An efficient global optimization algorithm for maximizing the sum of two generalized Rayleigh quotients

Xiaohui Wang1 · Longfei Wang2 · Yong Xia2

Received: 19 May 2017 / Revised: 8 January 2018 / Accepted: 9 January 2018 / Published online: 29 January 2018
© SBMAC - Sociedade Brasileira de Matemática Aplicada e Computacional 2018

Abstract Maximizing the sum of two generalized Rayleigh quotients (SRQ) can be reformulated as a one-dimensional optimization problem, where the function value evaluations are reduced to solving semi-definite programming (SDP) subproblems. In this paper, we first use the dual SDP subproblem to construct an explicit overestimation and then propose a branch-and-bound algorithm to globally solve (SRQ). Numerical results demonstrate that it is even more efficient than the recent SDP-based heuristic algorithm.

Keywords Fractional programming · Rayleigh quotient · Semidefinite programming · Branch and bound

Mathematics Subject Classification 90C32 · 90C26 · 90C22

Communicated by Ernesto G. Birgin.

This research was supported by NSFC under Grants 11571029, 11471325 and 11771056, and by fundamental research funds for the Central Universities under Grant YWF-17-BJ-Y-52.

✉ Yong Xia
dearyxia@gmail.com
Xiaohui Wang
xhwang@buaa.edu.cn
Longfei Wang
kingdflying999@126.com

1 School of Astronautics, Beihang University, Beijing 100191, People’s Republic of China
2 LMIB of the Ministry of Education, School of Mathematics and System Sciences, Beihang University, Beijing 100191, People’s Republic of China
1 Introduction

The problem of maximizing the sum of two generalized Rayleigh quotients

$$\text{(SRQ)} \quad \max_{x \neq 0} \frac{x^T B x}{x^T W x} + \frac{x^T D x}{x^T V x}$$  \hspace{1cm} (1)

with positive definite matrices $W$ and $V$ has gained attention due to recent applications in the multi-user MIMO system by Primolevo et al. (2006) and the sparse Fisher discriminant analysis in pattern recognition (see Dundar et al. 2005; Fung and Michael 2007; Wu et al. 2009). Without loss of generality, we can assume that $V$ is the identity matrix. Otherwise, we reformulate (1) as a problem in terms of $y$ by substituting $x = V^{-\frac{1}{2}} y$. Moreover, since the objective function in (1) is homogeneous of degree 0, (SRQ) can be further recast as the following sphere-constrained optimization problem, which was proposed by Zhang (2013, 2014):

$$\text{(P)} \quad \max f(x) = \frac{x^T B x}{x^T W x} + x^T D x$$

s.t. $\|x\| = 1$,

where $\| \cdot \|$ denotes the $\ell_2$-norm throughout this paper. Problem (P) is a special case of the quadratically constrained sum-of-quadratic ratios recently studied by Jiao and Liu (2017).

The single generalized Rayleigh quotient optimization problem [(i.e., (SRQ) with $B = 0$)] is related to the classical eigenvalue problem and solved in polynomial time, see Parlett (1998). However, to our best knowledge, whether the general (SRQ) [or (P)] can be efficiently solved in polynomial time remains open. Actually, as shown in Example 1.1 due to Zhang (2013), there could exist a few local non-global maximizers of (P). Moreover, even finding the critical points of (P) is nontrivial, see Zhang (2013, 2014).

When $B$, $W$ and $D$ are all diagonal matrices, (P) can be recast as a problem that minimizes the sum of a linear function and a linear ratio over the standard simplex, by introducing new nonnegative variables $y_i$ to replace $x_i^2$ for $i = 1, \ldots, n$, respectively. This special case is known as the sum-of-linear-ratios problem in the literature and there are many algorithms, see, for example, Freund and Jarre (2001), Jiao and Liu (2015), Schaible and Shi (2003). However, Matsui (1996) has shown that minimizing the sum of two linear ratios over a polyhedron is already NP-hard.

Recently, Nguyen et al. (2016) reformulated (P) as the problem of maximizing the following one-dimensional function:

$$\text{(P1)} \quad \max_{\mu \in [\underline{\mu}, \bar{\mu}]} q(\mu) := \mu + g(\mu),$$  \hspace{1cm} (2)

where $g(\mu)$ is related to a non-convex quadratic optimization problem:

$$g(\mu) = \max x^T D x$$

s.t. $\|x\| = 1$

$$x^T (B - \mu W)x \geq 0$$  \hspace{1cm} (3)

and the lower and upper bounds

$$\underline{\mu} = \min_{\|x\| = 1} \frac{x^T B x}{x^T W x}, \quad \bar{\mu} = \max_{\|x\| = 1} \frac{x^T B x}{x^T W x}$$  \hspace{1cm} (4)

are the smallest and the largest generalized eigenvalues of the matrix pencil $(B, W)$, respectively. To solve the one-dimensional problem (2), a “two-stage” heuristic algorithm is...
proposed in Nguyen et al. (2016) by first subdividing \([\mu, \bar{\mu}]\) into coarse intervals such that each one contains a local maximizer of \(q(\mu)\) and then applying the quadratic fit line search used by Antoniou and Lu (2007), Bazarraa et al. (2006), Luenberger and Ye (2008) in each interval. For any given \(\mu\), \(g(\mu)\) (or \(q(\mu)\)) can be evaluated by solving an equivalent semi-definite programming (SDP) formulation, according to an extended version of S-Lemma in Proposition 4.1 due to Polyak (1998), see also Theorem 5.17 in Pólik and Terlaky (2007). Finally, for the returned optimal solution \(\mu^*\), an optimal vector solution of \((P)\) is recovered by a rank-one decomposition procedure, see Theorem 3 in Nguyen et al. (2016) for more details.

Though this “two-stage” algorithm can find the global solutions of the tested examples, it is still a heuristic algorithm since the function \(q(\mu)\) is not guaranteed to be quasi-concave. Besides, there is no meaningful stopping criterion for the “two-stage” algorithm. That is, we cannot estimate the gap between the obtained solution and the global maximizer of \((P_1)\).

In this paper, we propose an easy-to-evaluate function for upper bounding \(q(\mu)\). It provides new upper bounds for \(g(\mu)\) over \([\mu, \bar{\mu}]\), which are used to establish an efficient branch-and-bound algorithm. Numerical results show that the new algorithm is more efficient than the “two-stage” heuristic algorithm developed by Nguyen et al. (2016).

The remainder of this paper is organized as follows. In Sect. 2, we give some preliminaries on the evaluation of \(g(\mu)\). In Sect. 3, we propose an easy-to-compute upper bounding function, which provides new upper bounds for \(g(\mu)\). In Sect. 4, we establish a new branch-and-bound algorithm and estimate the worst-case computational complexity. In Sect. 5, we do numerical comparison experiments, which demonstrate the efficiency of our new algorithm. Conclusions are made in Sect. 6.

Throughout the paper, \(v(\cdot)\) denotes the optimal objective value of the problem \((\cdot)\). We write \(A \succeq 0\) (resp., \(A \preceq 0\)) to denote that \(A\) is a positive (resp., negative) semi-definite matrix. We write \(A > 0\) to denote that \(A\) is positive definite. Let \(\lambda_{\max}(A)\) and \(\lambda_{\min}(A)\) be the maximal and minimal eigenvalues of \(A\), respectively. The inner product of two matrices \(A\) and \(B\) is denoted by \(A \bullet B = \text{trace}(AB^T)\). For a real number \(a\), \(\lfloor a \rfloor\) returns the largest integer less than or equal to \(a\).

2 Preliminaries

In this section, we first show how to evaluate \(g(\mu)\). Then, we present the “two-stage” algorithm proposed by Nguyen et al. (2016) to maximize \(q(\mu)\). Finally, we discuss how to obtain an optimal vector solution of \((P)\) from the maximizer of \(q(\mu)\).

Lifting \(xx^T\) to \(X \in \mathbb{R}^{n \times n}\) (since \(x^T Ax = A \bullet (xx^T)\)) yields the primal SDP relaxation of the optimization problem to evaluate \(g(\mu)\) for any given \(\mu\):

\[
\begin{align*}
\text{(SDP}_\mu) \quad & \max D \bullet X \\
\text{s.t.} \quad & I \bullet X = 1 \\
& (B - \mu W) \bullet X \geq 0 \\
& X \succeq 0.
\end{align*}
\]

The conic dual problem of \((\text{SDP}_\mu)\) is

\[
\begin{align*}
\text{(SD}_\mu) \quad & \min v \\
\text{s.t.} \quad & D - v I + \eta (B - \mu W) \preceq 0 \\
& \eta \geq 0,
\end{align*}
\]

which coincides with the Lagrangian dual problem of \(g(\mu)\).
An efficient global optimization algorithm... 4415

It is trivial to see that \((\text{SDP}_\mu)\) has an interior feasible solution, i.e., Slater’s condition holds. We can verify that, for any \(\mu\) satisfying
\[
\mu \leq \bar{\mu}, 
\tag{5}
\]
Slater’s condition holds for \((\text{SDP}_\mu)\), i.e., there is an \(X > 0\) such that \(I \cdot X = 1\) and \((B - \mu W) \cdot X > 0\). Therefore, under the assumption (5), strong duality holds for \((\text{SDP}_\mu)\), that is, \(v(\text{SDP}_\mu) = v(\text{SDP}_\mu^*)\) and both optimal values are attained.

Under the assumption (5), by further applying the extended version of S-Lemma, see Proposition 4.1 in Polyak (1998) or Theorem 5.17 in Pólik and Terlaky (2007), we can show that strong duality holds for the optimization problem of evaluating \(g(\mu)\), i.e., \(g(\mu) = v(\text{SDP}_\mu)\). More details can be found in Nguyen et al. (2016).

Next, we present the “two-stage” algorithm proposed by Nguyen et al. (2016) for solving (2). Firstly, it subdivides \([\mu, \bar{\mu}]\) into a rather coarse mesh and then collects all subintervals containing an interior local maximizer. In the second stage, the quadratic fit method used by Bazarra et al. (2006), Antoniou and Lu (2007), Luenberger and Ye (2008) is applied to find a corresponding local maximizer in each subinterval that has been collected in the first stage. Finally, the optimal solution \(\mu^*\) is selected from all these obtained local maximizers. In this paper, we will not present the detailed quadratic fit line search subroutine, which can be found in Nguyen et al. (2016). One of the reasons is that the algorithm in the first stage is already quite time-consuming.

The “two-stage” scheme proposed by Nguyen et al. (2016)

Step 1. Given \(\delta > 0\). Let \(\mu_0 = \mu\) and \(\mu_i = \mu + (i - 1)\delta\) for \(i = 1, 2, \ldots, \lfloor \frac{\bar{\mu} - \mu}{\delta} \rfloor + 1\). If \(\frac{\bar{\mu} - \mu}{\delta}\) is not an integer, set \(\mu_k = \bar{\mu}\) for \(k = \lfloor \frac{\bar{\mu} - \mu}{\delta} \rfloor + 2\).

Step 2. For \(i = 1, 2, \ldots\), collect all the three-point pattern \([\mu_{i-1}, \mu_i, \mu_{i+1}]\) such that \(\max\{q(\mu_{i-1}), q(\mu_{i+1})\} \leq q(\mu_i)\).

Step 3. Call the quadratic fit line search subroutine (with a smaller tolerance than \(\delta\)) to find a corresponding local maximizer in each three-point pattern \([\mu_{i-1}, \mu_i, \mu_{i+1}]\).

Step 4. Select the best maximizer \(\mu^*\) among \(\mu, \bar{\mu}\), and all the local maximizers found in Step 3.

Suppose (2) is solved, let \(\mu^*\) be the returned maximizer. If \(\mu^* = \bar{\mu}\), the feasible region of (3) is reduced to
\[
\|x\| = 1, \ (B - \mu^* W)x = 0,
\]
which contains only the unit eigenvectors corresponding to the maximal eigenvalue. In this case, \(g(\mu^*)\) is actually a maximum eigenvalue problem. On the other hand, suppose \(\mu^* < \bar{\mu}\), an optimal vector solution of (P) is recovered from the equivalent \((\text{SDP}_{\mu^*})\) based on the rank-one constraint, using a rank-one procedure similar to that in Sturm and Zhang (2003), Ye and Zhang (2003), see details in Nguyen et al. (2016).

There is an alternative approach to recover an optimal solution of (P). Let \(x^*\) be an optimal solution of (P). Then \(x^*\) and \(x^* x^* T\) are optimal solutions of \(g(\mu^*)\) and \((\text{SDP}_{\mu^*})\), respectively. Let \((v^*, \eta^*)\) be an optimal solution of the dual problem \((\text{SDP}_{\mu^*})\). Based on the strong duality between \((\text{SDP}_{\mu^*})\) and \((\text{SDP}_{\mu^*})\), it is not difficult to verify that
\[
\eta^* \left( (B - \mu^* W) \cdot (x^* x^* T) \right) = 0 \text{ (complementarity)},
\]
\[
g(\mu^*) = \max_{\|x\| = 1} x^T (D + \eta^* (B - \mu^* W)) x = \lambda_{\text{max}} (D + \eta^* (B - \mu^* W)).
\]
Therefore, we obtain
\[ x^* T (D + \eta^* (B - \mu^* W)) x^* = x^* T Dx^* = g(\mu^*) = \lambda_{\text{max}}(D + \eta^* (B - \mu^* W)). \]

Notice that \(|x^*| = 1\). It turns out that \(x^*\) is a unit-eigenvector corresponding to the maximum eigenvalue of \(D + \eta^* (B - \mu^* W)\). Let \(\{u_1, \ldots, u_k\}\) be an orthonormal basis of the eigenspace corresponding to the maximum eigenvalue of \(D + \eta^* (B - \mu^* W)\). Then, any unit-vector \(x\) in the range space of \(U = [u_1, \ldots, u_k]\) satisfying \(x^T (B - \mu^* W)x \geq 0\) is an optimal vector solution of (P). It is sufficient to solve the following maximum eigenvalue problem:
\[
\max_{x = U y, \|y\| = 1} x^T (B - \mu^* W)x = \max_{\|y\| = 1} y^T U (B - \mu^* W) U y = \lambda_{\text{max}}(U^T (B - \mu^* W) U).
\]

### 3 New upper bounds

In this section, we propose an easy-to-evaluate upper bounding function, which provides new upper bounds for \(q(\mu)\) over \([\mu, \tilde{\mu}]\).

Let \(\bigcup_{i=1}^{k}[\mu_i, \mu_{i+1}]\) be a subdivision of \([\mu, \tilde{\mu}]\), where \(\mu_1 = \mu\) and \(\mu_{k+1} = \tilde{\mu}\).

Consider the interval \([\mu_i, \mu_{i+1}]\) with \(i \leq k - 1\) (so that \(\mu_{i+1} < \tilde{\mu}\)). Solve (SD\(_\mu\)) with \(\mu = \mu_i, \mu_{i+1}\) and denote the optimal solutions by \((v_i, \eta_i)\) and \((v_{i+1}, \eta_{i+1})\), respectively. Then, we have \(\eta_i \geq 0, \eta_{i+1} \geq 0,\) and
\[
q(\mu_i) = \mu_i + v_i, \quad q(\mu_{i+1}) = \mu_{i+1} + v_{i+1}.
\]

For any \(\mu \in [\mu_i, \mu_{i+1}]\), it follows from strong duality that
\[
q(\mu) = \mu + \min_{\|x\| = 1} \max_{\|x\| = 1} x^T D x + \eta_i (x^T (B - \mu W)x)
\]
\[
\leq \mu + \max_{\|x\| = 1} x^T D x + \eta_i (x^T (B - \mu W)x)
\]
\[
= \mu_i + \max_{\|x\| = 1} \{x^T D x + \eta_i (x^T (B - \mu W)x) + \mu - \mu_i + \eta_i (\mu_i - \mu) x^T W x \}
\]
\[
\leq q(\mu_i) + \mu - \mu_i + \max_{\|x\| = 1} \eta_i (\mu_i - \mu) x^T W x
\]
\[
= q(\mu_i) + \mu - \mu_i + \eta_i (\mu_i - \mu) \min_{\|x\| = 1} x^T W x
\]
\[
= q(\mu_i) + (1 - \eta_i \lambda_{\text{min}}(W))(\mu - \mu_i)
\]
\[
=: q_1(\mu). \tag{6}
\]

Similarly, we have
\[
q(\mu) \leq q(\mu_{i+1}) + (1 - \eta_{i+1} \lambda_{\text{max}}(W))(\mu - \mu_{i+1}) =: q_2(\mu). \tag{8}
\]

Now, we obtain an upper bounding function of \(q(\mu)\) over \([\mu_i, \mu_{i+1}]\):
\[
\tilde{q}(\mu) = \min\{q_1(\mu), q_2(\mu)\}, \tag{9}
\]
which is a concave function as \(q_1(\mu)\) and \(q_2(\mu)\) are both linear functions. It provides the following upper bound of \(q(\mu)\) over \([\mu_i, \mu_{i+1}]\):
\[
U_i = \max_{\mu \in [\mu_i, \mu_{i+1}]} \tilde{q}(\mu). \tag{10}
\]

Problem (10) is a convex program. Moreover, it has a closed-form solution.
Theorem 1 Under the assumption $\mu_{i+1} < \bar{\mu}$, an upper bound of $q(\mu)$ over $[\mu_i, \mu_{i+1}]$ is given by

$$U_i = \begin{cases} 
q(\mu_i), & \text{if } 1 - \eta_i \lambda_{\min}(W) \leq 0, \\
q(\mu_{i+1}), & \text{if } 1 - \eta_{i+1} \lambda_{\max}(W) \geq 0, \\
q_1(\mu_0), & \text{otherwise,}
\end{cases}$$

(11)

where

$$\mu_0 = \frac{q(\mu_{i+1}) - \mu_{i+1} + \eta_{i+1} \mu_{i+1} \lambda_{\max}(W) - q(\mu_i) + \mu_i - \eta_i \mu_i \lambda_{\min}(W)}{\eta_{i+1} \lambda_{\max}(W) - \eta_i \lambda_{\min}(W)}.$$ 

Proof We first assume $1 - \eta_i \lambda_{\min}(W) \leq 0$. For any $\mu \in [\mu_i, \mu_{i+1}]$, we have

$$q(\mu) \leq q_1(\mu) = q(\mu_i) + (1 - \eta_i \lambda_{\min}(W))(\mu - \mu_i) \leq q(\mu_i).$$

(12)

Suppose it holds that $1 - \eta_{i+1} \lambda_{\max}(W) > 0$, we obtain

$$q(\mu) \leq q_2(\mu) < q(\mu_{i+1}), \quad \forall \mu \in [\mu_i, \mu_{i+1}).$$

(13)

Then, we have the following contradiction:

$$\frac{13}{12} < q(\mu_{i+1}) \leq q(\mu_i).$$

Therefore, it must hold that $1 - \eta_{i+1} \lambda_{\max}(W) \leq 0$. Since both $q_1(\mu)$ and $q_2(\mu)$ are decreasing, according to the definition (9), $\bar{q}(\mu)$ is also a decreasing function. It follows from (10) and (8) that

$$U_i = \bar{q}(\mu_i) = \min\{q_1(\mu_i), q_2(\mu_i)\} = \min\{q(\mu_i), q_2(\mu_i)\} = q(\mu_i).$$

Similarly, under the assumption $1 - \eta_{i+1} \lambda_{\max}(W) \geq 0$, we can show that $\bar{q}(\mu)$ is increasing and then

$$U_i = \bar{q}(\mu_{i+1}) = \min\{q_1(\mu_{i+1}), q_2(\mu_{i+1})\} = \min\{q_1(\mu_{i+1}), q(\mu_{i+1})\} = q(\mu_{i+1}).$$

The last case is $1 - \eta_i \lambda_{\min}(W) > 0$ and $1 - \eta_{i+1} \lambda_{\max}(W) < 0$. It implies that $q_1(\mu)$ and $q_2(\mu)$ are strictly increasing and decreasing, respectively. The maximizer of $\bar{q}(\mu)$, denoted by $\mu_0$, is clearly the unique solution of the equation $q_1(\mu) = q_2(\mu)$. \hfill \square

Finally, we also have a simple estimation of the upper bound $U_i$.

Theorem 2 For any $\mu \geq \mu_i$, we have

$$\max\{q(\mu), \bar{q}(\mu)\} \leq q(\mu_i) + \mu - \mu_i.$$ 

(14)

Proof The inequality (14) follows from the definition $q_1(\mu)$ and the facts that $\eta_i \geq 0$ and $\lambda_{\min}(W) > 0$ (as $W > 0$). \hfill \square

Remark 1 The estimation (14) is independent of $\mu_{i+1}$. Therefore, it can be used for the extended case $\mu_{i+1} = \bar{\mu}$.
4 A branch-and-bound algorithm

In this section, we first propose a branch-and-bound algorithm based on the new upper bounds and then establish the worst-case computational complexity of the new algorithm.

Our algorithm works on a list

$$\mu = \mu_1 < \cdots < \mu_{k+1} = \tilde{\mu}.$$  \hspace{1cm} (15)

The initial list is $\mu = \mu_1 < \mu_2 = \bar{\mu}$. In each iteration, we first select the interval $[\mu_i, \mu_{i+1}]$ from the $\{\mu\}$-list that provides the maximal upper bound $U_i$. Then, we insert the mid-point $\frac{\mu_i + \mu_{i+1}}{2}$ into the $\{\mu\}$-list and increase $k$ by one. The process is repeated until the stopping criterion is reached. The detailed algorithm is presented as follows.

The branch-and-bound algorithm

Step 0. Given the approximation error $\epsilon > 0$. Compute $\underline{\mu}, \bar{\mu}, \lambda_{\text{min}}(W)$ and $\lambda_{\text{max}}(W)$. Initialize the iteration number $k = 1$.

Let $\mu_1 = \underline{\mu}$. Solve (SD$\mu_1$) to obtain the optimal solution $(\nu_1, \eta_1)$. Then, $q(\mu_1) = \mu_1 + v_1$ and let $LB = q(\mu_1), \mu^* = \mu_1$.

Let $\mu_2 = \bar{\mu} - \epsilon$. If $\mu_2 \leq \mu_1$, stop and return $\mu^*$ as an approximate maximizer.

Otherwise, solve (SD$\mu_2$) to obtain the optimal solution $(\nu_2, \eta_2)$. Then, $q(\mu_2) = \mu_2 + v_2$. If $q(\mu_2) > LB$, update $LB = q(\mu_2)$ and $\mu^* = \mu_2$. Set $k = 2$ and $S = \emptyset$.

Step 1. Let $\overline{\mu} = \frac{1}{2}(\mu_1 + \mu_2)$. Solve (SD$\overline{\mu}$) and obtain the optimal solution $(\tilde{\nu}, \tilde{\eta})$. Then, $q(\tilde{\mu}) = \bar{\mu} + \tilde{\nu}$. If $q(\tilde{\mu}) > LB$, update $LB = q(\tilde{\mu})$ and $\mu^* = \tilde{\mu}$.

Step 2. According to Theorem 1, compute the upper bounds:

$$UB_1 = \max_{\mu \in [\mu_1, \mu_2]} \tilde{\nu}(\mu), \quad UB_2 = \max_{\mu \in [\mu_1, \mu_2]} \tilde{\eta}(\mu).$$

Update $S = S \cup \{(UB_1, \mu_1, \tilde{\mu})\} \cup \{(UB_2, \tilde{\mu}, \mu_2)\}$ and $k = k + 1$.

Step 3. Find $(UB^*, \mu_1, \mu_2) = \arg \max_{(t, s, *) \in S} t$. If $UB^* \leq LB + \epsilon$, stop and return $\mu^*$ as an approximate maximizer. Otherwise, update $S = S \setminus \{(UB^*, \mu_1, \mu_2)\}$ and go to Step 1.

Remark 1 There is a pruning operator in the typical branch and bound algorithm. That is, in Step 1, after updating the lower bound $LB$, we should remove from $S$ the elements $(t, s, *) \in S$ satisfying $t \leq LB$. However, according to our experiments, this pruning operator almost has no impact on the efficiency of the branch-and-bound algorithm.

Theoretically, we can show that our new algorithm returns an $\epsilon$-approximately optimal solution of (P$_1$) in at most $\left[\frac{\bar{\mu} - \underline{\mu}}{\epsilon}\right]$ iterations. Here, we call $\mu^*$ an $\epsilon$-approximately optimal solution of (P$_1$) if it is feasible and satisfies

$$v(P_1) \geq q(\mu^*) \geq v(P_1) - \epsilon.$$

Theorem 3 The above algorithm terminates in at most $\left[\frac{\bar{\mu} - \underline{\mu}}{\epsilon}\right]$ steps and returns an $\epsilon$-approximately optimal solution of (P$_1$).

Proof If the algorithm terminates at Step 0, that is,

$$\bar{\mu} - \epsilon \leq \underline{\mu},$$

then for any $\mu \in [\underline{\mu}, \bar{\mu}]$, it follows from the inequality (14) in Theorem 2 that

$$q(\mu) \leq q(\underline{\mu}) + \mu - \underline{\mu} \leq \bar{\mu} - \underline{\mu} \leq q(\bar{\mu}) + \epsilon.$$
Therefore, we have

\[ q(\mu^*) = q(\mu) \geq \max_{\mu \in [\mu, \bar{\mu}]} q(\mu) - \epsilon = v(P_1) - \epsilon. \]

It follows that \( \mu^* = \bar{\mu} \) is an \( \epsilon \)-approximately optimal solution of (P1).

Now, we suppose that the algorithm does not terminate at Step 0. Consider \( \{ UB, \mu_1, \mu_2 \} \in S \) in the \( k \)-th iteration of the algorithm. If \( UB < UB^* \), then the interval \([\mu_1, \mu_2]\) will be not selected to subdivide. It is sufficient to study the interval \([\mu_1, \mu_2]\) with \( UB = UB^* \). According to the inequality (14) in Theorem 2, for any \( \mu \in [\mu_1, \mu_2] \), we have

\[ UB = \max_{\mu \in [\mu_1, \mu_2]} \bar{q}(\mu) \leq \max_{\mu \in [\mu_1, \mu_2]} q(\mu) \leq q(\mu_1) + \mu - \mu_1 = q(\mu_1) + \mu_2 - \mu_1. \]

Therefore, if the length of the current interval is small enough, more precisely, \( \mu_2 - \mu_1 \leq \epsilon \), the stopping criterion is reached since

\[ UB^* = UB \leq q(\mu_1) + \mu_2 - \mu_1 \leq LB + \epsilon. \]

Consequently, there are at most \( \left\lceil \frac{\bar{\mu} - \mu}{\epsilon} \right\rceil \) elements in \( S \). Since the number of elements of \( S \) increases by one in each iteration, the algorithm terminates in at most \( \left\lceil \frac{\bar{\mu} - \mu}{\epsilon} \right\rceil \) steps.

Let \( \mu^* \) be the approximation solution returned by the algorithm. We have

\[ UB^* \leq q(\mu^*) + \epsilon. \]  

(16)

To show that \( \mu^* \) is an \( \epsilon \)-approximately optimal solution of (P1), it is sufficient to prove that

\[ q(\mu^*) \geq v(P_1) - \epsilon. \]  

(17)

Let \( \hat{\mu} = \bar{\mu} - \epsilon > \mu \). According to the inequality (14) in Theorem 2, for any \( \mu \in [\hat{\mu}, \bar{\mu}] \), we obtain

\[ q(\mu) \leq q(\hat{\mu}) + \mu - \hat{\mu} \leq q(\hat{\mu}) + \bar{\mu} - \hat{\mu} = q(\hat{\mu}) + \epsilon. \]

Therefore, we have

\[ v(P_1) \leq \max\{ UB^*, \max_{\mu \in [\hat{\mu}, \bar{\mu}]} q(\mu) \} \leq \max\{ UB^*, q(\hat{\mu}) + \epsilon \} \leq q(\mu^*) + \epsilon, \]  

(18)

where the equality (18) follows from (16). Then, we obtain (17). The proof is complete. \( \square \)

5 Computational experiments

We test the new branch-and-bound algorithm for solving (P1) on the same numerical examples as in Nguyen et al. (2016). All the experiments are carried out in MATLAB 2014a and run on a server with 2.6 GHz dual-core processor and 32 GB RAM. The SDP subproblems (SD mu) are solved by SDPT3 within CVX, see Grant and Boyd (2017). The default setting of the approximation error used in our algorithm is \( \epsilon = 1e - 6 \).

Nguyen et al. (2016) reported their computational results for the following five examples, where the setting \( \delta = 0.05 \) is used. We implement the “two-stage” heuristic algorithm by ourselves with the same \( \delta \) and then report the numbers of function evaluations (i.e., solving
In this case, the SDP subproblems) in the first stage, which are compared with the iterations used by our algorithm.

The first example is taken from Example 3.2 in Zhang (2013). It has many local non-global maximizers.

**Example 1**

Let $B = \begin{pmatrix} 2.3969 & 0.4651 & 4.6392 \\ 0.4651 & 5.4401 & 0.7838 \\ 4.6392 & 0.7838 & 10.1741 \end{pmatrix}$, $W = \begin{pmatrix} 0.8077 & 0.8163 & 1.0970 \\ 0.8163 & 4.1942 & 0.8457 \\ 1.0970 & 0.8457 & 1.8810 \end{pmatrix}$, and $D = \begin{pmatrix} 3.9104 & -0.9011 & -2.0128 \\ -0.9011 & 0.9636 & 0.6102 \\ -2.0128 & 0.6102 & 1.0908 \end{pmatrix}$.

In this case, $[\mu, \bar{\mu}] = [0.9882, 6.7322]$. The “two-stage” algorithm developed by Nguyen et al. (2016) gives an approximation solution $\mu^* = 6.5952$. The number of function evaluations in the first stage is 116. Our algorithm returns an $\epsilon$-approximately optimal solution, $\mu^* = 6.5952$, in 1463 iterations. We notice that, with a larger setting $\epsilon = 1e - 5$, we need only 141 iterations to obtain $\mu^* = 6.5952$.

The second example in Nguyen et al. (2016) is taken from Example 3.1 in Zhang (2013), where the optimal solution of (P_1) is achieved at the right-hand side end-point $\bar{\mu}$.

**Example 2**

Let $B = \text{diag}(1, 9, 2), W = D = \text{diag}(5, 2, 3)$.

In this case, $[\mu, \bar{\mu}] = [0.2, 4.5]$. The number of function evaluations in the first stage of the “two-stage” algorithm is 87, while our algorithm finds $\mu^* = 4.5$ in 4 iterations.

The next three examples are taken from Examples 3, 4, 5 in Nguyen et al. (2016), respectively.

**Example 3**

Let $B = \begin{pmatrix} 1 & 2 & 3 & 1 \\ 2 & 5 & 4 & -1 \\ 3 & 4 & 0 & 1 \\ 1 & -1 & 1 & 6 \end{pmatrix}$, $W = \text{diag}(2, 1, 5, 10)$, and $D = \begin{pmatrix} 5 & -1 & 0 & 3 \\ -1 & 9 & 1 & 0 \\ 0 & 1 & -2 & 0 \\ 3 & 0 & 0 & 8 \end{pmatrix}$.

In this case, $[\mu, \bar{\mu}] = [-0.8241, 6.0647]$. The “two-stage” algorithm gives an approximation solution $\mu^* = 5.8748$. The number of function evaluations in the first stage is 139. Our algorithm returns an $\epsilon$-approximately optimal solution, $\mu^* = 5.8825$, in 229 iterations.

**Example 4**

Let $n = 10$, $B = \text{diag}(1, 2, 8, 7, 9, 3, 10, 2, -1, 6)$, $W = \text{diag}(9, 8, 7, 6, 5, 4, 3, 2, 1, 10)$, and $D = \text{diag}(5, 20, 3, 4, 8, -1, 0, 6, 32, 10)$.

The searching interval is $[\mu, \bar{\mu}] = [-1, 3.3333]$. The optimal solution is the left end-point $-1$. The number of function evaluations in the first stage is 88. Our algorithm returns an $\epsilon$-approximately optimal solution, $\mu^* = -1$, in 18 iterations.

**Example 5**

Let $n = 20$,

$B = \text{diag}(1, 2, 20, 3, 50, 4, 6, 7, 8, 9, 100, 2, 3, 4, 5, 6, 7, 0, 10, 9)$, $W = \text{diag}(100, 1, 2, 30, 5, 7, 9, 7, 8, 9, 1, 2, 30, 1, 50, 8, 1, 10, 10, 9)$, and $D = \text{diag}(0, 1000, 20, 2, 5, 6, 7, 9, 50, 3, 4, 5, 100, 5, 2, 200, 4, 5, 9, 21)$.

The searching interval of this example is $[\mu, \bar{\mu}] = [0, 100]$. The “two-stage” algorithm gives an approximation solution $\mu^* = 2.0029$. The number of function evaluations in the first
stage is 2001. Our algorithm returns an $\epsilon$-approximately optimal solution, $\mu^* = 2.0000$, in 29 iterations.

In addition to Examples 2–5 reported above, our algorithm highly outperforms the “two-stage” algorithm developed by Nguyen et al. (2016). For Example 1, our algorithm is also competitive when $\epsilon \geq 1e - 5$. Notice that our algorithm is an exact algorithm and the “two-stage” algorithm is heuristic.

Finally, we test more examples where the data are chosen randomly as follows. Each component of the symmetric matrices $B$ and $D$ is uniformly distributed in $[-10, 10]$. We generate $W, V = 4LL^T + \eta I$, where $L$ is a randomly generated lower bi-diagonal matrix with each nonzero element being uniformly distributed in $[-10, 10]$ and $\eta > 0$ is a constant number to guarantee the positive definiteness of $W$ and $V$. For each dimension varying from 30 to 200, we independently run our new algorithm and the “two-stage” algorithm (with $\delta = 0.05$ in the first stage and the stopping criterion in the second stage is reset so that it returns a solution at the same approximation level as in our algorithm) ten times and report in Tables 1 and 2 (with $\epsilon = 1e - 4$ and $1e - 6$, respectively) the average numerical results including the time in seconds and the number of iterations. It follows from these limited numerical results that our new global optimization algorithm outperforms the “two-stage” heuristic algorithm when $\epsilon = 1e - 4$ and $\epsilon = 1e - 6$. It is also observed that the efficiency of the “two-stage” algorithm seems to depend only on the setting $\delta$ in the first stage, while the computational cost of our new algorithm obviously increases as the approximation error $\epsilon$ decreases.

| n    | “Two-stage” algorithm | Our new algorithm |
|------|-----------------------|-------------------|
|      | Time (s) | Iter. | Time (s) | Iter. |
| 30   | 59.81    | 242.1 | 6.06     | 26.2  |
| 50   | 87.05    | 286.8 | 8.99     | 31.5  |
| 80   | 206.05   | 425.0 | 14.30    | 31.3  |
| 100  | 313.19   | 481.4 | 21.48    | 34.7  |
| 120  | 469.97   | 511.1 | 29.38    | 33.5  |
| 150  | 891.39   | 592.2 | 51.12    | 35.3  |
| 180  | 1502.52  | 627.3 | 87.42    | 37.5  |
| 200  | 2157.65  | 684.3 | 104.77   | 34.6  |

| n    | “Two-stage” algorithm | Our new algorithm |
|------|-----------------------|-------------------|
|      | Time (s) | Iter. | Time (s) | Iter. |
| 30   | 57.82    | 233.1 | 33.82    | 143.4 |
| 50   | 89.95    | 276.7 | 53.68    | 167.6 |
| 80   | 225.27   | 400.5 | 91.71    | 174.0 |
| 100  | 362.16   | 477.1 | 151.48   | 209.0 |
| 120  | 566.16   | 543.3 | 201.87   | 203.2 |
| 150  | 991.96   | 594.4 | 405.50   | 253.9 |
| 180  | 1713.32  | 673.5 | 533.57   | 219.1 |
| 200  | 2379.29  | 703.6 | 733.46   | 226.2 |
6 Conclusions

The recent SDP-based heuristic algorithm for maximizing the sum of two generalized Rayleigh quotients (SRQ) is based on the one-dimensional parametric reformulation where each functional evaluation corresponds to solving a semi-definite programming (SDP) subproblem. In this paper, we propose a branch-and-bound algorithm to globally solve (SRQ) based on the newly developed overestimation approach. Numerical results demonstrate that it is much more efficient than the recent SDP-based heuristic algorithm developed by Nguyen et al. (2016).

Acknowledgements The authors are grateful to the two anonymous referees for their valuable comments and suggestions.

References

Antoniou A, Lu W-S (2007) Practical optimization: algorithms and engineering applications. Springer Science + Business Media, LLC, New York
Bazaraa M-S, Sherali H-D, Shetty C-M (2006) Nonlinear programming: theory and algorithms, 3rd edn. John Wiley and Sons Inc., Hoboken
Dundar M-M, Fung G, Bi J, Sandilya S, Rao B (2005) Sparse Fisher discriminant analysis for computer aided detection. In: Proceedings of SIAM international conference on data mining
Freund R-W, Jarre F (2001) Solving the sum-of-ratios problem by an interior-point method. J Glob Optim 19:83–102
Fung E, Michael K-Ng (2007) On sparse Fisher discriminant method for microarray data analysis. Bioinformation 2:230–234
Grant M, Boyd S (2017) CVX: MATLAB software for disciplined convex programming. Version 2.1. http://cvxr.com/cvx
Jiao H-W, Liu S-Y (2015) A practicable branch and bound algorithm for sum of linear ratios problem. Eur J Oper Res 243(3):723–730
Jiao H-W, Liu S-Y (2017) Range division and compression algorithm for quadratically constrained sum of quadratic ratios. Comput Appl Math 36(1):225–247
Luenberger D-G, Ye Y (2008) Linear and nonlinear programming, 3rd edn. Springer Science + Business Media, LLC, New York
Matsui T (1996) NP-hardness of linear multiplicative programming and related problems. J Glob Optim 9:113–119
Nguyen V-B, Sheu R-L, Xia Y (2016) Maximizing the sum of a generalized Rayleigh quotient and another Rayleigh quotient on the unit sphere via semidefinite programming. J Glob Optim 64(2):399–416
Parlet B-N (1998) The symmetric eigenvalue problem. In: Classics in applied mathematics, vol 20. SIAM, Philadelphia, PA
Pólik I, Terlaky T (2007) A survey of the S-lemma. SIAM Rev 49(3):371–418
Polyak B-T (1998) Convexity of quadratic transformations and its use in control and optimization. J Optim Theory App 99(3):553–583
Primolevo G, Simeone O, Spagnolini U (2006) Towards a joint optimization of scheduling and beamforming for MIMO downlink. In: IEEE ninth international symposium on spread spectrum techniques and applications. pp 493–497
Schaible S, Shi J-M (2003) Fractional programming: the sum-of-ratios case. Optim Methods Softw 18(2):219–229
Sturm J-F, Zhang S (2003) On cones of nonnegative quadratic functions. Math Oper Res 28:246–267
Wu M-C, Zhang L-S, Wang Z-X, Christiani D-C, Lin X-H (2009) Sparse linear discriminant analysis for simultaneous testing for the significance of a gene set/pathway and gene selection. Bioinformatics 25:1145–1151
Ye Y, Zhang S-Z (2003) New results on quadratic minimization. SIAM J Optim 14(1):245–267
Zhang L-H (2013) On optimizing the sum of the Rayleigh quotient and the generalized Rayleigh quotient on the unit sphere. Comput Optim Appl 54:111–139
Zhang L-H (2014) On a self-consistent-field-like iteration for maximizing the sum of the Rayleigh quotients. J Comput Appl Math 257:14–28