An algorithm for symmetric generalized inverse eigenvalue problems

Hua Dai

Department of Mathematics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, People's Republic of China

Received 23 November 1998; accepted 26 April 1999

Submitted by P. Lancaster

Abstract

Using QR-like decomposition with column pivoting and least squares techniques, we propose a new and efficient algorithm for solving symmetric generalized inverse eigenvalue problems, and give its locally quadratic convergence analysis. We also present some numerical experiments which illustrate the behaviour of our algorithm. © 1999 Elsevier Science Inc. All rights reserved.

AMS classification: 65F15; 65H15

Keywords: Eigenvalue; Inverse problems; QR-like decomposition; Least squares; Gauss–Newton method

1. Introduction

Let $A(c)$ and $B(c)$ be real $n \times n$ twice continuously differentiable symmetric matrix-valued functions depending on $c = (c_1, \ldots, c_n)^T \in \mathbb{R}^n$, and $B(c)$ be positive definite whenever $c \in \Omega$, an open subset of $\mathbb{R}^n$. The generalized inverse eigenvalue problem (GIEP) under consideration is as follows.

1 The research was supported by the National Natural Science Foundation of China and the Jiangsu Province Natural Science Foundation.

0024-3795/99 – see front matter © 1999 Elsevier Science Inc. All rights reserved.

PII: S 0 0 2 4 - 3 7 9 5 ( 9 9 ) 0 0 1 0 9 - 3
GIEP. Given \( n \) real numbers \( \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n \), find \( c \in \mathbb{R}^n \) such that the symmetric generalized eigenvalue problem \( A(c)x = \lambda B(c)x \) has the prescribed eigenvalues \( \lambda_1, \lambda_2, \ldots, \lambda_n \).

The GIEP arises in various areas of applications, such as the discrete analogue of inverse Sturm–Liouville problems [12] and structural design [13]. A special case of the GIEP is obtained when \( A(c) \) and \( B(c) \) are defined by

\[
A(c) = A_0 + \sum_{j=1}^n c_j A_j, \quad B(c) = I,
\]

where \( A_0, A_1, \ldots, A_n \) are real symmetric \( n \times n \) matrices, and \( I \) is the identity matrix. This problem is known as the algebraic inverse eigenvalue problem. There is a large literature on various aspects of the existence theory as well as numerical methods for the problem. See, for example, [1–4,10,14,17,18,20,23–25,27–30]. The well-known additive and multiplicative inverse eigenvalue problems [8,9] are also included in our formulation by taking

\[
A(c) = A + \text{diag}(c_1, \ldots, c_n), \quad B(c) = I
\]

and

\[
A(c) = A, \quad B(c) = \text{diag}(c_1, \ldots, c_n),
\]

respectively, where \( A \) is an \( n \times n \) constant matrix, about which there has been considerable discussion. See, for example, [30] and the references contained therein.

Recently, Dai and Lancaster [5] considered another special case of the GIEP, i.e., \( A(c) \) and \( B(c) \) are the following affine families

\[
A(c) = A_0 + \sum_{i=1}^n c_i A_i, \quad B(c) = B_0 + \sum_{i=1}^n c_i B_i, \tag{1}
\]

where \( A_0, A_1, \ldots, A_n, B_0, B_1, \ldots, B_n \) are the real symmetric \( n \times n \) matrices, and \( B(c) \) is positive definite whenever \( c \in \Omega \). Let \( \lambda_1(c) \leq \lambda_2(c) \leq \cdots \leq \lambda_n(c) \) be the eigenvalues of the symmetric generalized eigenvalue problem \( A(c)x = \lambda B(c)x \). Then there is a well-defined map \( \hat{\lambda} : \Omega \rightarrow \mathbb{R}^n \) given by \( \hat{\lambda}(c) = (\lambda_1(c), \ldots, \lambda_n(c))^\top \). The numerical algorithm analysed in [5] is Newton’s method for solving the following nonlinear system

\[
\hat{\lambda}(c) - \hat{\lambda}^* = 0, \tag{2}
\]

where \( \hat{\lambda}^* = (\lambda_1, \ldots, \lambda_n)^\top \). Each step in the numerical solution by Newton’s method of the system (2), however, involves the complete solution of the generalized eigenvalue problem \( A(c)x = \lambda B(c)x \). Based on determinant evaluations originating with Lancaster [15] and Biegler-König [2], an approach to the GIEP has been studied in [7] as well, but it is not computationally attractive for real symmetric matrices. When \( \lambda_1, \ldots, \lambda_n \) include multiple eigenvalues,
however, the eigenvalues \( \lambda_1(c), \ldots, \lambda_n(c) \) of the generalized eigenvalue problem 
\( A(c)x = \lambda B(c)x \) are not, in general, differentiable [26] at a solution \( c^* \).
Furthermore, the eigenvectors are not unique, and they cannot generally be defined
to be continuous functions of \( c \) at \( c^* \). The modification to the GIEP has been
considered in [5], but the number of given eigenvalues and their multiplicities
must satisfy a certain condition in the modified problem.

In this paper, we propose a new, efficient and reliable algorithm for solving
the GIEP, which consists of extension of ideas developed by [6,19,20]. The
problem formulation and our algorithm do not need to be changed when
multiple eigenvalues are present. The algorithm is suitable for both the distinct
and multiple eigenvalue cases. Each step in the iterative process does not in-
volve any solution of the generalized eigenvalue problem \( A(c)x = \lambda B(c)x \).

The paper is organized as follows. In Section 2 we recall some necessary
differentiability theory for \( QR \)-like decomposition of a matrix depending on
several parameters. In Section 3 a new algorithm based on \( QR \)-like decom-
position with column pivoting and least squares techniques is described, and its
locally quadratic convergence analysis is given. In Section 4 some numerical
examples are presented to illustrate the behaviour of our algorithm.

For our consideration, we shall need the following notation. A solution to
the GIEP will always be denoted by \( c^* \). \( \| \cdot \|_2 \) denotes the Euclidean vector
norm or induced spectral norm, and \( \| \cdot \|_F \) the Frobenius matrix norm. For
an \( n \times m \) matrix \( A = [a_1, \ldots, a_m] \), where \( a_i \) is the \( i \)th column vector of \( A \), we
define a vector \( \text{col} \ A \) by \( \text{col} \ A = [a_1^T, \ldots, a_m^T]^T \), and the norm
\( \| A \|_W := \max_{j=1,\ldots,m} (\| a_j \|_2) \). The symbol \( \otimes \) denotes the Kronecker product of matrices.

2. QR-like decomposition and differentiability

Following Li [19], we first define a \( QR \)-like decomposition of a real \( n \times n \)
matrix \( A \) with index \( m(1 \leq m \leq n) \) to be a factorization

\[
A = QR, \quad R = \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix},
\]

where \( Q \in \mathbb{R}^{n \times n} \) is orthogonal, \( R_{11} \) is \((n-m) \times (n-m)\) upper triangular, and
\( R_{22} \) is \( m \times m \) square. When \( m = 1 \), this is just a classical \( QR \) decomposition of
\( A \). The existence of a \( QR \)-like decomposition is obvious. In fact, we need only
construct a “partial” \( QR \) decomposition, see [11], for example. In general,
however, it is not unique as the following theorem shows.

**Theorem 2.1** (See [6,19]). Let \( A \) be an \( n \times n \) matrix whose first \( n-m \) columns
are linearly independent and let \( A = QR \) be a \( QR \)-like decomposition with index
\( m \). Then \( A = \tilde{Q} \tilde{R} \) is also a \( QR \)-like decomposition with index \( m \) if and only if
\[ Q = \hat{Q}D, \quad R = D^T \hat{R}, \]

where \( D = \text{diag}(D_{11}, D_{22}) \), \( D_{11} \) is an orthogonal diagonal matrix, and \( D_{22} \) is an \( m \times m \) orthogonal matrix.

The following is a theorem on perturbation of \( QR \)-like decomposition.

**Theorem 2.2** (See [6]). Let \( C_1 \in \mathbb{R}^{n \times n} \) have its first \( n - m \) columns linearly independent and let \( C_1 = Q_1R_1 \) be a \( QR \)-like decomposition with index \( m \). Let \( C_2 \in \mathbb{R}^{n \times n} \) be any matrix satisfying \( \|C_1 - C_2\|_2 < \varepsilon \). Then, for sufficiently small \( \varepsilon \), \( C_2 \) has a \( QR \)-like decomposition with index \( m \), \( C_2 = Q_2R_2 \), such that

\[ \|Q_1 - Q_2\|_2 \leq \kappa_1 \varepsilon, \quad \|R_1 - R_2\|_2 \leq \kappa_2 \varepsilon \]

where \( \kappa_1, \kappa_2 \) are constants independent on \( C_2 \).

Note that the linear independence hypothesis ensures that the \( R_{11} \) blocks of \( R \) and \( \hat{R} \) are nonsingular. In order to make the submatrix \( R_{11} \) of \( QR \)-like decomposition nonsingular, we admit a permutation of the columns of matrix \( A \). The \( QR \)-like decomposition with column pivoting of \( A \in \mathbb{R}^{n \times n} \) may be expressed as

\[ AP = QR, \]

where \( P \) is an \( n \times n \) permutation matrix, and \( R \) is similar to Eq. (3). If \( \text{rank}(A) = n - m \), then the permutation matrix \( P \) can be chosen such that the first \( n - m \) columns of the matrix \( AP \) are linearly independent and the block upper triangular matrix \( R \) satisfies

\[ |e_1^T R e_1| \geq |e_2^T R e_2| \geq \cdots \geq |e_{n-m}^T R e_{n-m}| > 0, \quad R_{22} = 0. \]

Now let \( A(c) = (a_{ij}(c)) \in \mathbb{R}^{n \times n} \) be a continuously differentiable matrix-valued function of \( c \in \mathbb{R}^n \). Here, the differentiability of \( A(c) \) with respect to \( c \) means, for any \( c^{(0)} \in \mathbb{R}^n \), we have

\[ A(c) = A(c^{(0)}) + \sum_{j=1}^n \frac{\partial A(c^{(0)})}{\partial c_j} (c_j - c_j^{(0)}) + o(\|c - c^{(0)}\|_2), \]

where \( c = (c_1, \ldots, c_n)^T, c^{(0)} = (c_1^{(0)}, \ldots, c_n^{(0)})^T \), and

\[ \frac{\partial A(c^{(0)})}{\partial c_j} = \left( \frac{\partial a_{ij}(c^{(0)})}{\partial c_j} \bigg|_{c=c^{(0)}} \right). \]

Note that if \( A(c) \) is twice differentiable, then \( o(\|c - c^{(0)}\|_2) \) in (8) may be replaced by \( O(\|c - c^{(0)}\|_2^2) \).
The next result which follows from Theorem 3.2 in [6] concerns the existence of a locally smooth QR-like decomposition of $A(c)$.

**Theorem 2.3.** Let $A(c) \in \mathbb{R}^{n \times n}$ be twice continuously differentiable at $c^{(0)} \in \mathbb{R}^n$ and assume that $\text{rank}(A(c^{(0)})) \geq n - m$. Let $P$ be a permutation matrix such that the first $n - m$ columns of $A(c^{(0)})P$ are linearly independent, and

$$
A(c^{(0)})P = Q^{(0)}R^{(0)}, \quad R^{(0)} = \begin{pmatrix} R_{11}^{(0)} & R_{12}^{(0)} \\ 0 & R_{22}^{(0)} \end{pmatrix}
$$

be a QR-like decomposition of $A(c^{(0)})P$ with index $m$. Then there exists a neighbourhood $N(c^{(0)})$ of $c^{(0)}$ in $\mathbb{R}^n$ such that, for any $c \in N(c^{(0)})$, there is a QR-like decomposition of $A(c)P$ with index $m$

$$
A(c)P = Q(c)R(c), \quad R(c) = \begin{pmatrix} R_{11}(c) & R_{12}(c) \\ 0 & R_{22}(c) \end{pmatrix}
$$

with the following properties:
1. $Q(c^{(0)}) = Q^{(0)}$, $R(c^{(0)}) = R^{(0)}$.
2. All elements of $Q(c)$ and $R(c)$ are continuous in $N(c^{(0)})$.
3. $R_{22}(c)$ and the diagonal elements $r_{jj}(c)$, $j = 1, \ldots, n - m$, of $R_{11}(c)$ are continuously differentiable at $c^{(0)}$. Moreover, if we write

$$
Q^{(0)^T} \frac{\partial A(c^{(0)})}{\partial c_j} P = \begin{pmatrix} A_{j,11} & A_{j,12} \\ A_{j,21} & A_{j,22} \end{pmatrix}, \quad A_{j,11} \in \mathbb{R}^{(n-m)\times(n-m)}, \quad j = 1, \ldots, n
$$

then

$$
R_{22}(c) = R_{22}^{(0)} + \sum_{j=1}^{n} \left( A_{j,22} - A_{j,21} R_{11}^{(0)^{-1}} R_{12}^{(0)} \right) \left( c_j - c_j^{(0)} \right) + O\left( \|c - c^{(0)}\|_2^2 \right).
$$

3. **A new algorithm**

We first present a new formulation of the GIEP, which is an extension of ideas developed in [6,19,20]. For convenience, we assume that only the first eigenvalue is multiple, with multiplicity $m$, i.e.,

$$
\lambda_1 = \cdots = \lambda_m < \lambda_{m+1} < \cdots < \lambda_n.
$$

There is no difficulty in generalizing all our results to an arbitrary set of given eigenvalues.
Compute $QR$-like decomposition with column pivoting of $A(c) - \lambda_1 B(c)$ with index $m$

$$
(A(c) - \lambda_1 B(c))P_1(c) = Q_1(c)R_1(c), \quad R_1(c) = \begin{pmatrix}
R_{11}^{(1)}(c) & R_{12}^{(1)}(c) \\
0 & R_{22}^{(1)}(c)
\end{pmatrix},
$$

(14)

where $R_{11}^{(1)}(c) \in \mathbb{R}^{(n-\lambda \times (n-\lambda)}$, and $n - m$ $QR$ decompositions with column pivoting of $A(c) - \lambda_i B(c)(i = m + 1, \ldots, n)$

$$
(A(c) - \lambda_i B(c))P_i(c) = Q_i(c)R_i(c), \quad R_i(c) = \begin{pmatrix}
R_{11}^{(i)}(c) & R_{12}^{(i)}(c) \\
0 & r_{nn}^{(i)}(c)
\end{pmatrix},
$$

(15)

$$
i = m + 1, \ldots, n,
$$

where $R_{11}^{(i)}(c) \in \mathbb{R}^{(n-\lambda \times (n-\lambda)}$.

We assume permutation matrices $P_i(c) \in \mathbb{R}^{n \times n} (i = 1, m + 1, \ldots, n)$ are constant matrices in a sufficiently small neighbourhood of $c$ for each $i$. If column pivoting is performed such that

$$
\det(R_{11}^{(1)}(c)) \neq 0, \quad |e_1^T R_1(c)e_1| \geq |e_2^T R_1(c)e_2| \geq \cdots \geq |e_n^T R_1(c)e_n| \geq \|R_{22}^{(1)}(c)\|_F
$$

(16)

and

$$
|e_1^T R_i(c)e_1| \geq |e_2^T R_i(c)e_2| \geq \cdots \geq |e_n^T R_i(c)e_n| = |r_{nn}^{(i)}(c)|, \quad i = m + 1, \ldots, n
$$

(17)

then the symmetric generalized eigenvalue problem $A(c)x = \lambda B(c)x$ has the eigenvalues $\lambda_1, \lambda_{m+1}, \ldots, \lambda_n$ in which $\lambda_1$ is a multiple eigenvalue with multiplicity $m$ if and only if

$$
R_{22}^{(1)}(c) = 0, \quad r_{nn}^{(i)}(c) = 0, \quad i = m + 1, \ldots, n.
$$

(18)

Thus we consider solving the following least squares problem

$$
\text{minimize } F(c)
$$

with

$$
F(c) = \frac{1}{2} \left\{ \|R_{22}^{(1)}(c)\|_F^2 + \sum_{i=m+1}^n (r_{nn}^{(i)}(c))^2 \right\}.
$$

(19)

In fact, the GIEP has a solution $c^*$ precisely when the function $F(c)$ defined by (19) has the minimal value zero at $c^*$.

It is worth noting that $F(c)$ may not be uniquely determined for any $c$ because of the nonuniqueness of $QR$-like and $QR$ decompositions. However, we
shall show that such nonuniqueness does not affect the effectiveness of our algorithm (see Theorem 3.1).

If \( m = 1 \), i.e., the given eigenvalues \( \lambda_1, \lambda_2, \ldots, \lambda_n \) are distinct, we may consider solving the nonlinear system

\[
\begin{pmatrix}
    r_{nn}^{(1)}(c) \\
    r_{nn}^{(2)}(c) \\
    \vdots \\
    r_{nn}^{(n)}(c)
\end{pmatrix} = 0.
\]  

(20)

The formulation (20) has been studied by Li [20] in the case of the algebraic inverse eigenvalue problem. If \( m = 1 \) and a solution of the GIEP exists, (19) and (20) are equivalent.

Let \( c^{(k)} \) be sufficiently close to \( c^* \). It follows from Theorem 2.3 that the matrix-valued function \( R_{nn}^{(i)}(c) \) and \( n - m \) functions \( r_{nn}^{(i)}(c)(i = m + 1, \ldots, n) \) are continuously differentiable at \( c^{(k)} \), and \( R_{nn}^{(i)}(c) \) and \( r_{nn}^{(i)}(c)(i = m + 1, \ldots, n) \) can be expressed as

\[
R_{nn}^{(1)}(c) = R_{nn}^{(1)}(c^{(k)}) + \sum_{j=1}^{n} \frac{\partial R_{nn}^{(1)}(c^{(k)})}{\partial c_j} (c_j - c_j^{(k)}) + O\left(\|c - c^{(k)}\|_2^2\right)
\]

\[
= R_{nn}^{(1)}(c^{(k)}) + \sum_{j=1}^{n} \left[ T_{j,21}^{(1)}(c^{(k)}) - T_{j,21}^{(i)}(c^{(k)})R_{11}^{(i-1)}(c^{(k)})R_{12}^{(i)}(c^{(k)}) \right] (c_j - c_j^{(k)})
\]

\[
+ O\left(\|c - c^{(k)}\|_2^2\right)
\]

(21)

\[
r_{nn}^{(i)}(c) = r_{nn}^{(i)}(c^{(k)}) + \sum_{j=1}^{n} \frac{\partial r_{nn}^{(i)}(c^{(k)})}{\partial c_j} (c_j - c_j^{(k)}) + O\left(\|c - c^{(k)}\|_2^2\right)
\]

\[
= r_{nn}^{(i)}(c^{(k)}) + \sum_{j=1}^{n} \left[ T_{j,21}^{(i)}(c^{(k)}) - t_{j,21}^{(i)}(c^{(k)})R_{11}^{(i-1)}(c^{(k)})R_{12}^{(i)}(c^{(k)}) \right] (c_j - c_j^{(k)})
\]

\[
+ O\left(\|c - c^{(k)}\|_2^2\right), \quad i = m + 1, \ldots, n,
\]  

(22)

where

\[
Q_i^T(c) \left[ \frac{\partial A(c)}{\partial c_j} - \lambda_j \frac{\partial B(c)}{\partial c_j} \right] P_i(c) = \begin{pmatrix}
    T_{j,11}^{(i)}(c) & T_{j,12}^{(i)}(c) \\
    T_{j,21}^{(i)}(c) & T_{j,22}^{(i)}(c)
\end{pmatrix}, \quad T_{j,11}^{(i)}(c) \in \mathbb{R}^{(n-m) \times (n-m)},
\]

\[
Q_i^T(c) \left[ \frac{\partial A(c)}{\partial c_j} - \lambda_j \frac{\partial B(c)}{\partial c_j} \right] P_i(c) = \begin{pmatrix}
    T_{j,11}^{(i)}(c) & t_{j,12}^{(i)}(c) \\
    t_{j,21}^{(i)}(c) & t_{j,22}^{(i)}(c)
\end{pmatrix}, \quad T_{j,11}^{(i)}(c) \in \mathbb{R}^{(n-1) \times (n-1)}.
\]  

(23)
Let

\[
f(c) = \begin{pmatrix}
\operatorname{col} R_{22}^{(1)}(c) \\
\vdots \\
\operatorname{col} R_{mn}^{(n)}(c)
\end{pmatrix}.
\]

Then

\[
F(c) = \frac{1}{2} f^T(c) f(c).
\]

We apply the Gauss–Newton method (see [21]) to solve the least squares problem (19). By use of (21) and (22), one step of Gauss–Newton method for the solution of (19) has the following form

\[
J_f^T(c^{(k)}) J_f(c^{(k)}) (c^{(k+1)} - c^{(k)}) = -J_f^T(c^{(k)}) f(c^{(k)}),
\]

where

\[
J_f(c) = \begin{pmatrix}
\frac{\partial R_{22}^{(1)}(c)}{\partial c_1} & \ldots & \frac{\partial R_{22}^{(1)}(c)}{\partial c_n} \\
\vdots & \ddots & \vdots \\
\frac{\partial R_{mn}^{(m+1)}(c)}{\partial c_1} & \ldots & \frac{\partial R_{mn}^{(m+1)}(c)}{\partial c_n}
\end{pmatrix}
\]

with

\[
\frac{\partial R_{22}^{(1)}(c)}{\partial c_j} = T_{j,22}^{(1)}(c) - T_{j,21}^{(1)}(c) R_{11}^{(1)}(c) R_{12}^{(1)}(c),
\]

\[
\frac{\partial R_{mn}^{(m+1)}(c)}{\partial c_j} = t_{j,22}^{(i)}(c) - t_{j,21}^{(i)}(c) R_{11}^{(i)}(c) R_{12}^{(i)}(c), \quad i = m + 1, \ldots, n.
\]

Thus the new method for solving the GIEP may be summarized as follows.

**Algorithm 3.1.**

1. Choose an initial approximation \( c^{(0)} \) to \( c^* \), and for \( k = 0, 1, 2 \ldots \)
2. Compute \( A(c^{(k)}) - \lambda_i B(c^{(k)}) (i = 1, m + 1, \ldots, n) \) and

\[
\frac{\partial A(c^{(k)})}{\partial c_j} - \lambda_i \frac{\partial B(c^{(k)})}{\partial c_j}, \quad (i = 1, m + 1, \ldots, n, \quad j = 1, \ldots, n).
\]
3. Compute QR-like decomposition with column pivoting of \( A(c^{(k)}) - \lambda_i B(c^{(k)}) \) with index \( m \):
\[(A(c^{(k)}) - \lambda_i B(c^{(k)}))P_i(c^{(k)}) = Q_i(c^{(k)})R_i(c^{(k)}),\]

\[R_i(c^{(k)}) = \begin{pmatrix} R_{11}^{(i)}(c^{(k)}) & R_{12}^{(i)}(c^{(k)}) \\ 0 & R_{22}^{(i)}(c^{(k)}) \end{pmatrix},\]

and \(n - m\) QR decompositions with column pivoting of \(A(c^{(k)}) - \lambda_i B(c^{(k)})(i = m + 1, \ldots, n)\):

\[(A(c^{(k)}) - \lambda_i B(c^{(k)}))P_i(c^{(k)}) = Q_i(c^{(k)})R_i(c^{(k)}),\]

\[R_i(c^{(k)}) = \begin{pmatrix} R_{11}^{(i)}(c^{(k)}) & R_{12}^{(i)}(c^{(k)}) \\ 0 & r_{ii}^{(i)}(c^{(k)}) \end{pmatrix}.\]

4. If

\[\sqrt{\|R_{22}^{(i)}(c^{(k)})\|_F^2 + \sum_{i=m+1}^{n} (r_{ii}^{(i)}(c^{(k)}))^2}\]

is small enough stop; Otherwise.

5. Form \(f_i(c^{(k)})\) and \(J_i(c^{(k)})\) using (24) and (27).

6. Find \(c^{(k+1)}\) by solving linear system (26).

7. Go to 2.

We first compare the computational requirements of Algorithm 3.1 and algorithms in [5]. Since in all the algorithms for solving GIEP, Step 2 and 6 are indispensable, our comparison does not include the computational requirements for Step 2 and 6. It is well-known that the QR decomposition with column pivoting for each \((A(c^{(k)}) - \lambda_i B(c^{(k)}))\) requires \(2/3n^3\) flops, while the QR-like decomposition with column pivoting of \((A(c^{(k)}) - \lambda_i B(c^{(k)}))\) with index \(m\) requires about \(2/3n^2(n - m)\) flops (see, for example, [11]). Therefore Step 3 requires approximately \(2/3(n^3 + n^2)(n - m)\) flops. It is easy to verify that Step 5 requires approximately \(n^4\) flops. Thus Algorithm 3.1 requires approximately \(n^4 + 2/3(n^3 + n^2)(n - m)\) flops per iteration, while algorithms in [5] require only about \(n^4 + 8n^3\) flops per iteration. However, our numerical experiments showed that Algorithm 3.1 took generally less iterations than algorithms in [5]. In the next section we will comment on the numerical behaviour of the algorithms.

Now we prove that the iterates \(c^{(k)}\) generated by Algorithm 3.1 do not vary with different QR-like decompositions of \((A(c^{(k)}) - \lambda_i B(c^{(k)}))P_i(c^{(k)})\) and different QR decompositions of \((A(c^{(k)}) - \lambda_i B(c^{(k))})P_i(c^{(k)}) (i = m + 1, \ldots, n)\).
Theorem 3.1. In Algorithm 3.1, for any fixed $k$, suppose

$$(A(c^{(k)}) - \lambda_1 B(c^{(k)}))P_1(c^{(k)}) = Q_1(c^{(k)})R_1(c^{(k)}) = \tilde{Q}_1(c^{(k)})\tilde{R}_1(c^{(k)}),$$

are two (different) QR-like decompositions with column pivoting of $A(c^{(k)}) - \lambda_1 B(c^{(k)})$ with index $m$, and

$$(A(c^{(k)}) - \lambda_i B(c^{(k)}))P_i(c^{(k)}) = Q_i(c^{(k)})R_i(c^{(k)}) = \tilde{Q}_i(c^{(k)})\tilde{R}_i(c^{(k)}),$$

are two (different) QR decompositions with column pivoting of $A(c^{(k)}) - \lambda_i B(c^{(k)}) (i = m + 1, \ldots, n)$, $J_f(c^{(k)})$, $f(c^{(k)})$ and $J_f(c^{(k)})$, $\tilde{f}(c^{(k)})$ are obtained in Step 5 of Algorithm 3.1 corresponding to two different decompositions of (29) and (30). Then

$$J_f^T(c^{(k)})J_f(c^{(k)}) = \tilde{J}_f^T(c^{(k)})\tilde{J}_f(c^{(k)}),$$

$$J_f^T(c^{(k)})f(c^{(k)}) = \tilde{J}_f^T(c^{(k)})\tilde{f}(c^{(k)}).$$

Proof. It follows from Theorem 2.1 that there exist a partitioned orthogonal matrix $D_1 = \text{diag}(D_{11}, D_{22})$ where $D_{11}$ is an orthogonal diagonal matrix, $D_{22}$ is an $m \times m$ orthogonal matrix, and $n - m$ orthogonal diagonal matrices $D_i = \text{diag}(\delta_1^{(i)}, \ldots, \delta_n^{(i)})$ where $\delta_j^{(i)} = \pm 1 (i = m + 1, \ldots, n)$ such that

$$Q_1(c^{(k)}) = \tilde{Q}_1(c^{(k)})D_1, \quad R_1(c^{(k)}) = D_1^T\tilde{R}_1(c^{(k)}),$$

$$Q_i(c^{(k)}) = \tilde{Q}_i(c^{(k)})D_i, \quad R_i(c^{(k)}) = D_i^T\tilde{R}_i(c^{(k)}), \quad i = m + 1, \ldots, n.$$  

By use of (23), (28) and (33), we have

$$R_{22}^{(i)}(c^{(k)}) = D_{22}^T\tilde{R}_{22}^{(i)}(c^{(k)}), \quad \frac{\partial R_{22}^{(i)}(c^{(k)})}{\partial c_j} = D_{22}^T\frac{\partial \tilde{R}_{22}^{(i)}(c^{(k)})}{\partial c_j}, \quad j = 1, \ldots, n,$$

$$r_m^{(i)}(c^{(k)}) = \delta_n^{(i)}\tilde{r}_m^{(i)}(c^{(k)}), \quad \frac{\partial r_m^{(i)}(c^{(k)})}{\partial c_j} = \delta_n^{(i)}\frac{\partial \tilde{r}_m^{(i)}(c^{(k)})}{\partial c_j}, \quad i = m + 1, \ldots, n.$$  

From (34) and the properties of the Kronecker product (see [16]), we obtain

$$\text{col} \frac{\partial R_{22}^{(i)}(c^{(k)})}{\partial c_j} = (I \otimes D_{22}^T) \text{col} \frac{\partial \tilde{R}_{22}^{(i)}(c^{(k)})}{\partial c_j}, \quad j = 1, \ldots, n.$$  

$$\text{col} \frac{\partial \tilde{R}_{22}^{(i)}(c^{(k)})}{\partial c_j} = (I \otimes D_{22}^T) \text{col} \frac{\partial \tilde{R}_{22}^{(i)}(c^{(k)})}{\partial c_j}, \quad j = 1, \ldots, n.$$
From (24), (27), (34) and (35), the matrices \( J_f(c^{(k)}) \), \( \tilde{J}_f(c^{(k)}) \) and the vectors \( f(c^{(k)}) \), \( \tilde{f}(c^{(k)}) \) obtained in Algorithm 3.1 satisfy

\[
J_f(c^{(k)}) = \text{diag}(I \otimes D_{22}^{T}, \delta_n^{(m+1)}, \ldots, \delta_n^{(n)}) \tilde{J}_f(c^{(k)}), \tag{36}
\]

\[
f(c^{(k)}) = \text{diag}(I \otimes D_{22}^{T}, \delta_n^{(m+1)}, \ldots, \delta_n^{(n)}) \tilde{f}(c^{(k)}). \tag{37}
\]

Hence (31) and (32) hold. \( \square \)

Although \( QR \)-like and \( QR \) decompositions are not unique, Theorem 3.1 shows that the iterates \( c^{(k)} \) generated by Algorithm 3.1 do not depend on the used decompositions.

**Theorem 3.2.** Suppose that the GIEP has a solution \( c^* \), and that in Algorithm 3.1 \( P_i(c^{(k)}) = P_i(c^*)(i = 1, m + 1, \ldots, n) \) are independent on \( k \) when \( \|c^* - c^{(k)}\|_2 \) is sufficiently small. Assume also that \( J_f(c^*) \in \mathbb{R}^{(m^2 + n - m) \times n} \) corresponding to a \( QR \)-like decomposition of \( (A(c^*) - \lambda_i B(c^*))P_i(c^*) \) with index \( m \) and to \( QR \) decompositions of \( (A(c^*) - \lambda_i B(c^*))P_i(c^*)(i = m + 1, \ldots, n) \) is of full rank. Then Algorithm 3.1 is locally quadratically convergent.

**Proof.** First form the \( QR \)-like decomposition of \( (A(c^*) - \lambda_i B(c^*))P_i(c^*) \) with index \( m \) and \( n - m \) \( QR \) decompositions of \( (A(c^*) - \lambda_i B(c^*))P_i(c^*) \) \( (i = m + 1, \ldots, n) \)

\[
(A(c^*) - \lambda_i B(c^*))P_i(c^*) = Q_i(c^*)R_i(c^*), \quad (i = m + 1, \ldots, n). \tag{38}
\]

Note that the matrix \( J_f(c^*) \) corresponding to the decompositions (38), by assumption, has full rank, and that \( J_f(c^*) \) is invertible.

Assuming that \( \|c^* - c^{(k)}\|_2 \) is sufficiently small, we can form a \( QR \)-like decomposition of \( (A(c^{(k)}) - \lambda_i B(c^{(k)}))P_i(c^*) \) with index \( m \) and \( n - m \) \( QR \) decompositions of \( (A(c^{(k)}) - \lambda_i B(c^{(k)}))P_i(c^*) \) \( (i = m + 1, \ldots, n) \)

\[
(A(c^{(k)}) - \lambda_i B(c^{(k)}))P_i(c^*) = \tilde{Q}_i(c^{(k)})\tilde{R}_i(c^{(k)}), \quad (i = m + 1, \ldots, n). \tag{39}
\]

It follows from Theorem 2.2 that

\[
\|Q_i(c^*) - \tilde{Q}_i(c^{(k)})\|_2 \leq \kappa_1^{(i)} \varepsilon, \quad i = 1, m + 1, \ldots, n, \tag{40}
\]

\[
\|R_i(c^*) - \tilde{R}_i(c^{(k)})\|_2 \leq \kappa_2^{(i)} \varepsilon, \quad i = 1, m + 1, \ldots, n.
\]
where 
\[ \varepsilon = \max_{i=1,m+1,\ldots,n} \left\{ \| [A(c^*^i) - A(c^{(k)})] - \lambda_i [B(c^*^i) - B(c^{(k)})] P_i(c^*) \|_2 \right\}. \]

Corresponding to the decompositions (39), we obtain a matrix \( \tilde{J}_f(c^{(k)}) \in \mathbb{R}^{m^2n+m\times n} \). From the definition of \( J_f(c) \) and (40) we know that 
\[ \|J_f(c^*) - \tilde{J}_f(c^{(k)})\|_2 \] 
is sufficiently small, and so is 
\[ \|J_f^T(c^*) J_f(c^*) - \tilde{J}_f^T(c^{(k)}) \tilde{J}_f(c^{(k)})\|_2 \], 
when \( c^{(k)} \) is sufficiently close to \( c^* \). Therefore, \( \tilde{J}_f^T(c^{(k)}) \tilde{J}_f(c^{(k)}) \) is invertible, \( \tilde{J}_f(c^{(k)}) \) has full rank, and \( \|\tilde{J}_f(c^{(k)})\|_2 \) is bounded.

The \( QR \)-like decomposition and \( n-m \) \( QR \) decompositions obtained in Algorithm 3.1 at \( c^{(k)} \) are not necessarily (39). Write them

\[ \begin{align*}
(A(c^{(k)}) - \hat{\lambda}_i B(c^{(k)}))P_i(c^*) &= Q_i(c^{(k)}) R_i(c^{(k)}), \\
(A(c^{(k)}) - \hat{\lambda}_i B(c^{(k)}))P_i(c^*) &= Q_i(c^{(k)}) R_i(c^{(k)}) \quad (i = m+1, \ldots, n).
\end{align*} \tag{41} \]

It follows from Theorem 4.1 and (36) that \( J_f(c^{(k)}) \) corresponding to the decompositions (41) also has full rank, \( \|J_f^T(c^*) J_f(c^*) - J_f^T(c^{(k)}) J_f(c^{(k)})\|_2 \) is sufficiently small, and \( \|J_f(c^{(k)})\|_2 \) is bounded if \( \|c^* - c^{(k)}\|_2 \) is small enough. Using the perturbation theory for the inversion of a matrix (see, for example, [22]), we have

\[ \left\| \left[ J_f^T(c^{(k)}) J_f(c^{(k)}) \right]^{-1} \right\|_2 \leq \left\| \left[ J_f^T(c^*) J_f(c^*) \right]^{-1} \right\|_2 + \omega(\varepsilon) \tag{42} \]

for the sufficiently small \( \|c^* - c^{(k)}\|_2 \), where \( \omega(\varepsilon) \geq 0 \) is a continuous function of \( \varepsilon \) and \( \omega(0) = 0 \).

Now we use Theorem 2.3 to extend smoothly the decompositions (41) to a neighbourhood of \( c^{(k)} \) which may be assumed to include \( c^* \). Then, abbreviating (21) and (22)

\[ R^{(1)}_{22}(c^*) = R^{(1)}_{22}(c^{(k)}) + \sum_{j=1}^{n} \frac{\partial R^{(1)}_{22}(c^{(k)})}{\partial c_j} (c_j - c^{(k)}) + O(\|c^* - c^{(k)}\|^2_2), \]

\[ r^{(i)}_{nn}(c^*) = r^{(i)}_{nn}(c^{(k)}) + \sum_{j=1}^{n} \frac{\partial r^{(i)}_{nn}(c^{(k)})}{\partial c_j} (c_j - c^{(k)}) + O(\|c^* - c^{(k)}\|^2_2). \]

But \( R^{(1)}_{22}(c^*) = 0, r^{(i)}_{nn}(c^*) = 0 \) \( (i = m+1, \ldots, n) \), and we have

\[ f(c^{(k)}) + J_f(c^{(k)})(c^* - c^{(k)}) = O(\|c^* - c^{(k)}\|^2_2). \]

Since \( \|J_f(c^{(k)})\|_2 \) is bounded, then

\[ J_f^T(c^{(k)}) J_f(c^{(k)})(c^* - c^{(k)}) = -J_f^T(c^{(k)}) f(c^{(k)}) + O(\|c^* - c^{(k)}\|^2_2). \tag{43} \]

Comparing (43) with Eq. (26) defining \( c^{(k+1)} \), we have

\[ J_f^T(c^{(k)}) J_f(c^{(k)})(c^* - c^{(k+1)}) = O(\|c^* - c^{(k)}\|^2_2). \]
It follows from this and (42) that
\[ \|c^* - c^{(k+1)}\|_2 = O(\|c^* - c^{(k)}\|_2^2). \]
as required. \(\square\)

4. Numerical experiments

Now we present some of our numerical experiments with Algorithm 3.1, and also give a numerical comparison between Algorithm 3.1 and algorithms in [5] for our examples. The following tests were made on a SUN workstation. Double precision arithmetic was used throughout. The starting points were chosen close to the solution, so that few iterations were required for convergence. We were interested in verifying that locally quadratic convergence takes place in practice, in both the distinct and multiple eigenvalue cases. The iterations were stopped when the norm \(\|f(c^{(k)})\|_2\) was less than \(10^{-9}\). For convenience, all vectors will be written as row-vectors. When specifying a symmetric matrix we will only write its upper triangular part.

**Example 4.1.** This is a generalized inverse eigenvalue problem with distinct eigenvalues. Let \(n = 5\),

\[ A_0 = \text{diag}(9, 11, 10, 8, 14), \quad B_0 = \text{diag}(11, 13, 15, 11, 10), \quad A_1 = B_1 = I, \]

\[
A_2 = \begin{pmatrix}
0 & 2 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}, \quad B_2 = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix},
\]

\[
A_3 = \begin{pmatrix}
0 & 0 & 3 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}, \quad B_3 = \begin{pmatrix}
0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix},
\]
The eigenvalues are prescribed to be

\[ \lambda^* = (0.43278721102, 0.66366274839, 0.94385900467, 1.10928454002, 1.49235323254) \]

With the starting point \( c^{(0)} = (1.25, 1.15, 1.05, 0.95, 0.85) \), Algorithm 3.1 converges to a solution

\[ c^* = (1, 1, 1, 1, 1) \]

and the results are displayed in Table 1.

**Example 4.2.** This is a generalized inverse eigenvalue problem with multiple eigenvalues. Let \( n = 6 \).

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Algorithm 3.1</th>
<th>Algorithm 2.1 in [5]</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k )</td>
<td>( | c^* - c^{(k)} |_2 )</td>
<td>( | f(c^{(k)}) |_2 )</td>
</tr>
<tr>
<td>0</td>
<td>0.34E + 00</td>
<td>0.83E + 00</td>
</tr>
<tr>
<td>1</td>
<td>0.14E + 00</td>
<td>0.51E - 01</td>
</tr>
<tr>
<td>2</td>
<td>0.15E - 02</td>
<td>0.49E - 03</td>
</tr>
<tr>
<td>3</td>
<td>0.40E - 06</td>
<td>0.81E - 07</td>
</tr>
<tr>
<td>4</td>
<td>0.14E - 12</td>
<td>0.68E - 14</td>
</tr>
</tbody>
</table>
\[
A_0 = \begin{pmatrix}
216 & 889.2 & -135 & -245.4 & 141.6 & 858.12 \\
66.3 & -483.75 & -820.39 & 413.12 & 1598.29 \\
204.375 & -2.425 & -131.5 & -541.325 \\
158.495 & -367.365 & 693.021 \\
209.035 & 154.057 \\
316.63
\end{pmatrix},
\]

\[B_0 = I\]

\[B_1 = \text{diag}(43.2222995816, 43.5245534248, 43.2978630424, 43.6484775005, 43.3099531961, 43.6635901927)\]

\[u_1 = \begin{pmatrix} 12 \\ -1 \\ 0.5 \\ 0.2 \\ 0.1 \end{pmatrix}, \quad u_2 = \begin{pmatrix} 0 \\ 12 \\ 0.5 \\ 0.2 \\ 0 \end{pmatrix}, \quad u_3 = \begin{pmatrix} 0 \\ 12 \\ -1 \\ 0.5 \\ -0.4 \end{pmatrix}, \quad u_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad u_5 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 12 \\ -1 \end{pmatrix}, \quad u_6 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},\]

\[V = \begin{pmatrix}
0 & -0.8467500959 & 0.4233750479 & -0.3387000384 & 0.1693500192 & -0.0846750096 \\
0 & -0.2246513095 & 2.8755367617 & -0.0898605238 & 1.1232565475 \\
0 & -0.9004508446 & 0.4502254223 & -0.3579661145 \\
0 & -0.6097095231 & 6.8750004851 \\
0 & -0.6911965633
\end{pmatrix},\]

\[A_k = u_k u_k^T, \quad B_k = \sum_{j=k}^{6} v_{k-1,j} (e_{k-1} e_j^T + e_j e_{k-1}^T), \quad k = 1, \ldots, 6,\]

\[A(c) = A_0 + \sum_{i=1}^{6} c_i A_i, \quad B(c) = B_0 + \sum_{i=1}^{6} c_i B_i.\]

The eigenvalues are prescribed by

\[\lambda^* = (1.5, 1.5, 1.5, 3.23032033004, 19.3004616769, 33.8547258306).\]
With the starting point $c^{(0)} = (3, 14, 3, 14, 1, 18)$, Algorithm 3.1 and Algorithm 2.1 in [5] converge the locally unique solution

$$
c^* = (3.30847736896, 14.1718318758, 2.2256714243, 13.5487684559, 0.95127266015, 17.6794860524).
$$

Choosing the target eigenvalues $\lambda^* = (1.5, 1.5, 1.5)$ and using the same starting point $c^{(0)}$, Algorithm 5.1 in [5] also converges the same solution $c^*$. Table 2 displays the residuals.

**Example 4.3.** This is also an example with multiple eigenvalues. Let $n = 8$, $A_0 = 0$, $B_0 = \text{diag}(99, 103, 100, 105, 101, 107, 101, 108)$, $B_1 = I$,

$$
A = \begin{pmatrix}
7400 & 11820 & -12060 & 1850 & -3020 & 460 & 710 & -2326 \\
27176 & -29158 & 7100 & -6116 & 2928 & 1878 & -4992.8 \\
32914 & -8005 & 6318 & -3854 & -1759 & 5681.4 \\
2950 & -1274 & 1466 & 827 & -1106 \\
2144 & -166 & 6 & 1334 \\
1523 & 593 & -351.2 \\
990 & 12.8 \\
1408.76
\end{pmatrix}
$$

$$
B = \begin{pmatrix}
0 & -20 & 10 & -10 & -10 & 10 & 0 & -10 \\
0 & -20 & 12 & 2 & -12 & 0 & 2 \\
0 & -21 & 9 & -9 & -10 & 9 \\
0 & -1 & 10 & 2 & -11 \\
0 & -22 & 9 & -8 \\
0 & -1 & 9 \\
0 & -22 \\
0
\end{pmatrix}
$$

Now we define the matrices $\{A_k\}$, $\{B_k\}$, $A(c)$ and $B(c)$ from $A$ and $B$ as follows:

$$
A_k = \sum_{j=1}^{k-1} a_{kj} \left( e_k e_j^T + e_j e_k^T \right) + a_{kk} e_k e_k^T, \quad k = 1, \ldots, 8,
$$

$$
B_k = \sum_{j=k}^{8} b_{kj} \left( e_{k-1} e_j^T + e_j e_{k-1}^T \right), \quad k = 2, \ldots, 8,
$$
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$|e^* - c^{(k)}|_2$</td>
<td>$|f(c^{(k)})|_2$</td>
<td>$|e^* - c^{(k)}|_2$</td>
</tr>
<tr>
<td>0</td>
<td>0.10E + 01</td>
<td>0.62E + 03</td>
<td>0.10E + 01</td>
</tr>
<tr>
<td>1</td>
<td>0.10E + 00</td>
<td>0.26E + 02</td>
<td>0.79E + 00</td>
</tr>
<tr>
<td>2</td>
<td>0.97E - 03</td>
<td>0.74E - 01</td>
<td>0.26E - 01</td>
</tr>
<tr>
<td>3</td>
<td>0.62E - 06</td>
<td>0.42E - 05</td>
<td>0.13E - 01</td>
</tr>
<tr>
<td>4</td>
<td>0.94E - 11</td>
<td>0.86E - 10</td>
<td>0.48E - 05</td>
</tr>
<tr>
<td>5</td>
<td>0.88E - 11</td>
<td>0.71E - 11</td>
<td>0.28E - 01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$|e^* - c^{(k)}|_2$</td>
<td>$|f(c^{(k)})|_2$</td>
<td>$|e^* - c^{(k)}|_2$</td>
</tr>
<tr>
<td>0</td>
<td>0.28E - 01</td>
<td>0.11E + 04</td>
<td>0.28E - 01</td>
</tr>
<tr>
<td>1</td>
<td>0.36E - 02</td>
<td>0.39E + 01</td>
<td>0.64E - 01</td>
</tr>
<tr>
<td>2</td>
<td>0.87E - 04</td>
<td>0.18E - 01</td>
<td>0.29E - 02</td>
</tr>
<tr>
<td>3</td>
<td>0.11E - 08</td>
<td>0.15E - 06</td>
<td>0.24E - 04</td>
</tr>
<tr>
<td>4</td>
<td>0.93E - 14</td>
<td>0.31E - 10</td>
<td>0.20E - 08</td>
</tr>
<tr>
<td>5</td>
<td>0.18E - 13</td>
<td>0.47E - 12</td>
<td></td>
</tr>
</tbody>
</table>
\[ A(c) = A_0 + \sum_{i=1}^{n} c_i A_i, \quad B(c) = B_0 + \sum_{i=1}^{n} c_i B_i. \]

The given eigenvalues are
\[ \lambda^* = (2, 2, 2, 3.1207536089, 10.2188681786, 18.2813657932, 36.7082186443, 723.6807937750). \]

With the starting point \( c^{(0)} = (0.99, 0.99, 0.99, 0.99, 1.01, 1.01, 1.01, 1.01) \), Algorithm 3.1 and Algorithm 2.1 in [5] converge to the exact solution
\[ c^* = (1, 1, 1, 1, 1, 1, 1, 1). \]

Specifying one eigenvalue of multiplicity 3 and 2 distinct eigenvalues, i.e.,
\[ \lambda^* = (2, 2, 2, 3.1207536089, 10.2188681786) \]
and using the same starting point \( c^{(0)} \), the solution found by Algorithm 5.1 in [5] is precisely \( c^* \). The results are displayed in Table 3.

**Example 4.4.** In this example we consider the usefulness of the GIEP in structural design. Fig. 1 shows a 10-bar truss structure where Young's modulus \( E = 6.95 \times 10^{10} \) N/m², weight density \( P = 2650 \) kg/m³, \( l = 10 \) m, acceleration of gravity \( g = 9.81 \) m/s², nonstructural mass at all nodes \( m_0 = 425 \) kg.

The design parameters are the areas of cross section of the bars. Since the number of design parameters exceeds the order of the global stiffness and mass matrices by two, the areas of cross section of the 7th and 8th bars are fixed with the values 0.000865 m² and 0.000165 m², respectively. The global stiffness and mass matrices of the structure can be expressed, respectively as

![Fig. 1. 10-bar Truss structure.](image-url)
\[ A(c) = A_0 + \sum_{i=1}^{8} c_i A_i, \quad B(c) = B_0 + \sum_{i=1}^{8} c_i B_i \]

where \( A_0, B_0, A_i \) and \( B_i (i = 1, \ldots, 8) \) are \( 8 \times 8 \) symmetric matrices, \( c_1, \ldots, c_8 \) are the areas of cross section of the bars 1-6, 9, 10, respectively.

The frequencies of the structure are prescribed to be \( \omega_j = 5j (j = 1, \ldots, 8) \), i.e., the given eigenvalues are \( \lambda_j = (2\pi\omega_j)^2 \) (\( j = 1, \ldots, 8 \)). With the starting point \( c^{(0)} = 10^{-3}(1.7, 0.4, 1.7, 1.6, 0.7, 1.1, 0.4, 1.1) \), Algorithm 3.1 and Algorithm 2.1 in [5] converge to a solution

\[ c^* = 10^{-3}(1.7024, 0.4203, 1.6994, 1.6010, 0.7056, 1.1164, 0.4391, 1.1192). \]

The results are displayed in Table 4.

These examples and our other numerical experiments with Algorithm 3.1 indicate that quadratic convergence indeed occurs in practice. We observed in most of our tests that Algorithm 3.1 took less iterations than algorithms in [5].

**Acknowledgements**

The author would like to thank Professor Peter Lancaster for his careful reading of a preliminary version of this paper. The author is also indebted to the referees for their comments and suggestions.

**References**


