Forward stable eigenvalue decomposition of rank-one modifications of diagonal matrices

N. Jakovčević Stor\textsuperscript{1}, I. Slapničar\textsuperscript{2}, J. L. Barlow\textsuperscript{3}

Abstract

We present a new algorithm for solving an eigenvalue problem for a real symmetric matrix which is a rank-one modification of a diagonal matrix. The algorithm computes each eigenvalue and all components of the corresponding eigenvector with high relative accuracy in $O(n)$ operations. The algorithm is based on a shift-and-invert approach. Only a single element of the inverse of the shifted matrix eventually needs to be computed with double the working precision. Each eigenvalue and the corresponding eigenvector can be computed separately, which makes the algorithm adaptable for parallel computing. Our results extend to Hermitian case.

Keywords: eigenvalue decomposition, diagonal-plus-rank-one update, high relative accuracy, forward stability

AMS subject classifications: 65F15, 65G50, 15-04, 15B99

1 Introduction and Preliminaries

In this paper we consider eigenvalue problem for a $n \times n$ real symmetric matrix $A$ of the form

$$A = D + \rho zz^T,$$

where

$$D = \text{diag}(d_1, d_2, \ldots, d_n)$$

is diagonal matrix of order $n$,

$$z = [\ z_1 \ z_2 \ \cdots \ z_n \ ]^T$$

is a vector and $\rho \neq 0$ is a scalar. Notice that $A$ is a rank-one modification of a diagonal matrix. Subsequently, we shall refer to such matrices as “diagonal-plus-rank-one” (DPR1) matrices.

DPR1 matrices arise, for example, in solving symmetric real tridiagonal eigenvalue problems with the divide-and-conquer method\textsuperscript{[10]}.\textsuperscript{1}

\textsuperscript{1}University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, R. Boškovića 32, 21000 Split, Croatia, nevena@fesb.hr
\textsuperscript{2}University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, R. Boškovića 32, 21000 Split, Croatia, ivan slapnicar@fesb.hr
\textsuperscript{3}Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802-6822, USA, barlow@cse.psu.edu
In this paper we present an algorithm which computes each eigenvalue and all components of the corresponding eigenvector with high relative accuracy in $O(n)$ operations.

Let us make some assumptions about the matrix $A$. First, without loss of generality, we may assume that $\rho > 0$, otherwise we may consider the matrix

$$A = -D - \rho zz^T.$$  

Further, without loss of generality, we may assume that $A$ is irreducible, that is,

$$\zeta_i \neq 0, \text{ for all } i$$  

and

$$d_i \neq d_j, \text{ for all } i \neq j, i, j = 1, \ldots, n.$$  

If $\zeta_i = 0$ for some $i$, then the diagonal element $d_i$ is an eigenvalue whose corresponding eigenvector is the $i$-th unit vector, and we can reduce the size of the problem by deleting the $i$-th row and column of the matrix, eventually obtaining a matrix for which all elements $\zeta_j$ are nonzero. If $d_i = d_j$, then $d_i$ is an eigenvalue of the matrix $A$: we can reduce the size of the problem by annihilating $\zeta_j$ with a Givens rotation in the $(i, j)$-plane and proceeding as in the previous case.

Also, by symmetric row and column pivoting, we can order elements of $D$ such that

$$d_1 > d_2 > \cdots > d_n.$$  

Finally, without loss of generality, we can also assume that $\zeta_i > 0$ for all $i$, which can be attained by pre- and post-multiplication of the matrix $A$ with the matrix $D_s = \text{diag}(\text{sign}(\zeta_1), \cdots, \text{sign}(\zeta_n), 1)$.

To summarize, we assume that $A$ is ordered irreducible DPR1 matrix of the form (1), where $\rho > 0$, elements of the diagonal matrix $D$ satisfy (3), and $\zeta_i > 0$ for all $i$.

Let

$$A = V \Lambda V^T$$  

be the eigenvalue decomposition of $A$. Here

$$\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$$  

is a diagonal matrix whose diagonal elements are the eigenvalues of $A$, and

$$V = \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix}$$  

is an orthonormal matrix whose columns are the corresponding eigenvectors.

The eigenvalue problem for a DPR1 matrix $A$ can clearly be solved by any of the standard methods for the symmetric eigenvalue problem (see, for example [21, 20]). However, due to the special structure of diagonal-plus-rank-one matrices, we can use the following approach. The eigenvalues of
A are the zeros of the secular equation (see, for example, [5] and [8] Section 8.5.3):

\[ f(\lambda) = 1 + \rho \sum_{i=1}^{n} \frac{\zeta_i^2}{d_i - \lambda} = 1 + \rho z^T (D - \lambda I)^{-1} z = 0, \quad (5) \]

and the corresponding eigenvectors are given by

\[ v_i = \frac{x_i}{\|x_i\|_2}, \quad x_i = (D - \lambda_i I)^{-1} z, \quad i = 1, \ldots, n. \quad (6) \]

Diagonal elements of the matrix \( D, d_i \), are called poles of the function \( f \).

Notice that

\[ f'(\lambda) = \rho \sum_{i=1}^{n} \frac{\zeta_i^2}{(d_i - \lambda)^2}. \]

Since \( \rho > 0 \), \( f \) is strictly increasing between poles implying the strict interlacing property

\[ \lambda_1 > d_1 > \lambda_2 > d_2 > \cdots > \lambda_n > d_n. \quad (7) \]

The approach given in (5) and (6) is conceptually simple and has been used to solve similar eigenvalue problems [2, 4, 5, 7]. However, maintaining orthogonality among the eigenvectors \( v_i \) in (6) requires all of the eigenvalues \( \lambda_i \) to be computed with high accuracy [10]. In other words, if the computed eigenvalues \( \lambda_i \) are not accurate enough, then the computed eigenvectors \( v_i \) may not be sufficiently orthogonal (see Example 3). The existing algorithms for DPR1 matrices [10, 5, 7] obtain orthogonal eigenvectors with the following procedure:

- compute the eigenvalues \( \tilde{\lambda}_i \) of \( A \) by solving (5);
- construct a new matrix

\[ \tilde{A} = D + \rho \tilde{z} \tilde{z}^T \quad (8) \]

by solving an inverse problem with the prescribed eigenvalues \( \tilde{\lambda} \), and diagonal matrix \( D \), that is, compute new \( \tilde{z} \) as

\[ \tilde{\zeta}_i = \sqrt{\left( \tilde{\lambda}_i - d_i \right) \prod_{j=1}^{i-1} \left( \tilde{\lambda}_j - d_i \right) \prod_{j=i+1}^{n} \left( \tilde{\lambda}_j - d_i \right) \prod_{j=i+1}^{n} \left( d_j - d_i \right)}. \]

- compute the eigenvectors of \( \tilde{A} \) by (6).

Since the formulas for \( \tilde{\zeta}_i \) involve only multiplications, division and subtractions of exact quantities, each \( \zeta_i \) is computed with relative error of \( O(\varepsilon_M) \), where \( \varepsilon_M \) denotes the machine precision. The machine precision
ε_M is defined as the smallest positive number such that in the floating-point arithmetic 1 + ε_M ≠ 1. In Matlab or FORTRAN REAL(8) arithmetic ε_M = 2^{-53} ≈ 1.1102 · 10^{-16}, thus the floating-point numbers have approximately 16 significant decimal digits. The term “double the working precision” means that the computations are performed with numbers having approximately 32 significant decimal digits, or with the machine precision less than or equal to ε_M^2.

Therefore, \( \tilde{A} \) in [8] satisfies \( \tilde{A} = A + δA, \) where \( ∥δA∥_2 = O(ε_M). \) Here \( ∥·∥_2 \) denotes the spectral matrix norm. We conclude that the computed eigenvalues \( \tilde{λ}_i \) satisfy standard perturbation bounds like those from [8 Corollary 8.1.6]. Further, since \( \tilde{λ}_i \) are the eigenvalues of the matrix \( \tilde{A} \) computed to higher relative accuracy, the eigenvectors computed by [8] are orthogonal to machine precision. For details see [10, 5, 7, 2]. This results in an algorithm which requires only \( O(n^2) \) computations and \( O(n) \) storage for eigenvalues and \( O(n) \) storage for each eigenvector. This algorithm is implemented in the LAPACK subroutine DLAED9 and its subroutines [1].

Our algorithm uses a different approach and is forward stable, that is, it computes all eigenvalues and all individual components of the corresponding eigenvectors of a given arrowhead matrix of floating-point numbers to almost full accuracy, a feature which no other method has. The accuracy of the eigenvectors and their numerical orthogonality follows from the high relative accuracy of the computed eigenvalues.

Even though we use bisection to compute the zeros of the secular equation, our FORTRAN implementation of our algorithm is, for larger matrices \( n = 4000 \), about four times slower than the optimal implementation of the LAPACK routine DLAED9 from the Intel Math Kernel Library [13].

The algorithm is based on a shift-and-invert technique. Basically, an eigenvalue \( λ \) is computed from the largest or the smallest eigenvalue of the inverse of the matrix shifted to the pole \( d_i \) which is nearest to \( λ \), that is,

\[
λ = \frac{1}{ν} + d_i, \tag{9}
\]

where \( ν \) is either largest or smallest (first or last) eigenvalue of the matrix

\[
A_i^{-1} \equiv (A - d_i I)^{-1}.
\]

The inverses of DPR1 matrices are structured as follows (here \( × \) stands for non-zero element): the inverse of an irreducible DPR1 matrix with one pole equal to zero is a permuted arrowhead matrix [14]:

\[
\begin{pmatrix}
  \times & \times \\
  \times & 0 & \times \\
  \times & \times & \times & \times \\
\end{pmatrix}^{-1} + ρ zz^T = \begin{pmatrix}
  \times & \times & \times & \times & \times & \times \\
  \times & \times & \times & \times & \times & \times \\
  \times & \times & \times & \times & \times & \times \\
\end{pmatrix}.
\]
The corresponding formulas are given in (10), (11) and (12). Inverse of a non-singular DPR1 matrix with \( d_i \neq 0, i = 1, \ldots, n \), is again a DPR1 matrix. The corresponding formulas are given in Remark 1.

Our algorithm is completely parallel, since the computation of one eigenvalue and its eigenvector is completely independent of the computation of other eigenvalues and eigenvectors.

The organization of the paper is the following. In Section 2, we describe the basic idea of our algorithm named \textit{dpr1eig} (DPR1 EIGenvalues). In Section 3, we discuss the accuracy of the algorithm. In Section 4, we present the complete algorithm which uses double the working precision, if necessary. In Section 4.1, we discuss fast secular equation solvers, and in Section 4.2 we discuss three implementations of the double the working precision. In Section 4.3 we extend our results to the Hermitian case. In Section 5, we illustrate algorithm with few examples. The proofs are given in Appendix A.

2 The basic shift-and-invert algorithm

Let \( \lambda \) be an eigenvalue of \( A \), let \( v \) be its eigenvector, and let \( x \) be the unnormalized version of \( v \) from (9). Let \( d_i \) be a pole which is closest to \( \lambda \). Clearly, from (7) it follows that either \( \lambda = \lambda_i \) or \( \lambda = \lambda_{i+1} \). Let \( A_i \) be the shifted matrix

\[
A_i = A - d_i I = \begin{bmatrix} D_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & D_2 \end{bmatrix} + \rho \begin{bmatrix} z_1 \\ \zeta_i \\ z_2 \end{bmatrix} \begin{bmatrix} z_1^T \\ \zeta_i \\ z_2^T \end{bmatrix},
\]

where

\[
D_1 = \text{diag}(d_1 - d_i, \ldots, d_{i-1} - d_i),
\]
\[
D_2 = \text{diag}(d_{i+1} - d_i, \ldots, d_n - d_i),
\]
\[
z_1 = \begin{bmatrix} \zeta_1 & \zeta_2 & \cdots & \zeta_{i-1} \end{bmatrix}^T,
\]
\[
z_2 = \begin{bmatrix} \zeta_{i+1} & \zeta_{i+2} & \cdots & \zeta_n \end{bmatrix}^T.
\]

Notice that \( D_1 \) (\( D_2 \)) is positive (negative) definite.

Obviously, \( \lambda \) is an eigenvalue of \( A \) if and only if

\[
\mu = \lambda - d_i
\]
is an eigenvalue of \( A_i \), and they share the same eigenvector. The inverse of \( A_i \) is a permuted arrowhead matrix

\[
A_i^{-1} = \begin{bmatrix} D_1^{-1} & w_1 & 0 \\ w_1^T & b & w_2^T \\ 0 & w_2 & D_2^{-1} \end{bmatrix},
\]

5
where

\[
\begin{align*}
    w_1 &= -D_1^{-1} z_1 \frac{1}{\zeta_i} , \\
    w_2 &= -D_2^{-1} z_2 \frac{1}{\zeta_i} , \\
    b &= \frac{1}{\zeta_i^2} \left( \frac{1}{\rho} + z_1^T D_1^{-1} z_1 + z_2^T D_2^{-1} z_2 \right) = \bar{f}(d_i) \rho \zeta_i^2 .
\end{align*}
\]

In (12),

\[
\bar{f}(d_i) = 1 + \rho \tilde{z}^T (\tilde{D} - d_i I)^{-1} \tilde{z} ,
\]

where \( \tilde{D} \) is the diagonal matrix \( D \) without \( d_i \) and \( \tilde{z} \) is \( z \) without \( \zeta_i \). The computation of the scalar \( b \) in (12), is critical to how well we are able to compute a desired eigenvalue.

The eigenvalues \( \nu = 1/\mu \) of a real symmetric arrowhead matrix \( A_i^{-1} \) from (11) are the zeros of the secular equation (see, for example \([19, 14]\))

\[
g(\nu) = b - \nu - w^T (\Delta - \nu I)^{-1} w = 0 ,
\]

where

\[
\Delta = \begin{bmatrix} D_1 & \\ D_2 \end{bmatrix} , \quad w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} .
\]

Once \( \nu \) is computed, the normalized and unnormalized eigenvectors \( v \) and \( x \) are computed by applying (6) to the matrix \( A_i \), that is,

\[
x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} (D_1 - \mu I)^{-1} z_1 \\ \zeta_i \\ -\mu \\ (D_2 - \mu I)^{-1} z_2 \end{bmatrix} .
\]

If \( \lambda \) is an eigenvalue of \( A \) which is closest to the pole \( d_i \), then \( \mu \) is the eigenvalue of matrix \( A_i \) which is closest to zero and

\[
\nu = \frac{1}{\mu} = \pm \| A_i^{-1} \|_2 .
\]

We say that \( \nu \) is the largest absolute eigenvalue of \( A_i^{-1} \). In this case, if all entries of \( A_i^{-1} \) are computed with high relative accuracy, then, according to standard perturbation theory, any reasonable algorithm can compute \( \nu \) to high relative accuracy. In Section 3, we show that all entries of \( A_i^{-1} \) are indeed computed to high relative accuracy, except possibly \( b \) in (12). If \( b \) is not computed to high relative accuracy and it influences \( \| A_i^{-1} \|_2 \), it is sufficient to compute it with double the working precision (see Section 4).

Further, if \( \mu \) is not the eigenvalue of \( A_i \) which is closest to zero, then \( |\nu| < \| A_i^{-1} \|_2 \), and the quantity

\[
K_\nu = \frac{\| A_i^{-1} \|_2}{|\nu|} .
\]

\( (15) \)
tells us how far ν is from the largest absolute eigenvalue of $A_i^{-1}$. If $K_ν \gg 1$, then standard perturbation theory does not guarantee that the eigenvalue $\mu$ will be computed with high relative accuracy. Remedies to this situation are described in Remark 3.

With this approach, the componentwise high relative accuracy of the eigenvectors computed by (14) follows from the high relative accuracy of the computed eigenvalues (see Theorem 3). Componentwise high relative accuracy of the computed normalized eigenvectors implies, in turn, their numerical orthogonality.

The described procedure is implemented in algorithm dpr1eig_basic (Algorithm 1). The computation of the inverse of the shifted matrix, $A_i^{-1}$, according to formulas (11) and (12), is implemented in Algorithm 2. Algorithm 3 computes the first or last zero of the secular equation (13) by bisection. Given eigenvalue $\lambda$, Algorithm 4 computes the corresponding eigenvector according to (6), or, if called for the shifted matrix, according to (14).

## 3 Accuracy of the algorithm

We now consider numerical properties of Algorithms 1, 2, 3, and 4. We assume the standard model of floating point arithmetic where subtraction is preformed with guard digit, such that [11, Section 2.2]

$$fl(a \text{ op } b) = (a \text{ op } b)(1 + \varepsilon_{\text{op}}), \quad |\varepsilon_{\text{op}}| \leq \varepsilon_M, \quad \text{op} \in \{+, -, *, /\},$$

where $a$ and $b$ are floating-point numbers, $\varepsilon_M$ is machine precision, and we assume that neither underflow nor overflow occurs. In the statements of the theorems and their proofs, we shall use the standard first order approximations, that is, we neglect the terms of order $O(\varepsilon_M^2)$ or higher.

We shall use the following notation:

| Matrix          | Exact eigenvalue | Computed eigenvalue |
|-----------------|------------------|---------------------|
| $A_i$           | $\lambda$       | $\lambda$           |
| $A_i^{-1}$      | $\mu$           | $\tilde{\mu} = fl(\hat{\mu})$ |
| $A_i^{-1}$      | $\nu$           | $\tilde{\nu} = fl(\hat{\nu})$ |
| $(A_i^{-1})^{-1}$ | $\mu$           | $\tilde{\mu} = fl(\hat{\mu})$ |

Here

$$\tilde{A}_i = fl(A_i) = \begin{bmatrix} D_1(I + E_1) & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & D_2(I + E_2) \end{bmatrix} + \rho \begin{bmatrix} z_1^T \\ \zeta_i \\ z_2^T \end{bmatrix},$$

where $E_1$ and $E_2$ are diagonal matrices whose elements are bounded by $\varepsilon_M$ in absolute values.
Algorithm 1
\[ \lambda, v = \text{dpr1eig\_basic}(D, z, \rho, k) \]
% Computes the \( k \)-th eigenpair of an irreducible DPR1 matrix \( A = \text{diag}(D) + \rho z z' \)
\( n = \text{length}(D) \)
% Determine the shift \( \sigma \), the shift index \( i \), and whether \( \lambda \) is on the left
% or the right side of the nearest pole.
% Exterior eigenvalue (\( k = 1 \)):
\begin{algorithmic}
\If {\( k == 1 \)}
\State \( \sigma = d_1 \)
\State \( i = 1 \)
\State \( \text{side} = 'R' \)
\Else
\State \( D_{\text{temp}} = D - d_k \)
\State \( \text{middle} = D_{\text{temp}}k - 1/2 \)
\State \( F_{\text{middle}} = 1 + \rho \sum (z. * z. / (D_{\text{temp}} - \text{middle})) \)
\If {\( F_{\text{middle}} > 0 \)}
\State \( \sigma = d_k \)
\State \( i = k \)
\State \( \text{side} = 'R' \)
\Else
\State \( \sigma = d_{k-1} \)
\State \( i = k - 1 \)
\State \( \text{side} = 'L' \)
\EndIf
\EndIf
% Compute the inverse of the shifted matrix, \( A_i^{-1} \)
\[ [\text{inv}D_1, \text{inv}D_2, w_1, w_2, b] = \text{invA}(D, z, \rho, i) \]
% Compute the leftmost or the rightmost eigenvalue of the arrowhead matrix \( A_i^{-1} \)
\( \nu = \text{bisect}([\text{inv}D_1; \text{inv}D_2], [w_1; w_2], b, \text{side}) \)
% Compute the corresponding eigenvector according to (14)
\( \mu = 1/\nu \)
\( v = \text{vect}(D - \sigma, z, \mu) \)
% Shift the eigenvalue back
\( \lambda = \mu + \sigma \)
Algorithm 2
\[
[\text{inv}D_1, \text{inv}D_2, w_1, w_2, b] = \text{invA} (D, \rho, i)
\]
% Computes the inverse of an irreducible DPR1 matrix \( A = \text{diag}(D - d_i) + \rho zz' \)
% according to [11] and [12].
n = \text{length}(D)
D = D - d_i
\text{inv}D_1 = 1./D_{1:i-1}
\text{inv}D_2 = 1./D_{i+1:n}
w_\zeta = 1/z_i
w_1 = -z_{1:i-1} \cdot \text{inv}D_1 \cdot w_\zeta
w_2 = -z_{i+1:n} \cdot \text{inv}D_2 \cdot w_\zeta
b = (1/\rho + \text{sum}(z_{1:i-1} \cdot 2 \cdot \text{inv}D_1) + \text{sum}(z_{i+1:n} \cdot 2 \cdot \text{inv}D_2)) \cdot w_\zeta \cdot w_\zeta

Algorithm 3
\[
\lambda = \text{bisect} (\Delta, w, b, \text{side})
\]
% Computes the leftmost (for \text{side}='L') or the rightmost (for \text{side}='R') eigenvalue
% of an arrowhead matrix \( A = [\text{diag}(\Delta) \, w; w' \, b] \) by solving [13] with bisection.
% Determine the starting interval for bisection, [left, right]
if \text{side} == 'L'
  left = \text{min}\{\Delta - |w|, b - \|w\|_1\}
  right = \text{min} \Delta
else
  right = \text{max}\{\Delta + |w|, b + \|w\|_1\}
  left = \text{max} \Delta
end
% Bisection
middle = (left + right)/2
while \((right - left)/\text{abs}(middle) > \varepsilon_M\)
  \( F_{\text{middle}} = b - \text{middle} - \text{sum}(w' \cdot 2 \cdot (\Delta - \text{middle})) \)
  if \( F_{\text{middle}} > 0\)
    left = middle
  else
    right = middle
end
middle = (left + right)/2
end
% Eigenvalue
\( \lambda = \text{right} \)
Algorithm 4

\[ v = \text{vect}(D, z, \lambda) \]
% Computes the eigenvector of a DPR1 matrix \( A = \text{diag}(D) + \rho z z' \)
% which corresponds to the eigenvalue \( \lambda \) by using \( \text{(6)} \).
\[ v = z ./ (D - \lambda) \]
\[ v = v / \| v \|_2 \]

Further we define the quantities \( \kappa_\lambda, \kappa_\mu \) and \( \kappa_b \) as follows:

\[ \tilde{\lambda} = \text{fl}(\lambda) = \lambda (1 + \kappa_\lambda \varepsilon_M), \]
\[ \tilde{\mu} = \text{fl}(\mu) = \mu (1 + \kappa_\mu \varepsilon_M), \]
\[ \tilde{b} = \text{fl}(b) = b (1 + \kappa_b \varepsilon_M). \]

We also define the quantity

\[ K_b = \frac{1 + \rho z_1^T D_1^{-1} z_1 - \rho z_2^T D_2^{-1} z_2}{1 + \rho z_1^T D_1^{-1} z_1 + \rho z_2^T D_2^{-1} z_2}. \]

In the next few sections, we show that keeping \( K_b \) modest allows us to compute \( b \) in \( \text{(12)} \) in a way that keeps \( \kappa_b \) in \( \text{(19)} \) modest. Before that, we show that if we keep \( \kappa_b \) modest, then it is always possible to compute highly accurate eigenvalues and eigenvectors.

### 3.1 Connection between accuracy of \( \lambda \) and \( \mu \)

Let

\[ \lambda = \mu + d_i \]

be an eigenvalue of the matrix \( A \), and let \( \mu \) be the corresponding eigenvalue of the shifted matrix \( A_i = A - d_i \). We compute \( \mu \) from \( A_i^{-1} \) and then \( \lambda \). Also, let

\[ \tilde{\lambda} = \text{fl}(\tilde{\mu} + d_i) \]

be the computed eigenvalue. Theorem 1 shows the relations between the accuracy of \( \tilde{\lambda} \) in \( \text{(17)} \) and the accuracy of \( \tilde{\mu} \) in \( \text{(18)} \).

**Theorem 1.** For \( \lambda \) and \( \tilde{\lambda} \) from \( \text{(17)} \) and \( \mu \) and \( \tilde{\mu} \) from \( \text{(18)} \) we have

\[ |\kappa_\lambda| \leq \frac{|d_i| + |\mu|}{|\lambda|} (|\kappa_\mu| + 1). \]

Proofs of this theorem and subsequent theorems are given in Appendix A.

From Theorem 1 we see that the accuracy of \( \tilde{\lambda} \) depends on \( \kappa_\mu \) and the size of the quotient

\[ \frac{|d_i| + |\mu|}{|\lambda|}. \]

10
Theorem 2 analyzes the quotient (22) with respect to the position of \( \lambda \), the sign of \( \mu \), and the signs of the neighboring poles. Figure 1 illustrates the assumptions made in Theorem 2.

**Theorem 2.** Let the assumptions of Theorem 1 hold.

(i) If (see Figure 1 (i)) \( \text{sign} \left( d_i \right) = \text{sign} \left( \mu \right) \), then

\[
\frac{|d_i| + |\mu|}{|\lambda|} = 1.
\]

(ii) If \( \lambda \) is between two poles of the same sign and \( \text{sign} \left( d_i \right) \neq \text{sign} \left( \mu \right) \) (see Figure 1 (ii)), then

\[
\frac{|d_i| + |\mu|}{|\lambda|} \leq 3.
\]

Figure 1: Typical situations from Theorem 2

Theorem 2 does not cover the cases when \( d_1 < 0 \), \( d_{i+1} < 0 < d_i \) and \( \mu < 0 \), or \( d_i < 0 < d_{i-1} \) and \( \mu > 0 \) (see e.g. Figure 2). Notice that these cases are mutually exclusive, so there can be at most one eigenvalue for which the quotient (22) is not bounded by Theorem 2. If one of these cases occurs, the quotient (22) may still be small. The quotient (22) will clearly be large if, additionally, \( \mu \) is such that \( \lambda \) is near zero. Then \( \lambda \) is computed as a difference of two close quantities and cancellation can occur.

Figure 2: Typical situations for special cases

If the quotient (22) is large, remedy is given in the following remark.

**Remark 1.** [Inverting the unshifted matrix] If, in one of the above cases, the quotient (22) is large, then \( \lambda \) is also an eigenvalue of \( A \) nearest to
zero, and we can accurately compute it from the inverse of $A$. Notice that
the inverse is of an unreduced DPR1 matrix $A = D + \rho z z^T$, with all poles
being non-zero, is a DPR1 matrix of the form

$$A^{-1} = D^{-1} + \gamma uu^T,$$

where

$$u = D^{-1}z,$$

$$\gamma = -\frac{\rho}{1 + \rho z^T D^{-1} z}.$$  \hspace{2cm} (23)

Eigenvalues of $A^{-1}$ are the zeros of the corresponding secular equation (5).
Since the absolutely largest eigenvalue of $A^{-1}$ is computed accurately accord-
ing to standard perturbation theory, and $1/|\lambda| = \|A^{-1}\|_2$, $\lambda$ is also computed
with high relative accuracy. In computing matrix $A^{-1}$, eventually $\gamma$
needs to be computed in higher precision. For more details see Remark \ref{rem:2}. If the
denominator in $\gamma$ is computed as zero, the matrix $A$ is numerically singular
and we can set $\lambda = 0$. Notice that all components of the corresponding
eigenvector are still computed accurately.

**Remark 2. [Additional accuracy in $\lambda$]** Notice that Algorithm 1 (and,
consequently, Algorithm 5 below) can be easily modified to return both
quantities, $d_i$ and $\mu$ such that $\lambda = d_i + \mu$. If none of the remedies from
Remark \ref{rem:1} were needed, these two quantities give additional information
about $\lambda$ (that is, they give a more accurate representation of $\lambda$). An example
is given in Example 2 in section 5.

We still need to bound the quantity $\kappa \mu$ from (21). This quantity depends
on the accuracy of $fl(b)$. The bound for $\kappa \mu$ is given in Theorem \ref{thm:3}.

### 3.2 Accuracy of the eigenvectors

Since the eigenvector is computed by (14), its accuracy depends on the
accuracy of $\tilde{\mu}$ as described by the following theorem:

**Theorem 3.** Let (13) hold and let

$$\tilde{x} = \begin{bmatrix} \tilde{x}_1 \\ \vdots \\ \tilde{x}_n \end{bmatrix} = fl \begin{bmatrix} (D_1 (I + E_1) - \tilde{\mu} I)^{-1} z_1 \\ -\frac{\zeta_i}{\mu} \\ (D_2 (I + E_2) - \tilde{\mu} I)^{-1} z_2 \end{bmatrix}$$  \hspace{2cm} (24)

be the computed un-normalized eigenvector corresponding to $\mu$ and $\lambda$. Then

$$\tilde{x}_j = x_j (1 + \varepsilon_{x_j}), \hspace{0.5cm} |\varepsilon_{x_j}| \leq 3 (|\kappa \mu| + 3) \varepsilon_M, \hspace{0.5cm} j = 1, \ldots, n.$$
In other words, if $\kappa_\mu$ is small, then all components of the unnormalized eigenvector $x$ are computed to high relative accuracy. Since computing the vector norm and the scaling induces only a small relative error in each component, all components of the corresponding normalized eigenvector $v$ are also computed to high relative accuracy. Componentwise high relative accuracy of the computed normalized eigenvectors implies, in turn, that they are mutually numerically orthogonal to the order $O(n\kappa_\mu \varepsilon_M)$.

Since the accuracy of $\tilde{\lambda}$ and $\tilde{x}$ depends on the accuracy of $\tilde{\mu}$ (that is, the size of $\kappa_\mu$), in the next three subsections, we discuss the accuracy of $\tilde{\mu}$.

Since $\tilde{\mu}$ is computed as the inverse of the eigenvalue of the matrix $f(A_i^{-1})$, we first discuss the accuracy of that matrix.

### 3.3 Accuracy of the matrix $A_i^{-1}$

We have the following theorem:

**Theorem 4.** For the computed elements of the matrix $A_i^{-1}$ from (11) and (12) for all $(j,k) \neq (i,i)$ we have

$$
(A_i^{-1})_{jk} = f(A_i^{-1})_{jk} = (A_i^{-1})_{jk}(1 + \varepsilon_{jk}), \quad |\varepsilon_{jk}| \leq 3\varepsilon_M.
$$

For the computed element $b \equiv (A_i^{-1})_{ii}$ from (19) we have

$$
|\kappa_b| \leq (n + 4)K_b,
$$

where $K_b$ is defined by (20).

The above theorem states that all elements of the matrix $A_i^{-1}$ are computed with high relative accuracy with the possible exception of $b = (A_i^{-1})_{ii}$. Therefore, the accuracy of our algorithm depends upon the accurate computation of $b$ as detailed in section 4.

### 3.4 Accuracy of bisection

Let $\lambda_{\text{max}}$ be the absolutely largest eigenvalue of a symmetric arrowhead matrix $A$, and let $\tilde{\lambda}_{\text{max}}$ be the eigenvalue computed by bisection as implemented in Algorithm 3. The error bound from [19, §3.1] immediately implies that

$$
\frac{|\tilde{\lambda}_{\text{max}} - \lambda_{\text{max}}|}{|\lambda_{\text{max}}|} = \kappa_{\text{bis}} \varepsilon_M, \quad \kappa_{\text{bis}} \leq 1.06n(\sqrt{n} + 1).
$$

(25)

Notice that the similar error bound holds for all eigenvalues which are of the same order of magnitude as $|\lambda_{\text{max}}|$.
3.5 Accuracy of exterior eigenvalues of $A_i^{-1}$

The desired interior eigenvalue and, in some cases, also absolutely smaller exterior eigenvalue $\lambda$ of $A$ is computed by (9), where $\nu$ is one of the exterior eigenvalues of the matrix $A_i^{-1}$.

The following theorem covers the case when $\nu$ is the largest absolute eigenvalue of $A_i^{-1}$, and gives two different bounds.

**Theorem 5.** Let $A_i^{-1}$ be defined by (11) and let $\nu$ be its eigenvalue such that

$$|\nu| = \|A_i^{-1}\|_2.$$  \hspace{1cm} (26)

Let $\tilde{\nu}$ be the exact eigenvalue of the computed matrix $(A_i^{-1}) = fl (A_i^{-1})$. Let $\tilde{\nu} = \nu (1 + \kappa_{\nu} \varepsilon_M)$. \hspace{1cm} (27)

Then

$$|\kappa_{\nu}| \leq \min \left\{ (n+4)\sqrt{n} K_b, 3\sqrt{n} + (n+4)(1 + \frac{2}{|\zeta_i|} \sum_{j=1\atop j \neq i}^n |\zeta_j|) \right\},$$ \hspace{1cm} (28)

where $K_b$ is defined by (20).

3.6 Final error bounds

All previous error bounds are summarized as follows.

**Theorem 6.** Let $\tilde{\lambda}$ be the computed eigenvalue of an unreduced DPR1 matrix $A$, let $\tilde{\mu}$ be computed eigenvalue of the matrix $A_i$ from (10), and let $\tilde{\nu}$ be the corresponding computed eigenvalue of the matrix $(A_i^{-1})$ from (11). If $\mu$ is the eigenvalue of $A_i$ closest to zero (or, equivalently, if (26) holds), then the error in the computed eigenvalue $\tilde{\lambda}$ is given by (17) with

$$|\kappa_{\lambda}| \leq 3(|\kappa_{\nu}| + \kappa_{\text{bis}}) + 4,$$ \hspace{1cm} (29)

and the error in the computed un-normalized eigenvector $\tilde{x}$ is given by Theorem 3 with

$$|\kappa_{\mu}| \leq |\kappa_{\nu}| + \kappa_{\text{bis}} + 1,$$ \hspace{1cm} (30)

where $|\kappa_{\nu}|$ is bounded by (28) and $\kappa_{\text{bis}}$ is defined by (25).

Since we are essentially using a shift-and-invert technique, we can guarantee high relative accuracy of the computed eigenvalue and high componentwise relative accuracy of the computed eigenvector if $\nu$ is such that $|\nu| = O(\|A_i^{-1}\|_2)$ and it is computed accurately. This is certainly fulfilled if the following conditions are met:
C1. The quantity $K_\nu$ from (17) is moderate, and

C2. (i) either the quantity $K_b$ from (20) is small, or

(ii) the quantity $\frac{1}{|\zeta_i|} \sum_{j=1, j \neq i}^n |\zeta_j|$ from (28) is of order $O(n)$.

The condition $C1$ implies that $\nu$ will be computed accurately according to the standard perturbation theory. The conditions $C2 (i)$ or $C2 (ii)$ imply that $\kappa_\nu$ from (28) is small, which, together with $C1$, implies that $\nu$ is computed accurately.

If the condition $C1$ does not hold, that is, if $K_\nu \gg 1$, remedies are given in Remark 2 above. If neither of the conditions $C2 (i)$ and $C2 (ii)$ holds, the remedy is to compute $b$ with double the working precision as described in section 4.

Remark 3. [Non-standard shifting] There are two possibilities:

(a) we can compute $\lambda$ by shifting to the neighboring pole on the other side if that gives a smaller value of $K_\nu$ (for example, by shifting to the pole $d_{i-1}$ instead of $d_i$ in Figure 3 (a)),

(b) if shifting to another neighboring pole is not possible (if $K_\nu \gg 1$, see Figure 3 (b)), then we can invert $A - \sigma I$, where the shift $\sigma$ is chosen near but not equal to $\lambda$, and not equal to the neighboring poles. This results in a DPR1 matrix

\[(A - \sigma I)^{-1} = (D - \sigma I)^{-1} + \gamma uu^T,\]

where $u$ and $\gamma$ are defined similarly as in Remark 1 (simply substitute $D$ by $D - \sigma I$), and the largest absolute eigenvalue is computed accurately. Similar to $b$ in (12), $\gamma$ may need to be computed in higher precision. If no floating-point numbers $\sigma$ lie between $\lambda$ and the neighboring poles, $\sigma$ and the corresponding DPR1 matrix must be computed in double the working precision.

4 Final algorithm

If neither of the conditions $C2 (i)$ and $C2 (ii)$ hold, in order to guarantee that $\lambda$ will be computed with high relative accuracy, the element $b$ from the matrix $A^{-1}_i$ needs to be computed in higher precision. The following

\[\text{Determining whether $\gamma$ needs to be computed in higher precision is done similarly as determining whether element $b$ of $A^{-1}_i$ needs to be computed in higher precision, which is described in section 4. Further, Theorem 7 implies that it suffices to compute $\gamma$ with double the working precision.}\]
Theorem 7. Set

\[ P = \frac{1}{\rho} + z_1^T D_1^{-1} z_1, \quad Q = -z_2^T D_2^{-1} z_2. \]

Notice that \( P, Q \geq 0 \) and \( b = (P - Q)/\zeta_i^2 \). Let \( \bar{P} = fl(P) \) and \( \bar{Q} = fl(Q) \) be evaluated in standard precision, \( \varepsilon_M \). Assume that \( P \neq Q \) and \( K_b \leq O(1/\varepsilon_M) \). If \( P, Q \) and \( b \) are all evaluated with double the working precision, \( \varepsilon_M^2 \), then (19) holds with \(|\kappa_b| \leq O(n)\).

Remark 4. [Summation techniques] Since the summands in (12) are computed quantities, neither compensated summation [11, Algorithm 4.2, p. 84] nor doubly compensated summation [11, Algorithm 4.3, pp. 87–88] nor any other summation method is guaranteed to achieve necessary accuracy without using double the working precision.

We summarize the above results in one complete algorithm, \textit{dpr1eig}. The algorithm first checks the components of the vector \( z \). If they are of the same order of magnitude, the eigenpair \((\lambda, v)\) is computed by Algorithm 1. If that is not the case, the quantity \( K_b \) is computed. If \( K_b \gg 1 \), the eigenpair \((\lambda, v)\) is computed by Algorithm 1, but with evaluation of \( b \) with double the working precision. At the end, the quantity \( K_\nu \) is computed, and if \( K_\nu \gg 1 \), one of the remedies from Remark 3 must be used.

4.1 Fast secular equation solvers

Instead of using bisection to compute zeros of secular equation (13) in Algorithm 3, we can use some fast zero finder with quadratic or even cubic convergence like those from [18, 3, 15]. Such zero finders compute zeros to
Algorithm 5

\[ [\lambda, v] = \text{dpr1eig} (D, z, \rho, k) \]

% Computes the \( k \)-th eigenpair of an ordered irreducible DPR1 matrix
% \( A = \text{diag} (D) + \rho zz' \)

compute the shift \( i \) as in the first part of Algorithm 1

if the quantity \( \left( \sum_{j=1}^{n} |\zeta_j| \right) / |\zeta_i| \) from (28) is of \( O(n) \)

% standard precision is enough

\[ [\lambda, v] = \text{dpr1eig\_basic} (D, z, \rho, k) \]

else

compute the quantity \( K_b \) from (20)

if \( K_b \gg 1 \)

% double the working precision is necessary

\[ [\lambda, v] = \text{dpr1eig\_basic} (D, z, \rho, k) \) with evaluation of \( b \) with double the working precision

else

% standard precision is enough

\[ [\lambda, v] = \text{dpr1eig\_basic} (D, z, \rho, k) \]

end

end

compute the quantity \( K_\nu \) from (15)

if \( K_\nu \gg 1 \)

apply one of the remedies from Remark 3

end

apply the remedy from Remark 4 if necessary
machine accuracy using a small number of direct evaluations of the Pick function and its derivative, where \( O(\log(\log(1/\varepsilon))) \) iterations are needed to obtain an \( \varepsilon \)-accuracy [15].

In particular, we tested the implementation of the cubically convergent zero finder by Borges and Gragg from [3, §3.3], with the stopping criterion defined by [3, p. 15]. From [3, (21)], it follows that the accuracy of the computed solution satisfies similar backward error bound as (25). This was indeed, true in all our tests. The number of iterations never exceeded 7.

Similarly, for the solution of the secular equation (4), which is needed in remedies according to Remarks 1 and 3, one can use the fast secular equation solver by Li [15]. This solver is implemented in the LAPACK routine DLAED4. The accuracy of the computed solution satisfied similar backward error bound as (25) and the number of iterations behaved as predicted.

Although the operation count of both fast zero finders is approximately half of the operations needed for bisection, we observed no speed-up in our Matlab implementation. This is due to the fact that the formulas of zero finders are more complex and have more memory references.

4.2 Implementation of the double the working precision

We tried three different implementations of the double the working precision:

- by converting all quantities in the formulas (12) or (23) to variable precision by Matlab command \texttt{sym} with parameter ’f’, and then performing the computations;
- by evaluating all parts of the formulas (12) or (23) using extended precision routines \texttt{add2}, \texttt{sub2}, \texttt{mul2}, and \texttt{div2} from [6]; and
- by converting all quantities in the formulas (12) or (23) from standard 64 bit double precision numbers, declared by \texttt{REAL(8)}, to 128 quadruple precision numbers, declared by \texttt{REAL(16)}, in Intel FORTRAN compiler \texttt{ifort} [12], and then performing the computations.

Having to invoke higher precision clearly slows the computation down. In Matlab, when using variable precision \texttt{sym} command, the computation may be slowed down by a factor of three hundred or more for each eigenvalue that requires formulas (12) or (23) to be evaluated in higher precision. This makes use of \texttt{sym} prohibitive for higher dimensions. Using Matlab implementations of the extended precision routines by Dekker [6], slows the computation down up to 8 times. The fastest implementation is the one in \texttt{ifort} which is only about three times slower. Thus, the algorithm benefits from a good implementation of higher precision.
4.3 Hermitian matrices

In this section we extend our results to Hermitian case. Let

\[ C = D + \rho z z^*, \]

where

\[ D = \text{diag}(d_1, d_2, \ldots, d_n), \]

is a real diagonal matrix of order \( n \),

\[ z = \begin{bmatrix} \zeta_1 & \zeta_2 & \cdots & \zeta_n \end{bmatrix}^*, \]

is a complex valued vector and \( \rho \) is a real scalar. Here \( z^* \) denotes the conjugate transpose of \( z \). As in Section 1, we assume that \( C \) is irreducible and \( \rho > 0 \). The eigenvalue decomposition of \( C \) is given by

\[ C = U \Lambda U^* \]

where \( \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^{n \times n} \) is a diagonal matrix of eigenvalues, and \( U = \begin{bmatrix} u_1 & u_2 & \cdots & u_n \end{bmatrix} \) is an unitary matrix of the corresponding eigenvectors.

To apply Algorithm 5 to the matrix \( C \) we first transform \( C \) to real symmetric DPR1 matrix \( A \) by diagonal unitary similarity:

\[ A = \Phi^* C \Phi = D + \rho |z| |z| T, \]

(31)

where

\[ \Phi = \text{diag} \left( \frac{\zeta_1}{|\zeta_1|}, \frac{\zeta_2}{|\zeta_2|}, \ldots, \frac{\zeta_n}{|\zeta_n|} \right). \]

We now compute the \( k \)-th eigenpair \((\lambda, v)\) of \( A \) by Algorithm 5, and set \( u = \Phi v \). Since we guarantee high relative accuracy of the eigenvalue decomposition of \( A \) computed by Algorithm 5, we also guarantee high relative accuracy of the eigenvalue decomposition of \( C \). Notice that, if double the working precision is needed to compute \( b \) in Algorithm 5, in order for the proof of Theorem 7 to hold, the moduli \(|\zeta_i|\) in (31) need to be computed in double the working precision, as well.

Similarly, for an irreducible real non-symmetric DPR1 matrix of the form

\[ G = D + \rho \hat{z} \hat{z}^T, \]

where \( \text{sign}(\hat{\zeta}_i) = \text{sign}(\zeta_i), \ i = 1, \ldots, n \), we define the diagonal matrix

\[ \Psi = \text{diag} \left( \sqrt{\frac{\text{sign}(\zeta_1)}{\zeta_1}}, \ldots, \sqrt{\frac{\text{sign}(\zeta_n)}{\zeta_n}} \right). \]

The matrix

\[ A = \Psi^{-1} G \Psi = D + \rho z z^T, \]

19
where \( \zeta_i = \sqrt{\zeta_i^2} \) is an irreducible DPR1 matrix. We now compute the \( k \)-th eigenpair \( (\lambda, v) \) of \( A \) by Algorithm 5. The eigenpair of \( G \) is then \( (\lambda, \Psi v) \). Since we guarantee high relative accuracy of the eigenvalue decomposition of \( A \), we also guarantee high relative accuracy of the eigenvalue decomposition of \( G \). Notice that, if double the working precision is needed to compute \( b \) in Algorithm 5, the elements \( \zeta_i \) need to be computed in double the working precision, as well.

5 Numerical Examples

We illustrate our algorithm with four numerically demanding examples. Examples 1 and 2 illustrate Algorithm 1, Example 3 illustrates the use of double precision arithmetic, and Example 4 illustrates an application to higher dimension.

Example 1. In this example quantities \( K_b \) from (20) are approximately 1 for all eigenvalues, so we guarantee that all eigenvalues and all components of their corresponding eigenvectors are computed with high relative accuracy by Algorithm 5, using only standard machine precision. Let \( A = D + zz^T \), where

\[
D = \text{diag} (10^{10}, 5 \cdot 10^{-3}, 0, -4 \cdot 10^{-3}, -5),
\]

\[
z = [10^{10} \ 1 \ 1 \ 10^{-7} \ 1 \ 1]^T.
\]

The eigenvalues computed by Matlab [17] routine \texttt{eig}, LAPACK routine \texttt{DLAED9}, and Algorithm 5 and Mathematica [22] with 100 digits of precision (properly rounded to 16 decimal digits), are, respectively:

\[
\lambda^{(\text{eig})} = 1.000000000100000 \cdot 10^{20}, 5.000000000000000 \cdot 10^{20}, 4.000000000000000 \cdot 10^{-3}, 1.66534536937735 \cdot 10^{-16}, 0, -25.000000000000000 \cdot 10^{-15},
\]

\[
\lambda^{(\text{dlaed9})} = 1.000000000100000 \cdot 10^{20}, 5.000000000000000 \cdot 10^{20}, 4.000000010000000 \cdot 10^{-3}, 1.000000023272195 \cdot 10^{-24}, -3.999999999999999 \cdot 10^{-15}, -4.999999999999999 \cdot 10^{-15},
\]

\[
\lambda^{(\text{dpr1eig,Math})} = 1.000000000100000 \cdot 10^{20}, 5.000000000000000 \cdot 10^{20}, 4.000000010000000 \cdot 10^{-3}, 9.999999999999999 \cdot 10^{-25}, -3.999999999999999 \cdot 10^{-15}, -4.999999999999999 \cdot 10^{-15}.
\]

We see that all eigenvalues computed by Algorithm 5 (including the tiniest ones), are exact to the machine precision. The eigenvalues computed by \texttt{DLAED9} are all accurate, except \( \lambda_4 \). The eigenvalues computed by \texttt{eig} are accurate according to the standard perturbation theory, but they have almost no relative accuracy.\(^6\) Due to the the accuracy of the computed eigenvalues, the eigenvectors computed by Algorithm 5 are componentwise

\(^6\)If, in the last column, the last digits computed by \texttt{dpr1eig} and Mathematica, respectively, differ, they are displayed in parentheses.

\(^7\)The displayed eigenvalues are the ones obtained by the command \([V,\text{Lambda}]=\text{eig}(A)\). The command \texttt{Lambda=eig(A)} produces different eigenvalues.
accurate up to machine precision, and therefore, orthogonal up to machine
precision. The eigenvectors computed by DLAED9 are also componentwise
accurate, except for \( v_4 \):

\[
\begin{array}{ccc}
\text{v}_4^{(dlaed9)} & \text{v}_4^{(dplaeg, Math)} \\
1.000000011586098 \cdot 10^{-17} & 9.9999998999989 \cdot 10^{-18} \\
2.000000023172195 \cdot 10^{-18} & 1.999999999800000 \cdot 10^{-18} \\
2.500000028965244 \cdot 10^{-15} & 2.49999999749999 \cdot 10^{-15} \\
-1.0000000000000000 \cdot 10^{-15} & -1.0000000000000000 \cdot 10^{-15} \\
-2.500000028965244 \cdot 10^{-15} & -2.49999999749999 \cdot 10^{-15} \\
-2.000000023172195 \cdot 10^{-18} & -1.999999999800000 \cdot 10^{-18}
\end{array}
\]

Example 2. In this example, despite very close diagonal elements, we again
guarantee that all eigenvalues and all components of their corresponding
eigenvectors are computed with high relative accuracy, without deflation.
Let \( A = D + zz^T \), where

\[
D = \text{diag}(1 + 40\varepsilon, 1 + 30\varepsilon, 1 + 20\varepsilon, 1 + 10\varepsilon), \\
z = \begin{bmatrix} 1 & 2 & 2 & 1 \end{bmatrix},
\]

and \( \varepsilon = 2^{-52} = 2\varepsilon_M \). For this matrix, the quantities \( K_b \) are again of order one for all eigenvalues, so Algorithm 5 uses only standard working precision.
The eigenvalues computed by Matlab, DLAED9, and Algorithm 5 are:

\[
\begin{array}{ccc}
\lambda^{(eig)} & \lambda^{(dlaed9)} & \lambda^{(dplaeg)} \\
11 + 32\varepsilon & 11 + 48\varepsilon & 11 + 16\varepsilon \\
1 + 38\varepsilon & 1 + 41\varepsilon & 1 + 39\varepsilon \\
1 + 31\varepsilon & 1 + 27\varepsilon & 1 + 25\varepsilon \\
1 + 8\varepsilon & 1 + 9\varepsilon & 1 + 11\varepsilon
\end{array}
\]

Notice that all computed eigenvalues are accurate according to standard perturbation theory. However, only the eigenvalues computed by Algorithm 5 satisfy the interlacing property. The eigenvalues computed by Mathematica with 100 digits of precision, properly rounded to 32 decimal digits are:

\[
\begin{array}{c}
\lambda^{(Math)} \\
11.000000000000005551115123125783 \\
1.00000000000000085712482686374087 \\
1.0000000000000055511151231257826 \\
1.0000000000000025309819717641565
\end{array}
\]

If, as suggested in Remark 2, Algorithm 5 is modified to return \( d_i \) and \( \mu \) (both in standard precision), then for the eigenvalues \( \lambda_2, \lambda_3 \) and \( \lambda_4 \) the corresponding pairs \( (d_i, \mu) \) give representations of those eigenvalues to 32 decimal digits. In our case, the exact values \( d_i + \mu \) properly rounded to 32 decimal digits are equal to the corresponding eigenvalues computed by Mathematica displayed above.

The eigenvectors \( v_2, v_3 \) and \( v_4 \) computed by Matlab span an invariant subspace of \( \lambda_2, \lambda_3 \) and \( \lambda_4 \), but their components are not accurate. Due
to the accuracy of the computed eigenvalues, the eigenvectors computed by Algorithm 5 are componentwise accurate up to the machine precision (they coincide with the eigenvectors computed by Mathematica with 100 digits precision), and are therefore orthogonal. Interestingly, in this example the eigenvectors computed by DLAED9 are also componentwise accurate, but there is no underlying theory for such high accuracy.

Example 3. In this example (see [9]) we can guarantee that all eigenvalues and eigenvectors will be computed with componentwise high relative accuracy only if \( b \) from (12) is for \( k \in \{2, 3, 4\} \) computed in double of the working precision. Let \( A = D + zz^T \), where

\[
D = \text{diag} (10/3, 2 + \beta, 2 - \beta, 1), \\
z = \begin{bmatrix} 2 & \beta & \beta \end{bmatrix}, \quad \beta = 10^{-7}.
\]

For \( k \in \{2, 3, 4\} \) the quantities \( \kappa \nu \) from (28) are of order \( O(10^7) \), so the element \( b \) in each of the matrices needs to be computed in double of the working precision. For example, for \( k = 2 \), the element \( b = (A^2)_{22} \) computed by Algorithm 2 in standard precision is equal to \( b = 5.749999751891721 \cdot 10^7 \), while Matlab routine \texttt{inv} gives \( b = 5.749999746046776 \cdot 10^7 \). Computing \( b \) in double of the working precision in Algorithm 2 gives the correct value \( b = 5.749999754927588 \cdot 10^7 \).

The eigenvalues computed by Matlab, DLAED9, Algorithm 5 and Mathematica with 100 digits precision, respectively, are all highly relatively accurate – they differ in the last or last two digits. However, the eigenvectors \( v_2, v_3 \) and \( v_4 \) computed by Algorithm 5, with double precision computation of \( b \)'s, are componentwise accurate to machine precision and therefore orthogonal. The eigenvectors computed by Matlab and DLAED9 are, of course, orthogonal, but are not componentwise accurate. For example,

\[
\begin{align*}
iv_2^{(\text{eig})} &= 2.088932176072975 \cdot 10^{-1} \\
iv_2^{(\text{dlaed9})} &= 2.088932143122528 \cdot 10^{-1} \\
iv_2^{(\text{dpr1eig,Math})} &= 2.088932138163857 \cdot 10^{-1} \\
-9.351941376557037 \cdot 10^{-1} &= -9.351941395201120 \cdot 10^{-1} \\
-6.48058628358029 \cdot 10^{-2} &= -6.480586288204153 \cdot 10^{-2} \\
-2.7852423414308628 \cdot 10^{-1} &= -2.785242297496694 \cdot 10^{-1} \\
\end{align*}
\]

Example 4. In this example we extend Example 3 to higher dimension, as in TEST 3 from [9, §6]. Here \( A = D + zz^T \in \mathbb{R}^{202 \times 202} \), where

\[
D = \text{diag} (1, 2 + \beta, 2 - \beta, 2 + 2\beta, 2 - 2\beta, \ldots, 2 + 100\beta, 2 - 100\beta, 10/3), \\
z = \begin{bmatrix} 2 & \beta & \beta \ldots & \beta \end{bmatrix}, \quad \beta \in \{10^{-3}, 10^{-8}, 10^{-15}\}.
\]

For each \( \beta \), we solved the eigenvalue problem with Algorithm 5 without using double the working precision, Algorithm 5, and DLAED9 from LAPACK. For \( \beta = 10^{-3} \), Algorithm 5 used double the working precision for computing 25
eigenvalues, and for $\beta = 10^{-8}$ and $\beta = 10^{-15}$ double the working precision was needed for all but the largest eigenvalue. As in [9] §6, for each algorithm we computed orthogonality and residue measures,

$$O = \max_{1 \leq i \leq n} \frac{\|V^T v_i - e_i\|_2}{n\varepsilon_M}, \quad R = \max_{1 \leq i \leq n} \frac{\|Av_i - \lambda_i v_i\|_2}{n\varepsilon_M\|A\|_2},$$

respectively. Here $V = [v_1 \ v_2 \ \cdots \ v_n]$ is the computed matrix of eigenvectors, and $e_i$ is the $i$-th column of the identity matrix.

Let $(\lambda(dpr1eig,nd), v(dpr1eig,nd))$, $(\lambda(dpr1eig), v(dpr1eig))$, and $(\lambda(dlaed9), v(dlaed9))$, denote the eigenpairs computed by Algorithm 5 without using double the working precision, Algorithm 5, and DLAED9, respectively. Since we proved the componentwise accuracy of eigenvectors computed by Algorithm 5, we take those as the ones of reference. Table 1 displays orthogonality measures, residue measures, relative errors in the computed eigenvalues and componentwise relative errors in the computed eigenvectors. From Table 1 we see

| $\beta$  | $10^{-3}$ | $10^{-8}$ | $10^{-15}$ |
|----------|-----------|-----------|------------|
| $O(dpr1eig,nd)$ | 1.47      | 5.8 $\cdot$ 10$^4$ | 2.1 $\cdot$ 10$^{11}$ |
| $O(dpr1eig)$ | 0.059     | 0.039     | 0.045      |
| $O(dlaed9)$ | 0.049     | 0.064     | 0.045      |
| $R(dpr1eig,nd)$ | 0.0086    | 0.033     | 0.0043     |
| $R(dpr1eig)$ | 0.0086    | 0.039     | 0.0043     |
| $R(dlaed9)$ | 0.029     | 0.03      | 0.013      |
| max $\frac{|\lambda(dpr1eig,nd) - \lambda(dpr1eig)|}{|\lambda(dpr1eig)|}$ | 2.2 $\cdot$ 10$^{-16}$ | 0         | 2.2 $\cdot$ 10$^{-16}$ |
| max $\frac{|\lambda(dlaed9) - \lambda(dpr1eig)|}{|\lambda(dpr1eig)|}$ | 1.5 $\cdot$ 10$^{-15}$ | 2.2 $\cdot$ 10$^{-16}$ | 0       |
| max $\frac{|v(dpr1eig,nd) - v(dpr1eig)|}{|v(dpr1eig)|}$ | 2.7 $\cdot$ 10$^{-13}$ | 2.8 $\cdot$ 10$^{-8}$ | 0.518   |
| max $\frac{|v(dlaed9) - v(dpr1eig)|}{|v(dpr1eig)|}$ | 2.2 $\cdot$ 10$^{-12}$ | 1.9 $\cdot$ 10$^{-8}$ | 0.043   |

Table 1: Orthogonality measures, residue measures, relative errors in computed eigenvalues, and componentwise relative errors in computed eigenvectors.

that all algorithms behave exactly as predicted by the theoretical analysis. All algorithms compute all eigenvalues to high relative accuracy because it is the same as normwise accuracy for this case. Algorithm 5 without use of double the working precision loses orthogonality as predicted by the respective condition numbers. The number of correct digits in the computed eigenvectors is approximately the same for Algorithm 5 without use of double the working precision and DLAED9, but there is no proof of such componentwise accuracy of the eigenvectors computed by DLAED9. When double the
working precision is used, the eigenvectors computed by Algorithm 5 are fully orthogonal, as a consequence of their componentwise accuracy.

Acknowledgment

We would like to thank Ren Cang Li for providing Matlab implementation of the LAPACK routine DLAED4 and its dependencies.

A Proofs

Proof of Theorem 1.

Let \( \tilde{\lambda} \) and \( \tilde{\mu} \) be defined by (17) and (18), respectively. Then

\[
\tilde{\lambda} \equiv f(l) (d_i + \tilde{\mu}) = (d_i + \tilde{\mu}) (1 + \varepsilon_1).
\]

By simplifying the equality

\[
(d_i + \mu (1 + \kappa_{\mu} \varepsilon_M)) (1 + \varepsilon_1) = \lambda (1 + \kappa_{\lambda} \varepsilon_M)
\]

and using \( \lambda = \mu + d_i \), we have

\[
d_i \varepsilon_1 + \mu (\kappa_{\mu} \varepsilon_M + \varepsilon_1) = \lambda \kappa_{\lambda} \varepsilon_M.
\]

Taking absolute value gives

\[
|\kappa_{\lambda}| \leq \frac{|d_i| + |\mu|}{|\lambda|} (|\kappa_{\mu}| + 1). \quad \Box
\]

Proof of Theorem 2.

(i) The assumption sign \( (d_i) = \text{sign} (\mu) \) immediately implies

\[
\frac{|d_i| + |\mu|}{|\lambda|} = \frac{|d_i + \mu|}{|d_i + \mu|} = 1.
\]

(ii) The assumptions imply that either

\[
0 < d_{i+1} < \lambda < d_i, \quad \mu < 0,
\]

or

\[
d_i < \lambda < d_{i-1} < 0, \quad \mu > 0.
\]

In the first case \( \lambda \) is closest to the pole \( d_i \) and

\[
\frac{|d_i| + |\mu|}{|\lambda|} \leq \frac{|d_i| + \frac{1}{2} |d_i - d_{i+1}|}{\frac{1}{2} |d_i + d_{i+1}|} \leq \frac{d_i + \frac{1}{2} d_i - \frac{1}{2} d_{i+1}}{\frac{1}{2} d_i + \frac{1}{2} d_{i+1}} \leq \frac{\frac{3}{2} d_i - \frac{1}{2} d_{i+1}}{\frac{1}{2} d_i + \frac{1}{2} d_{i+1}} \leq 3 d_i.
\]
Here we used the inequalities $|\mu| \leq \frac{1}{2} |d_i - d_{i+1}|$ and $|\lambda| \geq \frac{1}{2} |d_i + d_{i+1}|$ for the first inequality, $d_i - d_{i+1} > 0$ and $d_i + d_{i+1} > 0$ for the second inequality and $d_{i+1} > 0$ for the fourth inequality, respectively.

The proof for the second case is analogous. □

Proof of Theorem 3.

Let $x$ and $\tilde{x}$ be defined by (14) and (24), respectively. By using (18), for $\tilde{x}_i$ we have

$$\tilde{x}_i = ft\left(-\frac{\zeta_i}{\mu}\right) = -\frac{\zeta_i}{\mu(1 + \kappa \mu \varepsilon M)} (1 + \varepsilon_1) = x_i (1 + \varepsilon x_i),$$

and the first order approximation gives

$$|\varepsilon x_i| \leq (|\kappa \mu| + 1) \varepsilon M.$$ (32)

For $j \neq i$, by using (18), solving the equality

$$\tilde{x}_j = \frac{\zeta_j}{(d_j - d_i)(1 + \varepsilon_1) - \mu(1 + \kappa \mu \varepsilon M)(1 + \varepsilon_2)} (1 + \varepsilon_3) = \frac{\zeta_j}{d_j - \lambda} (1 + \varepsilon x)$$

for $\varepsilon_x$, using $\lambda = \mu + d_i$, and ignoring higher order terms, we have

$$\varepsilon_x = \frac{(d_j - d_i)(\varepsilon_1 + \varepsilon_2 + \varepsilon_3) - \mu(\kappa \mu \varepsilon M + \varepsilon_2 + \varepsilon_3)}{d_j - \lambda}.$$ (32)

Therefore,

$$|\varepsilon_x| \leq \frac{|d_j - d_i| + |\mu|}{|d_j - \lambda|} (|\kappa \mu| + 3) \varepsilon M.$$ (32)

To complete the proof we need to analyze two cases. If

$$\text{sign} (d_j - d_i) = -\text{sign} \mu,$$

then

$$\frac{|d_j - d_i| + |\mu|}{|d_j - \lambda|} = \frac{|d_j - d_i - \mu|}{|d_j - \lambda|} = \frac{|d_j - \lambda|}{|d_j - \lambda|} = 1.$$ (32)

If

$$\text{sign} (d_j - d_i) = \text{sign} \mu,$$

then, since $d_i$ is pole closest to $\lambda$, we have $|\mu| \leq 0.5 |d_j - d_i|$ and

$$\frac{|d_j - d_i| + |\mu|}{|d_j - \lambda|} \leq \frac{|d_j - d_i| + |\mu|}{|d_j - d_i| - |\mu|} \leq \frac{3}{2} |d_j - d_i| = 3.$$ (32)

Finally, the theorem follows by inserting this into (32). □
Proof of Theorem 4.

For the non-zero computed elements of the matrix $A^{-1}_i$ from (11) and (12) we have:

\[ fl([A^{-1}_i]_{jj}) = \frac{1}{(d_j - d_i)(1 + \varepsilon_1)} (1 + \varepsilon_2), \quad j \neq i, \]

\[ fl([A^{-1}_i]_{ji}) = fl([A^{-1}_i]_{ij}) = -\frac{\zeta_j}{(d_j - d_i)(1 + \varepsilon_3)} \frac{\zeta_i}{(1 + \varepsilon_4)} (1 + \varepsilon_5), \quad j \neq i, \]

where $|\varepsilon_k| \leq \varepsilon_M$ for all indices $k$. The first statement of the theorem now follows by using standard first order approximations.

Similar analysis of the formula (12) yields

\[ fl([A^{-1}_i]_{ii}) = \tilde{b} = b + \delta b, \]

where

\[ |\delta b| \leq \frac{1}{\zeta_i} \left( \frac{1}{\rho} + z_2^T D_1^{-1} z_1 - z_2^T D_2^{-1} z_2 \right) (n + 4)\varepsilon_M. \]  

(33)

This, in turn, implies (19) with

\[ |\kappa_b| \leq \frac{|\delta b|}{|b|} \leq \frac{1}{|\varepsilon_M|} = (n + 4)K_b, \]

where $K_b$ is defined by (20). □

Proof of Theorem 5.

Let

\[ \tilde{A}^{-1}_i = A^{-1}_i + \delta A^{-1}_i. \]

Therefore,

\[ |\tilde{\nu} - \nu| = ||\delta A^{-1}_i||_2, \]

which, together with (27), implies

\[ |\nu \kappa_b \varepsilon_M| \leq ||\delta A^{-1}_i||_2. \]  

(34)

Theorem 4 implies that

\[ ||\delta A^{-1}_i||_2 \leq (n + 4)||A^{-1}_i||_2 K_b \varepsilon_M. \]

Since $||A^{-1}_i||_2 \leq \sqrt{n}||A^{-1}_i||_2$ and $|\nu| = ||A^{-1}_i||_2$, from (34) we have

\[ |\kappa_{\nu}b| \leq (n + 4)\sqrt{n}K_b, \]  

(35)

which proves the first part of the bound (28).

For the second part of the proof, notice that Theorem 4 also implies
\[
\| \delta A_i^{-1} \|_2 \leq 3 \| A \|_2 \varepsilon_M + | \delta b |,
\]
where \( A \) is equal to the matrix \( A_i^{-1} \) without \( b \) (that is, with \( A_{ii} = 0 \)).

By bounding (34) with (36) and (33), and dividing the resulting inequality by \( | \nu \varepsilon_M | \), we have

\[
| \kappa_{\nu} | \leq 3 \sqrt{n} + (n + 4) \left( \frac{1}{| \nu |} + \frac{| 1/\rho |}{\zeta_i^2} + \frac{| z^T D_1^{-1} z_1 |}{\zeta_i^2} + \frac{| z_2^T D_2^{-1} z_2 |}{\zeta_i^2} \right).
\]

(37)

Since

\[
\frac{1/\rho}{\zeta_i^2} = \frac{1}{\zeta_i^2} \left( \frac{1}{\rho} + z_1^T D_1^{-1} z_1 + z_2^T D_2^{-1} z_2 - z_1^T D_1^{-1} z_1 - z_2^T D_2^{-1} z_2 \right)
\]

\[
\leq | b | + \frac{1}{\zeta_i^2} (| z_1^T D_1^{-1} z_1 | + | z_2^T D_2^{-1} z_2 |),
\]

from (37) it follows

\[
| \kappa_{\nu} | \leq 3 \sqrt{n} + (n + 4) \left( | b | + \frac{2}{| \nu |} \frac{| z_1^T D_1^{-1} z_1 | + | z_2^T D_2^{-1} z_2 |}{\zeta_i^2} \right).
\]

(38)

Since \( | b | \leq | \nu | \) and

\[
\| A_i^{-1} \|_2 = | \nu | = \max_{\| x \|_2 = 1} \| A_i^{-1} x \|_2 \geq \| A_i^{-1} e_k \|_2
\]

\[
= \sqrt{\frac{1}{(d_k - d_i)^2} + \frac{\zeta_k^2}{\zeta_i^2 (d_k - d_i)^2}} \geq \frac{| \zeta_k |}{| \zeta_i | d_k - d_i},
\]

by simply dividing each term

\[
\frac{\zeta_k^2}{\zeta_i^2 | d_k - d_i |}
\]

in (38) with the corresponding quotient

\[
\frac{| \zeta_k |}{| \zeta_i | d_k - d_i},
\]

we obtain

\[
| \kappa_{\nu} | \leq 3 \sqrt{n} + (n + 4) \left( 1 + \frac{2}{| \zeta_i |} \sum_{k=1}^{n} \frac{| \zeta_k |}{\zeta_i} \right).
\]

(39)

The bound (28) now follows from (35) and (39). \( \square \)
Proof of Theorem 6.

We first prove the bound (30). Since \( \tilde{\nu} = fl(\tilde{\nu}) \) is computed by bisection, from (25) we have
\[
\tilde{\nu} = \tilde{\nu}(1 + \kappa_{\text{bis}} \varepsilon_M).
\]
This and (27) imply
\[
\tilde{\nu} = \nu(1 + \kappa_{\nu} \varepsilon_M)(1 + \kappa_{\text{bis}} \varepsilon_M).
\]
Since \( \hat{\mu} = fl(1/\tilde{\nu}) \), the bound (30) follows by ignoring higher order terms. The bound (29) now follows by inserting (30) into Theorems 1 and 2. □

Proof of Theorem 7.

Let the assumptions of the theorem hold. Let \( b \) be computed in double of the working precision, \( \varepsilon_M^2 \), and then stored in the standard precision. The standard floating-point error analysis with neglecting higher order terms gives
\[
\frac{P (1 + \kappa_P \varepsilon_M)}{\zeta_i} - \frac{Q (1 + \kappa_Q \varepsilon_M)}{\zeta_i} = \frac{P - Q}{\zeta_i} (1 + \kappa_b \varepsilon_M) = b (1 + \kappa_b \varepsilon_M),
\]
where \(|\kappa_P|, |\kappa_Q| \leq (n + 2)\) and \(|\kappa_1| \leq 3\). Solving the above equality for \( \kappa_b \), neglecting higher order terms, and taking absolute values gives
\[
|\kappa_b| \leq \left( \frac{|P| + |Q|}{|P - Q|} \max \{|\kappa_P|, |\kappa_Q|\} + |\kappa_1| \right) \varepsilon_M \leq (K_b(n + 2) + 3) \varepsilon_M.
\]
Since, by assumption, \( K_b \leq O(1/\varepsilon_M) \), this implies
\[
|\kappa_b| \leq O(n),
\]
as desired. □

References

[1] E. Anderson et al., *LAPACK Users’ Guide*, SIAM 3rd ed., Philadelphia, (1999).

[2] J. L. Barlow, Error analysis of update methods for the symmetric eigenvalue problem, SIAM J. Matrix Anal. Appl., 14 (1993) 598-618.

[3] C. F. Borges, W. B. Gragg, A parallel Divide - and - Conquer Method for the Generalized Real Symmetric Definite Tridiagonal Eigenproblem, in *Numerical Linear Algebra and Scientific Computing*, L. Reichel, A. Ruttan and R. S. Varga, eds., de Gruyter, Berlin (1993) 11-29.
[4] J. R. Bunch and C. P. Nielsen, Rank-one modification of the symmetric eigenproblem, Numer. Math., 31 (1978) 31-48.

[5] J. J. M. Cuppen, A divide and conquer method for the symmetric tridiagonal eigenproblem, Numer. Math., 36 (1981) 177-195.

[6] T. J. Dekker, A floating-point technique for extending the available precision, Numer. Math., 18 (1971) 224-242.

[7] J. Dongarra and D. Sorensen, A fully parallel algorithm for the symmetric eigenvalue problem, SIAM J. Sci. Statist. Comput., 8 (1987) 139-154.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The John Hopkins University Press, Baltimore, 4th ed. (2013).

[9] M. Gu and S. C. Eisenstat, A stable and efficient algorithm for the rank-one modification of the symmetric eigenproblem, SIAM J. Matrix Anal. Appl., 15 (1994) 1266-1276.

[10] M. Gu and S. C. Eisenstat, A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem, SIAM J. Matrix Anal. Appl., 16 (1995) 79-92.

[11] N. Higham, *Accuracy and Stability of Numerical Algorithms*, Second Edition, SIAM, Philadelphia, 2002.

[12] Intel Fortran Compiler, [http://software.intel.com/en-us/fortran-compilers](http://software.intel.com/en-us/fortran-compilers)

[13] Intel Math Kernel Library, [http://software.intel.com/en-us/intel-mkl](http://software.intel.com/en-us/intel-mkl)

[14] N. Jakovčević Stor, I. Slapničar and J. L. Barlow, Accurate eigenvalue decomposition of real symmetric arrowhead matrices and applications, Lin. Alg. Appl., to appear, [http://dx.doi.org/10.1016/j.laa.2013.10.007](http://dx.doi.org/10.1016/j.laa.2013.10.007)

[15] R. C. Li, Solving secular equations stably and efficiently Tech. Report UCB/CSD-94851, Computer Science Division, University of California, Berkeley, CA (1994), Also: LAPACK Working Note 89.

[16] O. Livne and A. Brandt, N Roots of the secular equation in $O(N)$ operations, SIAM J. Matrix Anal. Appl., 24 (2002) 439453.

[17] *MATLAB. The MathWorks*, Inc., Natick, Massachusetts, USA, [http://www.mathworks.com](http://www.mathworks.com)

[18] A. Melman, Numerical solution of a secular equation, Numer. Math., 69 (1995) 483-493.
[19] D. P. O’Leary and G.W. Stewart, Computing the eigenvalues and eigenvectors of symmetric arrowhead matrices, J. Comput. Phys. 90, 2 (1990) 497-505.

[20] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, (1980).

[21] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, (1965).

[22] Wolfram Mathematica, Documentation Center,
http://reference.wolfram.com/mathematica/guide/Mathematica.html