SIAM J. MATRIX ANAL. APPL.
Vol. 41, No. 4, pp. 1771–1796

NONLINEAR EIGENVECTOR METHODS FOR CONVEX MINIMIZATION OVER THE NUMERICAL RANGE*

DING LU†

Abstract. We consider the optimization problem in which a continuous convex function is to be minimized over the joint numerical range of two Hermitian matrices. When those matrices are of large size, solving such problems by convex optimization can be computationally expensive. The goal of this paper is to present a novel nonlinear eigenvector method to accelerate the computation. We will show that the global minimizer of the optimization problem corresponds to a solution of a nonlinear eigenvalue problem with eigenvector nonlinearity (NEPv). The special structure of this NEPv allows for an efficient sequential subspace search algorithm, which is a nonlinear analogue to the NEPv of the commonly applied locally optimal conjugate gradient descent methods for Hermitian linear eigenvalue problems. Our new algorithm can be proven globally convergent to an eigenvector of the NEPv. Implementation details such as block iteration and preconditioning will be discussed. Numerical examples, with applications in computing the coercivity constant of boundary integral operators and solving multicast beamforming problems, show the effectiveness of our approach.

Key words. numerical range, Rayleigh quotient, nonlinear eigenvalue problem, eigenvector nonlinearity, sequential subspace method

AMS subject classifications. 15A18, 65F15, 47J10

DOI. 10.1137/18M1234473

1. Introduction.

In this paper we consider the following minimization problem:

$$\min_{y\in W(A,B)} F(y) \quad \text{and} \quad W(A,B) = \left\{ \begin{bmatrix} x^H A x \\ x^H B x \end{bmatrix} : x \in \mathbb{C}^n,\ \|x\|_2 = 1 \right\} \subset \mathbb{R}^2, \tag{1.1}$$

where $F : \mathbb{R}^2 \to \mathbb{R}$ is a continuous convex function, and A and B are n-by-n Hermitian matrices. The set $W(A,B)$ is known as the joint numerical range of the matrix pair (A,B), and it is a convex region in $\mathbb{R}^2$ by the famous Toeplitz–Hausdorff theorem; see, e.g., [16]. Therefore, the problem (1.1) is by nature a convex optimization.

A number of problems can be written into the form of (1.1). When F is a linear function, then the standard Rayleigh quotient minimization problem becomes a trivial special case in this framework; e.g., for $F(y) = y_1$ the minimizer is the smallest Rayleigh quotient $(x^H A x)/(x^H x)$ for $\|x\|_2 = 1$. For more general cases, a famous example is the Crawford number computation, where the objective function is $F(y) = \|y\|_2$. This problem has been studied since the 1970s [13] and is of particular interest in eigenvalue sensitivity analysis [13, 42] and the study of the coercivity constant for boundary operators [5]. Another example is the max-ratio minimization problem with $F(y) = \max\{y_1, y_2\}$. This objective function is also nondifferentiable. Such a problem arises in a class of homogeneous quadratic minimization [15], with applications in multicast beamforming problems in signal processing; see, e.g., [39].

Most of the existing methods for (1.1) were based on convex optimization. As a remarkable property of numerical range, the boundary of $W(A,B)$ can be approxi-

*Received by the editors December 20, 2018; accepted for publication (in revised form) by K. Meerbergen August 28, 2020; published electronically November 17, 2020. An earlier version of this paper was prepared while the author was affiliated with the Section of Mathematics, University of Geneva.

https://doi.org/10.1137/18M1234473

Funding: This work was supported by the SNSF under research project 169115.

†Department of Mathematics, University of Kentucky, Lexington, KY 40506 USA (Ding.Lu@uky.edu).

mated by a polygon, consisting of sampled supporting hyperplanes, each computed by solving a Hermitian eigenvalue problem of size n; see, e.g., [22]. The minimizer of (1.1) can then be estimated on the polygon by standard convex optimization [44]. For special F functions, this approach can lead to reformulations of problem (1.1) as eigenvalue optimization problems; see, e.g., [11] for the Crawford number computation and [15] for the max-ratio minimization. Such reformulations, however, are not available for a general F. In any case, including the eigenvalue optimization approaches, repeated Hermitian eigenvalue evaluations are typically required. This can be prohibitively expensive for problems with large matrices, hence necessitating the use of iterative methods as presented in the current paper for acceleration.

The main contribution of this work is to present a novel nonlinear eigenvector method for solving (1.1). Denote

$$\rho(x) := \left[\frac{x^H Ax}{x^H x}, \frac{x^H Bx}{x^H x}\right]^T \in \mathbb{R}^2; \tag{1.2}$$

we can rewrite (1.1) as the following composite minimization:

$$\min_{x\in\mathbb{C}^n\setminus\{0\}} \mathcal{F}(x) \quad \text{and} \quad \mathcal{F}(x) := F(\rho(x)). \tag{1.3}$$

We will show that the global minimizer of this problem is a solution to a *nonlinear eigenvalue problem with eigenvector nonlinearity* (NEPv). A sequential subspace search algorithm will be introduced to solve the NEPv. Such an algorithm involves only matrix-vector multiplication in each iteration and allows for block implementation; both are desirable for large-scale problems. The global convergence to an eigenvector can also be proven. This algorithm framework works for both smooth and nonsmooth F functions. For smooth problems, particular preconditioning and global verification schemes can be incorporated to further enhance the performance.

Our sequential subspace methods can be viewed as extensions to NEPv of the locally optimal block preconditioned conjugate gradient (LOBPCG) method commonly applied to Hermitian eigenvalue problems; see, e.g., [24, 25, 41, 30]. In each iteration, a reduced-size NEPv, obtained by subspace projection, is solved, and then the search subspace is updated by gradients. According to numerical experiments, these algorithms are as fast as LOBPCG in terms of the number of iterations, whereas the latter is only built for linear eigenvalue problems (LEPs) and hence is not applicable here. This convergence behavior seems to imply that the NEPv can be solved at a cost similar to a linear eigenvalue problem of the same size.

Although the algorithms are intended for general convex objective functions, they also provide, as a byproduct, new uses for the Crawford number computation, the max-ratio minimization, and related eigenvalue optimization. For those problems, the superior performance of the nonlinear eigenvector approaches, versus the state-of-the-art optimization methods, will be demonstrated by numerical experiments, with applications in coercivity constant computation for boundary operators and multicast transmit beamforming in signal processing.

The rest of this paper is organized as follows. In section 2, we present the NEPv characterization for the global minimizer of the optimization problem (1.3). In section 3, we develop and analyze the sequential subspace search algorithm for the NEPv with a smooth objective function. In section 4, extensions to nonsmooth objective functions are discussed. Implementation details are given in section 5, followed by numerical examples in section 6, and conclusions in section 7.

Notation. Throughout the paper, we follow the notation conventions in matrix analysis. We use $\mathbb{C}^{m\times n}$ for the set of m-by-n complex matrices, with $\mathbb{C}^n = \mathbb{C}^{n\times 1}$ and $\mathbb{C} = \mathbb{C}^1$. For real numbers we have $\mathbb{R}^{m\times n}$, $\mathbb{R}^n$, and $\mathbb{R}$, respectively. We use $\cdot^T$ for transpose and $\cdot^H$ for conjugate transpose. Let X be a matrix; $\mathrm{span}(X)$ is for the subspace spanned by the columns of X, $X(i:j,:)$ is for the submatrix consisting of row i to j of X, and $X(:,k:\ell)$ is for columns k to ℓ. For a Hermitian X, $\lambda_{\min}(X)$ denotes the smallest eigenvalue. The objective function $\mathcal{F}(x)$, which is a real-valued function in complex variables, is not differentiable in the holomorphic sense. Hence, we will work with the Wirtinger derivatives; see, e.g., [8, 27]. Namely, we view $\mathcal{F}(x) = \mathcal{F}(p,q)$ as a function in the real and imaginary components of $x = p + \jmath q$, and we define

$$\nabla^w \mathcal{F}(x) := \nabla_p \mathcal{F}(p,q) + \jmath \nabla_q \mathcal{F}(p,q), \tag{1.4}$$

where ∇_p and ∇_q are the standard partial derivatives. Clearly, $\nabla^w \mathcal{F}(x)$ has a one-to-one correspondence with the standard gradient $\nabla_{(p,q)}\mathcal{F}(p,q)$. So its angle with a vector $h = r + \jmath s \in \mathbb{C}^n$ is naturally defined as

$$\angle_w(\nabla^w \mathcal{F}(x), h) := \arccos \frac{\mathrm{Re}(h^H \cdot \nabla^w \mathcal{F}(x))}{\|\nabla^w \mathcal{F}(x)\|_2 \|h\|_2} = \angle \left(\nabla_{(p,q)} \mathcal{F}(p,q), \begin{bmatrix} r \\ s \end{bmatrix} \right), \tag{1.5}$$

where $\angle$ denotes the angle between two real vectors in the standard definition. So $\mathcal{F}(x)$ is descending in the direction of h if $\cos \angle_w(\nabla^w \mathcal{F}(x), h) < 0$.

2. Optimality conditions and nonlinear eigenvalue problems. In this section, we establish NEPv characterization for the solution of the optimization problem (1.3). We will use generalized gradient [12] to present the result, so that the case of nonsmooth F will also be included. For a continuous convex function $F(y)$, the generalized gradient, denoted by $\partial F(y)$, is the set of subgradients of F at y, and a vector $g \in \mathbb{R}^2$ is called a subgradient of F at y if it satisfies

$$F(z) \geq F(y) + g^T(z - y) \quad \forall z \in \mathbb{R}^2. \tag{2.1}$$

For the composite function $\mathcal{F}(x) = F(\rho(x))$ in (1.3), the generalized gradient is obtained by the chain rule (see, e.g., [12, Thm. 2.3.9] and [9, eq. (2.4)]) as

$$\begin{aligned}
\partial^w \mathcal{F}(x) &:= \left\{ g_1 \nabla^w \left(\frac{x^H A x}{x^H x} \right) + g_2 \nabla^w \left(\frac{x^H B x}{x^H x} \right) \;\middle|\; \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \in \partial F(\rho(x)) \right\} \\
&= \left\{ \frac{2g_1}{x^H x} \left(Ax - \frac{x^H A x}{x^H x} x \right) + \frac{2g_2}{x^H x} \left(Bx - \frac{x^H B x}{x^H x} x \right) \;\middle|\; \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \in \partial F(\rho(x)) \right\} \\
&= \left\{ \frac{2}{x^H x} \left(H(x)x - \frac{x^H H(x) x}{x^H x} x \right) \;\middle|\; H(x) = g_1 A + g_2 B, \; \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \in \partial F(\rho(x)) \right\},
\end{aligned} \tag{2.2}$$

where the w in ∂^w is to indicate that the elements are expressed in the Wirtinger style.[1]

By the standard nonsmooth analysis [12, Prop. 2.3.2], a necessary condition for $x \in \mathbb{C}^n$ to be a local optimizer of (1.3) is given by

$$0 \;\in\; \partial^w \mathcal{F}(x), \tag{2.3}$$

and any vector x satisfying (2.3) is called *stationary*. This optimality condition leads to the following NEPv characterization for the optimizers.

[1] Let $x = p + \jmath q \in \mathbb{C}^n$ and view $\mathcal{F}(x) = \mathcal{F}(p,q)$ as a function in the real and imaginary parts of x; then the standard generalized gradient $\partial \mathcal{F}(p,q) = \{ [\mathrm{Re}(z)^T, \mathrm{Im}(z)^T]^T : z \in \partial^w \mathcal{F}(x) \}$.

THEOREM 2.1 (NEPv characterization). *For the minimization problem* (1.3)*, the following hold:*

(a) *A vector x is stationary if and only if it satisfies the nonlinear eigenvalue problem with eigenvector nonlinearity (NEPv)*

$$H(x)x = \lambda x \qquad \text{and} \qquad x \neq 0, \tag{2.4}$$

where

$$H(x) := g_1 A + g_2 B \qquad \text{for some} \quad \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \in \partial F(\rho(x)). \tag{2.5}$$

(b) *A vector x is a global minimizer if and only if the assumptions of* (2.4) *hold with $\lambda = \lambda_{\min}(H(x))$.*

Proof. Result (a) is a direct consequence of the conditions (2.2) and (2.3) and the fact that any x satisfying the NEPv (2.4) has a corresponding $\lambda = x^H H(x)x/(x^H x)$.

For result (b), we first show the necessity. Denote by x_* a global minimizer, $y_* := \rho(x_*) \in W(A,B)$, and $f_{\min} := \mathcal{F}(x_*) = F(y_*)$. Define the convex sublevel set

$$L_f := \{y \in \mathbb{R}^2 \colon F(y) < f_{\min}\}.$$

If $L_f = \emptyset$, then the convex function F achieves minimum in $\mathbb{R}^2$ at y_*, which implies $g = 0 \in \partial F(y_*)$ by convex analysis [7]. So (2.4) holds with $H(x_*) = 0$ and $\lambda = 0$.

Now, consider L_f as being nonempty. Since $f_{\min}$ is a global minimizer, it holds that $F(y) \geq f_{\min}$ for all $y \in W(A,B)$. Hence, the two convex sets L_f and $W(A,B)$ are disjoint. We can separate the two sets by a hyperplane: $\exists g \neq 0 \in \mathbb{R}^2$ and $c \in \mathbb{R}$, s.t.

$$g^T z \leq c \text{ for all } z \in L_f \qquad \text{and} \qquad g^T z \geq c \text{ for all } z \in W(A,B). \tag{2.6}$$

By continuity of F, the $y_* \in W(A,B)$ lies in the completion $\overline{L}_f$. So by (2.6),

$$g^T y_* = c. \tag{2.7}$$

Geometrically, $g^T z - c = 0$ defines a supporting hyperplane of L_f at y_*. The vector g, which is an outer normal direction of the sublevel set L_f at y_*, satisfies $\alpha g \in \partial F(y_*)$ for some scalar $\alpha > 0$; see, e.g., [20, Thm. VI.1.3.5]. Since the inequalities in (2.6) can still hold for g and c multiplied by α, we can always assume that $g \in \partial F(y_*)$.

The condition (2.7), combined with the second equation of (2.6), implies

$$\min_{y \in W(A,B)} g^T y = g^T y_* \quad \Leftrightarrow \quad \min_{x \in \mathbb{C}^n} \frac{x^H (g_1 A + g_2 B)\, x}{x^H x} = \frac{x_*^H (g_1 A + g_2 B)\, x_*}{x_*^H x_*}.$$

Let $H(x_*) = g_1 A + g_2 B$; the Rayleigh quotient $(x^H H(x_*)x)/(x^H x)$ achieves minimum at x_*. Hence x_* is an eigenvector corresponding to the smallest eigenvalue of $H(x_*)$.

Next, consider the sufficiency. Let (2.4) hold with $\lambda = \lambda_{\min}(H(x))$. Since $g \in \partial F(y)$ for $y = \rho(x)$, the subgradient inequality (2.1) implies for all $\widetilde{y} \in W(A,B)$ that

$$F(\widetilde{y}) - F(y) \geq g^T(\widetilde{y} - y) \geq \min_{z \in W(A,B)} g^T z - g^T y \quad = \min_{\widetilde{x} \neq 0} \frac{\widetilde{x}^H H(x) \widetilde{x}}{\widetilde{x}^H \widetilde{x}} - \frac{x^H H(x) x}{x^H x} = 0,$$

where for the last equation we used the eigenvalue minimization principle for the smallest eigenvalue λ. Hence, $F(y) = \mathcal{F}(x)$ is a global minimizer. ∎

The NEPv (2.4) is a type of "self-consistent" eigenvalue problem, where the solution (λ_*, x_*) is an eigenpair of the Hermitian matrix $H(x_*)$ defined by x_* itself. Since $H(x_*)$ has n eigenvalues ordered as $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$, it holds that $\lambda_* = \lambda_\ell$ for some $1 \le \ell \le n$. As shown in Theorem 2.1(b), the index ℓ is crucial for the optimality of x_*: if $\ell = 1$, then the stationary condition (2.4) is also sufficient for global optimality.

We make two comments on the NEPv characterization. First, Theorem 2.1(b) provides a sufficient and necessary condition for the global optimality. This condition is different from the standard second order optimality conditions, which only serve as sufficient local optimality conditions; see, e.g., [34, Thm. 12.6] for differentiable problems and [9] for convex composite minimization problems. Second, the technique of NEPv characterization has also been explored for optimization problems in computational physics and chemistry and for many data analysis applications; see, e.g., [28, 32, 10, 46, 3]. Depending on the problem under consideration, particular NEPvs and their eigenvalue ordering conditions (the index ℓ) can be established. Those NEPvs, however, usually only serve as necessary conditions for optimality.

3. Sequential subspace search. In this section, we introduce an iterative method, based on sequential subspace search, to solve the NEPv (2.4). We assume F is a smooth function, and we will discuss the nonsmooth case in section 4. For a smooth function, the generalized gradient $\partial F(y)$ is simply the standard gradient, so the coefficient matrix $H(x)$ in (2.4) is given by

$$H(x) = F_1(\rho(x)) \cdot A + F_2(\rho(x)) \cdot B, \tag{3.1}$$

where $F_j(y) = \frac{\partial F}{\partial y_j}(y)$ for $j = 1, 2$ denotes the partial derivatives of F.

3.1. Nonlinear Rayleigh–Ritz procedure. Solving the NEPv (2.4) by a standard method, such as the self-consistent-field (SCF) iteration (see subsection 3.1.2), can be computationally expensive, especially for problems of large size. In what follows we describe a popular subspace search technique to accelerate the computation.

3.1.1. Subspace search. The basic idea of subspace search is as follows: Given a low-dimensional search subspace $\mathcal{U} \subset \mathbb{C}^n$, we find a vector $\widehat{x} \in \mathcal{U}$ that best approximates the desired eigenvector. In view of the minimization characterization (1.3), the best approximation can be defined as

$$\widehat{x} := \operatorname*{arg\,min}_{x \in \mathcal{U}} \; F\left(\rho(x)\right) = U \cdot \left(\operatorname*{arg\,min}_{v \in \mathbb{C}^k} \; \widehat{\mathcal{F}}(v) \right), \tag{3.2}$$

where $U \in \mathbb{C}^{n \times k}$ is a basis matrix (not necessarily orthogonal) of $\mathcal{U}$, and

$$\widehat{\mathcal{F}}(v) := F\left(\widehat{\rho}(v)\right) \quad \text{with} \quad \widehat{\rho}(v) = \left[\frac{v^H \widehat{A} v}{v^H \widehat{M} v}, \frac{v^H \widehat{B} v}{v^H \widehat{M} v} \right]^T \in \mathbb{R}^2,$$

and $\widehat{A} = U^H A U$, $\widehat{B} = U^H B U$, and $\widehat{M} = U^H U$. Here the second equation of (3.2) is obtained by a substitution of variables $x = Uv$.

Observe that the last problem in (3.2) is again a composite minimization in the same form[2] as (1.3), but for matrices of a reduced size k. According to Theorem 2.1, the stationary condition of this optimization problem is given by the projected NEPv

$$\widehat{H}(v) v = \lambda \widehat{M} v \quad \text{and} \quad v \neq 0, \tag{3.3}$$

[2] We can convert the Rayleigh quotients $\widehat{\rho}(v)$ into the standard form (1.2) by the factorization $\widehat{M} = \widehat{L}\widehat{L}^H$: Let $v = \widehat{L}^{-H} u$, $\widetilde{A} = \widehat{L}^{-1} \widehat{A} \widehat{L}^{-H}$, and $\widetilde{B} = \widehat{L}^{-1} \widehat{B} \widehat{L}^{-H}$; then $\widehat{\rho}(v)^T = \left[\frac{u^H \widetilde{A} u}{u^H u}, \frac{u^H \widetilde{B} u}{u^H u} \right]$.

where

$$\widehat{H}(v) := F_1\big(\widehat{\rho}(v)\big) \cdot \widehat{A} + F_2\big(\widehat{\rho}(v)\big) \cdot \widehat{B}. \tag{3.4}$$

Moreover, $\widehat{v}$ is a global minimizer of $\widehat{\mathcal{F}}$ if and only if it solves the NEPv (3.3), with λ being the smallest eigenvalue of the matrix pencil $\widehat{H}(v) - \lambda\widehat{M}$.

Since the projected NEPv (3.3) is of a much smaller size, the corresponding eigenvector $\widehat{v}$ can be computed by SCF iteration, or by convex optimization approaches mentioned in the introduction. Once $\widehat{v}$ is obtained, we have

$$\widehat{x} = U\widehat{v}. \tag{3.5}$$

The procedure from above is called a *nonlinear Rayleigh–Ritz procedure*, which is a natural generalization of the well-known Rayleigh–Ritz procedure [36] to the NEPv (2.4).

3.1.2. SCF for the reduced NEPv. SCF iteration is a fixed-point type iteration commonly applied to solve NEPvs in computational physics and chemistry [28]. For the reduced NEPv (3.3), an SCF iteration can be summarized as follows: Given an initial vector $v^{(0)}$, iteratively compute $v^{(1)}, v^{(2)}, \ldots$ satisfying

$$\widehat{H}(v^{(p)})v^{(p+1)} = \lambda^{(p+1)}\widehat{M}v^{(p+1)} \quad \text{for} \quad p = 0, 1, \ldots, \tag{3.6}$$

where $\lambda^{(p+1)}$ is the smallest eigenvalue of the matrix pencil $\widehat{H}(v^{(p)}) - \lambda\widehat{M}$. For reasons that will be be made clear shortly, we normalize the eigenvectors to satisfy

$$(v^{(p+1)})^H\widehat{M}v^{(p+1)} = 1 \quad \text{and} \quad (v^{(p)})^H\widehat{M}v^{(p+1)} \geq 0. \tag{3.7}$$

The iteration (3.6) is an SCF in its simplest form, also known as the pure SCF iteration. Such an algorithm, however, is not always convergent. To address this issue, we can add an extra line search step so that $\mathcal{F}(v^{(p+1)})$ is sufficiently reduced:

$$v^{(p+1)} := \alpha_p v^{(p+1)} + (1 - \alpha_p)v^{(p)} = v^{(p)} + \alpha_p d_p,$$

where $d_p = v^{(p+1)} - v^{(p)}$ and α_p is a damping factor obtained by a line search for $\alpha_p \in [0, 1]$, s.t. the following Armijo condition (see, e.g., [34, Chap. 3]) is satisfied:

$$\widehat{\mathcal{F}}\left(v^{(p)} + \alpha d_p\right) \leq \widehat{\mathcal{F}}\left(v^{(p)}\right) + \alpha_p \cdot c\,\mathrm{Re}\left(d_p^H \cdot \nabla^w \widehat{\mathcal{F}}(v^{(p)})\right),$$

where $c \in (0, 1)$ is a given constant. The overall algorithm of SCF with a line search is outlined in Algorithm 3.1, where a safeguard step described in Remark 3.1 has been included in line 5. Such a line search scheme is also applied in the SCF iteration [3].

By the normalization condition (3.7) and the safeguard scheme in Remark 3.1, the search directions d_p in Algorithm 3.1 are always descending and gradient-related; i.e., it holds for all $p \geq 0$ that $\cos\angle_w(-\nabla^w\mathcal{F}(v^{(p)}), d_p) > \gamma$ with a constant $\gamma > 0$. Hence, by a direct application of the convergence results for line search methods with gradient-related search directions (see, e.g., [1, Thm. 4.3]), we can conclude that the algorithm is globally convergent: any limiting point v of $\{v^{(p)}\}_{p=0}^{\infty}$ is stationary for the function $\widehat{\mathcal{F}}(v)$ in (3.2); namely, v is an eigenvector of the NEPv (3.3).

In addition to the line search from above, several other techniques developed in computational physics and chemistry can also be applied to stabilize the convergence of the SCF iteration—for example, level-shifting [38], direct inversion in the iterative subspace (DIIS) [37], and trust-region SCF [43, 45]. Since the reduced NEPv's are of a small size, those methods do not make a big difference in the overall performance of the subspace algorithm.

Algorithm 3.1. SCF iteration with line search for the NEPv (3.3).

Input: Starting vector $v^{(0)}$ with $v^{(0)H}\widehat{M}v^{(0)} = 1$, tolerance `tol_r`, `maxit`, and line search factors $\gamma, c, \tau \in (0,1)$ (e.g., $\gamma = c = \tau = 0.1$).

Output: Approximate eigenvector $\widehat{v}$.

1: **for** $p = 0, 1, \ldots,$ `maxit` **do**
2: Compute $\widehat{r}_p = \widehat{H}(v^{(p)})v^{(p)} - \widehat{\mu}_p\widehat{M}v^{(p)}$ with $\widehat{\mu}_p = v^{(p)H}\widehat{H}(v^{(p)})v^{(p)}$.
3: Check convergence: **if** $\|\widehat{r}_p\|_2 \le \|v^{(p)}\|_2 \cdot$ `tol_r` **then** break.
4: Solve the Hermitian eigenvalue problem $\widehat{H}(v^{(p)})v = \lambda\widehat{M}v$ for the smallest eigenvalue $\lambda^{(p+1)}$ and the eigenvector $v^{(p+1)}$ normalized as in (3.7).
5: Set $\alpha_p = 1$, $d_p = v^{(p+1)} - v^{(p)}$ (reset $d_p = -\widehat{r}_p/\|\widehat{r}_p\|_2$ if $\cos\angle_w(-\widehat{r}_p, d_p) \le \gamma$).
6: **while** $\widehat{\mathcal{F}}\left(v^{(p)}\right) - \widehat{\mathcal{F}}\left(v^{(p+1)}\right) < -\alpha_p \cdot c\,\mathrm{Re}(2d_p^H\widehat{r}_p)$ **do**
7: $\alpha_p := \tau\alpha_p$ and $v^{(p+1)} := v^{(p)} + \alpha_p d_p$. *% backtracking line search*
8: **end while**
9: **end for**
10: Return $\widehat{v} = v^{(p)}$.

Remark 3.1. Recall from (2.2) that $\nabla^w\widehat{\mathcal{F}}(v^{(p)}) = 2\widehat{r}_p$ with $\widehat{r}_p$ defined as in line 2 of Algorithm 3.1. For the SCF iteration (3.6), it holds for $d_p = v^{(p+1)} - v^{(p)}$ that

$$\begin{aligned}(\nabla^w\widehat{\mathcal{F}}(v^{(p)}))^H \cdot d_p &= 2 \cdot v^{(p)H}(\widehat{H}(v^{(p)}) - \widehat{\mu}_p \cdot \widehat{M})(v^{(p+1)} - v^{(p)}) \\ &= 2 \cdot v^{(p)H}\widehat{M}v^{(p+1)} \cdot (\lambda^{(p+1)} - \widehat{\mu}_p) \le 0,\end{aligned}$$

where we used (3.6) in the second equation and $\lambda^{(p+1)} = \min_v(v^H H(v^{(p)})v)/(v^H\widehat{M}v)$ and (3.7) in the last equation. Hence, if $(\nabla^w\widehat{\mathcal{F}}(v^{(p)}))^H \cdot d_p \ne 0$, then d_p is a descent direction. It can happen that d_p is nearly orthogonal to the gradient $\nabla^w\widehat{\mathcal{F}}(v^{(p)}) = 2\widehat{r}_p$; namely, $\cos\angle_w(-\widehat{r}_p, d_p) \le \gamma$ for a small constant $\gamma > 0$. In this case, we reset the search direction d_p as the negative gradient $d_p = -\widehat{r}_p / \|\widehat{r}_p\|_2$. By such a safeguard step, the search directions are gradient-related with $\cos\angle_w(-\widehat{r}_p, d_p) > \gamma$ for all $p \ge 0$.

3.2. Sequential subspace search. By the nonlinear Rayleigh–Ritz procedure, we can develop a sequential subspace search algorithm for the NEPv (2.4): In iteration k, unless x_k is already an optimizer with $\nabla^w\mathcal{F}(x_k) = 0$, we search for the next

$$x_{k+1} \in \mathrm{span}\Big\{ x_{k-1},\, x_k,\, r_k \Big\} \tag{3.8}$$

in a search subspace spanned by the current iterate x_k, the old x_{k-1}, and the gradient

$$r_k := \frac{\nabla^w\mathcal{F}(x_k)}{\|\nabla^w\mathcal{F}(x_k)\|_2} = \xi_k \cdot \Big(H(x_k)x_k - \mu(x_k) \cdot x_k \Big) \tag{3.9}$$

with $\mu(x_k) = (x_k^H H(x_k)x_k)/(x_k^H x_k)$, and $\xi_k \ge 0$ is a normalization factor, s.t. $\|r_k\|_2 = 1$. By discussions in subsection 3.1, the best approximation x_{k+1} can be computed by the nonlinear Rayleigh–Ritz procedure as

$$x_{k+1} = U_k\widehat{v}_k/\|U_k\widehat{v}_k\|_2 \quad \text{with} \quad U_k = [x_k, r_k, x_{k-1}], \tag{3.10}$$

where $\widehat{v}_k$ is a solution to the projected NEPv (3.3) by U_k. Here we have assumed x_{k-1} is linearly independent of x_k, r_k; otherwise we use $U_k = [x_k, r_k]$, which is always

orthogonal. Starting with $x_0 \in \mathbb{C}^n$ and $x_{-1} \in 0^n$, this process can be repeated for $k = 0, 1, \ldots$ until convergence is reached, e.g., with $\nabla^w \mathcal{F}(x_k)$ sufficiently small.

Since gradients are used for the search, each individual iteration of the subspace algorithm is at least as fast as gradient descent. We can therefore prove the global convergence in the following theorem. This result also shows that the NEPv (3.3) can be solved inexactly, and that accidental failures of convergence of SCF iteration within `maxit` will not terminate the overall algorithm. This provides us with great flexibility in choosing the parameters `tol_r` and `maxit` for Algorithm 3.1.

THEOREM 3.2 (global convergence). *Let F be a smooth convex function over $W(A, B)$, and let $\{x_k\}_{k=1}^{\infty}$ be produced by the sequential subspace search (without early convergence) using the nonlinear Rayleigh–Ritz procedure* (3.10)*. Assume each reduced NEPv* (3.3) *is solved either exactly or by running a few steps of the SCF iteration in Algorithm* 3.1 *starting with $v_k^{(0)} = e_1$. Then, it holds that $\mathcal{F}(x_{k+1}) < \mathcal{F}(x_k)$ for $k = 0, 1, \ldots$, and any limiting point $\widetilde{x}$ of $\{x_k\}_{k=1}^{\infty}$ is an eigenvector of the NEPv* (2.4)*.*

Proof. It is sufficient to consider the case where the NEPv is solved inexactly by the SCF iteration. We will show that the computed x_{k+1} satisfies $\mathcal{F}(x_{k+1}) \le \mathcal{F}(\widetilde{x}_{k+1})$, where $\widetilde{x}_{k+1}$ is a vector satisfying the Armijo condition

$$\mathcal{F}(\widetilde{x}_{k+1}) \le \mathcal{F}(x_k) + c \cdot \mathrm{Re}(\widetilde{d}_k^H \cdot \nabla^w \mathcal{F}(x_k)) \quad \text{with} \quad \widetilde{d}_k = \widetilde{x}_{k+1} - x_k, \tag{3.11}$$

and $\widetilde{d}_k$ is gradient-related, i.e., $\cos \angle_w(-\nabla^w \mathcal{F}(x_k), \widetilde{d}_k) > \gamma$. Here $c > 0$ and $\gamma > 0$ are constants independent of k. Then the global convergence is obtained by a direct application of [1, Thm. 4.3] for accelerated line search methods.

Consider x_{k+1} (3.10) with v_k computed by Algorithm 3.1. Let $v_k^{(1)} = e_1 + \alpha_0 d_0$ be the first iterate by Algorithm 3.1. Due to the line search in lines 5–8, $v_k^{(1)}$ satisfies

$$\widehat{\mathcal{F}}_k(v_k^{(1)}) \le \widehat{\mathcal{F}}_k(e_1) + c \cdot \alpha_0 \mathrm{Re}(d_0^H \cdot \nabla^w \widehat{\mathcal{F}}_k(e_1)), \tag{3.12}$$

where $\widehat{\mathcal{F}}_k$ is defined as in (3.2) with U_k. Because of the safeguarding in line 5, the search direction d_0 also satisfies $\cos \angle_w(-\nabla^w \widehat{\mathcal{F}}_k(e_1), d_0) > \gamma$.

Since $\widehat{\mathcal{F}}_k(v) = \mathcal{F}(U_k v)$ and $x_k = U_k e_1$, by the chain rules, $\nabla^w \widehat{\mathcal{F}}(e_1) = U_k^H \cdot \nabla^w \mathcal{F}(x_k)$. Let $\widetilde{x}_{k+1} := U_k v_k^{(1)} = x_k + \alpha_0 U_k d_0$. Then (3.11) follows from (3.12). Moreover, the increment $\widetilde{d}_k = \alpha_0 \cdot U_k d_0$ is also a gradient-related direction due to

$$\begin{aligned}\cos \angle_w(U_k d_0, -\nabla^w \mathcal{F}(x_k)) &= \frac{-\mathrm{Re}(d_0^H U_k^H \cdot \nabla^w \mathcal{F}(x_k))}{\|U_k d_0\|_2 \cdot \|\nabla^w \mathcal{F}(x_k)\|_2} = \frac{-\mathrm{Re}(d_0^H \cdot \nabla^w \widehat{\mathcal{F}}_k(e_1))}{\|U_k d_0\|_2 \cdot \|\nabla^w \mathcal{F}(x_k)\|_2} \\ &\ge \frac{\cos \angle_w(d_0, -\nabla^w \widehat{\mathcal{F}}_k(e_1))}{\sqrt{3}} \ge \frac{\gamma}{\sqrt{3}},\end{aligned}$$

where in the third equation we used $\|U_k d_0\|_2 \le \sqrt{3}\|d_0\|_2$ (as U_k has unitary columns) and $\|\nabla^w \mathcal{F}(x_k)\|_2 = |e_2^H U_k^H \cdot \nabla^w \mathcal{F}(x_k)| \le \|\nabla^w \widehat{\mathcal{F}}_k(e_1)\|_2$ (as $U_k e_2 = r_k = \zeta_k \nabla^w \mathcal{F}(x_k)$). Finally, due to monotonicity of Algorithm 3.1, it holds that $\mathcal{F}(x_{k+1}) \le \mathcal{F}(\widetilde{x}_{k+1})$. ∎

For Hermitian eigenvalue problems, the sequential subspace search using the subspace (3.8) has been applied in the well-known LOBPCG method [25]. The algorithm presented is therefore a formal extension of LOBPCG to the NEPv (2.4). In particular, when $H(x) = H$ is a constant matrix, the NEPv reduces to a Hermitian eigenvalue problem, and the algorithm coincides with LOBPCG in its simplest form. Apart from eigenvalue computation, the idea of sequential subspace search has also been applied in [18] for minimizing a quadratic function over a sphere, in [33] for unconstrained optimization problems, and in [1] for Riemannian optimization.

3.3. Other issues.

3.3.1. Preconditioned expansion. The sequential subspace search, as a gradient descent method, is linearly convergent. For acceleration, we can extend the search subspace (3.8) with extra vectors. Motivated by Newton's methods (see Appendix A for details), we propose to extend the subspace as

$$\text{span}\left\{ x_{k-1}, x_k, r_k, \;\middle|\; p_k^{(a)}, p_k^{(b)} \right\}, \tag{3.13}$$

where

$$\left[p_k^{(a)}, p_k^{(b)}\right] := (H(x_k) - \lambda_k I)^{-1} [Ax_k,\ Bx_k], \tag{3.14}$$

and λ_k is a prescribed shift. This formula resembles inverse iteration in eigenvalue computation. With a properly chosen λ_k, such an extended subspace can lead to local quadratic convergence.

THEOREM 3.3 (quadratic convergence). *Assume F is with continuous second derivatives, and (λ_*, x_*) is an eigenpair of the NEPv (2.4), with $\lambda_* = \lambda_{\min}(H(x_*))$ being a simple eigenvalue and $b^H x_* \neq 0$ for a normalization vector $b \in \mathbb{C}^n$. Define the shift parameter in (3.14) by $\lambda_k = (b^H H(x_k)x_k)/(b^H x_k)$. Then, provided $\lambda_k \neq 0$ and $H(x_k) - \lambda_k I$ is nonsingular, the subspace (3.13) contains a vector $\widehat{x}$ satisfying $\tan\angle(\widehat{x}, x_*) = \mathcal{O}(|\tan\angle(x_k, x_*)|^2)$ when $\tan\angle(x_k, x_*)$ is sufficiently small.*

Proof. The proof is deferred to Appendix A. ∎

To obtain the vectors in (3.14), we need to solve a linear system of equations with two right-hand sides. For structured matrices, this can be done efficiently by direct solvers, e.g., by sparse LU. Since those extra vectors are only for acceleration, their exact solution are usually not needed. Hence, we can also apply iterative methods or use a preconditioner $T_k \approx (H(x_k) - \lambda_k I)^{-1}$ for inexact solutions.

When an inexact solver is applied, it is more appealing to consider the following alternative for (3.14):

$$\left[p_k^{(a)}, p_k^{(b)}\right] := (H(x_k) - \lambda_k I)^{-1} [r_A(x_k),\ r_B(x_k)], \tag{3.15}$$

where

$$r_A(x) = \left(A - \frac{x^H Ax}{x^H x} I\right) x \qquad \text{and} \qquad r_B(x) = \left(B - \frac{x^H Bx}{x^H x} I\right) x \tag{3.16}$$

are residual vectors corresponding to $A - \lambda I$ and $B - \lambda I$, respectively. Given that λ_k is not an exact nonlinear Rayleigh quotient $(x_k^H H(x_k)x_k)/(x_k^H x_k)$, an easy calculation shows that the extended subspaces (3.13) by (3.14) and (3.15) are identical. Compared to (3.14), the new formulas (3.15) apply inversion on the residuals, i.e., the gradients of the Rayleigh quotients. When a preconditioner $T_k \approx (H(x_k) - \lambda_k I)^{-1}$ is applied, this scheme resembles the preconditioned residual technique and the generalized Davidson methods that are commonly used in linear eigenvalue computation; see, e.g., [41, 2].

3.3.2. Global verification and restarting. From Theorem 3.2, the sequential subspace algorithm is globally convergent to an eigenvector of the NEPv (2.4). In practice, the converged eigenvector is usually the one corresponding to the smallest eigenvalue and hence is a global minimizer of $\mathcal{F}$. Such a global convergence is partially

because of the "global search" within the subspace, which makes it easier to escape local optimizers and saddle points than when using those standard optimization techniques based on line search.

Even so, the rare but possible convergence to a local optimizer is undesirable. In applications where global optimality is a crucial concern, it is necessary to verify that the computed x_* is indeed a global solution. In view of Theorem 2.1(b), this can be done by computing the smallest eigenvalue $\lambda_{\min}$, and the corresponding eigenvector $\widehat{x}$, of the matrix $H(x_*)$

$$H(x_*)\widehat{x} = \lambda_{\min} \cdot \widehat{x}. \tag{3.17}$$

Then x_* is a global minimizer if $\lambda_{\min} = (x_*^H H(x_*)x_*)/(x_*^H x_*)$. The eigenvalue problem (3.17) can be solved by an iterative method such as LOBPCG or `eigs` in MATLAB.

If the global verification fails, namely, $\lambda_{\min} \neq (x_*^H H(x_*)x_*)/(x_*^H x_*)$, we need to restart the algorithm for a better solution. Then the eigenvector $\widehat{x}$ in (3.17) provides an ideal restarting vector. As justified in Lemma 3.4, by restarting with

$$x_0 := \widehat{x} \quad \text{and} \quad x_{-1} := x_*, \tag{3.18}$$

we can always obtain an x_1 with $\mathcal{F}(x_1) < \mathcal{F}(x_*)$ and escape the local optimizer x_*.

LEMMA 3.4. *Let (λ_*, x_*) be an eigenpair of the NEPv (2.4), and let $(\lambda_{\min}, \widehat{x})$ be a solution to the linear eigenvalue problem (LEP) (3.17). If $\lambda_* > \lambda_{\min}$, then it holds strictly that*

$$\min_{x \in \operatorname{span}([x_*, \widehat{x}])} \mathcal{F}(x) < \mathcal{F}(x_*). \tag{3.19}$$

Proof. Let $U = [x_*, \widehat{x}]$; we have that U is of full column rank since $\lambda_* \neq \lambda_{\min}(H(x_*))$. Hence, (3.19) can be written as

$$\min_{y \in W} F(y) < F(\rho(x_*)) \qquad \text{with} \qquad W := \{[v^H \widehat{A} v, v^H \widehat{B} v]^H : v^H \widehat{M} v = 1\},$$

where $\widehat{A} = U^H A U$, $\widehat{B} = U^H B U$, and $\widehat{M} = U^H U$. Due to the convexity of the numerical range W and that both $\rho(x_*), \rho(\widehat{x}) \in W$, we have the entire line segment $[\rho(x_*), \rho(\widehat{x})] \subset W$. Moreover, F is descending along the line segment $[\rho(x_*), \rho(\widehat{x})]$ as

$$\nabla F(y)|_{y=\rho(x_*)} \cdot (\rho(\widehat{x}) - \rho(x_*)) = \widehat{x}^H H(x_*)\widehat{x} - x_*^H H(x_*)x_* = \lambda_{\min}(H(x_*)) - \lambda_* < 0,$$

where we assumed x_*, $\widehat{x}$ are unitary. Hence, $\exists y \in [\rho(x_*), \rho(\widehat{x})]$, s.t. $F(y) < F(\rho(x_*))$. ∎

4. Extension to nonsmooth objective functions. We now describe sequential subspace methods for nondifferentiable objective functions. One major issue to be addressed is how to choose the search subspace. A particular element in $\partial^w \mathcal{F}(x)$ will not necessarily be descending and hence cannot be used immediately as a search direction. Fortunately, the generalized gradient (2.2), due to the special composite formulation of $\mathcal{F}$, always satisfies the inclusion relation

$$\partial^w \mathcal{F}(x) \subset \operatorname{span}\{r_A(x),\ r_B(x)\}, \tag{4.1}$$

where $r_A(x)$ and $r_B(x)$ are residuals, defined as in (3.16), for matrix pencils $A - \lambda I$ and $B - \lambda I$, respectively. Therefore, we can use the two-dimensional "descent subspace"

for the local search. More precisely, for sequential subspace search, we apply the following search subspace in analogy to (3.8):

$$x_{k+1} \in \operatorname{span}\Big\{ x_{k-1},\ x_k,\ r_A(x_k),\ r_B(x_k) \Big\}, \tag{4.2}$$

which contains the current iterate x_k, the old iterate x_{k-1}, and the descent subspace (4.1).

The best approximation x_{k+1} can be defined in the same way as (3.2) so that

$$x_{k+1} = U_k \widehat{v}_k / \|U_k \widehat{v}_k\|_2, \tag{4.3}$$

where U_k is a basis matrix of the subspace (4.2), and $\widehat{v}_k$ is a solution to the following projected NEPv by $U = U_k$:

$$\widehat{H}(v)v = \lambda \widehat{M} v \quad \text{with} \quad \widehat{H}(v) = g_1 \widehat{A} + g_2 \widehat{B} \qquad \text{for some} \quad \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \in \partial F(\widehat{\rho}(v)), \tag{4.4}$$

where λ is the smallest eigenvalue of the matrix pencil $\widehat{H}(v) - \lambda \widehat{M}$. This NEPv, unlike (3.3) in the smooth case, cannot be immediately solved by SCF iteration since $\widehat{H}(v)$ has no closed-form expression. Nevertheless, it can be written back to the convex minimization in the form of (1.1) and solved by convex optimization. For the moment, we assume the solution $\widehat{v}_k$ is always available but do not make any assumption on how it is computed. Despite the nonsmoothness in F, we can show a global convergence result similar to that of the smooth version in Theorem 3.2.

THEOREM 4.1. *The sequence $\{x_k\}_{k=1}^{\infty}$ produced by (4.3) is monotonic in function values $\mathcal{F}(x_{k+1}) \le \mathcal{F}(x_k)$ for $k = 1, \ldots,$ where equality holds only if x_k is stationary. In addition, any limiting point $\widetilde{x}$ of $\{x_k\}_{k=1}^{\infty}$ is an eigenvector of the NEPv (2.4).*

Proof. Since the current iterate x_k is included in the search subspace (4.2), the monotonicity $\mathcal{F}(x_{k+1}) \le \mathcal{F}(x_k)$ follows directly from the definition (3.2). As a result, we have $F(\rho(\widetilde{x})) = \mathcal{F}_\infty$ with $\mathcal{F}_\infty \le \lim_{k\to\infty} \mathcal{F}(x_k)$.

For global convergence, it suffices to consider the case where $\rho(\widetilde{x})$ is a nonsmooth point of F, since otherwise Theorem 3.2 applies.

We first show that for any nonstationary $\widetilde{x}$, it holds strictly that $\mathcal{F}(\widetilde{x}) < \min_{x\in\mathcal{U}} \mathcal{F}(x)$ with $\mathcal{U} := \operatorname{span}\{\widetilde{x}, r_A(\widetilde{x}), r_B(\widetilde{x})\}$. Assume otherwise; then $\mathcal{F}(\widetilde{x}) = \min_{x\in\mathcal{U}} \mathcal{F}(x)$. Let U be an orthogonal basis of $\mathcal{U}$; then $\widetilde{v} = U^H \widetilde{x}$ is a global minimizer of the projected $\widehat{\mathcal{F}}(v)$ in (3.2). By Theorem 2.1, $\widetilde{v}$ satisfies the NEPv $\widehat{H}(\widetilde{v})\widetilde{v} = \widetilde{\lambda}\widetilde{v}$; namely, it holds for some $g \in \partial F(\rho(\widetilde{x}))$ that

$$0 = (g_1 \cdot U^H A U + g_2 \cdot U^H B U - \widetilde{\lambda} I)\widetilde{v} = U^H (g_1 A + g_2 B - \widetilde{\lambda} I)\widetilde{x}.$$

Since $(g_1 A + g_2 B - \widetilde{\lambda} I)\widetilde{x} = g_1 r_A(\widetilde{x}) + g_2 r_B(\widetilde{x}) \in \mathcal{U}$, we have $(g_1 A + g_2 B - \widetilde{\lambda} I)\widetilde{x} = 0$. Hence $\widetilde{x}$ is an eigenvector of the NEPv (2.4), contradicting $\widetilde{x}$ being nonstationary.

Therefore, for a nonstationary $\widetilde{x}$, it holds that $\exists\ \widehat{x} \in \operatorname{span}\{\widetilde{x}, r_A(\widetilde{x}), r_B(\widetilde{x})\}$, s.t. $\|\widehat{x}\|_2 = 1$ and $\mathcal{F}(\widehat{x}) = \mathcal{F}(\widetilde{x}) - \delta$ for some constant $\delta > 0$. Since $\widetilde{x}$ is a limiting point, we obtain by the continuity of $r_A(x)$, $r_B(x)$ that

$$\forall \varepsilon > 0,\ \exists\, x_k, \quad \text{s.t.} \quad [\widetilde{x}, r_A(\widetilde{x}), r_B(\widetilde{x})] = [x_k, r_A(x_k), r_B(x_k)] + E \quad \text{and} \quad \|E\|_2 \le \varepsilon,$$

we have $\widehat{x} = \widehat{x}_k + \mathcal{O}(\varepsilon)$ for some $\widehat{x}_k \in \operatorname{span}\{[x_k, r_A(x_k), r_B(x_k)]\}$. On the other hand,

$$\mathcal{F}(x_{k+1}) \le \min_{x\in\operatorname{span}\{[x_k, r_A(x_k), r_B(x_k)]\}} \mathcal{F}(x) \le \mathcal{F}(\widehat{x}_k) \xrightarrow{\varepsilon\to 0} \mathcal{F}(\widehat{x}) = \mathcal{F}_\infty - \delta,$$

where we used the continuity of $\mathcal{F}(x)$. This contradicts $\lim_{k\to\infty} \mathcal{F}(x_k) \ge \mathcal{F}_\infty$. ∎

We make two comments on the search scheme (4.2), in comparison with its smooth counterpart (3.8). First, for a differentiable F function, we can also apply the extended subspace (4.2). In practice, this results in only a marginal improvement over the single vector formula (3.8) regarding the approximation error but increases the computation cost due to a larger projection basis. For efficiency considerations, the search subspace (3.8) is always recommended in the smooth case. Second, since the matrix $H(x_k)$ is not uniquely defined at a nonsmooth point, it is generally difficult to apply the global verification for a computed eigenvector x_k as in subsection 3.3.2.

5. Implementation issues. In this section, we discuss implementation details for the proposed algorithms. The recurrence relations (3.8) and (4.2) allow for efficient implementation. Many techniques developed for LOBPCG (see, e.g., [25]) can be applied in parallel to our nonlinear version, including the use of difference vectors to improve the condition number of basis matrices, and block iteration to enhance performance. For simplicity of exposition, those techniques will be explained for the smooth function F, and their application to the nonsmooth case is straightforward.

Difference vectors. In the basis matrix U_k of (3.10), the vectors x_{k-1} and x_k tend to be linearly dependent upon convergence. To avoid ill-conditioned basis matrices, we can introduce the difference vector $p_k = x_k - x_{k-1}$ and use $U_k = [x_k, p_k, r_k]$ instead. We can keep p_k in the memory and update the next p_{k+1} from p_k directly.

Block iteration. Block iterations are widely used in eigenvalue computation. The idea is to iterate over ℓ vectors $X_k = [x_k^{(1)}, \ldots, x_k^{(\ell)}]$ simultaneously rather than a single x_k. This is appealing from an efficiency perspective for particular computer architectures; in addition there are several other benefits such as ease of parallelization and faster convergence by extended search subspace. For a straightforward block implementation of (3.8), we take the leading $x_k^{(1)}$ to serve as the approximate eigenvector, and we take the remaining $x_k^{(2)}, \ldots, x_k^{(\ell)}$ as auxiliary vectors. The recurrence (3.8) can be written as

$$X_{k+1} \in \operatorname{span}\{[X_{k-1}, X_k, R_k]\},$$

where "$\in$" holds columnwisely, and $R_k = [r_k^{(1)}, \ldots, r_k^{(\ell)}]$ consists of the residuals

$$r_k^{(j)} = H\left(x_k^{(1)}\right) \cdot x_k^{(j)} - \mu_k\left(x_k^{(j)}\right) \cdot x_k^{(j)} \quad \text{with} \quad \mu_k(x) = \frac{x^H H(x_k^{(1)}) x}{x^H x} \tag{5.1}$$

for $j = 1, \ldots, \ell$. This is a natural generalization of the formula (3.9) to the block version, with the leading $r_k^{(1)} = \nabla^w \mathcal{F}(x_k^{(1)})/2$ corresponding to the gradient at the optimizer $x_k^{(1)}$. In analogy to the single vector version, we can apply the nonlinear Rayleigh–Ritz procedure with $U = [X_{k-1}, X_k, R_k]$ and define the new block as

$$X_{k+1} = [X_{k-1}, X_k, R_k] V_{k+1}, \tag{5.2}$$

where V_{k+1} consists of eigenvectors for the ℓ smallest eigenvalue of the linear problem $\widehat{H}(\widehat{v}_k)v = \lambda \widehat{M} v$, defined by the solution $\widehat{v}_k$ of the reduced NEPv. The first column of V_{k+1} is set to $\widehat{v}_k$, so that $x_{k+1}^{(1)}$ is the best approximation by the Rayleigh–Ritz procedure.

We summarize in Algorithm 5.1 the actual sequential subspace algorithm, with a few implementation details listed as follows.

(a) Matrix-vector multiplications (MatVec) $G[X_k, P_k, R_k]$ for $G \in \{A, B\}$ are required in lines 2, 5, 10, and 13. To avoid recalculations, we can precompute and save,

Algorithm 5.1. Sequential subspace search for NEPv (2.4).

Input: Coefficient matrices A and B, block size ℓ, starting vectors $X_0 \in \mathbb{C}^{n\times\ell}$ and $P_0 = 0^{n\times\ell}$, tolerance `tol`.

Output: Approximate eigenvector $x_k^{(1)} = X_k(:,1)$.

1: **for** $k = 0, \ldots$ **do**
2: Compute Rayleigh quotients $y_k^{(j)} = \rho(x_k^{(j)})$ for $j = 1, \ldots, \ell$.
3: **if** F is differentiable **then** {% smooth case}
4: Compute gradient $g_k = \nabla F(y_k^{(1)})$, and $\mu_k^{(j)} = g_k^T y_k^{(j)}$ for $j = 1, \ldots, \ell$.
5: Compute residual vectors $r_k^{(j)} = \big(H(x_k^{(1)}) - \mu_k^{(j)} I\big)x_k^{(j)}$ for $j = 1, \ldots, \ell$.
6: Convergence check: **if** $\|r_k^{(1)}\|_2 \le$ `tol`, **then** break.
7: (*Optional*, $\ell \ge 2$) Preconditioned expansion: $R_k := [r_k^{(1)}, \ldots, r_k^{(\ell-2)} | \, p_k^{(a)}, p_k^{(b)}]$ by (3.14) or (3.15).
8: **else** {% nonsmooth case; require $\ell = 2s$ is even}
9: Convergence check: **if** $k \ge 1$ and $F(y_k^{(1)}) \ge F(y_{k-1}^{(1)})$ **then** return.
10: Set residuals $R_k = [r_A(x_k^{(1)}), r_B(x_k^{(1)}), \ldots, r_A(x_k^{(s)}), r_B(x_k^{(s)})]$ by (3.16).
11: **end if**
12: Orthogonalize $R_k = \mathrm{orth}(R_k)$.
13: Let $U_k = [X_k, P_k, R_k]$, set $\widehat{A} = U_k^H A U_k$, $\widehat{B} = U_k^H B U_k$, and $\widehat{M} = U_k^H U_k$.
14: Solve the reduced NEPv ((3.3) for smooth F and (4.4) for nonsmooth F) for the eigenvector $\widehat{v}_k$ and $\widehat{H}(\widehat{v}_k)$.
15: Find eigenvectors $V_{k+1} = [\widehat{v}_k, \widehat{v}_k^{(2)}, \ldots, \widehat{v}_k^{(\ell)}]$ corresponding to the ℓ smallest eigenvalues of $\widehat{H}(\widehat{v}_k) - \lambda\widehat{M}$.
16: Update $X_{k+1} = [X_k, P_k, R_k] \cdot V_{k+1}$.
17: Update $P_{k+1} = [P_k, R_k] \cdot V_{k+1}(\ell+1 : 3\ell, :)$.
18: **end for**
19: (*Optional*) Global verification: compute the smallest eigenvalue $\lambda_{\min}$ and the eigenvector x of $H(x_k^{(1)})$. If $\lambda_{min} < \mu_k^{(1)}$ then restart the process with X_0 and P_0 satisfying $X_0(:,1) = x$ and $P_0(:,1) = x - x_k^{(1)}$.

with 4ℓ auxiliary vectors, the results of $A_k := A[X_k, P_k]$ and $B_k := B[X_k, P_k]$. Then the projection in line 13 only needs 2ℓ extra MatVecs:

$$A_k^r := A \cdot R_k \qquad \text{and} \qquad B_k^r := B \cdot R_k. \tag{5.3}$$

In the next iteration, A_k and B_k can be updated (using lines 16 and 17) by

$$A_{k+1} = [A_k, A_k^r]X_k \qquad \text{and} \qquad B_{k+1} = [B_k, B_k^r]X_k \tag{5.4}$$

with

$$X_k = \begin{bmatrix} V_{k+1}, & \begin{bmatrix} 0 \\ V_{k+1}(\ell+1 : 3\ell, :) \end{bmatrix} \end{bmatrix}.$$

In this way, only 2ℓ MatVecs in (5.3) are required in each iteration. In practice, to keep numerical error from accumulating in the updating formula (5.4) as k increases, both matrices are recalculated explicitly every N_{rec} iterations; e.g., $N_{rec} = 50$, as used in our numerical experiments.

(b) In line 6, the residual norm of $r_k^{(1)} = \nabla^w \mathcal{F}(x_k^{(1)})/2$ (i.e., the gradient) is used for the stopping criteria. As is common practice in numerical optimization, we can take `tol` $= \mathcal{O}(\sqrt{\epsilon_{mach}})$ to be on the order of square-root machine precision ϵ_{mach}.

But this scheme cannot be applied for the nonsmooth case, where the gradient is not defined. In line 9, we stop the algorithm when the function values $\mathcal{F}(x_k)$ lose monotonicity—an indication of accuracy close to machine precision.

(c) For efficiency considerations, we implement the preconditioned extension in line 7 by replacing the last two columns of R_k with $p_k^{(a)}$ and $p_k^{(a)}$ (assuming the block size $\ell \geq 3$). In this way, the size of search subspace (i.e., $[X_k, P_k, R_k]$) averaged by the number of MatVecs (see (a) and (5.3)) in each iteration is

$$\frac{\text{subspace size}}{\#\text{MatVec}} = \frac{3\ell}{2\ell} = \frac{3}{2}. \tag{5.5}$$

In comparison, for the naive implementation of appending $p_k^{(a)}$, $p_k^{(a)}$ directly to $R_k := [R_k, p_k^{(a)}, p_k^{(a)}]$, the algorithm would use a search subspace $[X_k, P_k, R_k]$ of size $3\ell + 2$ and require $2(\ell + 2)$ MatVecs in (5.3). So the ratio in (5.5) becomes $(3\ell + 2)/2(\ell + 2) < 3/2$. Namely, by the same number of MatVecs, the latter strategy can only search in a smaller subspace.

(d) For the same reason as in (c), we use the s leading r_A and r_B vectors to define R_k for nonsmooth F (line 10), assuming the block size $\ell = 2s$ is an even number.

(e) For a smooth F, the reduced NEPv in line 14 can be solved by the SCF iteration in Algorithm 3.1. For a nonsmooth F, convex optimization can be employed, where the exact algorithm is case dependent but usually cheap to apply for small-size problems; see, e.g., subsection 6.2 for discussions on the max-ratio minimization.

6. Numerical experiments. In this section, we demonstrate the performance of the proposed algorithms with several numerical examples. In subsection 6.1, we discuss a p-norm distance problem of numerical range, with applications to coercivity constant computation for boundary element operators. In subsection 6.2, we consider a nondifferentiable max-ratio minimization problem. All experiments are done in MATLAB 2017b and run on an HP machine with Intel(R) Xeon(R) CPU E5–2620 v3 @ 2.40GHz and 264 GB memory. No parallelization is applied.

6.1. Smooth objective function. Consider the minimization problem (1.1) with the objective function $F(y) = \|y\|_p$ with $p > 1$. Geometrically, the minimal value defines the p-norm distance from the origin $(0, 0)$ to the numerical range $W(A, B)$. For the case of $p = 2$, the problem is known as the Crawford number computation, where the problem also admits the eigenvalue optimization formula

$$\min_{y \in W(A,B)} \|y\|_2 = \left(\max \left\{ \max_{\theta \in (0, 2\pi]} \lambda_{\min}(A \sin\theta + B \cos\theta), \quad 0 \right\} \right); \tag{6.1}$$

see, e.g., [11, Thm. 2.1] and [19, eq. (2.8)]. In our experiment, we will also consider a general $p = 1.1$ problem, where no eigenvalue optimization formula such as (6.1) is available. It is clear that both F's are convex and differentiable for all $y \neq 0$. To apply Algorithm 5.1, we set the tolerance $\texttt{tol} = 10^{-8} \approx \sqrt{\epsilon_{mach}}$ and solve the reduced NEPv in line 14 by the SCF iteration in Algorithm 3.1 with $\texttt{tol\_r} = \texttt{tol}$ and $\texttt{maxit} = 30$.

Example 1. This is a small-size example to show the convergence of Algorithm 5.1 without preconditioned expansion. The testing matrices are given by

$$A = \cos(\pi/3)G - 4I_n, \qquad B = \sin(\pi/3)G - 2I_n,$$

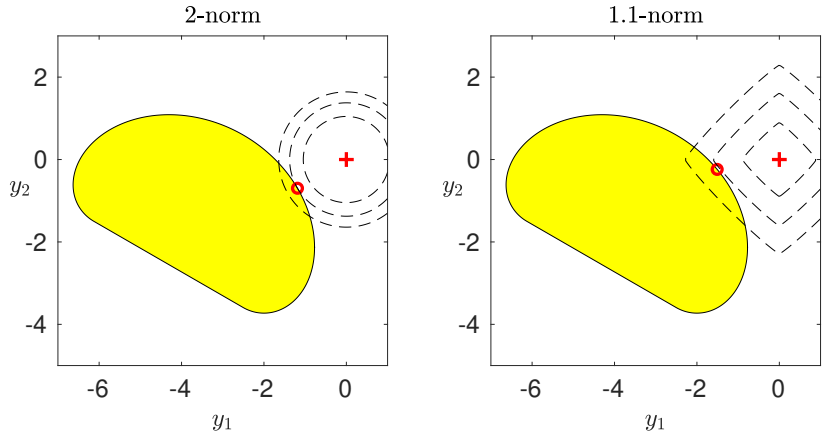

FIG. 1. *Numerical range and the p-norm distance to* $(0,0)$*: 2-norm on the left and* 1.1*-norm on the right. The nearest points are marked as* $\circ$*, with dashed lines being the contour plot of* F*.*

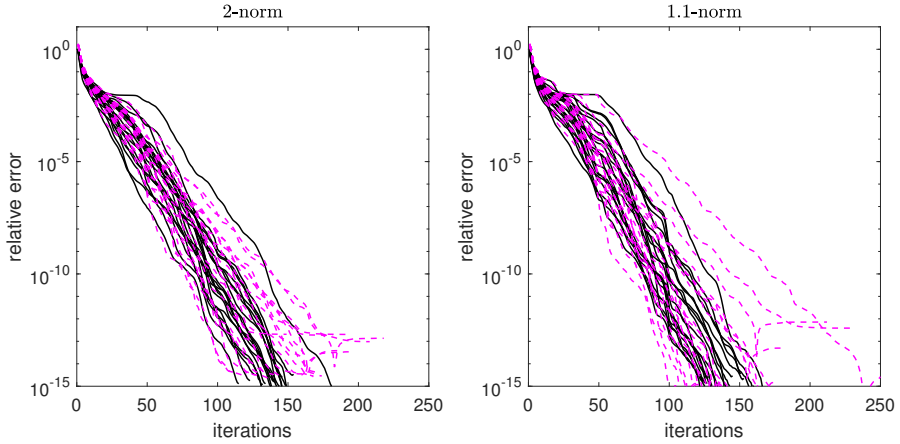

FIG. 2. *The p-norm distance problem in Example* 1*. Convergence history for* 20 *randomly generated starting vectors: Algorithm* 5.1 *for NEPv* (2.4) *(solid) and LOBPCG for LEP* (3.17) *(dashed).*

where G is a Grcar matrix of size $n = 120$. The numerical range $W(A, B)$ and the contours of F corresponding to $p = 2$ and 1.1 are depicted in Figure 1.

In the first experiment, we apply Algorithm 5.1 with the block size $\ell = 1$ and randomly generated starting vectors (normally distributed elements with 0 mean and unit variance). The convergence history is reported in Figure 2, where the relative error is measured by

$$|\mathcal{F}(x_k) - \mathcal{F}(x_*)|/|\mathcal{F}(x_*)|,$$

and the "exact" minimizer x_* is computed by SCF iterations applied to the NEPv (2.4).

For comparison, we also depict the convergence history of LOBPCG[3] for solving the LEP (3.17), where the relative error is measured by $|\lambda_k - \lambda_*|/|\lambda_*|$ with λ_* computed by MATLAB function `eig`. Note that this can be regarded as verification

[3]MATLAB code available at https://mathworks.com/matlabcentral/fileexchange/48-lobpcg-m.

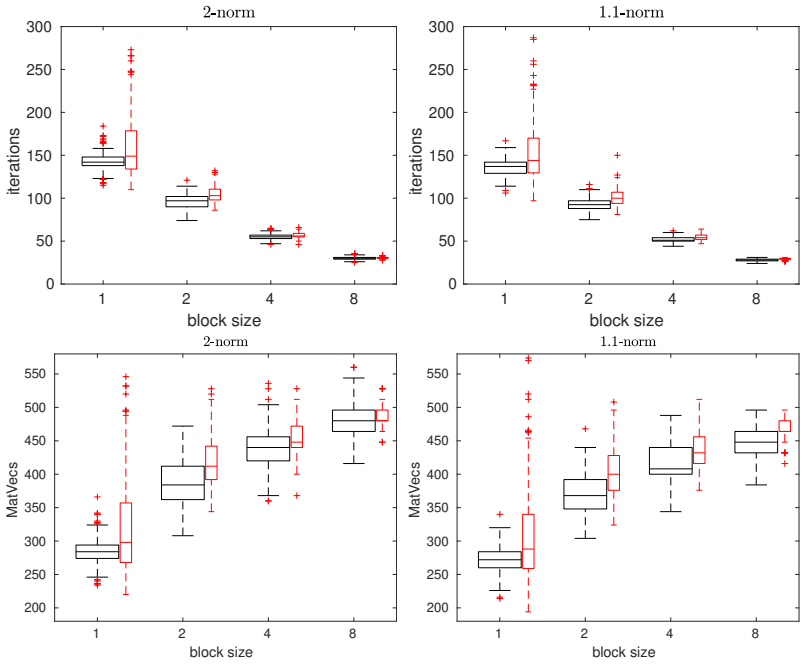

FIG. 3. *The p-norm distance problem in Example* 1. *Boxplots of number of iterations (top) and number of matrix vector multiplications (bottom) for Algorithm* 5.1 *(marked as black larger boxes), and LOBPCG (marked as red smaller boxes) with block size* $p = 1, 2, 4, 8$. *Statistics collected from* 200 *repeated experiments for each* p *with random starting vectors.*

of global optimality for a given x_*; see, e.g., subsection 3.3.2. From the reported result, Algorithm 5.1 solved an NEPv (2.4) in a number of iterations comparable to LOBPCG for an LEP, and the convergence of Algorithm 5.1 also seems less sensitive to the choice of initial vectors.

In our second experiment, we test with block size $\ell = 1, 2, 4, 8$. For each ℓ, the algorithms are repeatedly applied 200 times using random starting vectors. The computation results are reported in Figure 3. We can observe a great reduction in the numbers of iterations and their variance as the block size ℓ grows, whereas the total number of MatVecs slightly increases since each iteration needs more MatVecs. The performances are also similar to LOBPCG for the LEP (3.17).

Example 2. We consider the preconditioned expansion schemes discussed in subsection 3.3.1. For the testing, Algorithm 5.1 is applied to the same problem as in Example 1, with the *preconditioned expansion* in line 7 turned on. In Figure 4, we illustrate in the left plot the superlinear convergence when (3.14) is applied, which is consistent with our theoretical analysis in Theorem 3.3. In the other two plots, we report the performance of inexact expanding vectors, computed by a few steps of MATLAB `gmres` (only the 2-norm case is reported, the performance for $p = 1.1$ is similar). Both formulas (3.14) and (3.15) are tested. The former seems more affected by the error in the expanding vectors; it generally converges faster when the accuracy increases.

Example 3. In this example, we consider the problem of computing the coercivity constant for boundary integral operators in acoustic scattering [5, 26]. The major

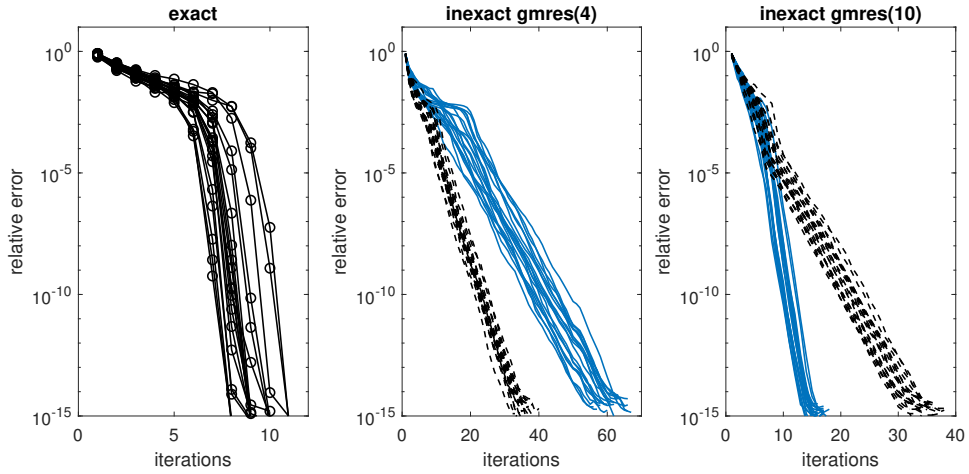

FIG. 4. *Convergence history of preconditioned expansion for the* 2*-norm problem with* 20 *random starting vectors. Left: Formula* (3.14)*. Middle and right: Formulas* (3.14) *(solid) and* (3.15) *(dashed), with expansion vectors computed inexactly by running* 4 *and* 10 *steps of GMRES, respectively.*

computation task is to evaluate

$$\gamma(L) := \min_{u \neq 0} \frac{|u^H L u|}{u^H u} = \min_{u \neq 0} \left(\left| \frac{u^H A u}{u^H u} \right|^2 + \left| \frac{u^H B u}{u^H u} \right|^2 \right)^{1/2},$$

where $A = (L + L^H)/2$, $B = (L - L^H)/2\jmath$, and L is a discrete boundary integral operator (matrix) of

$$a_\nu(u, v) = \int_\Gamma B_\nu u(y) \cdot \overline{v(y)} \, \mathrm{d}s(y), \quad \text{with} \quad B_\nu = I + K_\nu - \jmath \nu S_\nu,$$

where $\nu > 0$ is the wave number, Γ is the boundary of a sound-soft bounded obstacle in $\mathbb{R}^3$, $u(x), v(x) \in L^2(\Gamma)$, I is the identity operator, and K_ν and S_ν are defined by

$$K_\nu u(x) = 2 \int_\Gamma \frac{\partial \Phi(x, y)}{\partial n(x)} u(y) \, \mathrm{d}s(y), \qquad S_\nu u(x) = 2 \int_\Gamma \Phi(x, y) u(y) \, \mathrm{d}s(y), \qquad x \in \Gamma.$$

Here, $\Phi(x, y) = e^{\jmath \nu |x-y|}/(2\pi |x - y|)$ for $x, y \in \mathbb{R}^3$, $x \neq y$, and $n(x)$ is the outpoint unit normal at Γ.

In the following test, we consider two different geometries of Γ shown in Figure 5 (smooth and nonsmooth) as well as three wave numbers $\nu = 1, 2, 5$. Each coefficient matrix $L^{(\nu)}$ is generated by the Galerkin boundary element library `BEM++`,[4] with triangular mesh (generated by `gmsh`[5] with Delaunay's algorithm) and piecewise linear basis functions; see [40] for details about the discretization method. For convenience, we use a relatively coarse mesh size $h = 0.1$.

To solve the problem, we applied Algorithm 5.1 with block size $\ell = 3$. Both the standard and preconditioned versions (labeled `pcd`) are tested, where the preconditioning formula (3.15) is applied with $T_k \approx (H(x_k) - \lambda_k I)^{-1}$ using

$$T_k = (A_{\mathcal{H}} \cdot F_1(x_k) + B_{\mathcal{H}} \cdot F_2(x_k) - \lambda_k I)^{-1},$$

[4]The software is available from http://www.bempp.org/.
[5]The software is available from http://gmsh.info/.

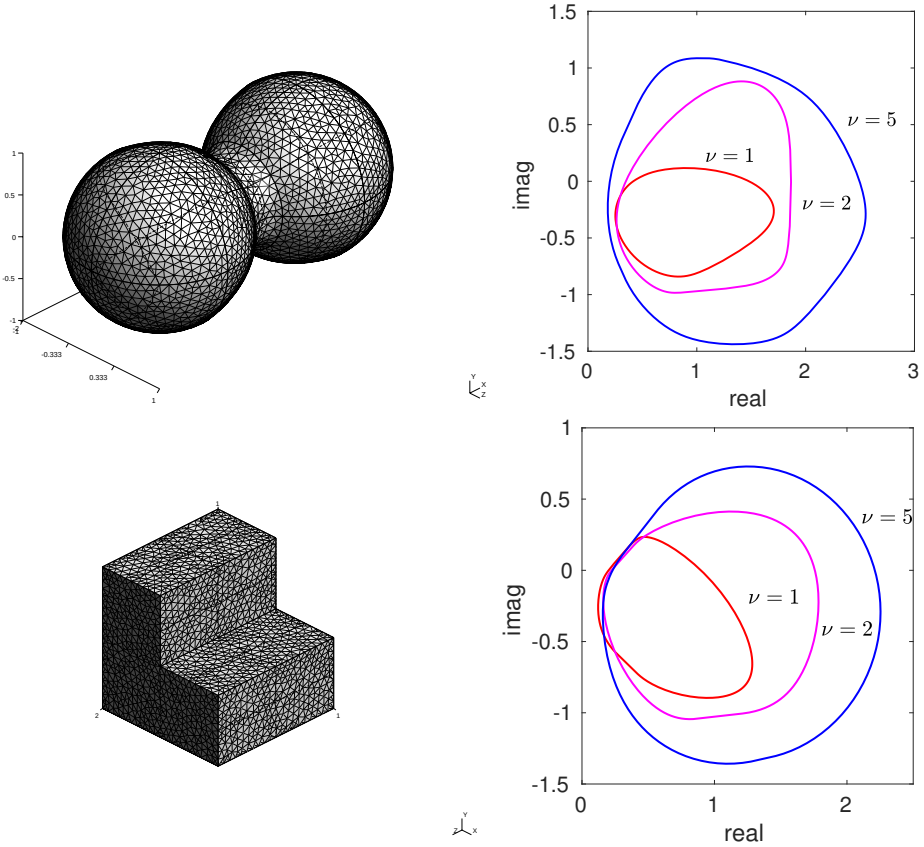

FIG. 5. *The geometry of* Γ *and the corresponding numerical range of the discrete operator* $L^{(\nu)}$.

where $\lambda_k = (x_k^H H(x_k)x_k)/(x_k^H x_k)$, and $A_{\mathcal{H}}$ and $B_{\mathcal{H}}$ are hierarchical matrix approximations [17] of A and B (generated by `BEM++` using a truncation threshold 0.1). To reduce cost, we apply $T_k x$ by running 10 steps of GMRES instead of solving exactly. This can be done very efficiently since each $A_{\mathcal{H}}x$ (and $B_{\mathcal{H}}x$) takes around 10% of the timing of Ax (and Bx). We repeated the experiment 15 times with random starting vectors. The iteration number and timing statistics (mean and max deviation over the repeated experiments) are reported in Tables 1 and 2.

For comparison, we have also applied the full subspace algorithm developed in [26], which is based on the eigenvalue optimization formula in (6.1). In each iteration, it solves a Hermitian LEP for the smallest eigenvalue, done by LOBPCG[6] using the same block size, $\ell = 3$, as Algorithm 5.1. For comparison, we have also applied LOBPCG with preconditioner (using T_k from above with λ_k replaced by an underestimate of the Crawford number available in the full subspace algorithm). From Tables 1 and 2, the algorithm is significantly accelerated by preconditioning although still slower than Algorithm 5.1. In the best case, the full subspace algorithm converges in fewer MatVecs

[6]For efficiency of the full subspace algorithm, we use the eigenvectors from the last iteration to initialize each call to LOBPCG. Since the inner eigenvalue problems need not be solved accurately, we start with a low tolerance 10^{-6} for LOBPCG and gradually increase it to 10^{-12} upon convergence of the full subspace algorithm.

TABLE 1
Computation results for Example 3: Peanut-shaped domain; size $n = 18,060$.

ν		$\gamma(L^{(\nu)})$	its	MatVec	Timing (s)
1	Alg. 1 in [26]	3.231643542030261E-01	5	1188	1810
	Alg. 1 in [26](pcd)	3.231643542030258E-01	5	366	839
	Alg. 5.1	3.231643542030256E-01 (±1E-15)	47(±4)	286(±28)	426 (±40)
	Alg. 5.1(pcd)	3.231643542030258E-01 (±3E-16)	15(±4)	88(±22)	175 (±40)
2	Alg. 1 in [26]	3.227569492274415E-01	5	822	1277
	Alg. 1 in [26](pcd)	3.227569492274416E-01	5	252	592
	Alg. 5.1	3.227569492274416E-01 (±4E-16)	62(±6)	378(±36)	567 (±54)
	Alg. 5.1(pcd)	3.227569492274415E-01 (±3E-16)	15(±2)	94(±16)	182 (±23)
5	Alg. 1 in [26]	1.985483883517590E-01	6	1032	1168
	Alg. 1 in [26](pcd)	1.985483883517590E-01	6	504	597
	Alg. 5.1	1.985483883517627E-01 (±3E-15)	56(±6)	342(±36)	503 (±61)
	Alg. 5.1(pcd)	1.985483883517589E-01 (±3E-16)	15(±2)	93(±12)	183 (±24)

TABLE 2
Computation results for Example 3: L-shaped domain; size $n = 16,116$.

ν		$\gamma(L^{(\nu)})$	its	MatVec	Timing (s)
1	Alg. 1 in [26]	1.719991140659281E-01	6	2562	3108
	Alg. 1 in [26](pcd)	1.719991140659281E-01	6	474	907
	Alg. 5.1	1.719991140659368E-01(±7E-15)	81(±13)	490(±76)	588(±89)
	Alg. 5.1 (pcd)	1.719991140659283E-01(±3E-16)	22(±3)	130(±16)	212(±26)
2	Alg. 1 in [26]	1.950697449793223E-01	6	3126	3767
	Alg. 1 in [26](pcd)	1.950697449793218E-01	6	528	1022
	Alg. 5.1	1.950697449793266E-01(±3E-15)	104(±13)	634(±82)	762(±95)
	Alg. 5.1 (pcd)	1.950697449793223E-01(±4E-16)	23(±2)	138(±12)	231(±19)
5	Alg. 1 in [26]	2.100416346643941E-01	7	1896	2334
	Alg. 1 in [26](pcd)	2.100416346643939E-01	7	456	897
	Alg. 5.1	2.100416346643952E-01(±6E-16)	80(±8)	486(±48)	585(±57)
	Alg. 5.1 (pcd)	2.100416346643941E-01(±3E-16)	27(±3)	160(±16)	267(±25)

than Algorithm 5.1 (no preconditioning), but each of its MatVecs is more costly due to the extra preconditioning step.

6.2. Nonsmooth objective function. We consider the max-ratio minimization problem with $F(y) := \max\{y_1, y_2\}$, which is nondifferentiable at y with $y_1 = y_2$. To apply Algorithm 5.1, the reduced problem in line 14 will be solved by the equivalent eigenvalue optimization problem in the following lemma, for which a similar result can be found in [15].

LEMMA 6.1. *Given Hermitian matrices A and $B \in \mathbb{C}^{n\times n}$, the following hold.*

(a) *It holds that*

$$\min_{x\in\mathbb{C}^n} \max\left\{\frac{x^HAx}{x^Hx}, \frac{x^HBx}{x^Hx}\right\} = \max_{t\in[0,1]} \lambda_{\min}\Big(tA + (1-t)B\Big). \tag{6.2}$$

(b) *Let t_* be an optimizer of (6.2) and x_* be the eigenvector corresponding to the smallest eigenvalue λ_* of*

$$H(x_*) := A(1-t_*) + Bt_*. \tag{6.3}$$

It holds that $\left[\begin{smallmatrix} t_* \\ 1-t_* \end{smallmatrix}\right] \in \partial F(\rho(x_*))$ for $F(y) := \max\{y_1, y_2\}$.

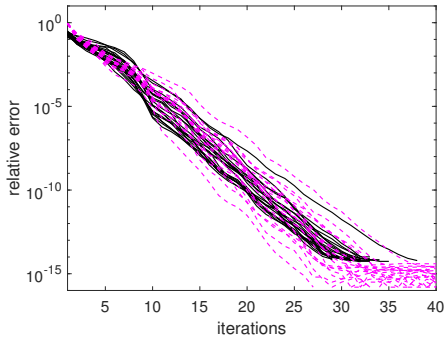

FIG. 6. *Convergence history with* 20 *randomly generated starting vectors: Algorithm* 5.1 *for the NEPv* (2.4) *(solid); LOBPCG for computing the smallest eigenvalue of* $H(x_*)$ *in* (6.3) *(dashed).*

Proof. The proof is deferred to Appendix B. ∎

The eigenvalue optimization (6.2) is convex in $t \in \mathbb{R}$ [35]; therefore, it can be conveniently solved, at least for problems with a small-size n, by golden section search (e.g., the MATLAB function `fminbd`) or other eigenvalue optimization techniques (see, e.g., [31]). In our implementation, we adopt the level-set based criss-cross search [6] for the maximizer, which works quite well in the numerical experiments.

For the testing problem, we consider a multicast transmit beamforming problem [39] in a simple setting in which the base station sends a common signal to two receivers a and b using n antennas, where a max-ratio minimization problem has to be solved. A detailed description of the coefficient matrices can be found in Appendix C. For convenience, we use the block size $\ell = 2$ in Algorithm 5.1 for the computation.

Example 4. In the first experiment, we consider a small-size problem of $n = 120$. The Rayleigh quotients at the optimizer $\widehat{x}$, computed by solving the eigenvalue optimization (6.2) directly, are given by

$$\frac{\widehat{x}^H A\widehat{x}}{\widehat{x}^H\widehat{x}} = -11.27112794653678, \quad \frac{\widehat{x}^H B\widehat{x}}{\widehat{x}^H\widehat{x}} = -11.27112794653939.$$

Hence, $\widehat{x}$ is a numerically nonsmooth point of $\mathcal{F}(x)$. The convergence history of Algorithm 5.1 is reported in Figure 6, where for the relative error the "exact" eigenvalue is computed using the eigenvalue optimization formula (6.2). For comparison we also applied LOBPCG (with the same block size of 2) to compute the smallest eigenvalue of the Hermitian matrix $H(x_*)$ at the optimizer in (6.3). Again we observe a similar convergence rate for both algorithms.

Example 5. We now test with larger problem sizes $n = 1000, 2000, 4000$. In the experiment, we treat matrices A and B as linear operators and only allow for matrix-vector (or matrix-matrix) multiplication in computation. The testing results for 20 repeated experiments with random starting vectors are reported in Table 3. Recall that the max-ratio minimization problem can also be reformulated and solved as an eigenvalue optimization problem in (6.2). For comparison, we applied the `leigopt`[7] to obtain the solution (with convergence tolerance `tol` $= 10^{-13}$). This algorithm is based on the subspace framework presented in [23], and in each iteration it needs to solve a large-scale LEP, which is done by the MATLAB function `eigs`.

[7] Available at http://home.ku.edu.tr/~emengi/software.html. We slightly modified the calling of the `eigs` function to accept linear operators as input.

In all testing cases, Algorithm 5.1 used fewer MatVecs, which accounts for the dominant cost for both algorithms as the problem size n grows. For `leigopt`, the percentages of time spent for MatVecs are about 72%, 84%, and 90% for n to be 1000, 2000, and 4000, respectively. From Table 3, we can also observe that the saving of Algorithm 5.1 in computing time is more than that in the number of MatVecs. Take $n = 2000$, for example; the ratio between the MatVecs of Algorithm 5.1 and `leigopt` is $1770/3082 \approx 0.57$, whereas that for the computation time is $7.7/23.3 \approx 0.33$. This difference is largely because of the block operations: The 1770 MatVecs (on average) in Algorithm 5.1 are executed as approximately $1770/2 = 885$ number of matrix-matrix multiplications $A \cdot R_k$ and $B \cdot R_k$, rather than 1770 sequential MatVecs. On a hierarchical memory machine, the block multiplication is more efficient since it uses a level-3 BLAS operation. Such a block operation, however, is not exploited by the MATLAB function `eigs` in `leigopt`.

TABLE 3
Computation results for Example 5.

n		Optimal value	its	MatVec	Timing (s)
1000	leigopt	-1.15337555620605E+01	6	1772	3.0
	Alg. 5.1	-1.15337555620603E+01(±1E-13)	229(±87)	903(±357)	1.4 (±0.5)
2000	leigopt	-1.15372647515872E+01	6	3082	23.3
	Alg. 5.1	-1.15372647515869E+01(±4E-13)	435(±81)	1770(±330)	7.7 (±1.7)
4000	leigopt	-1.15381560642041E+01	7	5760	159.8
	Alg. 5.1	-1.15381560642033E+01(±1E-12)	809(±258)	3295(±1053)	49.7(±16)

7. Conclusions. We studied a convex minimization problem over the joint numerical range of a pair of Hermitian matrices. For such problems, we have established a nonlinear eigenvalue problem characterization for the global optimizer. Iterative methods based on locally optimal subspace search were introduced, with both smooth and nonsmooth objective functions considered. The convergence of the algorithms, as well as their implementation details, were also discussed. The effectiveness and efficiency of the proposed nonlinear eigenvector approach have been demonstrated by numerical examples for computing the coercivity constant of boundary integral operators and solving multicast beamforming problems.

The theory and algorithms considered in this paper can be naturally extended to convex minimization over the joint numerical range of a d-tuple of Hermitian matrices $A_i \in \mathbb{C}^{n\times n}$ for $i = 1, \ldots, d$,

$$W(A_1, A_2, \ldots, A_d) = \left\{ \left(x^H A_1 x, x^H A_2 x, \ldots, x^H A_d x\right) : x \in \mathbb{C}^n,\ \|x\|_2 = 1 \right\},$$

provided that the considered joint numerical range is a convex set in $\mathbb{R}^d$. Such an assumption holds in particular in the case of $d = 3$ and $n \geq 3$ as shown in [14], while for more general cases of $d \geq 3$, the convexity can hold in certain conditions (see, e.g., [29, 16]). Under the convexity assumption, most of the results in this paper can be applied, but a detailed technical treatment is beyond the scope of this paper and is left for future research.

Appendix A. Proof of Theorem 3.3. We will use boldface letters for the

augmented real variables of a complex vector $x \in \mathbb{C}^n$ and a matrix $X \in \mathbb{C}^{n\times n}$:

$$\text{(A.1)}\quad \mathbf{x} = \begin{bmatrix}\mathrm{Re}(x)\\ \mathrm{Im}(x)\end{bmatrix},\ \overline{\mathbf{x}} = \begin{bmatrix}-\mathrm{Im}(x)\\ \mathrm{Re}(x)\end{bmatrix} \in \mathbb{R}^{2n},\ \text{and } \mathbf{X} = \begin{bmatrix}\mathrm{Re}(X) & -\mathrm{Im}(X)\\ \mathrm{Im}(X) & \mathrm{Re}(X)\end{bmatrix} \in \mathbb{R}^{2n\times 2n}.$$

Let $y = Xx$ and $z = \jmath Xx$; it is straightforward to verify that $\mathbf{y} = \mathbf{X}\mathbf{x}$ and $\mathbf{z} = \mathbf{X}\overline{\mathbf{x}}$.

Let $\lambda = \alpha + \jmath\beta$. By separating the real and imaginary parts of the NEPv (2.4) and the normalization $x_*^H x = 1$, the eigenpair (λ_*, x_*) satisfies the augmented real system

$$\mathbf{G}(\mathbf{x}, \alpha, \beta) := \begin{bmatrix}\mathbf{H}(\mathbf{x})\mathbf{x} - \mathbf{\Lambda}\mathbf{x}\\ 1 - \mathbf{x}_*^T\mathbf{x}\\ 0 - \overline{\mathbf{x}}_*^T\mathbf{x}\end{bmatrix} = 0,$$

where

$$\mathbf{\Lambda} = \begin{bmatrix}\alpha I_n & -\beta I_n\\ \beta I_n & \alpha I_n\end{bmatrix} \quad \text{and} \quad \mathbf{H}(\mathbf{x}) = F_1(\rho(\mathbf{x}))\cdot\mathbf{A} + F_2(\rho(\mathbf{x}))\cdot\mathbf{B}$$

with $\rho(\mathbf{x}) = \left[\mathbf{x}^T\mathbf{A}\,\mathbf{x}, \mathbf{x}^T\mathbf{B}\,\mathbf{x}\right]^T$. Let $(x_k, \alpha_k + \jmath\beta_k)$ with $x_*^H x_k = 1$ be an approximate eigenpair. Starting with $(\mathbf{x}_k, \alpha_k, \beta_k)$, a Newton iteration applied to $\mathbf{G}(\mathbf{x}, \alpha, \beta)$ yields

$$\text{(A.2)}\quad \mathbf{J}_k \cdot \left(\begin{bmatrix}\mathbf{x}_{\mathrm{nt}}\\ \alpha_{\mathrm{nt}}\\ \beta_{\mathrm{nt}}\end{bmatrix} - \begin{bmatrix}\mathbf{x}_k\\ \alpha_k\\ \beta_k\end{bmatrix}\right) = -\begin{bmatrix}\mathbf{r}_k\\ 0\\ 0\end{bmatrix} \text{ with } \mathbf{J}_k := \begin{bmatrix}\mathbf{H}(\mathbf{x}_k) - \mathbf{\Lambda}_k + \mathbf{S}(\mathbf{x}_k) & -\mathbf{x}_k & -\overline{\mathbf{x}}_k\\ -\mathbf{x}_*^T & 0 & 0\\ -\overline{\mathbf{x}}_*^T & 0 & 0\end{bmatrix},$$

where $\mathbf{r}_k = \mathbf{H}(\mathbf{x}_k)\mathbf{x}_k - \mathbf{\Lambda}_k\mathbf{x}_k$ and $\mathbf{S}(\mathbf{x}) := 2[\mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{x}]\cdot\nabla^2 F(\rho(\mathbf{x}))\cdot[\mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{x}]^T \succeq 0$. Provided $H(x_k) - \lambda_k I$ is nonsingular and $\lambda_k \neq 0$, we can obtain from the leading $2n$ elements that

$$\begin{aligned}\mathbf{x}_{\mathrm{nt}} &= -\big(\mathbf{H}(\mathbf{x}_k) - \mathbf{\Lambda}_k\big)^{-1}\cdot\big((\mathbf{\Lambda}_k - \mathbf{\Lambda}_{\mathrm{nt}})\mathbf{x}_k + \mathbf{S}(\mathbf{x}_k)\cdot(\mathbf{x}_{\mathrm{nt}} - \mathbf{x}_k)\big)\\ &\in \mathrm{span}\big\{(\mathbf{H}(\mathbf{x}_k) - \mathbf{\Lambda}_k)^{-1}[\mathbf{x}_k, \overline{\mathbf{x}}_k, \mathbf{A}\mathbf{x}_k, \mathbf{A}\overline{\mathbf{x}}_k, \mathbf{B}\mathbf{x}_k, \mathbf{B}\overline{\mathbf{x}}_k]\big\}\\ &= \mathrm{span}\big\{\mathbf{x}_k, \overline{\mathbf{x}}_k, \mathbf{p}_k^{(a)}, \overline{\mathbf{p}}_k^{(a)}, \mathbf{p}_k^{(b)}, \overline{\mathbf{p}}_k^{(b)}\big\},\end{aligned} \qquad \text{(A.3)}$$

where in the second equation we used $\mathrm{Range}(\mathbf{S}(\mathbf{x}_k) \subset \mathrm{span}([\mathbf{A}\mathbf{x}_k, \mathbf{B}\mathbf{x}_k])$ as well as $\mathrm{Range}(\mathbf{\Lambda}\mathbf{x}) \subset \mathrm{span}([\mathbf{x}, \overline{\mathbf{x}}])$. In the last equation, the trailing four vectors are the augmented real vectors of $p_k^{(a)}, p_k^{(b)}$ defined in (3.14), respectively. Equation (A.3) can be verified by a left multiplication of $\mathbf{H}(\mathbf{x}_k) - \mathbf{\Lambda}_k$; notice that this matrix is the real augmentation of $H(x_k) - \lambda_k I$. By writing the real $\mathbf{x}_{\mathrm{nt}}$ back to a length-n complex vector x_{nt}, we obtain $x_{nt} \in \mathrm{span}\big\{x_k, p_k^{(a)}, p_k^{(b)}\big\}$ included in the subspace (3.13).

For the quadratic convergence of Newton's methods, it remains to show the Jacobian $\mathbf{J}(\mathbf{x}_*, \alpha_*, \beta_*)$ is nonsingular; see, e.g., [34, Thm. 11.2]. By the simplicity of λ_*, the matrix $H(x_*) - \lambda_* I$ is positive semidefinite with the null space spanned by x_*. Hence, the real augmentation $\mathbf{H}(\mathbf{x}_*) - \mathbf{\Lambda}_*$ is positive semidefinite with the null space $\mathrm{span}\{\mathbf{x}_*, \overline{\mathbf{x}}_*\}$. Let $\mathbf{z} = [\mathbf{z}_1; z_2] \in \mathbb{R}^{2n+2}$, with $\mathbf{z}_1 \in \mathbb{R}^{2n}$ and $z_2 \in \mathbb{R}^2$, be a solution to $\mathbf{J}(\mathbf{x}_*, \alpha_*, \beta_*)\mathbf{z} = 0$. We obtain immediately $\mathbf{z}_1 \perp \mathrm{span}\{\mathbf{x}_*, \overline{\mathbf{x}}_*\}$. It follows that

$$0 = \mathbf{z}^T\mathbf{J}(\mathbf{x}_*, \alpha_*, \beta_*)\mathbf{z} = \mathbf{z}_1^T(\mathbf{H}(\mathbf{x}_*) - \mathbf{\Lambda}_* + \mathbf{S}(x_*))\mathbf{z}_1 \geq \mathbf{z}_1^T(\mathbf{H}(\mathbf{x}_*) - \mathbf{\Lambda}_*)\mathbf{z}_1 \geq 0.$$

Then the positive semidefiniteness of $\mathbf{H}(\mathbf{x}_*) - \mathbf{\Lambda}_*$ implies $\mathbf{z}_1 \in \mathrm{span}\{\mathbf{x}_*, \overline{\mathbf{x}}_*\}$; hence, $\mathbf{z}_1 = 0$. Plugging this into $\mathbf{J}(\mathbf{x}_*, \alpha_*, \beta_*)\mathbf{z} = 0$, we obtain $[\mathbf{x}_*, \overline{\mathbf{x}}_*]z_2 = 0$; hence, $z_2 = 0$. So $\mathbf{z} = 0$ is the only solution to $\mathbf{J}(\mathbf{x}_*, \alpha_*, \beta_*)\mathbf{z} = 0$, and the Jacobian is nonsingular.

Recall that the subspace (3.13) is invariant if we multiply x_k by a nonzero scalar. For $\tan\angle(x_k, x_*)$ sufficiently small, we can assume $x_k = x_* + \mathcal{O}(|\tan\angle(x_k, x_*)|)$,

and the shift $\lambda_k = \frac{b^H H(x_k) x_k}{b^H x_k} = \lambda_* + \mathcal{O}(|\tan\angle(x_k, x_*)|)$ by the smoothness of H. So the real augmentation $(\mathbf{x}_k, \alpha_k, \beta_k) = (\mathbf{x}_*, \alpha_*, \beta_*) + \mathcal{O}(|\tan\angle(x_k, x_*)|)$. It follows that $(\mathbf{x}_{\rm nt}, \alpha_{\rm nt}, \beta_{\rm nt}) = (\mathbf{x}_*, \alpha_*, \beta_*) + \mathcal{O}(|\tan\angle(x_k, x_*)|^2)$ by the quadratic convergence of Newton's methods. Writing $\mathbf{x}_{\rm nt} \in \mathbb{R}^{2n}$ back to its complex representation $x_{\rm nt} \in \mathbb{C}^n$, we complete the proof.

Appendix B. Proof of Lemma 6.1. A result similar to Lemma 6.1 was established in [15], where the optimizing parameter t is on $\mathbb{R}$. By exploiting the convexity of the numerical range $W(A, B)$, we can simplify the proof and obtain a bounded region for the parameter t, rendering the optimization problem more tractable in practice.

(a) We can reformulate the left-hand side as

$$\begin{aligned}\min_{x\in\mathbb{C}^n} \max\left\{\frac{x^HAx}{x^Hx}, \frac{x^HBx}{x^Hx}\right\} &= \min_{y\in W(A,B)} \max\Big\{y_1,\ y_2\Big\} \\ &= \min_{y\in W(A,B)} \max_{t\in[0,1]}\ ty_1 + (1-t)y_2.\end{aligned}$$

Note that $W(A, B)$ is convex, and $f(t, y) = ty_1 + (1 - t)y_2$ is linear in t and y, respectively. By von Neumann's minimax theorem, the min and max in the equation above can switch positions; namely

$$\min_{y\in W(A,B)} \max_{t\in[0,1]}\ f(t,y) = \max_{t\in[0,1]} \min_{y\in W(A,B)} f(t,y). \tag{B.1}$$

The inner minimization satisfies

$$\min_{y\in W(A,B)} f(t,y) = \min_{x\in\mathbb{R}^n} \frac{x^T\left(tA + (1-t)B\right)x}{x^Tx} = \lambda_{\min}\Big(tA + (1-t)B\Big),$$

where the last equation is by the eigenvalue minimization principle.

(b) Due to the minimax relation (B.1) above, we have $(t_*, \rho(x_*))$ as an equilibrium point of the minimax problem. Therefore, it holds that

$$f(t_*, \rho(x_*)) = \min_{y\in W(A,B)} \max_{t\in[0,1]}\ f(t,y) = \max_{t\in[0,1]}\ f(t, \rho(x_*)) = F(\rho(x_*)).$$

It is straightforward to verify that for all y it holds that

$$\begin{aligned}F(y) &= \max_{t\in[0,1]} f(t,y) \ge f(t_*, y) = f(t_*, \rho(x_*)) + f(t_*, y - \rho(x_*)) \\ &= F(\rho(x_*)) + [t_*, (1-t_*)]\cdot(y - \rho(x_*)).\end{aligned}$$

Hence, by the definition of the subgradient, it holds that $[t_*, (1-t_*)]^T \in \partial F(\rho(x_*))$.

Appendix C. Data matrix generation for subsection 6.2. Following the problem setting in [21, sect. C] we arrive at a quadratic optimization problem,

$$\min_{w\in\mathbb{C}^n} w^Hw, \qquad \text{s.t. } w^HR_aw \ge \sigma_a^2\tau_a,\ w^HR_bw \ge \sigma_b^2\tau_b,$$

where τ_a, τ_b, σ_a, τ_b are prescribed parameters, and in the case when the antenna array is linear and the receivers are located at θ_a and θ_b relative array broadside, the covariance matrices $R_i \in \mathbb{C}^{n\times n}$ are defined with (ℓ, p) elements

$$[R_i]_{\ell p} = \exp\Big(\pi\jmath(\ell-p)\sin\theta_i\Big)\cdot\exp\left(-\frac{(\pi(\ell-p)s_i\cos\theta_i)^2}{2}\right) \quad \text{for} \quad i = \{a, b\}, \tag{C.1}$$

where s_i is the spread angle of local scatterers for user i; see, e.g., [4] for details. For convenience, we use parameters $\theta_a = -5^\circ$, $\theta_b = 10^\circ$, $s_i = 2^\circ$, and $\tau_i = 1/\sigma_i^2$ in our experiment.

A straightforward derivation shows that the quadratic optimization from above can be reformulated as[8]

$$\min_{w\in\mathbb{C}^n} \left(\max \left\{ \frac{w^H A w}{w^H w}, \ \frac{w^H B w}{w^H w} \right\} \right) \quad \text{with} \quad A = -\frac{R_a}{\sigma_a^2 \tau_a}, \ B = -\frac{R_b}{\sigma_b^2 \tau_b}. \tag{C.2}$$

Acknowledgments. The author would like to thank Zhaojun Bai, Daniel Kressner, and Bart Vandereycken for fruitful discussions and their valuable comments on preliminary results related to this work. He is also grateful to the anonymous referees for their constructive feedback.

REFERENCES

[1] P.-A. Absil and K. A. Gallivan, *Accelerated line-search and trust-region methods*, SIAM J. Numer. Anal., 47 (2009), pp. 997–1018, https://doi.org/10.1137/08072019X.

[2] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, Software Environ. Tools 11, SIAM, Philadelphia, 2000, https://doi.org/10.1137/1.9780898719581.

[3] Z. Bai, D. Lu, and B. Vandereycken, *Robust Rayleigh quotient minimization and nonlinear eigenvalue problems*, SIAM J. Sci. Comput., 40 (2018), pp. A3495–A3522, https://doi.org/10.1137/18M1167681.

[4] M. Bengtsson and B. Ottersten, *Optimal downlink beamforming using semidefinite optimization*, in Proceedings of the 37th Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, 1999, pp. 987–996.

[5] T. Betcke and E. A. Spence, *Numerical estimation of coercivity constants for boundary integral operators in acoustic scattering*, SIAM J. Numer. Anal., 49 (2011), pp. 1572–1601, https://doi.org/10.1137/100788483.

[6] S. Boyd and V. Balakrishnan, *A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its L_∞-norm*, Systems Control Lett., 15 (1990), pp. 1–7.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

[8] D. Brandwood, *A complex gradient operator and its application in adaptive array theory*, Proc. IEE-H, 130 (1983), pp. 11–16.

[9] J. V. Burke, *Second order necessary and sufficient conditions for convex composite NDO*, Math. Program., 38 (1987), pp. 287–302.

[10] E. Cancès, R. Chakir, and Y. Maday, *Numerical analysis of nonlinear eigenvalue problems*, J. Sci. Comput., 45 (2010), pp. 90–117.

[11] S. H. Cheng and N. J. Higham, *The nearest definite pair for the Hermitian generalized eigenvalue problem*, Linear Algebra Appl., 302 (1999), pp. 63–76.

[12] F. H. Clarke, *Optimization and Nonsmooth Analysis*, Classics Appl. Math. 5, SIAM, Philadelphia, 1990, https://doi.org/10.1137/1.9781611971309.

[13] C. R. Crawford, *A stable generalized eigenvalue problem*, SIAM J. Numer. Anal., 13 (1976), pp. 854–860, https://doi.org/10.1137/0713067.

[14] M. K. H. Fan and A. L. Tits, *On the generalized numerical range*, Linear Multilinear Algebra, 21 (1987), pp. 313–320.

[15] D. D. Gaurav and K. V. S. Hari, *A fast eigen solution for homogeneous quadratic minimization with at most three constraints*, IEEE Signal Process. Lett., 20 (2013), pp. 968–971.

[16] E. Gutkin, E. A. Jonckheere, and M. Karow, *Convexity of the joint numerical range: Topological and differential geometric viewpoints*, Linear Algebra Appl., 376 (2004), pp. 143–171.

[17] W. Hackbusch, *Hierarchical Matrices: Algorithms and Analysis*, Springer Ser. Comput. Math. 49, Springer, Heidelberg, 2015.

[8] Let $\gamma = -(w^H w)^{-1}$; the minimization problem is equivalent to $\min \gamma$ s.t. $(w^H A w)/(w^H w) \le \gamma$ and $(w^H B w)/(w^H w) \le \gamma$.

[18] W. W. Hager, *Minimizing a quadratic over a sphere*, SIAM J. Optim., 12 (2001), pp. 188–208, https://doi.org/10.1137/S1052623499356071.
[19] N. J. Higham, F. Tisseur, and P. M. Van Dooren, *Detecting a definite Hermitian pair and a hyperbolic or elliptic quadratic eigenvalue problem, and associated nearness problems*, Linear Algebra Appl., 351/352 (2002), pp. 455–474.
[20] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms* I: *Fundamentals*, Grundlehren Math. Wiss. 305, Springer-Verlag, Berlin, 1993.
[21] Y. Huang and D. P. Palomar, *Randomized algorithms for optimal solutions of double-sided QCQP with applications in signal processing*, IEEE Trans. Signal Process., 62 (2014), pp. 1093–1108.
[22] C. R. Johnson, *Numerical determination of the field of values of a general complex matrix*, SIAM J. Numer. Anal., 15 (1978), pp. 595–602, https://doi.org/10.1137/0715039.
[23] F. Kangal, K. Meerbergen, E. Mengi, and W. Michiels, *A subspace method for large-scale eigenvalue optimization*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 48–82, https://doi.org/10.1137/16M1070025.
[24] A. V. Knyazev, *A preconditioned conjugate gradient method for eigenvalue problems and its implementation in a subspace*, in Numerical Treatment of Eigenvalue Problems, Vol. 5 (Oberwolfach, 1990), Internat. Ser. Numer. Math. 96, Birkhäuser, Basel, 1991, pp. 143–154.
[25] A. V. Knyazev, *Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method*, SIAM J. Sci. Comput., 23 (2001), pp. 517–541, https://doi.org/10.1137/S1064827500366124.
[26] D. Kressner, D. Lu, and B. Vandereycken, *Subspace acceleration for the Crawford number and related eigenvalue optimization problems*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 961–982, https://doi.org/10.1137/17M1127545.
[27] K. Kreutz-Delgado, *The Complex Gradient Operator and the CR-Calculus*, Tech. report, Course Lecture Suppl. ECE275A., Dept. Elect. Comput. Eng., University of California San Diego, San Diego, CA, 2005; available at http://dsp.ucsd.edu/~kreutz/.
[28] C. Le Bris, *Computational chemistry from the perspective of numerical analysis*, Acta Numer., 14 (2005), pp. 363–444.
[29] C.-K. Li and Y.-T. Poon, *Convexity of the joint numerical range*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 668–678, https://doi.org/10.1137/S0895479898343516.
[30] R.-C. Li, *Rayleigh quotient based optimization methods for eigenvalue problems*, in Matrix Functions and Matrix Equations, World Scientific, Singapore, 2015, pp. 76–108.
[31] E. Mengi, E. A. Yildirim, and M. Kiliç, *Numerical optimization of eigenvalues of Hermitian matrix functions*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 699–724, https://doi.org/10.1137/130933472.
[32] R. Meyer, *Nonlinear eigenvector algorithms for local optimization in multivariate data analysis*, Linear Algebra Appl., 264 (1997), pp. 225–246.
[33] G. Narkiss and M. Zibulevsky, *Sequential Subspace Optimization Method for Large-Scale Unconstrained Problems*, Tech. report CCIT 559, EE Dept., Technion, Haifa, Israel, 2005.
[34] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer, New York, 2006.
[35] M. L. Overton, *On minimizing the maximum eigenvalue of a symmetric matrix*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 256–268, https://doi.org/10.1137/0609021.
[36] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Classics Appl. Math. 20, SIAM, Philadelphia, 1998, https://doi.org/10.1137/1.9781611971163.
[37] P. Pulay, *Convergence acceleration of iterative sequences. The case of SCF iteration*, Chem. Phys. Lett., 73 (1980), pp. 393–398.
[38] V. R. Saunders and I. H. Hillier, *A "level–shifting" method for converging closed shell Hartree–Fock wave functions*, Int. J. Quantum Chem., 7 (1973), pp. 699–705.
[39] N. D. Sidiropoulos, T. N. Davidson, and Z.-Q. Luo, *Transmit beamforming for physical-layer multicasting*, IEEE Trans. Signal Process., 54 (2006), pp. 2239–2251.
[40] W. Śmigaj, T. Betcke, S. Arridge, J. Phillips, and M. Schweiger, *Solving boundary integral problems with BEM++*, ACM Trans. Math. Softw., 41 (2015), 6.
[41] A. Stathopoulos and J. R. McCombs, *PRIMME: Preconditioned iterative multimethod eigensolver—methods and software description*, ACM Trans. Math. Softw., 37 (2010), 21.
[42] G. W. Stewart, *Perturbation bounds for the definite generalized eigenvalue problem*, Linear Algebra Appl., 23 (1979), pp. 69–85.
[43] L. Thøgersen, J. Olsen, D. Yeager, P. Jørgensen, P. Sałek, and T. Helgaker, *The trust-region self-consistent field method: Towards a black-box optimization in Hartree–Fock and Kohn–Sham theories*, J. Chem. Phys., 121 (2004), pp. 16–27.

[44] F. Uhlig, *On computing the generalized Crawford number of a matrix*, Linear Algebra Appl., 438 (2013), pp. 1923–1935.
[45] C. Yang, J. C. Meza, and L.-W. Wang, *A trust region direct constrained minimization algorithm for the Kohn–Sham equation*, SIAM J. Sci. Comput., 29 (2007), pp. 1854–1875, https://doi.org/10.1137/060661442.
[46] L.-H. Zhang, *On optimizing the sum of the Rayleigh quotient and the generalized Rayleigh quotient on the unit sphere*, Comput. Optim. Appl., 54 (2013), pp. 111–139.