The results are shown in Fig. 4. As can be seen, Algorithm 2 can refine the computed eigenvalues until their relative accuracy obtains approximately $\mathbf{u}_h$. Both the orthogonality and diagonality of the computed eigenvectors are improved until approximately $\beta\mathbf{u}_h$. This result is consistent with Remark 1. For $\beta \in \{10^8, 10^{14}\}$, Algorithm 1 cannot work because $X^{(0)}$ is insufficiently accurate and the assumption (3) is not satisfied. We confirm that this problem can be resolved by Algorithm 2.

6.2 Computational speed

To evaluate the computational speed of the proposed algorithm (Algorithm 2), we first compare the computing time of Algorithm 2 to that of an approach that uses multiple-precision arithmetic (MP-approach). Note that the timing should be observed for reference because the computing time for Algorithm 2 strongly depends on the implementation of accurate matrix multiplication. Thus, we adopt an efficient method proposed by Ozaki et al. [19] that utilizes fast matrix multiplication routines such as xGEMM in BLAS. To simulate multiple-precision numbers and arithmetic in Algorithm 2, we represent $\widehat{X} = \widehat{X}_1 + \widehat{X}_2 + \cdots + \widehat{X}_m$ with $\widehat{X}_k$, $k = 1, 2, \ldots, m$, being floating-point matrices in working precision, such as "double-double" ($m = 2$) and "quad-double" ($m = 4$) precision format [10] and use the concept of error-free transformations [16].

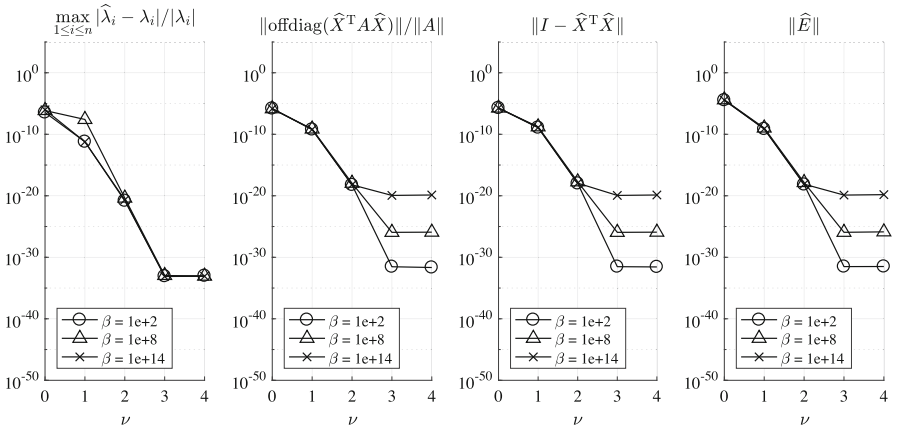

Fig. 4 Results of iterative refinement by Algorithm 2 (RefSyEvCL) in IEEE 754 binary128 arithmetic ($\mathbf{u}_h = 2^{-113} \approx 10^{-34}$) for symmetric matrices with $n = 100$ and various β

In the multiple-precision toolbox [1], the MRRR algorithm [6] via Householder reduction is implemented sophisticatedly with parallelism to solve symmetric eigenvalue problems.

For comparison of timing, we generate a pseudo-random real symmetric $n \times n$ matrix with clustered eigenvalues in a similar way to Sect. 6.1.2. To construct several eigenvalue clusters, we change diagonal elements of D to

$$d_{ii} = \begin{cases} 1 - \left\lfloor \dfrac{i-1}{k} \right\rfloor c^{-1} - (i-1)\beta^{-1} & \text{for } i = 1, 2, \ldots, ck \\ -1 + \dfrac{n-i}{n-ck-1} 2^{-1} & \text{for } i = ck+1, ck+2, \ldots, n \end{cases}.$$

Then, there are c clusters close to $1/c, 2/c, \ldots, 1$ with k eigenvalues and the gap β^{-1} in each cluster, and $n - ck$ eigenvalues are distributed equally in $[-1, -\frac{1}{2}]$. We set $n = 1000$, $c = 5$, $k = 100$, and $\beta = 10^{12}$, i.e., the generated 1000×1000 matrix has five clusters with 100 eigenvalues and the gap 10^{-12} in each cluster. We compare the measured computing time of Algorithm 2 to that of the MP-approach, which is shown in Table 1 together with $\|\widehat{E}\|$ and $n_{\mathcal{J}}$, where $n_{\mathcal{J}}$ is the number of eigenvalue clusters identified in Algorithm 2. At $\nu = 2$, Algorithm 2 successfully identifies five eigenvalue clusters as $n_{\mathcal{J}} = 5$ corresponding to $c = 5$.

For comparison of timing on a lower performance computer, we also conducted numerical experiments using MATLAB R2017b on our laptop PC with a 2.5 GHz Intel Core i7-7660U (2 cores) CPU and 16 GB of main memory. In a similar way to the previous example, we set $n = 500$, $c = 5$, $k = 10$, and $\beta = 10^{12}$, i.e., the generated 500×500 matrix has five clusters with 10 eigenvalues and the gap 10^{-12} in each cluster. As can be seen from Table 2, the result is similar to that in Table 1.

Next, we address more large-scale problems. The test matrices are generated using the MATLAB function randn with $n \in \{2000, 5000, 10{,}000\}$, such as `B = randn(n)` and `A = B + B'`. We aim to compute all the eigenvectors of a given

Table 1 Results for a pseudo-random real symmetric matrix with clustered eigenvalues on our workstation: $n = 1000$, 5 clusters, 100 eigenvalues, and $\beta = 10^{12}$ in each cluster

Algorithm 2	`eig` (binary64)	$\nu = 1$	$\nu = 2$	$\nu = 3$
$\|\widehat{E}\|$	6.4×10^{-4}	2.1×10^{-7}	8.4×10^{-25}	1.1×10^{-38}
$n_{\mathcal{J}}$		0	5	0
Elapsed time (s)	0.15	6.63	23.14	45.19
(accumulated)	0.15	6.77	29.91	75.10
MP-approach	`mp.Digits(d)`	`d` = 34	`d` = 36	`d` = 49
Elapsed time (s)		16.21	256.67	285.33

Table 2 Results for a pseudo-random real symmetric matrix with clustered eigenvalues on our laptop PC: $n = 500$, 5 clusters, 10 eigenvalues, and $\beta = 10^{12}$ in each cluster

Algorithm 2	`eig` (binary64)	$\nu = 1$	$\nu = 2$	$\nu = 3$
$\|\widehat{E}\|$	4.5×10^{-4}	1.4×10^{-7}	5.8×10^{-26}	2.6×10^{-39}
$n_{\mathcal{J}}$		0	5	0
Elapsed time (s)	0.07	1.41	5.99	9.20
(accumulated)	0.07	1.48	7.47	16.67
MP-approach	`mp.Digits(d)`	`d` = 34	`d` = 37	`d` = 50
Elapsed time (s)		5.12	33.89	35.93

real symmetric $n \times n$ matrix A with the maximum accuracy allowed by the binary64 format. To make the results more illustrative, we provide a less accurate initial guess $X^{(0)}$ using `eig` in binary32, and we then refine $X^{(\nu)}$ by Algorithm 2. For efficiency, we use binary64 arithmetic for $\nu = 1, 2$, and accurate matrix multiplication based on error-free transformations [19] for $\nu = 3$. As numerical results, we provide $\|\widehat{E}\|$, $n_{\mathcal{J}}$, and the measured computing time. The results are shown in Table 3. As can be seen, Algorithm 2 improves the accuracy of the computed results up to the limit of binary64 ($\mathbf{u} = 2^{-53} \approx 10^{-16}$). For $n \in \{5000, 10{,}000\}$, Algorithm 2 requires much computing time in total compared with `eig` in binary32 as $n_{\mathcal{J}}$ increases. This is because the problems generally become more ill-conditioned for larger n. In fact, on the minimum gap between eigenvalues, $\beta = 2.71 \cdot 10^{-5}$ for $n = 2000$, $\beta = 2.31 \cdot 10^{-6}$ for $n = 5000$, and $\beta = 2.26 \cdot 10^{-6}$ for $n = 10{,}000$. Thus, it is likely that binary32 arithmetic cannot provide a sufficiently accurate initial guess $X^{(0)}$ for a large-scale random matrix.

6.3 Application to a real-world problem

Finally, we apply the proposed algorithm to a quantum materials simulation that aims to understand electronic structures in material physics. The problems can be reduced to generalized eigenvalue problems, where eigenvalues and eigenvectors correspond to electronic energies and wave functions, respectively. To understand properties of

Table 3 Results of iterative refinement by Algorithm 2 (RefSyEvCL) for pseudo-random symmetric matrices on a workstation

$n = 2000$	`eig` (binary32)	$\nu = 1$	$\nu = 2$	$\nu = 3$
$\|\widehat{E}\|$	2.2×10^{-3}	2.6×10^{-6}	9.5×10^{-12}	9.4×10^{-17}
$n_{\mathcal{J}}$		0	0	0
Elapsed time (s)	0.35	1.68	0.98	1.76
(accumulated)	0.35	2.03	3.01	4.77
$n = 5000$	`eig` (binary32)	$\nu = 1$	$\nu = 2$	$\nu = 3$
$\|\widehat{E}\|$	3.0×10^{-3}	4.6×10^{-6}	3.0×10^{-11}	9.4×10^{-17}
$n_{\mathcal{J}}$		80	16	0
Elapsed time (s)	1.69	15.09	10.67	13.68
(accumulated)	1.69	16.78	27.45	41.13
$n = 10{,}000$	`eig` (binary32)	$\nu = 1$	$\nu = 2$	$\nu = 3$
$\|\widehat{E}\|$	3.9×10^{-3}	8.6×10^{-6}	9.4×10^{-11}	9.4×10^{-17}
$n_{\mathcal{J}}$		1276	587	0
Elapsed time (s)	15.81	406.52	211.36	82.93
(accumulated)	15.81	422.33	633.68	716.61

materials correctly, it is crucial to determine the order of eigenvalues [12] and to obtain accurate eigenvectors [24,25].

We deal with a generalized eigenvalue problem $Ax = \lambda Bx$ arising from a vibrating carbon nanotube within a supercell with s, p, d atomic orbitals [4]. The matrices A and B are taken from ELSES matrix library [7] as `VCNT22500`, where A and B are real symmetric $n \times n$ matrices with B being positive definite and $n = 22500$. Our goal is to compute accurate eigenvectors and separate all the eigenvalues of the problem for determining their order. To this end, we use a numerical verification method in [14] based on the Gershgorin circle theorem (cf. e.g. [9, Theorem 7.2.2] and [23, pp. 71ff]), which can rigorously check whether all eigenvalues are separated and determine an existing range of each eigenvalue.

Let $\Lambda(B^{-1}A)$ be the set of the eigenvalues of $B^{-1}A$. Here, all the eigenvalues of $B^{-1}A$ are real from the assumption of A and B. Let $\widehat{X} \in \mathbb{R}^{n \times n}$ be an approximate eigenvector matrix of $B^{-1}A$ with $\widehat{X}$ being nonsingular. Then, it is expected that $C := \widehat{X}^{-1}B^{-1}A\widehat{X}$ is nearly diagonal. Although it is not possible, in general, to calculate $C = (c_{ij})$ exactly in finite precision arithmetic, we can efficiently obtain an enclosure of C. Note that we compute neither an enclosure of B^{-1} nor that of $\widehat{X}^{-1}$ explicitly. Instead, we compute an approximate solution $\widehat{C}$ of linear systems $(B\widehat{X})C = A\widehat{X}$ and then verify the accuracy of $\widehat{C}$ using Yamamoto's method [26] with matrix-based interval arithmetic [18,21] for obtaining an enclosure of C. Suppose $\widehat{D} = \mathrm{diag}(\widehat{\lambda}_i)$ is a midpoint matrix and $G = (g_{ij})$ is a radius matrix with $g_{ij} \geq 0$ satisfying

$$c_{ij} \in \begin{cases} [\widehat{\lambda}_i - g_{ii}, \widehat{\lambda}_i + g_{ii}] & \text{if } i = j \\ [-g_{ij}, g_{ij}] & \text{otherwise} \end{cases} \quad \text{for all } (i, j).$$

Then, the Gershgorin circle theorem implies

$$\Lambda(B^{-1}A) \subseteq \bigcup_{i=1}^{n} [\widehat{\lambda}_i - e_i, \widehat{\lambda}_i + e_i], \quad e_i := \sum_{j=1}^{n} g_{ij}.$$

It can also be shown that if all the disks $[\widehat{\lambda}_i - e_i, \widehat{\lambda}_i + e_i]$ are isolated, then all the eigenvalues are separated, i.e., each disk contains precisely one eigenvalue of $B^{-1}A$ [23, pp. 71ff].

We first computed an approximate eigenvector matrix $\widehat{X}$ of $B^{-1}A$ using the MATLAB function $\mathsf{eig}(A, B)$ in binary64 arithmetic as an initial guess, and $\widehat{X}$ was obtained in 235.17 s. Then, we had $\max_{1\le i\le n} e_i = 2.75 \times 10^{-7}$, and 10 eigenvalues with 5 clusters could not be separated due to relatively small eigenvalue gaps. We next applied Algorithm 2 to A, B, and $\widehat{X}$ in higher precision arithmetic in a similar way to Sect. 6.2, and obtained a refined approximate eigenvector matrix $\widehat{X}'$ in 597.52 s. Finally, we obtained $\max_{1\le i\le n} e_i = 1.58 \times 10^{-14}$ and confirmed that all the eigenvalues can successfully be separated.

Acknowledgements The first author would like to express his sincere thanks to Professor Chen Greif at the University of British Columbia for his valuable comments and helpful suggestions.

References

1. Advanpix: Multiprecision Computing Toolbox for MATLAB, Code and documentation. http://www.advanpix.com/ (2016)
2. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide, 3rd edn. SIAM, Philadelphia (1999)
3. Atkinson, K., Han, W.: Theoretical Numerical Analysis, 3rd edn. Springer, New York (2009)
4. Cerdá, J., Soria, F.: Accurate and transferable extended Hückel-type tight-binding parameters. Phys. Rev. B **61**, 7965–7971 (2000)
5. Davis, C., Kahan, W.M.: The rotation of eigenvectors by a perturbation. III. SIAM J. Numer. Anal. **7**, 1–46 (1970)
6. Dhillon, I.S., Parlett, B.N.: Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices. Linear Algebra Appl. **387**, 1–28 (2004)
7. ELSES matrix library, Data and documentation. http://www.elses.jp/matrix/ (2018)
8. GMP: GNU Multiple Precision Arithmetic Library, Code and documentation. http://gmplib.org/ (2018)
9. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th edn. The Johns Hopkins University Press, Baltimore (2013)
10. Hida, Y., Li, X. S., Bailey, D. H.: Algorithms for quad-double precision floating point arithmetic. In: Proceedings of the 15th IEEE Symposium on Computer Arithmetic, pp. 155–162. IEEE Computer Society Press (2001)
11. Higham, N.J.: Accuracy and Stability of Numerical Algorithms, 2nd edn. SIAM, Philadelphia (2002)
12. Lee, D., Hoshi, T., Sogabe, T., Miyatake, Y., Zhang, S.-L.: Solution of the k-th eigenvalue problem in large-scale electronic structure calculations. J. Comput. Phys. **371**, 618–632 (2018)

13. Li, X.S., Demmel, J.W., Bailey, D.H., Henry, G., Hida, Y., Iskandar, J., Kahan, W., Kang, S.Y., Kapur, A., Martin, M.C., Thompson, B.J., Tung, T., Yoo, D.: Design, implementation and testing of extended and mixed precision BLAS. ACM Trans. Math. Softw. **28**, 152–205 (2002)
14. Miyajima, S.: Numerical enclosure for each eigenvalue in generalized eigenvalue problem. J. Comput. Appl. Math. **236**, 2545–2552 (2012)
15. MPFR: The GNU MPFR Library, Code and documentation. http://www.mpfr.org/ (2018)
16. Ogita, T., Rump, S.M., Oishi, S.: Accurate sum and dot product. SIAM J. Sci. Comput. **26**, 1955–1988 (2005)
17. Ogita, T., Aishima, K.: Iterative refinement for symmetric eigenvalue decomposition. Jpn. J. Ind. Appl. Math. **35**(3), 1007–1035 (2018)
18. Oishi, S.: Fast enclosure of matrix eigenvalues and singular values via rounding mode controlled computation. Linear Algebra Appl. **324**, 133–146 (2001)
19. Ozaki, K., Ogita, T., Oishi, S., Rump, S.M.: Error-free transformations of matrix multiplication by using fast routines of matrix multiplication and its applications. Numer. Algorithms **59**, 95–118 (2012)
20. Parlett, B.N.: The Symmetric Eigenvalue Problem. Classics in Applied Mathematics, vol. 20, 2nd edn. SIAM, Philadelphia (1998)
21. Rump, S.M.: Fast and parallel interval arithmetic. BIT Numer. Math. **39**, 534–554 (1999)
22. Rump, S.M., Ogita, T., Oishi, S.: Accurate floating-point summation part II: sign, K-fold faithful and rounding to nearest. SIAM J. Sci. Comput. **31**, 1269–1302 (2008)
23. Wilkinson, J.H.: The Algebraic Eigenvalue Problem. Clarendon Press, Oxford (1965)
24. Yamamoto, S., Fujiwara, T., Hatsugai, Y.: Electronic structure of charge and spin stripe order in $La_{2-x}\,Sr_x\,NiO_4$ ($x = \frac{1}{3}, \frac{1}{2}$). Phys. Rev. B **76**, 165114 (2007)
25. Yamamoto, S., Sogabe, T., Hoshi, T., Zhang, S.-L., Fujiwara, T.: Shifted conjugate-orthogonal-conjugate-gradient method and its application to double orbital extended Hubbard model. J. Phys. Soc. Jpn. **77**, 114713 (2008)
26. Yamamoto, T.: Error bounds for approximate solutions of systems of equations. Jpn. J. Appl. Math. **1**, 157–171 (1984)