SIAM J. MATRIX ANAL. APPL.
Vol. 34, No. 2, pp. 709–737

FAST APPROXIMATION OF THE H_∞ NORM VIA OPTIMIZATION OVER SPECTRAL VALUE SETS*

NICOLA GUGLIELMI†, MERT GÜRBÜZBALABAN‡, AND MICHAEL L. OVERTON‡

Abstract. The H_∞ norm of a transfer matrix function for a control system is the reciprocal of the largest value of ε such that the associated ε-spectral value set is contained in the stability region for the dynamical system (the left half-plane in the continuous-time case and the unit disk in the discrete-time case). After deriving some fundamental properties of spectral value sets, particularly the intricate relationship between the singular vectors of the transfer matrix and the eigenvectors of the corresponding perturbed system matrix, we extend an algorithm recently introduced by Guglielmi and Overton [*SIAM J. Matrix Anal. Appl.*, 32 (2011), pp. 1166–1192] for approximating the maximal real part or modulus of points in a matrix pseudospectrum to spectral value sets, characterizing its fixed points. We then introduce a Newton-bisection method to approximate the H_∞ norm, for which each step requires optimization of the real part or the modulus over an ε-spectral value set. Although the algorithm is guaranteed only to find lower bounds on the H_∞ norm, it typically finds good approximations in cases where we can test this. It is much faster than the standard Boyd–Balakrishnan–Bruinsma–Steinbuch algorithm to compute the H_∞ norm when the system matrices are large and sparse and the number of inputs and outputs is small. The main work required by the algorithm is the computation of the spectral abscissa or radius of a sequence of matrices that are rank-one perturbations of a sparse matrix.

Key words. H_∞ norm, large and sparse systems, spectral value sets, spectral value set abscissa, spectral value set radius, Newton-bisection method, pseudospectra, transfer matrix, stability radius, linear dynamical systems

AMS subject classifications. 93B40, 93B25, 93B60, 93C05, 93C80, 65F15, 15A18, 90C06, 93C55, 65F50

DOI. 10.1137/120875752

1. Introduction.

Consider the continuous-time linear dynamical system with input and output defined by

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t), \end{aligned} \tag{1.1}$$

where $A \in \mathbb{C}^{n,n}$, $B \in \mathbb{C}^{n,p}$, $C \in \mathbb{C}^{m,n}$, and $D \in \mathbb{C}^{m,p}$. The discrete time analogue is

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k + Du_k. \end{aligned} \tag{1.2}$$

In this paper we present new methods for computing the H_∞ norm of the transfer matrix function associated with these systems, a well-known important quantity for measuring robust stability [17, 31]. We build on two foundations. The first is the theory of spectral value sets presented in [17], as the H_∞ norm can be viewed as

*Received by the editors May 2, 2012; accepted for publication (in revised form) by P.-A. Absil January 25, 2013; published electronically June 13, 2013.
http://www.siam.org/journals/simax/34-2/87575.html

†Dipartimento di Matematica Pura e Applicata, Università dell'Aquila, I-67010 L'Aquila, Italy (guglielm@univaq.it). This author was supported in part by the Italian Ministry of Education, Universities and Research (M.I.U.R.).

‡Courant Institute of Mathematical Sciences, New York University, New York, NY 10012 (mert@cims.nyu.edu, overton@cs.nyu.edu). These authors were supported in part by National Science Foundation grant DMS-1016325.

the reciprocal of the largest value of ε such that the associated ε-spectral value set is contained in the stability region for the dynamical system (the left half-plane in the continuous-time case and the unit disk in the discrete-time case). The second is an algorithm recently introduced by Guglielmi and Overton [14] for computing the rightmost point (or the largest point in modulus) in the ε-pseudospectrum of a matrix A. We extend this algorithm from pseudospectra to spectral value sets and then give a Newton-bisection method to approximate the H_∞ norm. Although the algorithm is guaranteed only to find lower bounds on the H_∞ norm, it typically finds good approximations in cases where we can test this. The algorithm is much faster than the standard Boyd–Balakrishnan–Bruinsma–Steinbuch (BBBS) algorithm [4, 5] to compute the H_∞ norm when $n \gg \max(m, p)$ and the matrix A is sparse.

The paper is organized as follows. In the next section we establish the fundamental properties of spectral value sets that we will need, including a detailed explanation of the relationship between the singular vectors of the transfer matrix and the eigenvectors of a corresponding perturbed system matrix. We then define the H_∞ norm and explain its relationship with spectral value sets. In section 3 we generalize the algorithm of [14] for computing the pseudospectral abscissa of a matrix A to a spectral value set abscissa for (A, B, C, D). We briefly discuss local convergence properties of this method, including the characterization of fixed points of the iteration, and we give a variation for the spectral value set radius. Then in section 4 we introduce a Newton-bisection method to approximate the H_∞ norm. Every step of this method requires the approximation of a spectral value set abscissa (or radius) and each of these is carried out by an iteration which requires only the computation of the rightmost eigenvalue (or eigenvalue with largest modulus) of a sequence of matrices that are rank-one perturbations of A. In sections 5.1 and 5.2 we present numerical examples, and in section 6 we discuss some open questions for future research.

2. Spectral value sets. The first part of this section follows the development in [17, section 5.1]; more detailed attribution appears below. Given A, B, C, D defining the linear dynamical system (1.1), consider the *perturbed system matrix*

$$M(E) = A + BE(I - DE)^{-1}C \quad \text{for } E \in \mathbb{C}^{p\times m} \tag{2.1}$$

assuming $I - DE$ is invertible and the associated *transfer matrix*

$$G(\lambda) = C(\lambda I - A)^{-1}B + D \quad \text{for } \lambda \in \mathbb{C}\backslash\sigma(A),$$

where $\sigma(\cdot)$ denotes spectrum. The following fundamental theorem relates the norm of the transfer matrix evaluated at eigenvalues of the perturbed system matrices to the norms of the underlying perturbations E. Here and throughout the paper, $\|\cdot\|$ denotes the vector or matrix 2-norm (maximum singular value). The dimension of the identity matrix I depends on the context.

THEOREM 2.1. *Let $\varepsilon \in \mathbb{R}$ with $\varepsilon > 0$ and $\varepsilon\|D\| < 1$ so that $I - DE$ is invertible when $\|E\| \le \varepsilon$. Then for $\lambda \notin \sigma(A)$ the following are equivalent:*

$$\|G(\lambda)\| \ge \varepsilon^{-1} \qquad \text{and} \qquad \lambda \in \sigma(M(E)) \text{ for some } E \text{ with } \|E\| \le \varepsilon. \tag{2.2}$$

Proof. Suppose the first statement holds with $\xi = \|G(\lambda)\|^{-1} \le \varepsilon$. Let u and v respectively be right and left singular vectors of $G(\lambda)$ corresponding to the largest singular value ξ^{-1}, so that $\xi G(\lambda)u = v$, $\xi v^* G(\lambda) = u^*$, and $\|u\| = \|v\| = 1$. Set $E = \xi uv^*$ so that $\|E\| = \xi \le \varepsilon$. We have $G(\lambda)E = vv^*$, so

$$(C(\lambda I - A)^{-1}B + D)Ev = v. \tag{2.3}$$

Define $Y = (I - DE)^{-1}C$ and $Z = (\lambda I - A)^{-1}BE$, so we have $YZv = v$. It follows that $ZYx = x$ with $x = Zv \neq 0$ an eigenvector of ZY. Multiplying through by $\lambda I - A$, we have

$$BE(I - DE)^{-1}Cx = (\lambda I - A)x, \tag{2.4}$$

proving the second statement in (2.2).

Conversely, suppose that the second statement holds. Then $\exists x \neq 0$ such that (2.4) holds. We have $ZYx = x$, so x is an eigenvector of ZY corresponding to the eigenvalue 1. Consequently, $YZw = w$ where $w = Yx \neq 0$ is an eigenvector of YZ. Multiplying by $I - DE$ and rearranging we have

$$(C(\lambda I - A)^{-1}B + D)Ew = w$$

so

$$\varepsilon\|G(\lambda)\| \geq \|G(\lambda)E\| \geq 1,$$

establishing the first statement in (2.2). ∎

Remark 2.2. Equivalence (2.2) also holds if we restrict E in the second statement to have rank one. The proof remains unchanged. Note that u and v are each uniquely defined up to a unimodular scalar if and only if the maximum singular value ξ^{-1} is simple.

DEFINITION 2.3. *Let $\varepsilon \in \mathbb{R}$ with $\varepsilon \geq 0$ and $\varepsilon\|D\| < 1$, and define the* spectral value set

$$\sigma_\varepsilon(A, B, C, D) = \bigcup \left\{\sigma(M(E)) : E \in \mathbb{C}^{p\times m}, \|E\| \leq \varepsilon\right\}.$$

Note that $\sigma_\varepsilon(A, B, C, D) \supset \sigma_0(A, B, C, D) = \sigma(A)$. The following corollary of Theorem 2.1 and Remark 2.2 is immediate.

COROLLARY 2.4. *Let $\varepsilon \in \mathbb{R}$ with $\varepsilon > 0$ and $\varepsilon\|D\| < 1$. Then*

$$\begin{aligned}\sigma_\varepsilon(A, B, C, D)\backslash\sigma(A) &= \bigcup \left\{\lambda \in \mathbb{C}\backslash\sigma(A) : \|G(\lambda)\| \geq \varepsilon^{-1}\right\} \\ &= \bigcup \left\{\sigma(M(E)) : E \in \mathbb{C}^{p\times m}, \|E\| \leq \varepsilon, \mathit{rank}(E) = 1\right\}.\end{aligned}$$

Remark 2.5. Theorem 2.1 is implied by the more general development in [17, Theorem 5.2.9], where the norm need not be the 2-norm and the admissible perturbations E may be restricted to have a specified structure; see also [20] for the case $D = 0$. The basic idea of our proof is from [17, Lemma 5.2.7], but the relationship between eigenvectors of $M(E)$ and singular vectors of $G(\lambda)$ revealed by our proof and developed further below is essential for this paper. Remark 2.2 may also be found in [17, Remark 5.2.20(iii)]; this observation does not generally apply when structure is imposed on E. The sets σ_ε are called spectral value sets in [17, 20] and are also sometimes known as structured pseudospectra. In the case $B = C = I, D = 0$, the σ_ε are called pseudospectra [27]. In all the references just mentioned, the sets are defined with strict inequalities instead of the nonstrict inequalities used above. Using our definition, the set σ_ε is compact for fixed ε.

DEFINITION 2.6. *An eigenvalue λ of A is* observable *if all its corresponding right eigenvectors x (with $Ax = \lambda x$, $x \neq 0$) satisfy $Cx \neq 0$, and it is* controllable *if all its corresponding left eigenvectors y (with $y^*A = \lambda y^*$, $y \neq 0$) satisfy $y^*B \neq 0$ [2, Corollary 6.9].*

Remark 2.7. If an eigenvalue λ of A is either unobservable or uncontrollable, that is, $Cx = 0$ or $y^*B = 0$ for some right eigenvector x or left eigenvector y, then from (2.1) we have either $M(E)x = \lambda x$ or $y^*M(E) = \lambda y^*$ for all E for which $M(E)$ is defined. Therefore, λ is an eigenvalue of $M(E)$ for all such E, so $\lambda \in \sigma_\varepsilon(A, B, C, D)$ for all ε with $\varepsilon\|D\| < 1$. If in addition λ is a simple eigenvalue of A, then by eigenvalue continuity, λ must be an isolated point of $\sigma_\varepsilon(A, B, C, D)$ for all sufficiently small ε.

Next, we show that as long as E is chosen to have rank one, $E(I - DE)^{-1}$ can be simplified.

LEMMA 2.8. *Let $\varepsilon \in \mathbb{R}$, with $\varepsilon > 0$ and $\varepsilon\|D\| < 1$. Then for all E with $\|E\| \le \varepsilon$, we have $E(I - DE)^{-1} = (I - ED)^{-1}E$, and if $E = \varepsilon uv^*$, where $u \in \mathbb{C}^p$ and $v \in \mathbb{C}^m$ are arbitrary vectors with unit norm, we have*

$$E(I - DE)^{-1} = \frac{1}{1 - \varepsilon v^* Du} E.$$

Proof. To see that the first statement holds, multiply both sides on the left by $I - ED$ and on the right by $I - DE$. For the second, by the Sherman–Morrison–Woodbury formula [13], we have

$$\begin{aligned} E(I - DE)^{-1} &= \varepsilon uv^*(I - \varepsilon Duv^*)^{-1} = \varepsilon uv^* \left(I + \frac{\varepsilon}{1 - \varepsilon v^* Du} Duv^* \right) \\ &= \varepsilon uv^* + \frac{\varepsilon^2 v^* Du}{1 - \varepsilon v^* Du} uv^* = \frac{1}{1 - \varepsilon v^* Du} E. \qquad \square \end{aligned}$$

We now show that again provided E is rank-one, there is a key relationship between the right and left eigenvectors of $M(E)$ and the right and left singular vectors of $G(\lambda)$.

THEOREM 2.9. *Let $\varepsilon \in \mathbb{R}$ with $\varepsilon > 0$ and $\varepsilon\|D\| < 1$, and suppose that $u \in \mathbb{C}^p$ and $v \in \mathbb{C}^m$ with $\|u\| = \|v\| = 1$ satisfy*

$$\varepsilon G(\lambda)u = v \quad \text{and} \quad \varepsilon v^* G(\lambda) = u^*, \tag{2.5}$$

i.e., that u and v are respectively right and left singular vectors of $G(\lambda)$ corresponding to a singular value ε^{-1}. Then, defining $E = \varepsilon uv^*$, we have

$$M(E)x = \lambda x \quad \text{and} \quad y^* M(E) = \lambda y^* \tag{2.6}$$

with

$$x = \varepsilon(\lambda I - A)^{-1}Bu \quad \text{and} \quad y = \varepsilon(\lambda I - A)^{-*}C^*v \tag{2.7}$$

both nonzero, so that x and y are respectively right and left eigenvectors of $M(E)$ corresponding to the eigenvalue λ. Furthermore,

$$Cx + \varepsilon Du = v \quad \text{and} \quad B^*y + \varepsilon D^*v = u \tag{2.8}$$

and

$$u = \left(I - \varepsilon^2 D^*D\right)^{-1} \left(B^*y + \varepsilon D^*Cx\right), \tag{2.9}$$

$$v = \left(I - \varepsilon^2 DD^*\right)^{-1} \left(Cx + \varepsilon DB^*y\right). \tag{2.10}$$

Conversely, suppose $E = \varepsilon uv^*$ for some $u \in \mathbb{C}^p$ and $v \in \mathbb{C}^m$ with $\|u\| = \|v\| = 1$ and that (2.6) holds with x and y nonzero. Then we can scale x and y so that

$$v^*Cx + \varepsilon v^*Du = 1 \quad \text{and} \quad u^*B^*y + \varepsilon u^*D^*v = 1 \tag{2.11}$$

and so that (2.7) *also holds, and if we assume further that* (2.8) *holds, it follows that* (2.5) *holds.*

Proof. Suppose that (2.5) holds, so $G(\lambda)E = vv^*$ and hence (2.3) holds. Defining $Y = (I - DE)^{-1}C$ and $Z = (\lambda I - A)^{-1}BE$ as in the proof of the first part of Theorem 2.1 and using the same argument given there, we have (2.4) with $x = Zv \neq 0$, proving the first statement in (2.6). Hence

$$x = Zv = (\lambda I - A)^{-1}BEv = \varepsilon(\lambda I - A)^{-1}Bu$$

giving the first part of (2.7). Furthermore, we have $EG(\lambda) = uu^*$, so

$$u^*E(C(\lambda I - A)^{-1}B + D) = u^*.$$

Defining $\widetilde{Z} = EC(\lambda I - A)^{-1}$ and $\widetilde{Y} = B(I - ED)^{-1}$, we have $u^*\widetilde{Z}\widetilde{Y} = u^*$, so u is a left eigenvector of $\widetilde{Z}\widetilde{Y}$. Hence $y^*\widetilde{Y}\widetilde{Z} = y^*$ with $y = \widetilde{Z}^*u \neq 0$ a left eigenvector of $\widetilde{Y}\widetilde{Z}$. Multiplying through by $(\lambda I - A)$ on the right, we find

$$y^*B(I - ED)^{-1}EC = y^*(\lambda I - A) \tag{2.12}$$

with

$$y = \widetilde{Z}^*u = (\lambda I - A)^{-*}C^*E^*u = \varepsilon(\lambda I - A)^{-*}C^*v$$

so y is a left eigenvector of $A + B(I - ED)^{-1}EC$ and hence by Lemma 2.8 a left eigenvector of $M(E)$. This proves the second statement in (2.6) and the second part of (2.7). Also, (2.7) implies that

$$Cx = \varepsilon C(\lambda I - A)^{-1}Bu \quad \text{and} \quad B^*y = \varepsilon B^*(\lambda I - A)^{-*}C^*v$$

and combining this with (2.5) we obtain (2.8). Solving (2.8) for u and v gives (2.9) and (2.10), which are well defined as $\varepsilon\|D\| < 1$. Note that the right-hand sides of (2.9) and (2.10) must have unit norm as we assumed a priori that u and v have unit norm.

Conversely, given (2.6), it follows that (2.4) holds, and hence using Lemma 2.8 we have

$$x = \psi(v^*Cx)(\lambda I - A)^{-1}Bu \quad \text{with} \quad \psi = \frac{\varepsilon}{1 - \varepsilon v^*Du},$$

giving the first parts of (2.11) and (2.7) by scaling x so that $\psi v^*Cx = \varepsilon$. Similarly, we have (2.12), which implies

$$y = \bar{\psi}(u^*B^*y)(\lambda I - A)^{-*}C^*v,$$

giving the second parts of (2.11) and (2.7) by scaling y so that $\bar{\psi}u^*B^*y = \varepsilon$. Note that scaling x and y does not change the norms of u and v which are one by assumption. It follows from (2.7) that

$$Cx + \varepsilon Du = \varepsilon G(\lambda)u \quad \text{and} \quad B^*y + \varepsilon D^*v = \varepsilon G(\lambda)^*v.$$

So, if u and v satisfy (2.8), then (2.5) must hold. ∎

Remark 2.10. Theorem 2.9 generalizes the far more trivial Lemma 1.1 of [14]. In the case $D = 0$, (2.8), (2.9), and (2.10) simplify considerably to $u = B^*y$, $v = Cx$, and if we also assume $B = C = I$, then they simplify to $u = y$ and $v = x$. It may be

helpful to consider the converse result in this case. If we take $E = \epsilon uv^*$ and assume that the eigenvector equations (2.6) hold, then we can scale x, y so that $v^*x = 1$, $u^*y = 1$, and (2.7) holds, that is, $x = \epsilon(\lambda I - A)^{-1}u$, $y = \epsilon(\lambda I - A)^{-*}v$ (the proof is as given, with obvious simplifications). However, only if we further assume that $x = v$ and $y = u$ can we conclude that the singular vector equations (2.5) hold.

Remark 2.11. If either $Cx = 0$ or $y^*B = 0$, the normalization (2.11) is not possible, given the assumption $\varepsilon\|D\| < 1$, and consequently neither the assumptions of the theorem nor its converse can hold.

2.1. The H_∞ norm for continuous-time systems. We start by defining spectral abscissa and spectral value set abscissa.

DEFINITION 2.12. *The* spectral abscissa *of the matrix A is*

$$\alpha(A) = \max\{\text{Re } \lambda : \lambda \in \sigma(A)\}$$

with A (Hurwitz) stable *if $\alpha(A) < 0$. For $\varepsilon \geq 0$, $\varepsilon\|D\| < 1$, the* spectral value set abscissa *is*

$$\alpha_\varepsilon(A, B, C, D) = \max\{\text{Re } \lambda : \lambda \in \sigma_\varepsilon(A, B, C, D)\} \tag{2.13}$$

with $\alpha_0(A, B, C, D) = \alpha(A)$.

DEFINITION 2.13. *A* rightmost *point of a set $S \subset \mathbb{C}$ is a point where the maximal value of the real part of the points in S is attained.*

Remark 2.14. Since $\sigma_\varepsilon(A, B, C, D)$ is compact, its rightmost points, that is, the maximizers of the optimization problem in (2.13), lie on its boundary. There can be only a finite number of these; otherwise, the boundary would need to contain an infinite number of points with the same real part, which can be ruled out by an argument similar to [14, Lemma 2.5], exploiting [17, Lemma 5.3.30].

We now define the H_∞ norm.

DEFINITION 2.15. *The H_∞ norm of the transfer matrix function G for continuous-time systems is*

$$\|G\|_\infty^c = \sup_{\delta > \|D\|} \left\{\delta : \delta = \varepsilon^{-1} \text{ and } \alpha_\varepsilon(A, B, C, D) \geq 0\right\}. \tag{2.14}$$

Remark 2.16. The reciprocal of the H_∞ norm, which can be characterized as the largest ε such that $\sigma_\varepsilon(A, B, C, D)$ is contained in the left half-plane, is called the *complex stability radius* [17, section 5.3] (complex because complex perturbations are admitted even if the data are real, and radius in the sense of the perturbation space, not the complex plane). When $B = C = I$ and $D = 0$ this is also known as the *distance to instability* [29] for the matrix A.

The following lemma states an equivalent definition of the H_∞ norm which is actually the standard one.

LEMMA 2.17.

$$\|G\|_\infty^c = \begin{cases} \infty \text{ if } \alpha(A) \geq 0, \\ \sup_{\omega \in \mathbb{R}} \|G(\mathbf{i}\omega)\| \text{ otherwise.} \end{cases} \tag{2.15}$$

Proof. Clearly, the supremum in (2.14) is bounded if and only if A is stable. For stable A and sufficiently small ε, rightmost points of the spectral value set are in the open left half-plane. If $\sigma_\varepsilon(A, B, C, D)$ does not intersect the imaginary axis for any $\varepsilon > 0$ with $\varepsilon\|D\| < 1$, then we take the supremum in (2.14) to be $\|D\|$ as

no $\delta > \|D\|$ satisfies the conditions, while by Corollary (2.4), $\|G(\mathbf{i}\omega)\| < \varepsilon^{-1}$ for all $\omega \in \mathbb{R}$ and all ε with $\varepsilon\|D\| < 1$, and hence the supremum in (2.15) is at most $\|D\|$ (and therefore equal to $\|D\|$ as seen by letting $\omega \to \pm\infty$). Otherwise, there must exist a smallest $\tilde{\varepsilon}$ for which a rightmost point $\tilde{\lambda}$ in $\sigma_{\tilde{\varepsilon}}(A, B, C, D)$ is on the imaginary axis, and by choosing E to have rank one as explained in Remark 2.2 we have $\|E\| = \tilde{\varepsilon}$ and $\|G(\tilde{\lambda})\| = \tilde{\varepsilon}^{-1}$. Furthermore, supposing that there is another point on the imaginary axis with a norm larger than $\tilde{\varepsilon}$ leads immediately to a contradiction. ∎

The standard method for computing the H_∞ norm is the BBBS algorithm [4, 5], which generalizes and improves an algorithm of Byers [9] for computing the distance to instability for A. The method relies on Lemma 2.17: for stable A, it needs only to maximize $\|G(\mathbf{i}\omega)\|$ for $\omega \in \mathbb{R}$. The key idea is that given any $\delta > 0$, it is possible to determine whether $\omega \in \mathbb{R}$ exists such that $\|G(\mathbf{i}\omega)\| = \delta$ by computing all eigenvalues of an associated $2n \times 2n$ Hamiltonian matrix and determining whether any are imaginary. The algorithm is quadratically convergent, but the computation of the eigenvalues and the evaluation of the norm of the transfer matrix both require on the order of n^3 operations, which is not practical when n is sufficiently large.

Our new algorithm is *not* based on evaluating the norm of the transfer matrix. Instead, it works directly with spectral value sets. The first step is to generalize the algorithm of [14] for approximating the pseudospectral abscissa of a matrix to the more general setting of the spectral value set abscissa $\alpha_\varepsilon(A, B, C, D)$ defined in (2.13), as explained in the next section. For this we will need the following concept.

DEFINITION 2.18. *A* locally rightmost *point of a set $S \subset \mathbb{C}$ is a point λ which is a rightmost point of $S \cap \mathcal{N}$ for some neighborhood $\mathcal{N}$ of λ.*

In order to analyze conditions for a point to be a locally rightmost one, we need to make a basic assumption.

Assumption 2.19. Let $\varepsilon \in \mathbb{R}$ with $\varepsilon > 0$ and $\varepsilon\|D\| < 1$, and let $\lambda \notin \sigma(A)$ be a locally rightmost point of $\sigma_\varepsilon(A, B, C, D)$. Then

1. the largest singular value ε^{-1} of $G(\lambda)$ is simple,
2. letting u and v be corresponding right and left singular vectors and setting $E = \varepsilon uv^*$, the eigenvalue λ of $M(E)$ is simple. (That λ is an eigenvalue of $M(E)$ follows from Theorem 2.9.)

We shall assume throughout the paper that Assumption 2.19 holds.

Remark 2.20. Some comments on the assumption are in order. It can be shown by similar arguments to those used in [6, section 2] that generically, that is, for almost all quadruples (A, B, C, D), the largest singular value of $G(\lambda)$ is simple for all $\lambda \in \mathbb{C}\backslash\sigma(A)$. However, examples for which the first part of the assumption does not hold can be constructed easily: take $A = \mathrm{diag}(-2, -1)$, $B = \mathrm{diag}(2, 1)$, $C = I$, $D = 0$, and $\varepsilon = 1$, for which the rightmost point of $\sigma_\varepsilon(A)$ is $\lambda = 0$ and $G(\lambda) = I$ [28]. If the first part does not hold, then u and v are not uniquely defined and so the second part of the assumption is not well defined. In the special case $B = C = I$, $D = 0$, as pointed out in [14], it is clear that if A has simple eigenvalues and ε is sufficiently small, the first part of the assumption must hold, and it seems to be an open question as to whether this is true for all ε. Furthermore, as stated in [14, Lemma 2.6], when $B = C = I$, $D = 0$, the second part of the assumption is implied by the first, but this observation relies on a nontrivial result from [1] that may not carry over to the general case. Finally, again in the case $B = C = I$, $D = 0$, it is also known that regardless of the multiplicity of the largest singular value of $G(\lambda)$, all pairs of corresponding right and left singular vectors u and v satisfy $u^*v \neq 0$ [15].

Although we do not use the transfer matrix $G(\lambda)$ as a computational tool, we use it to characterize maxima of the optimization problem on the right-hand side of (2.13). First, note that it follows from Corollary 2.4 that for $\varepsilon > 0$, $\varepsilon\|D\| < 1$, the definition of the spectral value set abscissa in (2.13) is equivalent to

$$\alpha_\varepsilon(A, B, C, D) = \max\left\{\text{Re } \lambda : \lambda \in \sigma(A) \text{ or } \|G(\lambda)\| \geq \varepsilon^{-1}\right\}. \tag{2.16}$$

The set of admissible λ must include $\sigma(A)$ because of the possibility that the spectral value set $\sigma_\varepsilon(A, B, C, D)$ has isolated points. Excluding such points, we obtain local optimality conditions for (2.16) as follows.

LEMMA 2.21. *Under Assumption* 2.19*, a necessary condition for* $\lambda \notin \sigma(A)$ *to be a local maximizer of the optimization problem in* (2.16) *is*

$$\|G(\lambda)\| = \varepsilon^{-1} \quad \text{and} \quad v^* C\left(\lambda I - A\right)^{-2} Bu \in \mathbb{R}^{++}, \tag{2.17}$$

where $\mathbb{R}^{++}$ *denotes the positive real numbers and* u *and* v *are respectively right and left singular vectors corresponding to the largest singular value* ε^{-1} *of* $G(\lambda)$*.*

Proof. We have already observed that by compactness of $\sigma_\varepsilon(A, B, C, D)$, maximizers must lie on the boundary, and hence the first statement in (2.17) holds. The standard first-order necessary condition for $\hat{\zeta} \in \mathbb{R}^2$ to be a local maximizer of an optimization problem $\max\{f(\zeta) : g(\zeta) \leq 0,\ \zeta \in \mathbb{R}^2\}$, when f, g are continuously differentiable and $g(\hat{\zeta}) = 0$, $\nabla g(\hat{\zeta}) \neq 0$, is the existence of a Lagrange multiplier $\mu \geq 0$ such that $\nabla f(\hat{\zeta}) = \mu \nabla g(\hat{\zeta})$. In our case, identifying $\lambda \in \mathbb{C}$ with $\zeta \in \mathbb{R}^2$, the gradient of the maximization objective is $[1, 0]^T$, while the constraint

$$\frac{1}{\varepsilon} - \|C\left(\lambda I - A\right)^{-1} B + D\|$$

is differentiable with respect to λ because of the first part of Assumption 2.19, and it has gradient

$$\begin{bmatrix} \text{Re}(v^* C(\lambda I - A)^{-2} Bu) \\ \text{Im}(v^* C(\lambda I - A)^{-2} Bu) \end{bmatrix}$$

using standard perturbation theory for singular values [14, Lemma 2.3]. Defining $E = \varepsilon uv^*$ and applying Theorem 2.9 we know that x and y as defined in (2.7) are respectively right and left eigenvectors of $M(E)$ with inner product

$$y^* x = \varepsilon^2 v^* C(\lambda I - A)^{-2} Bu. \tag{2.18}$$

By the second part of Assumption 2.19, λ is a simple eigenvalue of $M(E)$ and so $y^* x \neq 0$. Therefore, the constraint gradient is nonzero, implying that the Lagrange multiplier $\mu \geq 0$ exists with $v^* C(\lambda I - A)^{-2} Bu = 1/\mu \in \mathbb{R}^{++}$. ∎

COROLLARY 2.22. *Let* $\lambda \notin \sigma(A)$ *be a local maximizer of the optimization problem in* (2.13) *and let* u*,* v *be respectively right and left singular vectors of* $G(\lambda)$ *corresponding to the largest singular value* ε^{-1}*. Let* $E = \varepsilon uv^*$*. Define* x *and* y *to be eigenvectors of* $M(E)$ *corresponding to the eigenvalue* λ *and scaled as in* (2.7)*. Then, under Assumption* 2.19*,* $y^* x$ *must be real and positive.*

Proof. Since the optimization problems in (2.13) and (2.16) are equivalent, the result follows directly from Lemma 2.21 using (2.18). ∎

For this reason the following definition is very useful.

DEFINITION 2.23. *A pair of complex vectors* x *and* y *is called* RP-compatible *if* $\|x\| = \|y\| = 1$ *and* $y^* x \in \mathbb{R}^{++}$ *and therefore in the interval* $(0, 1]$*.*

2.2. The H_∞ norm for discrete-time systems. We have analogous definitions relevant to discrete-time systems.

DEFINITION 2.24. *The* spectral radius *of the matrix A is*

$$\rho(A) = \max\{|\lambda| : \lambda \in \sigma(A)\}$$

with A (Schur) stable *if $\rho(A) < 1$. For $\varepsilon \geq 0$, the* spectral value set radius *is*

$$\rho_\varepsilon(A, B, C, D) = \max\{|\lambda| : \lambda \in \sigma_\varepsilon(A, B, C, D)\}. \tag{2.19}$$

DEFINITION 2.25. *An* outermost *point of a set $S \subset \mathbb{C}$ is a point where the maximal value of the modulus of the points in S is attained.*

DEFINITION 2.26. *The H_∞ norm of the transfer matrix function G for discrete-time systems is*

$$\|G\|_\infty^d = \sup_{\delta > \|D\|} \left\{\delta : \delta = \varepsilon^{-1} \text{ and } \rho_\varepsilon(A, B, C, D) \geq 1\right\}. \tag{2.20}$$

The more standard equivalent definition of the H_∞ norm is given by the following lemma.

LEMMA 2.27.

$$\|G\|_\infty^d = \begin{cases} \infty \text{ if } \rho(A) \geq 1, \\ \sup_{\theta \in \mathbb{R}} \|G(e^{\mathbf{i}\theta})\| \text{ otherwise.} \end{cases} \tag{2.21}$$

We omit the proof.

There is a variant of the BBBS algorithm for computing the discrete-time H_∞ norm, based on computing eigenvalues of symplectic pencils instead of Hamiltonian matrices [18, 12].

DEFINITION 2.28. *A* locally outermost *point of a set $S \subset \mathbb{C}$ is a point λ which is an outermost point of $S \cap \mathcal{N}$ for some neighborhood $\mathcal{N}$ of λ.*

From Corollary 2.4, for $\varepsilon > 0$, the definition of the spectral value set radius in (2.19) is equivalent to

$$\rho_\varepsilon(A, B, C, D) = \max\left\{|\lambda| : \lambda \in \sigma(A) \text{ or } \|G(\lambda)\| \geq \varepsilon^{-1}\right\}. \tag{2.22}$$

Excluding possibly isolated points in $\sigma(A)$, we obtain local optimality conditions for (2.22) as follows.

LEMMA 2.29. *Extending Assumption* 2.19 *to locally outermost points in addition to locally rightmost points, a necessary condition for $\lambda \notin \sigma(A)$ to be a local maximizer of the optimization problem in* (2.22) *is*

$$\|G(\lambda)\| = \varepsilon^{-1} \quad \text{and} \quad \lambda\left(v^* C \left(\lambda I - A\right)^{-2} B u\right) \in \mathbb{R}^{++}, \tag{2.23}$$

where u and v are respectively right and left singular vectors corresponding to the largest singular value ε^{-1} of $G(\lambda)$.

The proof is the same as the proof of Lemma 2.21, except that the derivative of the complex modulus replaces the derivative of the real part.

So, we generalize the definition of RP-compatibility as follows.

DEFINITION 2.30. *A pair of complex vectors x and y is called* RP(λ)-compatible *if $\|x\| = \|y\| = 1$ and y^*x is a positive real multiple of λ. When the argument λ is omitted, it is understood to be* 1.

3. Approximating the spectral value set abscissa and radius. We now show how to generalize the algorithm of [14] to approximate the spectral value set abscissa $\alpha_\varepsilon(A,B,C,D)$. We address the spectral value set radius $\rho_\varepsilon(A,B,C,D)$ in section 3.6 below. We write *approximate*, not *compute*, because the algorithm aims to find local maximizers of the optimization problem in (2.13). There will be no assurance that these are global maximizers, but in practice this is often the case, and even if it is not, we obtain guaranteed lower bounds on α_ε. We remark that we could easily extend the criss-cross algorithm of [6] to compute the global optimum, but, like the BBBS algorithm, this would require repeated computation of all the eigenvalues of a $2n \times 2n$ Hamiltonian matrix.

We have seen from Theorem 2.1 and Remark 2.2 that without loss of generality, we can restrict the perturbation matrix $E \in \mathbb{C}^{p\times m}$ parameterizing $\sigma_\varepsilon(A,B,C,D)$ to have rank one. The idea of the algorithm is to generate a sequence of rank-one perturbations with norm ε, say, $\varepsilon u_k v_k^*$, $k = 0,1,2,\ldots$, with $u_k \in \mathbb{C}^p$, $v_k \in \mathbb{C}^m$, and $\|u_k\| = \|v_k\| = 1$. The goal is to choose the sequence so that $M(\varepsilon u_k v_k^*)$ converges to a matrix $M(E)$ with an eigenvalue that is a rightmost point of $\sigma_\varepsilon(A,B,C,D)$. Assuming $\max(m,p) \ll n$, the primary matrix operation needed by the algorithm is the computation of eigenvalues with largest real part and their corresponding right and left eigenvectors, which can be done efficiently using an iterative method assuming A is sparse.

We know from Lemma 2.8 that

$$M(\varepsilon u_k v_k^*) = A + BF_kC, \quad \text{where} \quad F_k = \frac{\varepsilon u_k v_k^*}{1-\varepsilon v_k^* D u_k}.$$

The first step of the algorithm is to compute the rightmost eigenvalue λ_0 of A and corresponding RP-compatible right and left eigenvectors x_0, y_0. Assume that λ_0 is simple, controllable, and observable, and consider the matrix-valued function

$$K(t) = A + tBF_0C = A + tB\frac{\varepsilon u_0 v_0^*}{1-\varepsilon v_0^* D u_0}C,$$

where u_0 and v_0 are to be determined. Let $\lambda(t)$ denote the eigenvalue of $K(t)$ converging to λ_0 as $t \to 0$. Using standard eigenvalue perturbation theory [19, Theorem 6.3.12], [14, Lemma 2.1], we have

$$\lambda'(0) := \left.\frac{d\lambda(t)}{dt}\right|_{t=0} = \frac{y_0^* B\left(\frac{u_0 v_0^*}{1-\varepsilon v_0^* D u_0}\right) C x_0}{y_0^* x_0}. \tag{3.1}$$

We choose (u_0, v_0) to maximize the real part of this expression, as this choice is the one that moves $\lambda(t)$ to the right as fast as possible as t is increased from zero. Since x_0, y_0 are RP-compatible, their inner product $y_0^* x_0$ is a fixed positive real number. Therefore, we choose u_0 and v_0 as maximizers of

$$\max_{\substack{u\in\mathbb{C}^p,\|u\|=1\\ v\in\mathbb{C}^m,\|v\|=1}} \operatorname{Re}\left(y_0^* B\left(\frac{uv^*}{1-\varepsilon v^* D u}\right) C x_0\right). \tag{3.2}$$

When $D = 0$, we obtain $u_0 = B^* y_0/\|B^* y_0\|$ and $v_0 = Cx_0/\|Cx_0\|$. We will discuss the case $D \neq 0$ in detail below.

Now, let us consider how to compute (u_k, v_k) from (u_{k-1}, v_{k-1}) for $k = 1,2,\ldots$. The algorithm computes the rightmost eigenvalue λ_k of $M(\varepsilon u_{k-1} v_{k-1}^*) = A + BF_{k-1}C$

and corresponding RP-compatible right and left eigenvectors x_k, y_k. Assume that λ_k is simple, controllable, and observable and consider the matrix-valued linear function

$$K(t) = A + BF_{k-1}C + tB\left(F_k - F_{k-1}\right)C$$

with $t \in \mathbb{R}$, which satisfies $K(0) = M(\varepsilon u_{k-1} v_{k-1}^*)$ and $K(1) = M(\varepsilon u_k v_k^*)$. Define $\lambda(t)$ to be the eigenvalue of $K(t)$ that converges to λ_k as $t \to 0$. Again using standard first-order eigenvalue perturbation theory we have

$$\lambda'(0) := \left.\frac{d\lambda(t)}{dt}\right|_{t=0} = \frac{y_k^* B\left(F_k - F_{k-1}\right)Cx_{k-1}}{y_{k-1}^* x_{k-1}} \tag{3.3}$$

$$= \varepsilon \frac{y_k^* B\left(\frac{u_k v_k^*}{1-\varepsilon v_k^* D u_k}\right)Cx_k}{y_k^* x_k} - \varepsilon \frac{y_k^* B\left(\frac{u_{k-1} v_{k-1}^*}{1-\varepsilon v_{k-1}^* D u_{k-1}}\right)Cx_k}{y_k^* x_k}. \tag{3.4}$$

The second term is fixed so we choose u_k, v_k to maximize the real part of the first term; clearly, the real part of the second term is a lower bound. Since x_k, y_k are RP-compatible, their inner product $y_k^* x_k$ is a fixed positive real number. Therefore, we choose u_k and v_k as maximizers of

$$\max_{\substack{u\in\mathbb{C}^p, \|u\|=1 \\ v\in\mathbb{C}^m, \|v\|=1}} \operatorname{Re}\left(y_k^* B\left(\frac{uv^*}{1-\varepsilon v^* D u}\right)Cx_k\right), \tag{3.5}$$

an optimization problem with the same form as (3.2). When $D = 0$, we obtain $u_k = B^* y_k / \|B^* y_k\|$ and $v_k = Cx_k / \|Cx_k\|$.

3.1. Solving the optimization subproblem when $D \neq 0$. We address here the following unconstrained optimization problem:

$$\max_{\substack{u\in\mathbb{C}^p, \|u\|=1 \\ v\in\mathbb{C}^m, \|v\|=1}} \operatorname{Re}\, g(u,v) \tag{3.6}$$

with

$$g(u,v) = \frac{g_1(u,v)}{g_2(u,v)}, \quad g_1(u,v) = b^* u v^* c, \quad g_2(u,v) = 1 - \varepsilon v^* D u,$$

which is equivalent to (3.2) when $b = B^* y_0$ and $c = Cx_0$ and to (3.5) when $b = B^* y_k$ and $c = Cx_k$. Assume furthermore that $b \neq 0$ and $c \neq 0$; otherwise the optimization problem (3.6) is trivial. Note that since we assume $\varepsilon\|D\| < 1$ the denominator $g_2(u,v)$ is always nonzero.

By compactness, a maximizer must exist. Let us define the Lagrangian

$$\mathcal{L}(u,v,\mu,\nu) = \operatorname{Re}\, g(u,v) - \frac{1}{2}\mu\left(u^* u - 1\right) - \frac{1}{2}\nu\left(v^* v - 1\right),$$

where $\mu \in \mathbb{R}$, $\nu \in \mathbb{R}$ are Lagrange multipliers, and impose the classical optimality conditions. Observe that replacing g by its complex conjugate $\overline{g}$ leaves the problem unchanged.

Denoting the jth component of u by $u^j = u_R^j + \mathbf{i}u_I^j$, let us consider the partial derivatives of g with respect to u_R^j and u_I^j and impose the conditions $\partial\mathcal{L}/\partial u_R^j = 0$

and $\partial\mathcal{L}/\partial u_I^j = 0$. Since the function $g(u,v)$ is holomorphic with respect to u^j, the Cauchy–Riemann equations yield

$$\mu u_R^j = \frac{\partial \text{Re } g(u,v)}{\partial u_R^j} = \frac{\partial \text{Im } g(u,v)}{\partial u_I^j},$$
$$\mu u_I^j = \frac{\partial \text{Re } g(u,v)}{\partial u_I^j} = -\frac{\partial \text{Im } g(u,v)}{\partial u_R^j},$$

which imply, using $\partial/\partial u^j = 1/2\big(\partial/\partial u_R^j - \mathbf{i}\partial/\partial u_I^j\big)$,

$$\frac{\partial \text{Re } g(u,v)}{\partial u^j} = \frac{1}{2}\mu\overline{u}^j \quad \text{and} \quad \mathbf{i}\frac{\partial \text{Im } g(u,v)}{\partial u^j} = \frac{1}{2}\mu\overline{u}^j$$

so that we can write

$$\frac{\partial g(u,v)}{\partial u^j} = \mu\overline{u}^j.$$

The gradients of g_1 and g_2 with respect to u are the row vectors

$$\nabla_u g_1(u,v) = v^* c b^* \quad \text{and} \quad \nabla_u g_2(u,v) = -v^*\varepsilon D.$$

Imposing $\nabla_u g(u,v) = \mu u^*$ we obtain

$$\frac{v^* c b^* \left(1 - \varepsilon v^* D u\right) - b^* u v^* c \left(-\varepsilon v^* D\right)}{\left(1 - \varepsilon v^* D u\right)^2} = \mu u^*. \tag{3.7}$$

A right multiplication by u gives a formula for the Lagrange multiplier

$$\mu = \frac{b^* u v^* c}{\left(1 - \varepsilon v^* D u\right)^2} \in \mathbb{R}. \tag{3.8}$$

At the maximal value of $g(u,v)$, we have

$$\text{Re}\left(\frac{b^* u v^* c}{1 - \varepsilon v^* D u}\right) > 0 \quad \text{and} \quad \text{Re}\left(1 - \varepsilon v^* D u\right) > 0$$

with the first inequality holding because we may take $u = b$, $v = c$ and the second holding because $\varepsilon\|D\| < 1$. Therefore, we have $\mu > 0$. Substituting (3.8) into (3.7), dividing through by $b^* u v^* c$, and conjugating gives

$$\beta b + \varepsilon D^* v = u \quad \text{with} \quad \beta = \frac{1 - \varepsilon u^* D^* v}{u^* b}. \tag{3.9}$$

In order to obtain a similar formula for the gradient with respect to v we replace g by $\overline{g}$ in (3.6), which has the same optimal solution. Doing so we obtain

$$\nabla_v \overline{g}_1(u,v) = u^* b c^* \quad \text{and} \quad \nabla_v \overline{g}_2(u,v) = 1 - u^*\varepsilon D^*.$$

Imposing $\nabla_v \overline{g}(u,v) = \nu v^*$ we get

$$\frac{u^* b c^* \left(1 - \varepsilon u^* D^* v\right) - c^* v u^* b \left(-\varepsilon u^* D^*\right)}{\left(1 - \varepsilon u^* D^* v\right)^2} = \nu v^*. \tag{3.10}$$

A right multiplication by v gives

$$\nu = \frac{c^* v u^* b}{(1 - \varepsilon u^* D^* v)^2} = \overline{\mu} = \mu. \tag{3.11}$$

Substituting (3.11) into (3.10), conjugating, and dividing through by $b^* u v^* c$ gives

$$\gamma c + \varepsilon D u = v \quad \text{with} \quad \gamma = \frac{1 - \varepsilon v^* D u}{v^* c}. \tag{3.12}$$

We have

$$\frac{\beta}{\gamma} = \frac{1 - \varepsilon u^* D^* v}{u^* b} \frac{v^* c}{1 - \varepsilon v^* D u} = \frac{\mu |1 - \varepsilon u^* D^* v|^2}{|b^* u|^2}, \tag{3.13}$$

a positive real number.

Now, combining (3.9) and (3.12), we find

$$u = \Delta \left(\beta b + \gamma \varepsilon D^* c \right) \quad \text{and} \quad v = \widetilde{\Delta} \left(\gamma c + \beta \varepsilon D b \right),$$

where

$$\Delta = \left(I - \varepsilon^2 D^* D \right)^{-1} \quad \text{and} \quad \widetilde{\Delta} = \left(I - \varepsilon^2 D D^* \right)^{-1}. \tag{3.14}$$

Note the equivalences

$$D\Delta = \widetilde{\Delta} D \quad \text{and} \quad \Delta D^* = D^* \widetilde{\Delta}.$$

Therefore, we have

$$u = \beta \widetilde{b} + \gamma \varepsilon D^* \widetilde{c} \quad \text{and} \quad v = \gamma \widetilde{c} + \beta \varepsilon D \widetilde{b},$$

where

$$\widetilde{b} = \Delta b \qquad \text{and} \qquad \widetilde{c} = \widetilde{\Delta} c. \tag{3.15}$$

From (3.13) we can assume

$$\gamma = \rho \beta \qquad \text{with } \rho > 0, \tag{3.16}$$

which implies

$$u = \beta \left(\widetilde{b} + \rho \varepsilon D^* \widetilde{c} \right) \quad \text{and} \quad v = \beta \left(\rho \widetilde{c} + \varepsilon D \widetilde{b} \right). \tag{3.17}$$

Substituting u, v given by (3.17) into the function $g(u, v)$ to be optimized, we observe that the argument of β does not play any role, that is, the function g depends only on $|\beta|$ whose purpose is to normalize the vectors u and v. So we can choose β real and positive. Note also that (3.9) and (3.12) remain unchanged if we scale u, v, β, and γ by any unimodular scalar $e^{\mathbf{i}\theta}$.

From (3.17) we require

$$0 = \|u\|^2 - \|v\|^2 = \beta^2 \left(\|\widetilde{b}\|^2 + \rho^2 \varepsilon^2 \|D^* \widetilde{c}\|^2 - \rho^2 \|\widetilde{c}\|^2 - \varepsilon^2 \|D \widetilde{b}\|^2 \right)$$

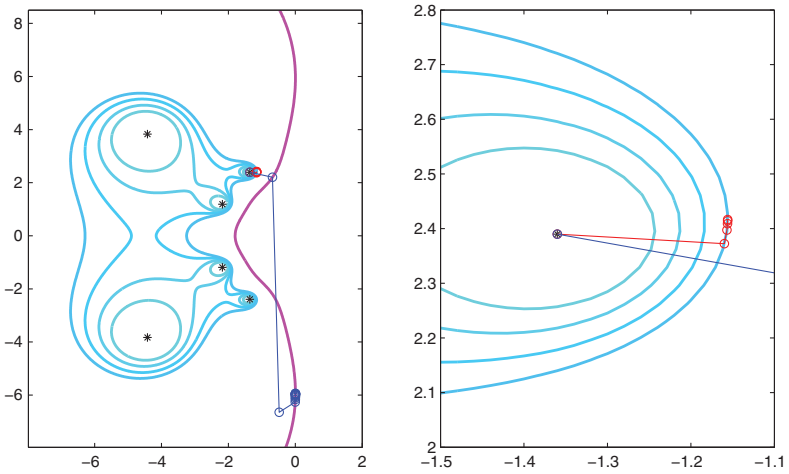

FIG. 3.1. *The iterates of Algorithm* SVSA0 *superimposed on a plot of spectral value sets from* [17, *Example* 5.2.21]. *The left panel shows convergence of the iterates (blue circles) for* $\varepsilon = 8.77$, *while the right panel gives a close-up view of the iterates (red circles) for* $\varepsilon = 1$. *The black asterisks plot the eigenvalues of* A.

so

$$\rho = \sqrt{\frac{\|\widetilde{b}\|^2 - \|\varepsilon D\widetilde{b}\|^2}{\|\widetilde{c}\|^2 - \|\varepsilon D^*\widetilde{c}\|^2}}. \tag{3.18}$$

The last step is to choose $\beta > 0$ such that $\|u\| = 1$, which yields

$$\beta = \frac{1}{\|\widetilde{b} + \rho\varepsilon D^*\widetilde{c}\|}. \tag{3.19}$$

Substituting the optimal values (3.18)–(3.19) into (3.17) we obtain a pair (u, v) that solves (3.6). This pair is unique up to multiplication of u and v by a unimodular factor $e^{\mathbf{i}\theta}$.

3.2. Basic algorithm statement. The derivation given above leads to the following algorithm. To make it well defined, we interpret "rightmost eigenvalue" below to mean the rightmost eigenvalue with largest imaginary part, in case there is more than one with largest real part, although in practice we make no attempt to break ties except in the case of complex conjugate pairs of eigenvalues of real matrices. We adopt the convention that the algorithm *breaks down* if it generates a rightmost eigenvalue λ_k which is not simple, controllable, and observable, as these properties are needed for the basic step of the algorithm to be well defined. For later use, we include as inputs to the algorithm the scalar ε and an initial pair of RP-compatible vectors x_0, y_0. In the absence of any other estimates, these should in principle be set to right and left eigenvectors corresponding to λ_0, the rightmost eigenvalue of A that is simple, controllable, and observable, although checking these conditions is not actually practical.

Algorithm SVSA0(ε, x_0, y_0). Set u_0, v_0 to u and v as defined by (3.17) using (3.18), (3.19), (3.15), and (3.14), where $b = B^*y_0$ and $c = Cx_0$. Set $F_0 = \varepsilon u_0 v_0^*/(1 - \varepsilon v_0^* D u_0)$. For $k = 1, 2, \ldots,$

Let λ_k be the rightmost eigenvalue of $A + BF_{k-1}C$ with corresponding RP-compatible right and left eigenvectors x_k and y_k. Set u_k, v_k to u and v as defined by (3.17) using (3.18), (3.19), (3.15), and (3.14), where $b = B^*y_k$ and $c = Cx_k$. Set $F_k = \varepsilon u_k v_k^*/(1 - \varepsilon v_k^* D u_k)$.

Figure 3.1 shows iterates of Algorithm SVSA0 for computing the spectral value set abscissa for a simple example, superimposing them on a spectral value set plot in the complex plane. The matrix data, with $n = 6$, $m = 6$, and $p = 1$, as well as the contour values $\varepsilon = 0.5, 0.66, 0.83, 1.0, 8.77$, are from [17, Example 5.2.21]. The left panel shows convergence of the iterates for $\varepsilon = 8.77$, while the right panel gives a close-up view for $\varepsilon = 1$. In both cases we initialize x_0 and y_0 to right and left eigenvectors for the rightmost eigenvalue of A.

By construction, the sequence $\{\mathrm{Re}\ \lambda_k\}$ is bounded above by $\alpha_\varepsilon(A, B, C, D)$. Also, the real part of the quantities $\lambda'(0)$ in (3.1) and (3.3) is nonnegative for all k. This is not enough to guarantee monotonicity of the sequence $\{\mathrm{Re}\ \lambda_k\}$; however, we discuss how to achieve monotonicity in section 3.5. First, we characterize fixed points of the iteration described by Algorithm SVSA0.

3.3. Fixed points. Now denote by $\mathcal{T}_\varepsilon$ the map that generates the pair (u_k, v_k) from the pair (u_{k-1}, v_{k-1}) as defined by Algorithm SVSA0. Equivalently, $\mathcal{T}_\varepsilon$ maps a rank-one matrix $u_{k-1}v_{k-1}^*$ with norm one to a rank-one matrix $u_k v_k^*$ with norm one.

DEFINITION 3.1. *The pair (u_{k-1}, v_{k-1}) is a* fixed point *of the map $\mathcal{T}_\varepsilon$ if λ_{k-1} is simple, controllable, and observable and $u_k = e^{\mathbf{i}\theta}u_{k-1}$, $v_k = e^{\mathbf{i}\theta}v_{k-1}$ for some $\theta \in \mathbb{R}$ or, equivalently, if $u_k v_k^* = u_{k-1}v_{k-1}^*$. It follows that $\lambda_k = \lambda_{k-1}$.*

THEOREM 3.2. *Assume $0 < \varepsilon\|D\| < 1$ and suppose that (u, v) is a fixed point of $\mathcal{T}_\varepsilon$ corresponding to the rightmost eigenvalue λ of $M(\varepsilon uv^*)$ that is simple, controllable, and observable. Then $G(\lambda)$ has a singular value equal to ε^{-1}, and furthermore, if it is the largest singular value, then λ satisfies the first-order necessary condition for a local maximizer of* (2.16) *given in* (2.17).

Conversely, assume $0 < \varepsilon\|D\| < 1$ and suppose that $\lambda \notin \sigma(A)$ satisfies (2.17)*, and let u and v denote unit right and left singular vectors corresponding to the largest singular value ε^{-1} of $G(\lambda)$. Then λ is an eigenvalue of $M(\varepsilon uv^*)$, and if it is the rightmost eigenvalue and is simple, then (u, v) is a fixed point of $\mathcal{T}_\varepsilon$.*

Proof. Suppose (u, v) is a fixed point. This means that u and v satisfy (3.9) and (3.12) with $b = B^*y$, $c = Cx$, and x, y, respectively, right and left RP-compatible eigenvectors of $M(\varepsilon uv^*)$. By definition of x and y, it follows that (2.6) holds with $E = \varepsilon uv^*$, and by replacing x and y by βx and γy, respectively, we have (2.8). Therefore, from the second part of Theorem 2.9, it follows that (2.5) also holds, that is, u and v are respectively right and left singular vectors of $G(\lambda)$ corresponding to the singular value ε^{-1}, and if this is the largest singular value, then Lemma 2.21 shows that the first-order optimality conditions hold, using (2.18) and the positivity of y^*x. The latter is not changed by the scaling of x by β and y by γ because β/γ is real and positive, as shown in (3.13).

Conversely, if λ satisfies the first-order necessary conditions, then ε^{-1} is the largest singular value of $G(\lambda)$ and the corresponding unit right and left singular vectors u and v satisfy the inequality in (2.17). Applying the first part of Theorem 2.9 with $E = \varepsilon uv^*$ we see that (2.6) holds for nonzero x and y defined by (2.7) so λ is an eigenvalue of $M(\varepsilon uv^*)$, and furthermore u and v satisfy (2.8) and therefore also (3.9) and (3.12) with $\beta = \gamma = 1$. Also, y^*x is real and positive using (2.17) and (2.18). Thus, if λ is the rightmost eigenvalue of $M(\varepsilon uv^*)$ and it is simple, then (x, y) is a

fixed point of $\mathcal{T}_\varepsilon$. Note that Remark 2.11 shows that λ must be controllable and observable. ∎

As in [14, section 4], we conjecture that the only *attractive* fixed points for Algorithm SVSA0 correspond to points λ that are local maximizers of (2.16).

3.4. Local convergence analysis. We have established that for sufficiently small ε, Algorithm SVSA0 converges locally to rightmost points of the spectral value set with a linear rate of convergence. We omit the details since the proof is a rather lengthy generalization of the development in [14, section 5] but offers little additional insight. In practice, we find that the algorithm almost always converges to a locally rightmost point, without assuming that ε is small, although convergence to global rather than local maximizers of (2.13) cannot be guaranteed.

3.5. A monotonic variant. Algorithm SVSA0 does not always generate a monotonically increasing sequence $\{\mathrm{Re}\ \lambda_k\}$, so we now derive a variant that does. Consider the continuous matrix family

$$N(t) = A + BF(t)C, \quad \text{where} \quad F(t) = \frac{\varepsilon u(t)v(t)^*}{1 - \varepsilon v(t)^* Du(t)}, \tag{3.20}$$

with

$$u(t) = \frac{tu_k + (1-t)u_{k-1}}{\|tu_k + (1-t)u_{k-1}\|} \quad \text{and} \quad v(t) = \frac{tv_k + (1-t)v_{k-1}}{\|tv_k + (1-t)v_{k-1}\|}. \tag{3.21}$$

The idea is that in case the rightmost eigenvalue of $N(1)$ does not have real part greater than that of λ_k, the rightmost eigenvalue of $N(0) = A + BF_{k-1}C$, we may instead choose $t \in (0,1)$ so that the rightmost eigenvalue of $N(t)$ has this property. As in [14, section 6], using $'$ to denote differentiation with respect to t, we have

$$\begin{aligned}
N'(t) &= B\left(\frac{\varepsilon u(t)v(t)^*}{1 - \varepsilon v(t)^* Du(t)}\right)' C = N_1'(t) + N_2'(t),\\
N_1'(t) &= \frac{1}{1 - \varepsilon v(t)^* Du(t)} B\Big(\varepsilon u(t)v(t)^*\Big)' C,\\
N_2'(t) &= \varepsilon B u(t)v(t)^* C\Big(\frac{1}{1 - \varepsilon v(t)^* Du(t)}\Big)'.
\end{aligned}$$

Evaluating these at $t = 0$, we find

$$N'(0) = N_1'(0) + N_2'(0)$$

with

$$\begin{aligned}
N_1'(0) = {} & \frac{\varepsilon}{1 - \varepsilon v_{k-1}^* Du_{k-1}} B\Big(\big(u_k - \mathrm{Re}(u_k^* u_{k-1})u_{k-1}\big)v_{k-1}^*\\
& + u_{k-1}\big(v_k - \mathrm{Re}(v_k^* v_{k-1})v_{k-1}\big)^*\Big)C,\\
N_2'(0) = {} & \frac{\varepsilon^2}{(1 - \varepsilon v_{k-1}^* Du_{k-1})^2}\Big(v_k^* Du_{k-1} - (v_{k-1}^* Du_{k-1})\mathrm{Re}(v_k^* v_{k-1})\\
& + v_{k-1}^* Du_k - (v_{k-1}^* Du_{k-1})\mathrm{Re}(u_k^* u_{k-1})\Big) B u_{k-1} v_{k-1}^* C.
\end{aligned}$$

Now let $\lambda(t)$ denote the eigenvalue of $N(t)$ converging to λ_k as $t \to 0$. From standard eigenvalue perturbation theory

$$\lambda'(0) = \frac{y_k^* N'(0) x_k}{y_k^* x_k} = \frac{\psi_k}{y_k^* x_k}, \tag{3.22}$$

where

$$\begin{aligned}\psi_k = \frac{\varepsilon}{1-\varepsilon v_{k-1}^* D u_{k-1}} \Big(& (v_{k-1}^* C x_k)\big(y_k^* B u_k - (y_k^* B u_{k-1})\mathrm{Re}(u_k^* u_{k-1})\big) \\ & + (y_k^* B u_{k-1})\big(v_k^* C x_k - (v_{k-1}^* C x_k)\mathrm{Re}(v_k^* v_{k-1})\big)\Big) \\ & + \frac{\varepsilon^2 (y_k^* B u_{k-1})(v_{k-1}^* C x_k)}{(1-\varepsilon v_{k-1}^* D u_{k-1})^2}\Big(v_k^* D u_{k-1} - (v_{k-1}^* D u_{k-1})\mathrm{Re}(v_k^* v_{k-1}) \\ & + v_{k-1}^* D u_k - (v_{k-1}^* D u_{k-1})\mathrm{Re}(u_k^* u_{k-1})\Big).\end{aligned} \tag{3.23}$$

We know from the RP-compatibility of x_k, y_k that the denominator of (3.22) is real and positive. Furthermore, if Re $\psi_k < 0$, we can change the sign of both u_k and v_k so that Re $\psi_k > 0$. Excluding the unlikely event that Re $\psi_k = 0$, defining u_k, v_k in this way guarantees that Re $\lambda(t) >$ Re λ_k for sufficiently small t, so that the following algorithm generates monotonically increasing {Re λ_k}. As before, we say that the algorithm breaks down if it generates a rightmost eigenvalue λ_k that is not simple, controllable, and observable. Note, however, that provided x_0 and y_0 are RP-compatible right and left eigenvectors corresponding to a rightmost eigenvalue λ_0 that is simple, controllable, and observable, and provided that Re $\lambda_1 >$ Re λ_0 and Re $\psi_k \neq 0$ for all k, then for $k > 1$, as long as λ_k is simple, it must also be controllable and observable.

Algorithm SVSA1(ε, x_0, y_0). Set u_0, v_0 to u and v as defined by (3.17) using (3.18), (3.19), (3.15), and (3.14), where $b = B^* y_0$ and $c = C x_0$. Set $F_0 = \varepsilon u_0 v_0^*/(1 - \varepsilon v_0^* D u_0)$, and let λ_1 be the rightmost eigenvalue of $A + BF_0C$. For $k = 1, 2, \ldots,$

1. Set x_k and y_k to be right and left eigenvectors of $A + BF_{k-1}C$ corresponding to the eigenvalue λ_k, normalized so they are RP-compatible. Set u_k, v_k to u and v as defined by (3.17) using (3.18), (3.19), (3.15), and (3.14), where $b = B^* y_k$ and $c = C x_k$. Furthermore, compute ψ_k defined in (3.23). If Re $\psi_k < 0$, then replace u_k by $-u_k$ and v_k by $-v_k$. Set $t = 1$ and set z to the rightmost eigenvalue of $N(t)$ as defined in (3.20), (3.21).
2. Repeat the following zero or more times until Re $z >$ Re λ_k: replace t by $t/2$ and set z to the rightmost eigenvalue of $N(t)$ as defined in (3.20), (3.21).
3. Set $F_k = F(t)$ as defined in (3.20) and set $\lambda_{k+1} = z$.

Note that if t is always 1, then Algorithm SVSA1 generates the same iterates as Algorithm SVSA0, and if we omit step 2, Algorithm SVSA1 reduces to Algorithm SVSA0.

Remark 3.3. In the case $B = C = I$, $D = 0$, Algorithm SVSA1 reduces to a monotonically increasing algorithm for the pseudospectral abscissa as derived by a similar argument in [14]. However, the analogous Algorithm PSA1 stated there contains several errors: in step 1, z should be set to the rightmost eigenvalue of $A + \varepsilon y x^*$; in step 2, the stopping criterion should be Re $z >$ Re z_k; and in step 3, z_{k+1}, not z_k, should be set to z. The errors in the algorithm statement did not affect the experimental results, except that Table 8.2 of [14] should show that the two Boeing examples need just one bisection each, not two.

By compactness, the monotonically increasing sequence $\{\mathrm{Re}\ \lambda_k\}$ generated by Algorithm SVSA1 must converge. As noted in [14], this does not imply that $\{\lambda_k\}$ converges to an eigenvalue associated with a fixed point of Algorithm SVSA0, but this is typical in practice.

3.6. Approximating the spectral value set radius. Algorithms for the spectral value set radius $\rho_\varepsilon(A, B, C, D)$, defined in (2.19) and (2.22), are obtained by simple variants of Algorithms SVSA0 and SVSA1. Observe that

$$\left.\frac{d\left(|\lambda(t)|^2\right)}{dt}\right|_{t=0} = 2\,\mathrm{Re}\left(\overline{\lambda(0)}\lambda'(0)\right).$$

Thus, in order to maximize the modulus of the left-hand side of (3.1) or (3.3) instead of the real part, we will obtain the same optimization problems (3.2) and (3.5) as before if we simply require x_k and y_k to be $\mathrm{RP}(\overline{\lambda}_k)$-compatible, using Definition 2.30. (Note the conjugate.)

Likewise, in order to ensure that the modulus of the left-hand side of (3.22) is positive we again need only that $\mathrm{Re}\ \psi_k$ is positive, assuming that x_k and y_k are $\mathrm{RP}(\overline{\lambda}_k)$-compatible. This leads to the following algorithm. To ensure that it is well defined we say that if there is a tie for the outermost eigenvalue, the one whose nonnegative complex argument is closest to zero is used. We say that the algorithm breaks down if it generates an outermost eigenvalue that is not simple, controllable, and observable. In the absence of other estimates, x_0 and y_0 are to be set to an $\mathrm{RP}(\overline{\lambda}_0)$-compatible pair of right and left eigenvectors for the outermost eigenvalue λ_0 that is simple, controllable, and observable.

Algorithm SVSR1(ε, x_0, y_0). Set u_0, v_0 to u and v as defined by (3.17) using (3.18), (3.19), (3.15), and (3.14), where $b = B^*y_0$ and $c = Cx_0$. Set $F_0 = \varepsilon u_0 v_0^*/(1 - \varepsilon v_0^* D u_0)$, and let λ_1 be the outermost eigenvalue of $A + BF_0C$. For $k = 1, 2, \ldots,$

1. Set x_k and y_k to be right and left eigenvectors of $A + BF_{k-1}C$ corresponding to the eigenvalue λ_k, normalized so they are $\mathrm{RP}(\overline{\lambda}_k)$-compatible. Set u_k, v_k to u and v as defined by (3.17) using (3.18), (3.19), (3.15), and (3.14), where $b = B^*y_k$ and $c = Cx_k$. Furthermore, compute ψ_k defined in (3.23). If $\mathrm{Re}\ \psi_k < 0$, then replace u_k by $-u_k$ and v_k by $-v_k$. Set $t = 1$ and set z to the outermost eigenvalue of $N(t)$ as defined in (3.20), (3.21).
2. Repeat the following zero or more times until $|z| > |\lambda_k|$: replace t by $t/2$ and set z to the outermost eigenvalue of $N(t)$ as defined in (3.20), (3.21).
3. Set $F_k = F(t)$ as defined in (3.20) and set $\lambda_{k+1} = z$.

Remark 3.4. In the case $B = C = I$, $D = 0$, Algorithm SVSR1 reduces to a monotonically increasing algorithm for the pseudospectral radius as derived by a similar argument in [14]. However, Algorithm PSR1 stated in [14] contains the same errors as Algorithm PSA1 as described in Remark 3.3.

Let us also define Algorithm SVSR0, a variant of Algorithm SVSA0 for the spectral value set radius, as Algorithm SVSR1 with step 2 omitted. The fixed point theorem for Algorithm SVSA0, Theorem 3.2, extends in a straightforward way to Algorithm SVSR0, replacing "rightmost" by "outermost" and using the first-order optimality conditions for (2.22) given in (2.23). The local convergence results mentioned in section 3.4 also apply.

4. Approximating the H_∞ norm. Recall that the H_∞ norm was defined for the continuous-time and discrete-time cases, respectively, in sections 2.1 and 2.2.

4.1. The continuous-time case. We wish to compute $\|G\|_\infty^c$, defined in (2.14) and (2.15). Assume that A is Hurwitz stable, so the norm is finite. We start by observing that since the spectral value set abscissa $\alpha_\varepsilon(A,B,C,D)$ is a monotonically increasing function of ε, we need only to solve the equation

$$f(\varepsilon) \equiv \alpha_\varepsilon(A,B,C,D) = 0 \tag{4.1}$$

for $\varepsilon \in \mathbb{R}^{++}$. The first step is to characterize how α_ε depends on ε.

THEOREM 4.1. *Let $\lambda(\varepsilon)$ denote the rightmost point of $\sigma_\varepsilon(A,B,C,D)$ for $\varepsilon > 0$, $\varepsilon\|D\| < 1$, and assume that Assumption* 2.19 *holds for all such ε. Define $u(\varepsilon)$ and $v(\varepsilon)$ as right and left singular vectors with unit norm corresponding to ε^{-1}, the largest singular value of $G(\lambda(\varepsilon))$, and applying Theorem* 2.9 *with $E(\varepsilon) = \varepsilon u(\varepsilon)v(\varepsilon)^*$, define $x(\varepsilon)$ and $y(\varepsilon)$ by* (2.6) *and* (2.7). *Furthermore, assume that for a given value $\hat\varepsilon$, the rightmost point $\lambda(\hat\varepsilon)$ is unique. Then λ is continuously differentiable at $\hat\varepsilon$ and its derivative is real with*

$$\frac{d}{d\varepsilon}\alpha_\varepsilon(A,B,C,D)\bigg|_{\varepsilon=\hat\varepsilon} = \frac{d}{d\varepsilon}\lambda(\hat\varepsilon) = \frac{1}{y(\hat\varepsilon)^* x(\hat\varepsilon)} \in \mathbb{R}^{++}. \tag{4.2}$$

Proof. For the purposes of differentiation, we identify $\lambda \in \mathbb{C}$ with $\zeta \in \mathbb{R}^2$ as in the proof of Lemma 2.21. The first part of Assumption 2.19 ensures that the largest singular value of $G(\lambda)$ is differentiable with respect to λ and that the singular vectors $v(\varepsilon)$ and $u(\varepsilon)$ are well defined up to multiplication of both by a unimodular scalar and that $E(\varepsilon)$ is not only well defined but differentiable with respect to ε. The second part ensures that $y(\varepsilon)^* x(\varepsilon)$ is nonzero, while the assumption that $\lambda(\hat\varepsilon)$ is unique ensures that $\lambda(\varepsilon)$ is unique in a neighborhood of $\hat\varepsilon$ and, as an eigenvalue of $M(\varepsilon)$, is differentiable at $\hat\varepsilon$ using standard eigenvalue perturbation theory. As in the proof of Lemma 2.21, observe that

$$\frac{1}{\varepsilon} - \|C\,(\lambda I - A)^{-1}\,B + D\| = 0,$$

so differentiating this with respect to ε at $\hat\varepsilon$ and using the chain rule yields

$$\frac{d\lambda(\varepsilon)}{d\varepsilon}\bigg|_{\varepsilon=\hat\varepsilon} = \frac{1}{\varepsilon^2 v^* C(\lambda(\varepsilon)I - A)^{-2}Bu}.$$

Furthermore, (2.18) follows (for $\lambda = \lambda(\varepsilon)$) from (2.7). Combining these with the first-order optimality conditions for (2.16) in (2.17) gives the result. ∎

COROLLARY 4.2. *Make the same assumptions as in Theorem* 4.1, *except normalize $x(\varepsilon)$ and $y(\varepsilon)$ so that they are RP-compatible. This is equivalent to scaling $x(\varepsilon)$ and $y(\varepsilon)$ by $1/\beta(\varepsilon)$ and $1/\gamma(\varepsilon)$, respectively, where these are defined as in* (3.9) *and* (3.12) *or equivalently in* (3.19), (3.16), *and* (3.18). *So*

$$\frac{d}{d\varepsilon}\alpha_\varepsilon(A,B,C,D)\bigg|_{\varepsilon=\hat\varepsilon} = \frac{d}{d\varepsilon}\lambda(\hat\varepsilon) = \frac{1}{\beta(\hat\varepsilon)\gamma(\hat\varepsilon)\big(y(\hat\varepsilon)^* x(\hat\varepsilon)\big)} \in \mathbb{R}^{++}. \tag{4.3}$$

Remark 4.3. If A,B,C,D are all real, then $\sigma_\varepsilon(A,B,C,D)$ is symmetric with respect to the real axis and hence its rightmost points must either be real or part of a conjugate pair. In the latter case, the assumption that $\lambda(\hat\varepsilon)$ is unique does not hold but the result still holds as long as there is no third rightmost point.

The derivative formula (4.3) naturally leads to a formulation of Newton's method for computing $\|G\|_\infty^c$. We first state this in an idealized form.

Algorithm NHC0(ε^1) (Newton's method for H_∞ norm for continuous-time systems). For $j = 1, 2, \ldots,$

1. Compute the spectral value set abscissa $\alpha_{\varepsilon^j}(A, B, C, D)$, along with the rightmost point λ^j and corresponding RP-compatible right and left eigenvectors x^j, y^j and scalars β^j, γ^j defined as in (3.9) and (3.12), or equivalently (3.19), (3.16) using (3.14), (3.15), and (3.18), where $b = B^* y^j$ and $c = Cx^j$.
2. Set
$$\varepsilon^{j+1} = \varepsilon^j - \big(\text{Re } \lambda^j\big)\beta^j\gamma^j\big((y^j)^* x^j\big).$$

Let $\varepsilon_{\text{opt}} = (\|G\|_\infty^c)^{-1}$, so that the rightmost point of $\sigma_{\varepsilon_{\text{opt}}}(A, B, C, D)$ lies on the imaginary axis, and suppose this rightmost point is unique or part of a complex conjugate pair if A, B, C, D are real. It follows from Definition 2.15, Assumption 2.19, Theorem 4.1, and Remark 4.3 that $\alpha_\varepsilon(A, B, C, D)$ is differentiable with respect to ε at $\varepsilon = \varepsilon_{\text{opt}}$ and that the derivative is positive. Thus, the nonzero derivative condition for Newton's method to converge quadratically holds, so the sequence $\{\varepsilon^j\}$ defined by Algorithm NHC0 converges quadratically to ε_{opt} if $|\varepsilon^1 - \varepsilon_{\text{opt}}|$ is sufficiently small.

In practice, each step of Algorithm NHC0 requires a call to Algorithm SVSA1 to compute the spectral value set abscissa via an "inner iteration." It is generally desirable to "warm start" this computation by providing as input to Algorithm SVSA1 not only the new value of ε, but also the final right and left eigenvectors already computed for the previous value of ε, as opposed to repeatedly initializing Algorithm SVSA1 with right and left eigenvectors corresponding to a rightmost eigenvalue of A. In the absence of any other estimates, x_0 and y_0 are to be set to an RP-compatible pair of right and left eigenvectors for the rightmost eigenvalue λ_0 that is simple, controllable, and observable.

Algorithm NHC1(ε^1, x^0, y^0). For $j = 1, 2, \ldots,$

1. Call Algorithm SVSA1(ε^j,x^{j-1},y^{j-1}) to compute the spectral value set abscissa $\alpha_{\varepsilon^j}(A, B, C, D)$, also returning rightmost point λ^j, corresponding RP-compatible right and left eigenvectors x^j, y^j, and corresponding scalars β^j, γ^j defined as in (3.9) and (3.12), or equivalently (3.19), (3.16) using (3.14), (3.15), and (3.18), where $b = B^* y^j$ and $c = Cx^j$.
2. Set
$$\varepsilon^{j+1} = \varepsilon^j - \big(\text{Re } \lambda^j\big)\beta^j\gamma^j\big((y^j)^* x^j\big).$$

Since Newton's method may not converge if it is not initialized near the solution, it is standard practice to combine it with a bisection method to enforce convergence. While there are many variants of Newton-bisection methods in the literature, a good choice is the `rtsafe` routine [26, 16], a hybrid Newton-bisection method that maintains an interval known to contain the root, bisecting when the Newton step is either outside the interval or does not yield a sufficient decrease in the absolute function value (in this case, $|f(\varepsilon^j)| \equiv |\alpha_{\varepsilon^j}(A, B, C, D)| = |\text{Re } \lambda^j|$). This safeguard is also useful in the unlikely event that f is not differentiable at some values of ε^j. If one has nonnegative lower and upper bounds h_{lb} and h_{ub} on $\|G\|_\infty^c$, then these can be used to define initial lower and upper bounds ε_{lb} and ε_{ub} on ε_{opt} by

$$\varepsilon_{\text{lb}} = \frac{1}{h_{\text{ub}}} \le \varepsilon_{\text{opt}} \le \varepsilon_{\text{ub}} = \frac{1}{h_{\text{lb}}}.$$

When such bounds are not known, we use the trivial initial lower bound $\varepsilon_{\text{lb}} = 0$ and first call SVSA1 for $\varepsilon = \min(d_N, \delta/2)$, where d_N is the Newton step from 0 as

defined above and $\delta = \|D\|^{-1}$ (with $\delta = \infty$ if $\|D\| = 0$). The rationale for this is that $\|M(E)\|$ potentially blows up as $\|DE\| \to 1$, so it is inadvisable to try to compute $\alpha_\varepsilon(A, B, C, D)$ with ε closer to δ than needed. We then repeatedly do the substitution $\varepsilon \leftarrow \min(2\varepsilon, (\varepsilon + \delta)/2)$ until we find $\alpha_\varepsilon(A, B, C, D) > 0$ giving an upper bound $\varepsilon_{\rm ub}$. Putting all this together, we call the resulting algorithm NBHC1 (Newton-bisection method for the H_∞ norm for continuous-time systems). We emphasize, however, that this is still an idealized algorithm because there is no guarantee that Algorithm SVSA1 will return the correct value of $\alpha_{\varepsilon^j}(A, B, C, D)$.

4.2. The discrete-time case. In this case, the H_∞ norm is the quantity $\|G\|_\infty^d$ defined in (2.20) and (2.21). Assume that A is Schur stable so that the norm is finite. Equation (4.1) is replaced by

$$f(\varepsilon) \equiv \rho_\varepsilon(A, B, C, D) - 1 = 0,$$

where, as in the continuous-time case, f is a monotonically increasing function of ε. Defining $\lambda(\varepsilon)$ as the outermost point of $\sigma_\varepsilon(A, B, C, D)$, and assuming that it is nonzero and unique for a given value $\hat\varepsilon$, (4.2) is replaced by

$$\frac{d}{d\varepsilon}\rho_\varepsilon(A, B, C, D)\bigg|_{\varepsilon=\hat\varepsilon} = \frac{d}{d\varepsilon}|\lambda(\hat\varepsilon)|, \qquad \frac{d}{d\varepsilon}\lambda(\hat\varepsilon) = \frac{1}{y(\hat\varepsilon)^* x(\hat\varepsilon)} \tag{4.4}$$

when the eigenvectors are normalized by (2.7). Applying the first-order optimality conditions for (2.22) given in (2.23), we find that the right-hand side of (4.4) is a multiple of $\lambda(\hat\varepsilon)$. Equation (4.3) is replaced by

$$\frac{d}{d\varepsilon}\rho_\varepsilon(A, B, C, D)\bigg|_{\varepsilon=\hat\varepsilon} = \frac{d}{d\varepsilon}|\lambda(\hat\varepsilon)|, \quad \frac{d}{d\varepsilon}\lambda(\hat\varepsilon) = \frac{1}{\beta(\hat\varepsilon)\gamma(\hat\varepsilon)\big(y(\hat\varepsilon)^* x(\hat\varepsilon)\big)}$$

when the eigenvectors are normalized to be $\mathrm{RP}(\overline{\lambda}(\hat\varepsilon))$-compatible, with the right-hand side again a multiple of $\lambda(\hat\varepsilon)$. Algorithm NHC0 is replaced by the following.

Algorithm NHD0($\boldsymbol{\varepsilon^1}$) (Newton's method for H_∞ norm for discrete-time systems). For $j = 1, 2, \ldots,$

1. Compute the spectral value set radius $\rho_{\varepsilon^j}(A, B, C, D)$, along with the outermost point λ^j, corresponding $\mathrm{RP}(\overline{\lambda}^j)$-compatible right and left eigenvectors x^j, y^j and corresponding scalars β^j, γ^j defined as in (3.9) and (3.12), or equivalently (3.19), (3.16) using (3.14), (3.15), and (3.18), where $b = B^* y^j$ and $c = Cx^j$.
2. Set

$$\varepsilon^{j+1} = \varepsilon^j - \big(|\lambda^j| - 1\big)\beta^j\gamma^j\big|(y^j)^* x^j\big|.$$

This algorithm is quadratically convergent. A less idealized version is the following, where x^0, y^0 are set, in the absence of any other estimates, to right and left eigenvectors for λ_0, the outermost eigenvalue of A, normalized to be $\mathrm{RP}(\overline{\lambda}_0)$-compatible.

Algorithm NHD1($\boldsymbol{\varepsilon^1}$, $\boldsymbol{x_0}$, $\boldsymbol{y_0}$). For $j = 1, 2, \ldots,$

1. Call Algorithm SVSR1(ε^j, x^{j-1}, y^{j-1}) to compute the spectral value set radius $\rho_{\varepsilon^j}(A, B, C, D)$, also returning outermost point λ^j, corresponding RP $(\overline{\lambda^j})$-compatible right and left eigenvectors x^j, y^j and corresponding scalars

β^j, γ^j defined as in (3.9) and (3.12), or equivalently (3.19), (3.16) using (3.14), (3.15), and (3.18), where $b = B^* y^j$ and $c = Cx^j$.

2. Set

$$\varepsilon^{j+1} = \varepsilon^j - \big(|\lambda^j| - 1\big)\beta^j\gamma^j\big|(y^j)^* x^j\big|.$$

This algorithm has only local convergence guarantees but can be globalized under the assumption that Algorithm SVSR1 returns the correct answer for $\rho_{\varepsilon^j}(A, B, C, D)$ by combining it with the `rtsafe` routine, using an initial interval for ε as described above, giving an algorithm that we call NBHD1 (Newton-bisection method for H_∞ norm for discrete-time systems).

5. Numerical results. Our MATLAB codes SVSAR and HINFNORM implementing the spectral value set abscissa and radius and H_∞ norm algorithms (Algorithms SVSA1, SVSR1, NBHC1, and NBHD1) are freely available at the website http://cims.nyu.edu/~mert/software/hinfinity.html. We have tested these on many examples from the Compleib [23] and EigTool [30] collections. The example data and the script used to generate results shown in this section are also available on the website, together with the codes.

We use the following stopping condition in step 1 of the SVSA1 and SVSR1 algorithms: termination takes place at iteration $k \geq 1$ if

$$|\phi(\lambda_k) - \phi(\lambda_{k-1})| < \max(1, |\phi(\lambda_{k-1})|)\ \texttt{ftol},$$

where ϕ is the real part or modulus function, respectively, and we use the value $\texttt{ftol} = 10^{-12}$. We also set the maximum number of iterations of SVSA1 and SVSR1 to 100. The `rtsafe` routine, which is used to implement the NBHC1 and NBHD1 algorithms as explained above, uses both relative and absolute tolerances. The termination condition is

$$\Big|\varepsilon_k - \varepsilon_{k-1}\Big| \leq \max(\texttt{atol}, |\varepsilon_{k-1}|\ \texttt{rtol}),$$

where `rtol` and `atol` were both set to 10^{-10}. The results were obtained using MATLAB Release 2012b on a MacBook Air with a 1.8 GHz dual-core CPU (i7-2677M).

5.1. Small dense problems. We first consider small dense problems for which we can compare our results with a standard implementation of the BBBS algorithm for computing the H_∞ norm, namely, the `getPeakGain`[1] function in the Control Systems Toolbox of MATLAB [24] with tolerance 10^{-10}. For these problems, in Algorithms SVSA1 and SVSR1, all eigenvalues and right eigenvectors of the matrices $M_k := M(\varepsilon u_k v_k^*) = M + BF_kC$ are computed by calling the standard MATLAB eigenvalue routine `eig`. To compute the left eigenvectors of M_k, we make a second call to `eig`, computing the right eigenvectors of the transposed matrix M_k^T instead of inverting the possibly ill-conditioned matrix of right eigenvectors. Once right and left eigenvectors are computed, they are normalized to satisfy the RP-compatibility condition.

Compleib [23] is a database of continuous-time control-design examples, many from real applications, collected from the engineering literature. Each of these examples defines an "open-loop plant," described by a system of the form (1.1). In most

[1] This code, which calls routines from the SLICOT library to compute the eigenvalues of Hamiltonian matrices using a method that preserves Hamiltonian structure, supersedes the older function ss/norm starting with Release 2012a of MATLAB.

TABLE 5.1

Results for dense continuous-time problems from Compleib. The rel diff column shows the relative difference between the $\|G\|_\infty^c$ norm computed by Algorithm NBHC1 *and that computed by the BBBS algorithm implemented in* MATLAB. *The last two columns show the number of Newton iterates and the number of other iterates (doubling and bisection steps) in Algorithm* NBHC1.

Example	n	m	p	$\|\|G\|\|_\infty^c$	rel diff	ni	oi
CBM	351	2	1	$2.630e-01$	$-1.5e-15$	4	4
CSE2	63	32	1	$2.034e-02$	$+2.3e-11$	9	6
CM1	23	3	1	$8.165e-01$	$+7.7e-14$	3	16
CM3	123	3	1	$8.203e-01$	$-3.5e-06$	0	34
CM4	243	3	1	$1.411e+00$	$-9.3e-02$	0	33
HE6	23	16	6	$4.929e+02$	$0.0e+00$	25	16
HE7	23	16	9	$3.465e+02$	$0.0e+00$	4	5
ROC1	12	2	2	$1.217e+00$	$-4.7e-04$	3	5
ROC2	13	1	4	$1.334e-01$	$0.0e+00$	3	4
ROC3	14	11	11	$1.723e+04$	$-3.0e-11$	2	4
ROC4	12	2	2	$2.957e+02$	$-2.3e-06$	3	3
ROC5	10	2	3	$9.800e-03$	$0.0e+00$	5	26
ROC6	8	3	3	$2.576e+01$	$0.0e+00$	3	5
ROC7	8	3	1	$1.122e+00$	$-2.9e-10$	16	3
ROC8	12	7	1	$6.599e+00$	$-5.7e-11$	5	4
ROC9	9	5	1	$3.294e+00$	$-6.5e-13$	5	4
ROC10	9	2	2	$1.015e-01$	$+3.6e-12$	4	4

cases, this open-loop system is not Hurwitz stable, and hence its H_∞ norm is $+\infty$, according to our definition (2.15). However, by designing an appropriate controller, the open-loop system can typically be stabilized, defining a "closed-loop plant" associated with a different system of the form (1.1), one with a finite H_∞ norm. We obtained these stabilized closed-loop systems by computing third-order controllers using the HIFOO package [7].

Table 5.1 compares the results for the new NBHC1 algorithm with the BBBS algorithm for computing the H_∞ norm of the closed-loop systems obtained in this way for 17 different examples from Compleib. The columns headed n, m, and p specify the dimension of the state space and the number of outputs and inputs in (1.1). The column headed $\|G\|_\infty^c$ shows the value of the norm computed by the new algorithm. The column headed rel diff shows the relative difference between the value of the H_∞ norm computed by Algorithm NBHC1 and that obtained using `getPeakGain`; this is clarified below. The small values shown in this column for 13 of the 17 examples indicate that our algorithm converged to a global maximizer of the optimization problem in (2.14) and (2.15). In three of the other four cases, plots (not shown here) of the spectral values sets and the norm function $\|G(\mathbf{i}\omega)\|$ for $\omega \in \mathbb{R}$ indicate that Algorithm NBHC1 found only a local maximizer of (2.15) because Algorithm SVSA1 repeatedly returned a local maximizer of (2.13) for the sequence ε^j. However, the example CM3 was different. In this case, on a single occasion Algorithm SVSA1 returned a locally maximal value for $\alpha_\varepsilon(A, B, C, D)$ which was negative when the globally optimal value was positive, resulting in an invalid update to the lower bound $\varepsilon_{\mathrm{lb}}$ in Algorithm NBHC1. Subsequently, Algorithm SVSA1 returned globally optimal values, but the consequence was convergence of ε to the invalid lower bound, resulting in a final value of ε^{-1} that is not a stationary value of $\|G(\mathbf{i}\cdot)\|$. Spectral value set plots for CM3 and CM4 show that they have many locally rightmost points and that the norm function $\|G(\mathbf{i}\cdot)\|$ has many local maximizers, making these problems particularly difficult. However, in cases where the global maximizer was not found, restarting Algorithm NBHC1 at an eigenvector pair corresponding to a

different eigenvalue of A (not a rightmost one) sometimes resulted in convergence to the global maximizer.

The columns headed ni and oi show the number of Newton iterates and the number of other iterates (bisection steps inside `rtsafe` and initial steps taken to find an upper bound $\varepsilon_{\rm ub}$) in Algorithm NBHC1, so the sum of these two numbers is the number of calls to Algorithm SVSA1 for different values of ε^j. Once an upper bound is obtained and a first bisection step is taken, we observe that in many of the examples only Newton steps are taken subsequently and termination takes place rapidly.

According to (2.15), for stable A the norm $\|G\|_\infty^c$ is the maximum of $\|G(\lambda)\|$ over the imaginary axis. Algorithm NBHC1 does not verify the norm computation explicitly but returns a value for $\hat{\varepsilon}$ for which the rightmost point $\hat{\lambda}$ of $\sigma_{\hat{\varepsilon}}(A,B,C,D)$ is estimated to lie on the imaginary axis, and hence $\hat{\varepsilon}^{-1}$ is an estimate of the norm. Thus, for validation purposes, we need to actually compute $\|G(\mathbf{i}\omega_{\rm NBHC1})\|$ where $\omega_{\rm NBHC1} = \mathrm{Im}\ \hat{\lambda}$ to obtain a guaranteed lower bound for $\|G\|_\infty^c$, neglecting rounding errors in the computation of the largest singular value. Similarly, the BBBS algorithm implemented in `getPeakGain` returns a value for the norm, along with a second output argument, which we denote $\omega_{\rm BBBS}$, which is the algorithm's estimate of the corresponding point on the imaginary axis where the maximum is attained. So, again for validation purposes, we compute $\|G(\mathbf{i}\omega_{\rm BBBS})\|$ to obtain a guaranteed lower bound on the norm. The relative difference reported in the sixth column is therefore

$$\text{rel diff} = \frac{\|G(\mathbf{i}\omega_{\rm NBHC1})\| - \|G(\mathbf{i}\omega_{\rm BBBS})\|}{\max(\|G(\mathbf{i}\omega_{\rm NBHC1})\|, \|G(\mathbf{i}\omega_{\rm BBBS})\|)}.$$

The Compleib examples correspond to physical control systems that are all posed in continuous time. In order to create discrete-time examples of the form (1.2), we sampled these systems with sampling time $T_s = 1$, but these are usually not Schur stable. So, we attempted to stabilize these discrete-time open-loop systems with the HIFOOd package [25], which is an extension of HIFOO to discrete-time systems. In these examples, the order of the controller was taken to be 5 except for some of the smaller dimensional examples with $n < 10$, where we used a fourth-order controller. Since the examples in Table 5.1 are posed in continuous time, some of them could not be stabilized in discrete time by HIFOOd, so we added some new examples instead of these. The results for these discrete-time problems are shown in Table 5.2. The relative differences reported in the sixth column are computed in the same way as previously, except that now the norm must be evaluated at points on the unit circle returned by the two algorithms. The results are not as favorable as in Table 5.1, but the new algorithm still obtains accurate estimates of the global maximizer for 9 out of 17 examples. In most other cases the algorithm apparently found a local maximizer, with the exception of AC15, where all iterates were real and converged to $\lambda = 1$, a local minimizer of $\|G(e^{\mathbf{i}\theta})\|$ over $\theta \in [0, 2\pi)$.

5.2. Large sparse matrices. As in [14], our MATLAB implementation supports three kinds of matrix input: dense matrices, sparse matrices, and function handles, which specify the name of a MATLAB file implementing matrix-vector products. In the last two cases, we compute the rightmost eigenvalue of M_k and corresponding right eigenvector with the MATLAB routine `eigs`, which is an interface for ARPACK, a well-known code implementing the implicitly restarted Arnoldi method [22]. This was called with its default parameters, except that the number of eigenpairs requested was increased from the default of 6 to 8. Since `eigs` does not require M_k explicitly but needs only the ability to do matrix-vector products with M_k, it also accepts as input

TABLE 5.2

Results for dense discrete-time version of problems from Compleib. The rel diff column shows the relative difference between the $\|G\|_\infty^d$ norm computed by Algorithm NBHD1 *and that computed by the BBBS algorithm implemented in* MATLAB. *The last two columns show the number of Newton iterates and the number of other iterates (doubling and bisection steps) in Algorithm* NBHD1.

Example	n	m	p	$\|\|G\|\|_\infty^d$	rel diff	ni	oi
AC5	8	4	4	$7.626e+01$	$-1.3e-14$	2	4
AC12	8	1	3	$1.082e+01$	$-3.6e-11$	3	3
AC15	8	6	4	$2.369e+01$	$-3.1e-04$	2	4
AC16	8	6	4	$1.818e+01$	$-6.1e-03$	4	5
AC17	8	4	4	$3.001e+05$	$-2.2e-11$	2	4
REA1	8	4	4	$7.438e+02$	$-2.1e-11$	3	4
AC1	10	2	3	$1.500e-01$	$-2.1e-03$	5	3
AC2	10	5	3	$3.056e-01$	$-1.2e-12$	4	4
AC3	10	5	5	$1.912e+01$	$-5.4e-11$	4	4
AC6	12	7	7	$5.294e+07$	$-1.4e-07$	3	6
AC11	10	5	5	$2.185e+07$	$-2.8e-08$	2	4
ROC3	16	11	11	$2.337e+01$	$-4.2e-11$	3	6
ROC5	12	2	3	$3.911e+03$	$-1.1e-09$	3	4
ROC6	10	3	3	$1.720e+01$	$-1.6e-05$	8	10
ROC7	10	3	1	$1.109e+00$	$-1.0e-07$	9	6
ROC8	14	7	1	$6.283e+04$	$-3.3e-11$	3	4
ROC9	11	5	1	$2.861e+01$	$-5.6e-05$	3	4

either a sparse matrix or a function handle. The last is crucial, because we must avoid computing the dense matrix $M_k = A + BF_kC$ explicitly. On the other hand, writing an efficient function to compute matrix-vector products with M_k is straightforward, and it is a handle for this function that we pass to `eigs`, which computes the largest eigenvalue with respect to real part or modulus, respectively. As in the dense case, we compute the left eigenvector of M_k by a second call to the eigenvalue routine, in this case `eigs`, to find the right eigenvector of the transposed matrix M_k^T. Thus, when the input to our implementation is a function handle, it must implement matrix-vector products with M_k^T as well as with M_k.

The results for Algorithms NHBC1 and NHBD1 on large sparse continuous-time and discrete-time problems are summarized in Tables 5.3 and 5.4, respectively. In Table 5.3 the first example `NN18` is from Compleib; in the other examples the A matrix is obtained from EigTool with B, C, and D generated randomly. The Shift column in Table 5.3 shows the multiple of I that we subtracted from the EigTool matrix to make it Hurwitz stable. Similarly, the Scale column in Table 5.4 shows the scale factor that we divided into the EigTool matrix to make it Schur stable. The `tolosa`

TABLE 5.3

Results of Algorithm NBHC1 *on sparse continuous-time problems from Compleib and EigTool. The last seven columns show the computed $\|G\|_\infty^c$ norm, the number of Newton iterates, the number of other iterates, the number of pairs of calls to eigs, the time required in seconds by Algorithm* NBHC1*, the time required in seconds for the BBBS algorithm, and the relative difference in the norm.*

Example	Shift	n	m	p	$\|\|G\|\|_\infty^c$	ni	oi	`eigs`	t-nbhc1	t-bbbs	rel diff
NN18	0	1006	2	1	1.0234	3	3	40	62	93	$-1.3e-15$
dwave	$-$I	2048	6	4	38020	4	3	21	116	694	$0.0e+00$
markov	-2I	5050	6	4	6205.5	2	4	21	107	-	-
pde	-10I	2961	6	4	368.75	4	3	50	67	-	-
rdbrusselator	$-$I	3200	6	4	1868.3	3	3	54	305	-	-
skewlap3d	0	24389	6	4	217.4	3	36	511	17855	-	-
sparserandom	-3I	10000	6	4	141905	2	3	13	11	-	-

TABLE 5.4

Results of Algorithm NBHD1 *on sparse discrete-time problems from EigTool. The last seven columns show the computed $\|G\|_\infty^d$ norm, the number of Newton iterates, the number of other iterates, the number of pairs of calls to eigs, the time required in seconds by Algorithm* NBHD1*, the time required in seconds for the BBBS algorithm, and the relative difference in the norm.*

Example	Scale	n	m	p	$\|\|G\|\|_\infty^d$	ni	oi	`eigs`	t-nbhd1	t-bbbs	rel diff
`dwave`	1	2048	6	4	39027	4	3	24	134	1689	0.0e+00
`markov`	2	5050	6	4	4112.7	4	9	54	252	-	-
`pde`	10	2961	6	4	3645.6	3	11	636	792	-	-
`rdbrusselator`	120	3200	6	4	3891.8	4	3	37	58	-	-
`skewlap3d`	11000	24389	6	4	30342	2	32	637	16476	-	-
`sparserandom`	3	10000	6	4	3.951e+06	2	3	12	9	-	-
`tolosa`	5000	4000	6	4	5.663e+06	4	6	424	2250	-	-

problem does not appear in Table 5.3 because `eigs` failed to compute the rightmost eigenvalue of A to the default required accuracy using the default input parameters, and the `NN18` problem does not appear in Table 5.4 because HIFOOd could not find a stable discrete-time closed-loop system. The column headed `eigs` in Tables 5.3 and 5.4 reports the number of pairs of calls made to `eigs` (to compute right and left eigenvectors). The next two columns report the timing in seconds for our algorithm and for the BBBS algorithm implemented in `getPeakGain` respectively. These timings are included only to give some idea of the efficiency of the new algorithm, which is implemented in MATLAB and far from optimized; in contrast the `getPeakGain` code calls compiled Fortran routines to solve the Hamiltonian eigenvalue problems but is not intended for large sparse problems and hence was run only for the smaller problems. The final column shows the relative difference in the computed value in the cases where we were able to run both codes.

For most of the problems reported in Tables 5.3 and 5.4, the work required for the H_∞ norm computation is just a few dozen times as much work as the computation of the spectral abscissa or spectral radius of A. The most notable exception is the largest problem `skewlap3d`, which required many iterations to find an upper bound $\varepsilon_{\rm ub}$. For this example, if we simply set the upper bound $\varepsilon_{\rm ub}$ a priori to $0.999/\|D\|$, the number of iterations is dramatically reduced, but for reasons mentioned in section 4.1, this does not seem to be a good strategy in general.

6. Open questions and future work. In this paper we have presented new efficient algorithms for approximating the spectral value set abscissa and radius for a linear dynamical system along with the H_∞ norm of the associated transfer matrix function. We conclude by briefly discussing various issues that appear to be worth investigating in future work.

6.1. Multiplicity, controllability, and observability. Algorithms SVSA0, SVSA1, SVSR0, and SVSR1 all break down if they generate a rightmost eigenvalue λ_k which is not simple, controllable, and observable, as these properties are needed for the basic step of the algorithm to be well defined. Except for trivial examples, in the context of floating point arithmetic it is simply not possible to check whether an eigenvalue is simple, controllable, or observable because it requires determining whether a computed quantity is exactly zero. For this reason, our codes do not attempt to check these properties. A natural question, however, is what effect *nearness* to multiplicity, uncontrollability, or unobservability has on algorithmic behavior.

If the eigenvalue λ_k is nearly multiple, then $y_k^* x_k$ may be very small (it is zero if the eigenvalue is defective, that is, has algebraic multiplicity greater than its geometric

multiplicity), while if the eigenvalue is nearly uncontrollable or nearly unobservable, $\|B^*y_k\|$ and $\|Cx_k\|$ may be very small (the former is zero if the eigenvalue is uncontrollable and the latter is zero if the eigenvalue is unobservable). Indeed, instead of focusing on the *rightmost* eigenvalue, one might want to choose an eigenvalue that balances nearness to being the rightmost eigenvalue with desirability of the quantities $y_k^*x_k$, $\|B^*y_k\|$, and $\|Cx_k\|$. Since $y_k^*x_k$ appears in the denominator of (3.4), it is actually desirable that it be small: this reflects the fact that nearly defective eigenvalues move rapidly under perturbation. On the other hand, B^*y_k and Cx_k appear in the numerator of (3.4) so it is undesirable that they have small norm; nearly uncontrollable or nearly unobservable eigenvalues move slowly under perturbation. The latter point is addressed in section 3.2 of a recent paper by Benner and Voigt [3], which presents an algorithm closely related to ours, as discussed further below.

6.2. Convergence of the spectral value set abscissa algorithms. As noted in section 3.4, we have established that for sufficiently small ε, Algorithm SVSA0 converges locally to rightmost points of the spectral value set with a linear rate of convergence. Yet, we typically see convergence for large values of ϵ as well. It does not seem likely that the proof techniques developed in [14] for small ε can be extended to large ε so a different approach may be needed.

Algorithm SVSA1 was introduced in section 3.5 to ensure that the iterates Re λ_k generated by the Algorithm are monotonically increasing. We find that in practice this is very effective in some cases, enabling convergence when the basic algorithm SVSA0 fails, although on most of our test examples there was no difference between the behavior of the two algorithms; in other words, the quantity t in Algorithm SVSA1 was always one. However, Algorithm SVSA1 could fail if the quantity Re ψ_k is zero, or, as is perhaps more likely, it converges to zero over a sequence of iterations resulting in stagnation of the algorithm. This needs further investigation.

Characterizing when the spectral value set abscissa algorithms converge to global rather than local maximizers of (2.13) would seem to be a hard problem, even though the maximization problem has only two real variables. As was observed for problem AC15 in the discrete-time case, convergence to a local minimizer is also possible but this seems unlikely unless all iterates are real, as was the case for AC15. The possibility that a local minimizer could correspond to a fixed point of Algorithm SVSA0 is covered by Theorem 3.2. However, as noted at the end of section 3.3, we conjecture that such fixed points are not attractive, meaning that reinitializing the algorithm with λ set to a small complex perturbation of the final real iterate will almost surely result in moving away from the local minimizer; see [14, Figure 4.1]. It would be interesting to prove or disprove this conjecture.

6.3. Convergence of the H_∞ norm algorithms. The hybrid Newton-bisection algorithms NBHC1 and NBHD1 are each guaranteed to find a root of the associated function $f(\varepsilon) = 0$ provided that Algorithms SVSA1 and SVSR1 compute the spectral value set abscissa and radius functions correctly for each ε^j, but in practice this cannot be guaranteed. Furthermore, a natural question is how accurately the spectral value set abscissa and radius computations should be done to obtain reasonable accuracy of the H_∞ norm computations. We saw in the results in Table 5.1 that Algorithm NBHC1 sometimes returns only a local maximizer of (2.14) and (2.15), but that it is also possible, as in the case of CM3, that the final value is not a stationary value of $\|G(\mathbf{i}\cdot)\|$ because of an invalid update to the lower bound $\varepsilon_{\rm lb}$. This could be avoided by reconsidering the validity of the lower bound if the ε iterates converge to it. On the positive side, when Algorithm SVSA1 returns a positive value that is not

globally optimal it still results in a valid update to the upper bound $\varepsilon_{\rm ub}$, suggesting that we should terminate the spectral value set and radius computations as soon as they return a positive value.

Recall from Theorem 4.1 and Remark 4.3 that the idealized algorithms NHC0 and NHD0 are quadratically convergent as long as the rightmost point of $\sigma_{\varepsilon_{\rm opt}}(A, B, C, D)$ is unique (or part of a complex conjugate pair in the real case). A natural question is, when might this assumption fail to hold? One answer is that it may not hold if the A, B, C, D matrices depend on parameters over which optimization has been done to minimize the corresponding H_∞ norm, as explained in [8] for the case $B = C = I$, $D = 0$ (maximizing the distance to instability for A). However, even then, the property will fail only in the limiting case of optimal choices of the parameters.

6.4. Relationship with other recent work. Several papers on related topics have appeared recently. Benner and Voigt [3] have also generalized the algorithm of [14] to compute the H_∞ norm of a transfer matrix. The basic algorithm is quite similar, but the specific problem addressed is different. Their linear dynamical system is a descriptor system, which introduces a (possibly singular) matrix factor on the left-hand side of the first equation $\dot{x} = Ax + Bu$ in (1.1) but sets $D = 0$ in the second equation $y = Cx + Du$. In addition, as mentioned above, they select the eigenvalue to which the basic algorithm is to be applied at each step by balancing the distance to rightmost eigenvalue with measures of the observability and controllability of the eigenvalue. Kressner and Vandereycken [21] have introduced a more efficient variant of the algorithm of [14] based on computing the pseudospectral abscissa of a small rectangular matrix pencil. It would be interesting to investigate whether similar ideas can be used for the spectral value set abscissa. Finally, Freitag and Spence [10] have introduced an efficient Newton method for computing the distance to instability for a large sparse matrix A and extensions of this to an algorithm to compute the H_∞ norm of a transfer matrix are in progress [11]. In conclusion, this is an active research area and it will be interesting to compare the advantages and disadvantages of various methods for estimating the H_∞ norm in the near future.

Acknowledgments. Special thanks to Daniel Kressner for suggesting the focus on spectral value sets. Thanks also to the referees for carefully reading the paper and making many useful suggestions. We are also very grateful to Tim Mitchell for his assistance at the revision stage of this paper. He made substantial improvements to our original MATLAB codes and helped us obtain a clearer understanding of the behavior of the algorithms in practice.

REFERENCES

[1] R. Alam, S. Bora, R. Byers, and M. L. Overton, *Characterization and construction of the nearest defective matrix via coalescence of pseudospectral components*, Linear Algebra Appl., 435 (2011), pp. 494–513.

[2] P. J. Antsaklis and A. N. Michel, *A Linear Systems Primer*, Birkhäuser Boston, Boston, MA, 2007.

[3] P. Benner and M. Voigt, *A Structured Pseudospectral Method for h_∞-Norm Computation of Large-Scale Descriptor Systems*, Tech. report MPIMD/12-10. Max Planck Institute, Magdeburg, 2012.

[4] S. Boyd and V. Balakrishnan, *A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its $\mathbf{L}_\infty$-norm*, Systems Control Lett., 15 (1990), pp. 1–7.

[5] N. A. Bruinsma and M. Steinbuch, *A fast algorithm to compute the H^∞-norm of a transfer function matrix*, Systems Control Lett., 14 (1990), pp. 287–293.

[6] J. V. Burke, A. S. Lewis, and M. L. Overton, *Robust stability and a criss-cross algorithm for pseudospectra*, IMA J. Numer. Anal., 23 (2003), pp. 359–375.

[7] J. V. Burke, D. Henrion, A. S. Lewis, and M. L. Overton, *HIFOO: a MATLAB package for fixed-order controller design and H_∞ optimization*, in Proceedings of the Fifth IFAC Symposium on Robust Control Design, Toulouse, France, 2006.
[8] J. V. Burke, A. S. Lewis, and M. L. Overton, *A nonsmooth, nonconvex optimization approach to robust stabilization by static output feedback and low-order controllers*, in Proceedings of the Fourth IFAC Symposium on Robust Control Design, Milan, Italy, 2003.
[9] R. Byers, *A bisection method for measuring the distance of a stable matrix to the unstable matrices*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 875–881.
[10] M. A. Freitag and A. Spence, *A Newton-based method for the calculation of the distance to instability*, Linear Algebra Appl., 435 (2011), pp. 3189–3205.
[11] M. Freitag, *private communication*, 2012, University of Bath.
[12] Y. Genin, P. Van Dooren, and V. Vermaut, *Convergence of the calculation of H_∞-norms and related questions*, in Proceedings of MTNS, 1998, pp. 429–432.
[13] G. H. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 1983.
[14] N. Guglielmi and M.L. Overton, *Fast algorithms for the approximation of the pseudospectral abscissa and pseudospectral radius of a matrix*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1166–1192.
[15] M. Gürbüzbalaban and M.L. Overton, *Some regularity results for the pseudospectral abscissa and pseudospectral radius of a matrix*, SIAM J. Optim., 22 (2012), pp. 281–285.
[16] N. J. Higham, *Computing a nearest symmetric positive semidefinite matrix*, Linear Algebra Appl., 103 (1988), pp. 103–118.
[17] D. Hinrichsen and A.J. Pritchard, *Mathematical Systems Theory* I*: Modelling, State Space Analysis, Stability and Robustness*, Springer, Berlin, 2005.
[18] D. Hinrichsen and N. K. Son, *The complex stability radius of discrete-time systems and symplectic pencils*, in Proceedings of the 28th IEEE Conference on Decision and Control, vol. 1–3, Tampa, FL, 1989, pp. 2265–2270.
[19] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1990.
[20] M. Karow, *Geometry of Spectral Value Sets*, Ph.D. thesis, Universität Bremen, Germany, 2003.
[21] D. Kressner and B. Vandereycken, *Subspace Methods for Computing the Pseudospectral Abscissa and the Stability Radius*, Tech. report, MATHICSE 13.2012, École Polytechnique Fédérale de Lausanne, Switzerland, 2012.
[22] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, Software Environ. Tools SIAM, Philadelphia, PA, 1997.
[23] F. Leibfritz, *Compleib: Constrained Matrix Optimization Problem Library*, http://www. compleib.de (2006).
[24] Matlab *Control System Toolbox, Release* 2012b, http://www.mathworks.com.
[25] A. P. Popov, H. Werner, and M. Millstone, *Fixed-structure discrete-time H_∞ controller synthesis with HIFOO*, in 49th IEEE Conference on Decision and Control, 2010, pp. 3152 –3155.
[26] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, Cambridge, UK, 1986.
[27] L. N. Trefethen and M. Embree, *Spectra and Pseudospectra: the Behavior of Nonnormal Matrices and Operators*, Princeton University Press, Princeton, NJ, 2005.
[28] P. Van Dooren, 2012. Private communication.
[29] C. Van Loan, *How near is a stable matrix to an unstable matrix?*, in Linear algebra and Its Role in Systems Theory (Brunswick, Maine, 1984), Contemp. Math. 47, AMS, Providence, RI, 1985, pp. 465–478.
[30] T. G. Wright, *Eigtool: A Graphical Tool for Nonsymmetric Eigen-Problems*, http://www.comlab.ox.ac.uk/pseudospectra/eigtool (2002).
[31] K. Zhou, K. Glover, and J. Doyle, *Robust and Optimal Control*, Prentice-Hall, Englewood Cliffs, NJ, 1995.