Approximating the Real Structured Stability Radius with Frobenius Norm Bounded Perturbations

N. Guglielmi* M. Gürbüzbalaban† T. Mitchell‡ M. L. Overton§

Dec. 30th, 2016

Abstract

We propose a fast method to approximate the real stability radius of a linear dynamical system with output feedback, where the perturbations are restricted to be real valued and bounded with respect to the Frobenius norm. Our work builds on a number of scalable algorithms that have been proposed in recent years, ranging from methods that approximate the complex or real pseudospectral abscissa and radius of large sparse matrices (and generalizations of these methods for pseudospectra to spectral value sets) to algorithms for approximating the complex stability radius (the reciprocal of the H_∞ norm). Although our algorithm is guaranteed to find only upper bounds to the real stability radius, it seems quite effective in practice. As far as we know, this is the first algorithm that addresses the Frobenius-norm version of this problem. Because the cost mainly consists of computing the eigenvalue with maximal real part for continuous-time systems (or modulus for discrete-time systems) of a sequence of matrices, our algorithm remains very efficient for large-scale systems provided that the system matrices are sparse.

1 Introduction

Consider a linear time-invariant dynamical system with output feedback defined, for continuous-time systems, by matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times p}$, $C \in \mathbb{R}^{m\times n}$ and $D \in \mathbb{R}^{m\times p}$ as

$$\dot{x} = Ax + Bw \tag{1}$$
$$z = Cx + Dw \tag{2}$$

where w is a disturbance feedback depending linearly on the output z [HP05, p. 538]. For simplicity, we restrict our attention to continuous-time systems for most of the paper, but we briefly explain how to extend our results and methods to discrete-time systems in Section 6.

The real stability radius, sometimes called the real structured stability radius, is a well known quantity for measuring robust stability of linear dynamical systems with output feedback [HP90a, HP90b, HK94, HP05, ZGD95, Kar03]. It measures stability under a certain class of real perturbations where the size of the perturbations are measured by a given norm $\|\cdot\|$. Most of the literature has

*Dipartimento di Matematica Pura e Applicata, Università dell'Aquila, via Vetoio (Coppito), I-67010 L'Aquila, Italy (guglielm@univaq.it). Supported in part by the Italian Ministry of Education, Universities and Research (M.I.U.R.) and by Istituto Nazionale di Alta Matematica - Gruppo Nazionale di Calcolo Scientifico (INdAM-G.N.C.S).

†Department of Management Science and Information Systems, Rutgers University, New Brunswick, NJ, USA (mgurbuzbalaban@business.rutgers.edu)

‡Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, 39106 Germany (mitchell@mpi-magdeburg.mpg.de). Previously supported by National Science Foundation Grant DMS-1317205 at New York University.

§Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA (overton@cims.nyu.edu). Supported in part by National Science Foundation Grant DMS-1620083.

focused on spectral norm bounded perturbations for which there exists a characterization in terms of an explicit formula [QBR+95] and a level-set algorithm [SVDT96]. This algorithm has been proven to be convergent; however, it is not practical for systems where large and sparse matrices arise as it requires a sequence of Hamiltonian eigenvalue decompositions, each with a complexity of $\mathcal{O}(n^3)$.

As an alternative to the spectral norm, Frobenius-norm bounded perturbations have also been of interest to the control community [LKL96, BS99, BS98, BB01, Bob99, BBD01]. It has been argued that the Frobenius norm is easier to compute and is more advantageous to consider in certain types of control systems [Bob99, BBD01], admitting natural extensions to infinite-dimensional systems [BB01]. In the special case $B = C = I, D = 0$, there exists an algorithm [Bob99, BBD01] that gives upper and lower bounds for the Frobenius-norm bounded real stability radius; however, there is no algorithm to our knowledge that is applicable in the general case. Indeed, [BV14] describes this as an unsolved research problem. In this paper, we present the first method to our knowledge that provides good approximations to the Frobenius-norm bounded real stability radius.

Our method relies on two foundations. The first is the theory of spectral value sets associated with the dynamical system (1)–(2) as presented in [HP05, Chapter 5]. The second is the appearance of a number of recent iterative algorithms that find rightmost points of spectral value sets of various sorts, beginning with the special case of matrix pseudospectra (the case $B = C = I, D = 0$) [GO11], followed by a related method for pseudospectra [KV14] and extensions to real-structured pseudospectra [GL13, GM15, Ros15, Gug16], and to spectral value sets associated with (1)–(2) [GGO13, MO16] and related descriptor systems [BV14].

The paper is organized as follows. In Section 2 we introduce spectral value sets, establishing a fundamental relationship between the spectral value set abscissa and the stability radius. In Section 3 we introduce an ordinary differential equation whose equilibria are generically associated with rightmost points of Frobenius-norm bounded real spectral value sets. In Section 4 we present a practical iterative method to compute these points. This leads to our method for approximating the Frobenius-norm bounded real stability radius, presented in Section 5. We outline extensions to discrete-time systems in Section 6, present numerical results in Section 7, and make concluding remarks in Section 8.

2 Fundamental Concepts

Throughout the paper, $\|\cdot\|_2$ denotes the matrix 2-norm (maximum singular value), whereas $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm (associated with the trace inner product). The usage $\|\cdot\|$ means that the norm may be either $\|\cdot\|_2$ or $\|\cdot\|_{\mathrm{F}}$, or both when they coincide, namely for vectors or rank-one matrices. We use the notation $\mathbb{C}_-$ to denote the open *left* half-plane $\{\lambda : \mathrm{Re}(\lambda) < 0\}$ and $H_{+=}$ to denote the closed *upper* half-plane $\{\lambda : \mathrm{Im}(\lambda) \geq 0\}$.

2.1 Spectral Value Sets and μ-Values

Given real matrices A, B, C and D defining the linear dynamical system (1)–(2), linear feedback $w = \Delta\, z$ leads to a *perturbed system matrix* with the linear fractional form

$$M(\Delta) = A + B\Delta(I - D\Delta)^{-1}C \quad \text{for } \Delta \in \mathbb{K}^{p\times m} \tag{3}$$

where the field $\mathbb{K}$ is either $\mathbb{R}$ or $\mathbb{C}$. Note that since

$$\|D\Delta\|_2 \leq \|D\|_2\|\Delta\|_2 \leq \|D\|_2\|\Delta\|_{\mathrm{F}} \tag{4}$$

we can ensure that $M(\Delta)$ is well defined by assuming $\|\Delta\| \leq \varepsilon$ and $\varepsilon\|D\|_2 < 1$, regardless of whether $\|\Delta\|$ is $\|\Delta\|_2$ or $\|\Delta\|_{\mathrm{F}}$.

Definition 2.1. *Let $\varepsilon \in \mathbb{R}$, with $\varepsilon\|D\|_2 < 1$. Define the* spectral value set *with respect to the norm $\|\cdot\|$ and the field $\mathbb{K}$ as*

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

Here σ denotes spectrum. Note that

$$\sigma_\varepsilon^{\mathbb{K},\|\cdot\|_2}(A,B,C,D) \supseteq \sigma_\varepsilon^{\mathbb{K},\|\cdot\|_{\mathrm{F}}}(A,B,C,D) \supseteq \sigma_0^{\mathbb{K},\|\cdot\|}(A,B,C,D) = \sigma(A).$$

It is well known that when $\mathbb{K} = \mathbb{C}$, the spectral value set can equivalently be defined as the set of points $s \in \mathbb{C}$ for which the spectral norm, i.e., the largest singular value, of the transfer matrix

$$G(s) = C(sI - A)^{-1}B + D$$

takes values at least $1/\varepsilon$ [HP05, Chap. 5]. Furthermore, it is well known that

$$\sigma_\varepsilon^{\mathbb{C},\|\cdot\|}(A,B,C,D) = \bigcup \left\{ \sigma(M(\Delta)) : \Delta \in \mathbb{C}^{p\times m}, \|\Delta\| \le \varepsilon \text{ and } \operatorname{rank}(\Delta) \le 1 \right\}. \tag{5}$$

As a consequence of this rank-one property, it is clear that

$$\sigma_\varepsilon^{\mathbb{C},\|\cdot\|_2}(A,B,C,D) = \sigma_\varepsilon^{\mathbb{C},\|\cdot\|_{\mathrm{F}}}(A,B,C,D).$$

In contrast, when perturbations are restricted to be real, the inclusion

$$\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_2}(A,B,C,D) \supseteq \sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$$

is generically strict. Instead of ordinary singular values, we must consider *real structured singular values* or *real μ-values*, that is with perturbations restricted to real matrices. This topic is discussed at length in [HP05, Section 4.4], allowing additional structure to be imposed on Δ beyond simply $\Delta \in \mathbb{K}^{p\times m}$, and treating a general class of operator norms, including the spectral norm, but not, however, the Frobenius norm. See also [Kar03, Chapter 6].

Definition 2.2. *The μ-value of a matrix $H \in \mathbb{C}^{m\times p}$ with respect to the field $\mathbb{K}$ and the norm $\|\cdot\|$ is defined by*

$$\mu_{\mathbb{K}}^{\|\cdot\|}(H) = \left[\inf\left\{\|\Delta\| \;:\; \Delta \in \mathbb{K}^{p\times m},\ \det(I - H\Delta) = 0\right\}\right]^{-1}. \tag{6}$$

We use the convention that taking the infimum over the empty set always yields ∞ and that $\infty^{-1} = 0$, so that $\mu_{\mathbb{K}}^{\|\cdot\|}(0)^{-1} = \infty$. Definition 2.2 defines the *real μ-value* when $\mathbb{K} = \mathbb{R}$ for complex matrices H as well as real matrices.

The following lemma is well known in the case of the spectral norm, where it is usually known as the Eckart-Young theorem. See [QBR+95, Lemma 1] and [HP05, Prop. 4.4.11] for extensions to other structures and other operator norms. A key point to note here is that while this result holds both for $\mathbb{K} = \mathbb{C}$ and $\mathbb{K} = \mathbb{R}$, it does *not* hold for the real μ value when H is complex.

Lemma 2.3. *Let $H \in \mathbb{K}^{m\times p}$ and let $\|\cdot\|$ be either $\|\cdot\|_2$ or $\|\cdot\|_{\mathrm{F}}$. Then, $\mu_{\mathbb{K}}^{\|\cdot\|}(H) = \|H\|_2$.*

Proof. If $H = 0$, the result is clear. Assume $H \neq 0$. If $\|\Delta\| < \|H\|_2^{-1}$, then we have from (4) that $\det(I - H\Delta) \neq 0$. This shows that $\mu_{\mathbb{K}}^{\|\cdot\|}(H) \le \|H\|_2$. For the reverse inequality, let H have the singular value decomposition $U\Sigma V^{\mathsf{T}}$ where U and V are unitary and

$$\Sigma = \operatorname{diag}\{\sigma_1(H), \sigma_2(H), \ldots, \sigma_{\min\{p,m\}}(H)\}$$

is a diagonal matrix with singular values on the diagonal in descending order by magnitude. Define

$$\Delta = V \operatorname{diag}\{\sigma_1(H)^{-1}, 0, \ldots, 0\} U^{\mathsf{T}}.$$

Then $\|\Delta\|_{\mathrm{F}} = \|\Delta\|_2 = \|H\|_2^{-1}$ and $\det(I - H\Delta) = 0$. Furthermore, if $\mathbb{K} = \mathbb{R}$ then since H is real, Δ is also real. This shows that $\mu_{\mathbb{K}}^{\|\cdot\|}(H) \ge \|H\|_2$. □

Combining [HP05, Lemma 5.2.7] with Lemma 2.3 results in the following corollary. This may be compared with [HP05, Corollary 5.2.8], which treats a more general class of structured perturbations, but is restricted to operator norms.

Corollary 2.4. *Let $\|\cdot\|$ be either $\|\cdot\|_2$ or $\|\cdot\|_{\mathrm{F}}$. Let $s \in \mathbb{C}\backslash\sigma(A)$ and $\|D\| < \mu_{\mathbb{K}}^{\|\cdot\|}(G(s))$ and define*

$$\varepsilon(s) = \min\left\{\|\Delta\| \ : \ \Delta \in \mathbb{K}^{p\times m}, s \in \sigma(M(\Delta))\right\}.$$

Then,

$$\varepsilon(s) = \left(\mu_{\mathbb{K}}^{\|\cdot\|}(G(s))\right)^{-1}.$$

This leads to the following theorem which may be compared to [HP05, Theorem 5.2.9], which again does not treat the Frobenius norm.

Theorem 2.5. *Let $\|\cdot\|$ be either $\|\cdot\|_2$ or $\|\cdot\|_{\mathrm{F}}$. Suppose $\varepsilon > 0$ and $\varepsilon\|D\|_2 < 1$. Then*

$$\sigma_\varepsilon^{\mathbb{K},\|\cdot\|}(A,B,C,D) = \sigma(A) \bigcup \left\{s \in \mathbb{C}\backslash\sigma(A) \ : \ \mu_{\mathbb{K}}^{\|\cdot\|}(G(s)) \geq \varepsilon^{-1}\right\}.$$

Proof. Suppose $s \in \sigma(M(\Delta)) \cap \{s \in \mathbb{C}\backslash\sigma(A)\}$, $\Delta \in \mathbb{K}^{p\times m}$ and $\|\Delta\| \leq \varepsilon$. By [HP05, Lemma 5.2.7], we have $\left(\mu_{\mathbb{K}}^{\|\cdot\|}(G(s))\right)^{-1} \leq \|\Delta\| \leq \varepsilon$. Conversely, if $s \in \mathbb{C}\backslash\sigma(A)$ and $\left(\mu_{\mathbb{K}}^{\|\cdot\|}(G(s))\right) \geq \varepsilon^{-1}$, then we have $\mu_{\mathbb{K}}^{\|\cdot\|}(G(s)) > \|D\|_2$ and by Corollary 2.4, there exists Δ with $\|\Delta\| = \left(\mu_{\mathbb{K}}^{\|\cdot\|}(G(s))\right)^{-1} = \varepsilon$ such that $s \in \sigma(M(\Delta))$. □

Note that even when A, B, C, D are real, the transfer function $G(s)$ is normally complex for $s \notin \mathbb{R}$ so it is not generally the case that $\mu_{\mathbb{R}}^{\|\cdot\|}(G(s)) = \|G(s)\|_2$. For real spectral value sets defined by the spectral norm, the optimal perturbation that appears in Definition 2.2 of the μ-value can in fact always be chosen to have rank at most two [QBR+95, Section 2], leading to the formula

$$\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_2}(A,B,C,D) = \bigcup\left\{\sigma(M(\Delta)) : \Delta \in \mathbb{R}^{p\times m}, \|\Delta\|_2 \leq \varepsilon \text{ and } \mathrm{rank}(\Delta) \leq 2\right\}.$$

We make the assumption in this paper that the same property holds for the Frobenius norm, but, for brevity, we leave a detailed justification of this to future work.

2.2 The Stability Radius

Because we are focusing on the continuous-time dynamical system (1)–(2), the stability region of interest is the open left half-plane $\mathbb{C}_-$. We say that A is *stable* if $\sigma(A) \in \mathbb{C}_-$, in which case, for sufficiently small ε, the spectral value set $\sigma_\varepsilon^{\mathbb{K},\|\cdot\|}(A,B,C,D)$ is also in $\mathbb{C}_-$. The stability radius $r_{\mathbb{K}}^{\|\cdot\|}$ measures the size of the minimal perturbation that destabilizes the matrix or results in $M(\Delta)$ being undefined [HP05, Def. 5.3.1].

Definition 2.6. *The stability radius $r_{\mathbb{K}}^{\|\cdot\|}$ is defined with respect to the field $\mathbb{K}$ and the norm $\|\cdot\|$ as*

$$r_{\mathbb{K}}^{\|\cdot\|}(A,B,C,D) = \inf\{\|\Delta\| : \Delta \in \mathbb{K}^{p\times m}, \det(I - D\Delta) = 0 \text{ or } \sigma(M(\Delta)) \not\subset \mathbb{C}_-\}.$$

The characterization

$$r_{\mathbb{K}}^{\|\cdot\|}(A,B,C,D) = \min\left(\left[\mu_{\mathbb{K}}^{\|\cdot\|}(D)\right]^{-1}, \inf_{\omega\in\mathbb{R}}\left[\mu_{\mathbb{K}}^{\|\cdot\|}(G(\mathbf{i}\omega))\right]^{-1}\right) \tag{7}$$

is well known for operator norms [HP05, Theorem 5.3.3]. Corollary 2.4 and Lemma 2.3 extend [HP05, Theorem 5.3.3] beyond operator norms to the Frobenius norm leading to a similar formula

$$r_{\mathbb{K}}^{\|\cdot\|}(A,B,C,D) = \min\left(\|D\|_2^{-1}, \inf_{\omega\in\mathbb{R}}\left[\mu_{\mathbb{K}}^{\|\cdot\|}(G(\mathbf{i}\omega))\right]^{-1}\right). \tag{8}$$

Remark 2.7. *As the $\mu_{\mathbb{R}}^{\|\cdot\|}$ function is upper semi-continuous both for operator norms and the Frobenius norm (see [Kar03, Lemma 1.7.1]), we have $\lim_{|\omega|\to\infty} G(iw) = D$ but*

$$\liminf_{|\omega|\to\infty} \left[\mu_{\mathbb{K}}^{\|\cdot\|}\left(G(\mathbf{i}\omega)\right)\right]^{-1} \geq \left[\mu_{\mathbb{K}}^{\|\cdot\|}(D)\right]^{-1} = \|D\|_2^{-1} \tag{9}$$

with a possible strict inequality (see [HP05, Remark 5.3.17 (i)] and [HP05, Example 5.3.18] for an example with $p = 1$). Therefore, when $D \neq 0$, we cannot eliminate the first term in (8)*. Either $r_{\mathbb{K}}^{\|\cdot\|}(A, B, C, D) = \|D\|_2^{-1}$ or the infimum in* (8) *is strictly less than $\|D\|_2^{-1}$ in which case it has to be attained at a finite ω; otherwise, we would obtain a contradiction as $|\omega| \to \infty$ by the inequality* (9)*. However, in the special case when $D = 0$, we can interpret $\left[\mu_{\mathbb{K}}^{\|\cdot\|}(D)\right]^{-1} = \|D\|_2^{-1} = \infty$ (see the paragraph after Definition 2.2) and dispense with the first term in* (8)*.*

In the complex case $\mathbb{K} = \mathbb{C}$, the spectral norm and Frobenius norms define the same stability radius $r_{\mathbb{C}}^{\|\cdot\|}$. In this case also we can eliminate the first term in (8), since (9) holds with equality, and the second term is simply the reciprocal of the H_∞ norm of the transfer matrix G on the boundary of the stability region. The standard method to compute it is the Boyd-Balakrishnan-Bruinsma-Steinbuch (BBBS) algorithm [BB90, BS90]. This algorithm is globally and quadratically convergent, but is not practical when n is large due to its computational complexity: it requires repeated computation of all eigenvalues of $2n \times 2n$ Hamiltonian matrices. The first constructive formula to compute $\mu_{\mathbb{R}}^{\|\cdot\|_2}$ and hence $r_{\mathbb{R}}^{\|\cdot\|_2}$, the real μ value and the real stability radius for the spectral norm, was given in [QBR+95]; this led to a practical level-set algorithm [SVDT96]. However, this is significantly more involved than the BBBS algorithm and hence is also impractical in the large-scale case. To our knowledge, no efficient algorithm to compute $r_{\mathbb{R}}^{\|\cdot\|_2}$ is known when n is large. As noted in the introduction, much less attention has been given to the Frobenius-norm case, though it is clearly of interest in applications. As far as we know, no constructive method has been given to approximate $\mu_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}$ or $r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}$, even if n is small.

2.3 The Spectral Value Set Abscissa

The spectral value set abscissa measures how far the spectral value set extends rightwards into the complex plane for a prescribed value of ε.

Definition 2.8. *For $\varepsilon \geq 0$, $\varepsilon\|D\|_2 < 1$, the* spectral value set abscissa *(w.r.t. the norm $\|\cdot\|$ and the field $\mathbb{K}$) is*

$$\alpha_\varepsilon^{\mathbb{K},\|\cdot\|}(A, B, C, D) = \max\{\mathrm{Re}\ \lambda : \lambda \in \sigma_\varepsilon^{\mathbb{K},\|\cdot\|}(A, B, C, D)\} \tag{10}$$

with $\alpha_0^{\mathbb{K},\|\cdot\|}(A, B, C, D) = \alpha(A)$, the spectral abscissa of A.

In the case $\mathbb{K} = \mathbb{R}$, $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|}(A, B, C, D)$ is called the *real spectral value set abscissa*.

Definition 2.9. *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. 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 $\mathcal{N}_\delta = \{s : |s - \lambda| < \delta\}$ of λ with $\delta > 0$.*

Remark 2.10. *Since $\sigma_\varepsilon^{\mathbb{K},\|\cdot\|}(A, B, C, D)$ is compact, its rightmost points, that is the maximizers of the optimization problem in* (10) *lie on its boundary. When $\mathbb{K} = \mathbb{C}$, there can only be a finite number of these [GGO13, Remark 2.14]. However, when $\mathbb{K} = \mathbb{R}$, there can be an infinite number of points on the boundary with the same real part, as shown by the following example.*

Example 2.11. *Let*

$$A = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \end{pmatrix}, \quad D = 0.$$

For $\varepsilon \in (0,1)$, $\sigma_\varepsilon^{\mathbb{R}}(A,B,C,D)$ consists of two line segments on the imaginary axis, which merge into one line segment $\left[\frac{\sqrt{2}}{2}\mathbf{i}, -\frac{\sqrt{2}}{2}\mathbf{i}\right]$ when $\varepsilon = 1$.

Remark 2.12. *Since A, B, C, D are real, $\sigma_\varepsilon^{\mathbb{K},\|\cdot\|}(A,B,C,D)$ is symmetric with respect to the real axis, so without loss of generality, when we refer to a rightmost λ in $\sigma_\varepsilon^{\mathbb{K},\|\cdot\|}$, we imply that $\lambda \in H_{+=}$, the closed upper half-plane, and when we say that the rightmost point λ is unique, we mean considering only points in $H_{+=}$. The same convention applies to the spectrum, so that a rightmost eigenvalue is understood to be in $H_{+=}$.*

There is a key relationship between the spectral value set abscissa and the stability radius that is a consequence of Theorem 2.5:

Corollary 2.13.

$$r_{\mathbb{K}}^{\|\cdot\|}(A,B,C,D) = \inf\left\{\varepsilon : \ \varepsilon\|D\|_2 < 1 \text{ or } \alpha_\varepsilon^{\mathbb{K},\|\cdot\|}(A,B,C,D) \geq 0\right\} \tag{11}$$

$$= \min\left(\|D\|_2^{-1}, \inf\left\{\varepsilon : \ \alpha_\varepsilon^{\mathbb{K},\|\cdot\|}(A,B,C,D) \geq 0\right\}\right). \tag{12}$$

Proof. That the right-hand sides of (11) and (12) are the same is immediate. Hence, it suffices to show that both of the infimum terms in (8) and (12) are attained and are equal when the upper bound $\|D\|_2^{-1}$ is not active. The infimum in (12) is attained because $\alpha_\varepsilon^{\mathbb{K},\|\cdot\|}(A,B,C,D)$ is a monotonically increasing continuous function of ε (see [Kar03, Chapter 2] for continuity properties of real spectral value sets) and the infimum in (8) is attained at a finite point by Remark 2.7. Finally, the infimal values are equal by Theorem 2.5. □

The algorithm developed in this paper for approximating the real stability radius $r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}$ when n is large depends on the fundamental characterization (12). This was also true of the recent algorithms developed in [GGO13, BV14, MO16] for the complex stability radius $r_{\mathbb{C}}^{\|\cdot\|}$ when n is large, for which the equivalence (12) is more straightforward and well known.[1]

3 An Ordinary Differential Equation

This section extends the method of Guglielmi and Lubich [GL13, Sec. 2.1] for approximating the real pseudospectral abscissa (the real spectral value set abscissa in the case $B = C = I$, $D = 0$) to the spectral value set abscissa $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}$ for general A, B, C, D, using the Frobenius norm. As we shall see, the extension is not straightforward as additional subtleties arise in the general case that are not present in the pseudospectral case.

We consider $\varepsilon > 0$ to be fixed with $\varepsilon\|D\|_2 < 1$ throughout this section and the next section. We start by considering the variational behavior of eigenvalues of the perturbed system matrix $M(\Delta)$ defined in (3). It is convenient to assume a smooth parametrization $t \mapsto \Delta(t)$ mapping $\mathbb{R}$ to $\mathbb{R}^{p\times m}$, with $\|\Delta(t)\|_{\mathrm{F}} = \varepsilon$ for all t. We will use $\dot{\Delta}$ to denote the derivative $(d/dt)\Delta(t)$.

We need the following lemma.

Lemma 3.1. *Given a smooth parametrization $\Delta(t)$ with $\|\Delta(t)\|_{\mathrm{F}} = 1$, we have*

$$\frac{d}{dt}\left(\Delta(t)\left(I - D\Delta(t)\right)^{-1}\right) = \left(I - \Delta(t)D\right)^{-1}\dot{\Delta}(t)\left(I - D\Delta(t)\right)^{-1}. \tag{13}$$

[1] For a different approach to approximating $r_{\mathbb{C}}^{\|\cdot\|}$ when n is large, namely the "implicit determinant" method, see [FSVD14].

Proof. For conciseness, we omit the dependence on t, differentiate and regroup terms as

$$\begin{aligned}\frac{d}{dt}\Big(\Delta\left(I-D\Delta\right)^{-1}\Big) &= \dot{\Delta}\left(I-D\Delta\right)^{-1}+\Delta\frac{d}{dt}\left(I-D\Delta\right)^{-1}\\ &= \dot{\Delta}\left(I-D\Delta\right)^{-1}+\Delta\left(I-D\Delta\right)^{-1}D\dot{\Delta}\left(I-D\Delta\right)^{-1}\\ &= \Big(I+\Delta\left(I-D\Delta\right)^{-1}D\Big)\dot{\Delta}\left(I-D\Delta\right)^{-1}. \end{aligned} \tag{14}$$

We then observe that

$$I+\Delta\left(I-D\Delta\right)^{-1}D = I+\Delta\Big(\sum_{k=0}^{\infty}(D\Delta)^k\Big)D = I+\sum_{k=1}^{\infty}(\Delta D)^k = \left(I-\Delta D\right)^{-1}. \tag{15}$$

Combining (14) and (15) yields the result. □

The following definition from [GO11, MO16] is useful.

Definition 3.2. *Let λ be a simple eigenvalue of a matrix M with associated right eigenvector x satisfying $Ax=\lambda x$ and left eigenvector y satisfying $y^*M=\lambda y^*$. We refer to $(\lambda,\ x,\ y)$ as an* RP-compatible *eigentriple of M if y^*x is real and positive and $\|x\|=\|y\|=1$, and as a rightmost eigentriple if λ is a rightmost eigenvalue of M in $H_{+=}$.*

Note that if (λ,x,y) is an RP-compatible eigentriple of M, so is $(\lambda,\ e^{\mathbf{i}\theta}x\ ,e^{\mathbf{i}\theta}y)$ for any $\theta\in[0,2\pi)$. Then we have:

Lemma 3.3. *Given a smooth parametrization $\Delta(t)$ with $\|\Delta(t)\|_{\mathrm{F}}=\varepsilon$, let $\lambda(t)$ be a continuously varying simple eigenvalue of*

$$M\left(\Delta(t)\right)=A+B\Delta(t)\left(I-D\Delta(t)\right)^{-1}C.$$

Then $\lambda(t)$ is differentiable with

$$\operatorname{Re}\dot{\lambda}(t)=\frac{1}{y(t)^*x(t)}\operatorname{Re}(u(t)^*\dot{\Delta}(t)v(t))$$

where $(\lambda(t),\ x(t),\ y(t))$ is an RP-compatible eigentriple of $M(\Delta(t))$ and

$$u(t)=\left(I-\Delta(t)D\right)^{-\mathsf{T}}B^{\mathsf{T}}y(t),\quad v(t)=\left(I-D\Delta(t)\right)^{-1}Cx(t).$$

Proof. Applying standard eigenvalue perturbation theory [HJ90, Theorem 6.3.12], together with Lemma 3.1, we find that λ is differentiable with

$$\dot{\lambda}=\frac{y^*\dot{M}\left(\Delta\right)x}{y^*x}\quad\text{and}\quad\dot{M}\left(\Delta\right)=B\left(I-\Delta D\right)^{-1}\dot{\Delta}\left(I-D\Delta\right)^{-1}C \tag{16}$$

where we omitted the dependence on t for conciseness. The result is then immediate. □

In what follows next it is convenient to define $E(t)=\Delta(t)/\varepsilon$, so that $\|E(t)\|_{\mathrm{F}}=1$ for all t. Consequently, we have

$$\operatorname{Re}\dot{\lambda}(t)=\frac{\varepsilon}{y(t)^*x(t)}\operatorname{Re}\Big(u(t)^*\dot{E}(t)v(t)\Big)=\frac{\varepsilon}{y(t)^*x(t)}\Big\langle\operatorname{Re}\left(u(t)v(t)^*\right),\dot{E}(t)\Big\rangle, \tag{17}$$

where for $R,S\in\mathbb{R}^{p\times m}$,

$$\langle R,S\rangle=\operatorname{Tr}R^{\mathsf{T}}S=\sum_{i,j}R_{ij}S_{ij},$$

the trace inner product on $\mathbb{R}^{p\times m}$ associated with the Frobenius norm. The condition that $\|E(t)\|_{\mathrm{F}}$ is constant is equivalent to

$$\frac{d}{dt}\|E(t)\|_{\mathrm{F}}^2 = 2\left\langle E(t), \dot{E}(t)\right\rangle = 0 \quad \forall t. \tag{18}$$

Our aim, given t, is to choose $\dot{E}(t)$ to maximize (17) subject to the constraint (18), leading to an optimization problem whose solution is given by the following lemma. The proof is a straightforward application of first-order optimality conditions; see also [GL13, Lemma 2.4].

Lemma 3.4. *Let $E \in \mathbb{R}^{p\times m}$ have unit Frobenius norm, and let $u \in \mathbb{C}^p, v \in \mathbb{C}^m$ be given complex vectors such that $\mathrm{Re}\,(uv^*) \neq 0$. A solution to the optimization problem*

$$\tilde{Z} = \arg\max_{Z\in\Omega} \ \mathrm{Re}\,(u^*Zv), \qquad \Omega = \{Z \in \mathbb{R}^{p\times m}, \|Z\|_{\mathrm{F}} = 1, \langle E, Z\rangle = 0\} \tag{19}$$

exists and it satisfies

$$\tau\tilde{Z} = \Big(\mathrm{Re}\,(uv^*) - \langle E, \mathrm{Re}\,(uv^*)\rangle\, E\Big), \tag{20}$$

where τ is the Frobenius norm of the matrix on the right-hand side in (20)*.*

This suggests consideration of the following ordinary differential equation (ODE) on the manifold of real $p \times m$ matrices of unit Frobenius norm:

$$\dot{E}(t) = \mathrm{Re}\,(u(t)v(t)^*) - \langle E(t), \mathrm{Re}\,(u(t)v(t)^*)\rangle\, E(t), \tag{21}$$

with $u(t)$ and $v(t)$ defined by

$$u(t) = (I - \varepsilon E(t)D)^{-\mathsf{T}} B^{\mathsf{T}} y(t), \quad v(t) = (I - \varepsilon DE(t))^{-1} Cx(t) \tag{22}$$

where $(\lambda(t),\ x(t),\ y(t))$ is a rightmost RP-compatible eigentriple for the matrix $M(\varepsilon E(t))$ (see Definition 3.2). Assume the initial condition $E(0) = E_0$, a given matrix with unit Frobenius norm, chosen so that $M(\varepsilon E_0)$ has a unique rightmost eigenvalue (considering only eigenvalues in $H_{+=}$), and that this eigenvalue, $\lambda(0)$, is simple.

3.1 Equilibrium Points of the ODE

We now focus on the properties of the ODE (21), in particular, characterizing equilibrium points.

Theorem 3.5. *Let $E(t)$ with unit Frobenius norm satisfy the differential equation* (21) *initialized as described above. There exists $t_{\max} \in (0, \infty]$ such that, for all $t \in [0, t_{\max})$*

(1) *The eigenvalue $\lambda(t)$ is the unique rightmost eigenvalue of $M(\varepsilon E(t))$ in $H_{+=}$ and this eigenvalue is simple, so the ODE is well defined.*

(2) $\|E(t)\|_{\mathrm{F}} = 1$.

(3) $\mathrm{Re}\,\dot{\lambda}(t) \geq 0$.

Furthermore, at a given value $t \in [0, t_{\max})$, the following three conditions are equivalent:

(i) $\mathrm{Re}\,\dot{\lambda}(t) = 0$.

(ii) *One of the following two mutually exclusive conditions holds:*

$$\mathrm{Re}\,(u(t)v(t)^*) = 0 \ \text{ or } \ E(t) = \frac{\mathrm{Re}\,(u(t)v(t)^*)}{\|\,\mathrm{Re}\,(u(t)v(t)^*)\,\|_{\mathrm{F}}}. \tag{23}$$

(iii) $\dot{E}(t) = 0$.

Finally, if the second alternative in (23) *holds at* $t_0 \in [0, t_{\max}]$*, then there does not exist any locally differentiable path* $F(t)$*, with* $\|F(t)\|_{\mathrm{F}} = 1$ *and* $F(0) = E(t_0)$*, for which the rightmost eigenvalue of* $M(\varepsilon F(t))$*, say* $\kappa(t)$*, has* $\operatorname{Re}\dot{\kappa}(0) > 0$*.*

Proof. (1) Because the rightmost eigenvalue is unique and simple for $t = 0$, the same property must hold for sufficiently small positive t, establishing the existence of $t_{\max} > 0$ such that the ODE is well defined for all $t \in [0, t_{\max}]$. (2) Taking the trace inner product of $E(t)$ with the ODE (21) we find that (18) holds for $\|E(t)\|_{\mathrm{F}} = 1$, so the norm is preserved by the ODE. (3) Substituting the ODE (21) into (17) we obtain from Cauchy-Schwartz that

$$\operatorname{Re}\dot{\lambda}(t) = \frac{\varepsilon}{y(t)^* x(t)} \left[\| \operatorname{Re}\left(u(t)v(t)^*\right) \|_{\mathrm{F}}^2 - \left\langle \operatorname{Re}\left(u(t)v(t)^*\right), E(t) \right\rangle^2 \right] \geq 0, \tag{24}$$

establishing (i). For (ii), equality holds in (24) at a given t if and only if one of the two alternatives in (23) holds. That (iii) is an equivalent condition follows directly from the ODE (21). The final statement follows from the optimality property of Lemma 3.4. □

Thus, equilibria of the ODE come in two flavors. When the second alternative in (23) holds, a first-order optimality condition for $\lambda(t)$ to be a rightmost point of $\sigma_\varepsilon^{\mathbb{K},\|\cdot\|_{\mathrm{F}}}$ holds, implying in particular that it is on the boundary of $\sigma_\varepsilon^{\mathbb{K},\|\cdot\|_{\mathrm{F}}}$. However, when the first alternative holds, we cannot make any such claim. In the special case of pseudospectra, that is with $B = C = I$ and $D = 0$, the outer product $u(t)v(t)^*$ reduces to $y(t)x(t)^*$, whose real part *cannot* be zero as was shown in [GL13, Sec. 2.1.6]: in fact, the proof given there shows that $\operatorname{Re}\left(y(t)x(t)^*\right)$ has rank one when $\lambda(t)$ is real and rank two when it is complex.

If $\operatorname{Re}\left(u(t)v(t)^*\right) = 0$, we say that $\lambda(t)$ is a *static point*. Note that this generalizes the notions of uncontrollability and unobservability, because if $\lambda(t)$ is unobservable, then $B^{\mathsf{T}} y(t) = 0$, implying $u(t) = 0$, while if it is uncontrollable, then $Cx(t) = 0$, implying $v(t) = 0$. Example 2.11 given earlier shows that it is possible that $\operatorname{Re}\left(u(t)v(t)^*\right) = 0$ even if $\lambda(t)$ is controllable and observable. In this example, for all $\varepsilon \in (0, 1)$, setting $E_0 = \pm 1$, we find that $u(0)$ and $v(0)$ are both nonzero but that $\operatorname{Re}\left(u(0)v(0)^*\right) = 0$. However, we do not know whether it is possible for the solution of the ODE to converge to a static point if it is not initialized there.

The next lemma explicitly states formulas for $\operatorname{Re}\left(u(t)v(t)^*\right)$ and bounds on its rank.

Lemma 3.6. *Fix* $t < t_{\max}$ *and let* $u \in \mathbb{C}^p, v \in \mathbb{C}^m$ *be defined by* (22) *for some vectors* $y(t) = y$ *and* $x(t) = x$*. If* $\lambda = \lambda(t) \in \mathbb{R}$*, then we can choose* y*,* x*,* u *and* v *to be real, with* $\operatorname{Re}(uv^*) = uv^{\mathsf{T}}$ *having rank 1. If* $\lambda \notin \mathbb{R}$*, set* $X = (\operatorname{Re} x, \operatorname{Im} x) \in \mathbb{R}^{n\times 2}$*,* $Y = (\operatorname{Re} y, \operatorname{Im} y) \in \mathbb{R}^{n\times 2}$*, so* $\operatorname{Re}(yx^*) = YX^{\mathsf{T}}$*. Then*

$$\operatorname{Re}(uv^*) = (I - \varepsilon ED)^{-\mathsf{T}} B^{\mathsf{T}} Y X^{\mathsf{T}} C^{\mathsf{T}} (I - \varepsilon DE)^{-\mathsf{T}}$$

with

$$\mathit{rank}\left(\operatorname{Re}\left(uv^*\right)\right) = \mathit{rank}\left(B^{\mathsf{T}} Y X^{\mathsf{T}} C^{\mathsf{T}}\right) \leq 2.$$

Furthermore, if $\min(p, m) = 1$*, then* $\mathit{rank}\left(\operatorname{Re}\left(uv^*\right)\right) \leq 1$*.*

Proof. The first statement follows from the definition (22), noting that E and D are real. The rank results follow from submultiplicativity. □

As already mentioned, the argument given in [GL13, Sec. 2.1.6] shows that when λ is not real, the matrix YX^{T} has rank two, so when $\min(p, m) \geq 2$, we can expect that UV^{T} will also have rank two for generic B and C.

If the ODE is initialized so that for all $t \geq 0$, the rightmost eigenvalue $\lambda(t)$ of $M(\varepsilon E(t))$ is unique (considering only eigenvalues in $H_{+=}$) and is simple, then we can take $t_{\max} = \infty$ in Theorem 3.5. If we further suppose that the number of equilibrium points of the ODE is finite, then since (21) is a gradient system (with Lyapunov function $\operatorname{Re}\lambda(t)$), we can apply Lasalle's Theorem [LR14] allowing us to state that $E(t)$ converges to an equilibrium point $\tilde{E}$ and hence $\lambda(t)$ converges to

some $\tilde{\lambda}$. Suppose that $\tilde{\lambda}$ is a unique rightmost eigenvalue of $M(\varepsilon\tilde{E})$ and is simple, with associated RP-compatible eigentriple $(\tilde{\lambda}, \tilde{x}, \tilde{y})$, and define

$$\tilde{u} = \left(I - \varepsilon\tilde{E}D\right)^{-\mathsf{T}} B^{\mathsf{T}}\tilde{y}, \quad \tilde{v} = \left(I - \varepsilon D\tilde{E}\right)^{-1} C\tilde{x},$$

by analogy with (22). Then, if $\operatorname{Re}(\tilde{u}\tilde{v}^*) \neq 0$, we have, by taking limits in Theorem 3.5, that $\tilde{E} = \operatorname{Re}(\tilde{u}\tilde{v}^*)/\|\operatorname{Re}(\tilde{u}\tilde{v}^*)\|_{\mathrm{F}}$, and that there does not exist any locally differentiable path $F(t)$, with $\|F(t)\|_{\mathrm{F}} = 1$ and $F(0) = \tilde{E}$, for which the rightmost eigenvalue of $M(\varepsilon F(t))$, say $\kappa(t)$, has $\operatorname{Re}\dot{\kappa}(0) > 0$.

To summarize this section, we have characterized equilibria of the ODE (21) as those which have a first-order local optimality property with respect to rightmost points of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A, B, C, D)$, with exceptions that are apparently nongeneric. A natural idea would be to attempt to approximate $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A, B, C, D)$ by integrating the ODE (21) numerically to determine its equilibria, guaranteeing monotonicity by step-size control. Such a method would generally find locally rightmost points of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A, B, C, D)$, although it could not be guaranteed to find globally rightmost points. However, a serious drawback of this approach is the fact that the solution $E(t)$ (and hence most likely its discretization) does not preserve the low rank-structure even if both the initial point E_0 and the limit of $E(t)$ as $t \to \infty$ both have rank two. Although it is possible to consider an ODE defined on the manifold of rank-two matrices, as done in [GL13] and [GM15] for the special case $B = C = I$, $D = 0$, we instead develop an efficient discrete iteration that is nonetheless based on the ODE (21).

4 An Iterative Method to Approximate the Frobenius-norm Real Spectral Value Set Abscissa

As in the previous section, assume ε is fixed with $\varepsilon\|D\|_2 < 1$. Following an idea briefly mentioned in [GM15], we consider the following implicit-explicit Euler discretization of (21) with a variable step-size h_k:

$$E_{k+1} = E_k + h_{k+1}\Big(\operatorname{Re}(u_{k+1}v_{k+1}^*) - \big\langle E_{k+1}, \operatorname{Re}(u_{k+1}v_{k+1}^*)\big\rangle E_k\Big), \tag{25}$$

where

$$u_{k+1} = \left(I - \varepsilon E_k D\right)^{-\mathsf{T}} B^{\mathsf{T}} y_k, \quad v_{k+1} = \left(I - \varepsilon D E_k\right)^{-1} C x_k$$

and (λ_k, x_k, y_k) is a rightmost RP-compatible eigentriple of $M(\varepsilon E_k)$. The method is clearly consistent and converges with order 1 with respect to h_k.

Lemma 4.1. *Let u_0, v_0 be given complex vectors with $\operatorname{Re}(u_0v_0^*) \neq 0$ and set $E_0 = \operatorname{Re}(u_0v_0^*)/\|\operatorname{Re}(u_0v_0^*)\|_{\mathrm{F}}$. Let $h_k = 1/\|\operatorname{Re}(u_kv_k^*)\|_{\mathrm{F}}$. Then the difference equation (25) has the solution*

$$E_{k+1} = \frac{\operatorname{Re}\left(u_{k+1}v_{k+1}^*\right)}{\|\operatorname{Re}\left(u_{k+1}v_{k+1}^*\right)\|_{\mathrm{F}}} \tag{26}$$

as long as the rightmost eigenvalue of $M(\varepsilon E_k)$ is simple and unique (considering only those in $H_{+=}$) and as long as $\operatorname{Re}\left(u_{k+1}v_{k+1}^*\right) \neq 0$, for all $k = 0, 1, \ldots$.

Proof. The result is easily verified by substituting (26) into (25). The assumptions ensure that the difference equation is well defined. □

Equivalently, let $E_k = U_kV_k^{\mathsf{T}}$ be the current perturbation, with $\|U_kV_k^{\mathsf{T}}\|_{\mathrm{F}} = 1$, and (λ_k, x_k, y_k) be a rightmost eigenvalue of $M(\varepsilon U_kV_k^{\mathsf{T}})$. Then by setting

$$X_k = (\operatorname{Re} x_k, \operatorname{Im} x_k),\ Y_k = (\operatorname{Re} y_k, \operatorname{Im} y_k)$$

we can write (26) in the form

$$E_{k+1} = U_{k+1}V_{k+1}^{\mathsf{T}} \text{ with } \|U_{k+1}V_{k+1}^{\mathsf{T}}\|_{\mathrm{F}} = 1 \tag{27}$$

where

$$\widehat{U}_{k+1} = \left(I - \varepsilon U_k V_k^{\mathsf{T}} D\right)^{-\mathsf{T}} B^{\mathsf{T}} Y_k, \tag{28}$$

$$\widehat{V}_{k+1} = \left(I - \varepsilon D U_k V_k^{\mathsf{T}}\right)^{-1} C X_k, \tag{29}$$

$$\beta_{k+1} = \|\widehat{U}_{k+1}\widehat{V}_{k+1}^{\mathsf{T}}\|_{\mathrm{F}}^{-1}, \tag{30}$$

$$U_{k+1} = \sqrt{\beta_{k+1}}\,\widehat{U}_{k+1}, \tag{31}$$

$$V_{k+1} = \sqrt{\beta_{k+1}}\,\widehat{V}_{k+1}. \tag{32}$$

Since $E_k = U_k V_k^{\mathsf{T}}$ has rank at most two, we can simplify these expressions using the Sherman-Morrison-Woodbury formula [GV83] as follows:

$$\left(I - \varepsilon U_k V_k^{\mathsf{T}} D\right)^{-1} = I + \varepsilon U_k \left(I - \varepsilon V_k^{\mathsf{T}} D U_k\right)^{-1} V_k^{\mathsf{T}} D, \tag{33}$$

$$\left(I - \varepsilon D U_k V_k^{\mathsf{T}}\right)^{-1} = I + \varepsilon D U_k \left(I - \varepsilon V_k^{\mathsf{T}} D U_k\right)^{-1} V_k^{\mathsf{T}}. \tag{34}$$

Note that $I - \varepsilon V_k^{\mathsf{T}} D U_k \in \mathbb{R}^{2\times 2}$ and is invertible since $\varepsilon\|D\|_2 < 1$ by assumption. The second formula (34) can be also used to simplify the definition of the perturbed system matrix in (3) as follows:

$$\begin{aligned} M(\Delta_k) = M(\varepsilon E_k) &= M(\varepsilon U_k V_k^{\mathsf{T}}) \\ &= A + \varepsilon B U_k V_k^{\mathsf{T}} (I - \varepsilon D U_k V_k^{\mathsf{T}})^{-1} C \\ &= A + \varepsilon B U_k V_k^{\mathsf{T}} (I + \varepsilon D U_k \left(I - \varepsilon V_k^{\mathsf{T}} D U_k\right)^{-1} V_k^{\mathsf{T}}) C \\ &= A + (\varepsilon B U_k) \left[I + \varepsilon \left(V_k^{\mathsf{T}} D U_k\right) \left(I - \varepsilon V_k^{\mathsf{T}} D U_k\right)^{-1}\right] \left(V_k^{\mathsf{T}} C\right) \end{aligned} \tag{35}$$

The product $U_k V_k^{\mathsf{T}}$ is never computed explicitly, but retained in factored form, so that the eigenvalues of $M(\varepsilon_k U_k V_k^{\mathsf{T}})$ with largest real part can be computed efficiently by an iterative method. The Frobenius norm of the product can be obtained using the following equivalence:

$$\|UV^{\mathsf{T}}\|_{\mathrm{F}} = \left[\mathrm{Tr}\left(VU^{\mathsf{T}}UV^{\mathsf{T}}\right)\right]^{\frac{1}{2}} = \left[\mathrm{Tr}\left(\left(U^{\mathsf{T}}U\right)\left(V^{\mathsf{T}}V\right)\right)\right]^{\frac{1}{2}}$$

which requires only inner products to compute the 2×2 matrices $U^{\mathsf{T}}U$ and $V^{\mathsf{T}}V$.

As with the spectral value set abscissa (SVSA) iteration for complex valued spectral values sets given in [GGO13], there is no guarantee that the full update step for the real Frobenius-norm bounded case will satisfy monotonicity, that is, $\mathrm{Re}(\lambda_{k+1}) > \mathrm{Re}(\lambda_k)$ may or may not hold, where λ_k is a rightmost eigenvalue of (35). However, the line search approach to make a monotonic variant [GGO13, Sec. 3.5] does extend to the real rank-2 iteration described above, although, as the derivation is quite lengthy [Mit14, Sec. 6.3.3], we only outline the essential components here. Let the pair U_k and V_k define the current perturbation, with $\|U_k V_k^{\mathsf{T}}\|_{\mathrm{F}} = 1$, and let the pair U_{k+1} and V_{k+1} be the updated perturbation described above, with $\|U_{k+1}V_{k+1}^{\mathsf{T}}\|_{\mathrm{F}} = 1$. Consider the evolution of a continuously varying simple rightmost eigenvalue $\lambda(t)$ defined on $t \in [0,1]$ of the perturbed system matrix. The interpolated perturbation is defined using $U_k V_k^{\mathsf{T}}$ and $U_{k+1}V_{k+1}^{\mathsf{T}}$ such that the interpolated perturbation also has unit Frobenius norm, that is, $\lambda(t)$ is an eigenvalue of

$$M(\Delta(t)) = A + B\Delta(t)(I - D\Delta(t))^{-1}C, \tag{36}$$

where

$$\Delta(t) = \frac{\varepsilon U(t)V(t)^{\mathsf{T}}}{\|U(t)V(t)^{\mathsf{T}}\|} \tag{37}$$

and $U(t) \coloneqq tU_{k+1} + (1-t)U_k$ and $V(t) \coloneqq tV_{k+1} + (1-t)V_k$. In [Mit14, Sec. 6.3.3] it is shown that as long $\mathrm{Re}\left(\lambda'(0)\right) = 0$ does not hold, then $\mathrm{Re}\left(\lambda'(0)\right) > 0$ can be ensured, though it may requiring flipping the signs of both U_k and V_k. This result allows a line search to be employed to find a

Algorithm SVSA-RF: (Spectral Value Set Abscissa: Real Frobenius-norm)

Purpose: to approximate $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$

Input: $\varepsilon \in (0, \|D\|_2^{-1})$, $U_0 \in \mathbb{R}^{n\times p}$ and $V_0 \in \mathbb{R}^{n\times m}$, such that $\|U_0V_0^{\mathsf{T}}\|_{\mathrm{F}} = 1$, along with eigentriple (λ_0, x_0, y_0), with λ_0 a rightmost eigenvalue of $M(\varepsilon U_0 V_0^{\mathsf{T}})$

Output: final iterates U_k, V_k with $\|U_kV_k^{\mathsf{T}}\|_{\mathrm{F}} = 1$ along with λ_k, a rightmost eigenvalue of $M(\varepsilon U_k V_k^{\mathsf{T}})$, certifying that $\mathrm{Re}\,(\lambda_k) \le \alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$

for $k = 0, 1, 2, \ldots$ **do**
  // Compute the new perturbation
  $X_k \coloneqq (\mathrm{Re}\; x_k, \mathrm{Im}\; x_k)$
  $Y_k \coloneqq (\mathrm{Re}\; y_k, \mathrm{Im}\; y_k)$
  $\widehat{U}_{k+1} \coloneqq \left(I + \varepsilon U_k \left(I - \varepsilon V_k^{\mathsf{T}} D U_k\right)^{-1} V_k^{\mathsf{T}} D\right)^{\mathsf{T}} B^{\mathsf{T}} Y_k$
  $\widehat{V}_{k+1} \coloneqq \left(I + \varepsilon D U_k \left(I - \varepsilon V_k^{\mathsf{T}} D U_k\right)^{-1} V_k^{\mathsf{T}}\right) C X_k$
  // Normalize the new perturbation
  $\beta_{k+1} \coloneqq \left[\mathrm{Tr}\left(\left(\widehat{U}_{k+1}^{\mathsf{T}}\widehat{U}_{k+1}\right)\left(\widehat{V}_{k+1}^{\mathsf{T}}\widehat{V}_{k+1}\right)\right)\right]^{-\frac{1}{2}}$
  $U_{k+1} \coloneqq \sqrt{\beta_{k+1}}\,\widehat{U}_{k+1}$
  $V_{k+1} \coloneqq \sqrt{\beta_{k+1}}\,\widehat{V}_{k+1}$
  // Attempt the full update step and, if necessary, do a line search
  $(\lambda_{k+1}, x_{k+1}, y_{k+1}) \coloneqq$ a rightmost eigentriple of $M(\varepsilon U_{k+1} V_{k+1}^{\mathsf{T}})$ using (35)
  if $\mathrm{Re}\,(\lambda_{k+1}) \le \mathrm{Re}\,(\lambda_k)$ **then**
    Find new λ_{k+1} via line search using (36) to ensure $\mathrm{Re}\,(\lambda_{k+1}) > \mathrm{Re}\,(\lambda_k)$
  end if
end for

NOTE: *The kth step of the iteration is well defined if $U_kV_k^{\mathsf{T}}$ is nonzero and the rightmost eigenvalue of* (35) *in $H_{+=}$ is unique and simple.*

$t \in (0,1)$ such that $\mathrm{Re}(\lambda_{k+1}) > \mathrm{Re}(\lambda_k)$ is guaranteed in an actual implementation of the iteration. We now have all the essential pieces necessary to describe approximating $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$; these are given in Algorithm SVSA-RF.

To summarize this section, we have proposed an efficient method, Algorithm SVSA-RF, to approximate $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$. Although it only guarantees finding a lower bound on $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$, the ODE (21) on which it is based has equilibrium points that typically satisfy a first-order optimality condition (see Section 3). The kth step of the iteration is well defined as long as the condition $U_kV_k^{\mathsf{T}} \neq 0$ holds and the rightmost eigenvalue of $M(\varepsilon U_k V_k^{\mathsf{T}})$ is unique and simple.

5 Approximating the Real Stability Radius by Hybrid Expansion-Contraction

Recall the relationship between the stability radius $r_{\mathbb{K}}^{\|\cdot\|}(A,B,C,D)$ and the spectral value set abscissa $\alpha_\varepsilon^{\mathbb{K},\|\cdot\|}(A,B,C,D)$ given in (12), which we write here for the real Frobenius-norm case:

$$r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}(A,B,C,D) = \min\left(\|D\|_2^{-1}, \inf\left\{\varepsilon :\ \alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D) \ge 0\right\}\right). \tag{38}$$

The interesting case is when the second term is the lesser of these two terms, and for the remainder of the paper we assume this is the case. It follows that $r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$ equals the infi-

mum in (38) and from Remark 2.7 that this infimum is attained. Hence $\alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D) = 0$ for $\varepsilon = r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$. For brevity, we henceforth use $\varepsilon_\star$ to denote the real stability radius $r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$.

Let

$$g(\varepsilon) = \alpha_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D). \tag{39}$$

We wish to find $\varepsilon_\star$, the root (zero) of the monotonically increasing continuous function g. However, we do not have a reliable way to evaluate g: all we have is Algorithm SVSA-RF which is guaranteed to return a lower bound on the true value. Consequently, if the value returned is negative we have no assurance that its sign is correct. On the other hand, if the value returned is positive, we *are* assured that the sign is correct. This observation underlies the hybrid expansion-contraction (HEC) algorithm recently introduced in [MO16] for approximating the complex stability radius, which we now extend to the real Frobenius-norm case.

5.1 Hybrid Expansion-Contraction

For any value of ε satisfying $\varepsilon_\star < \varepsilon < \|D\|_2^{-1}$, there exists a real pertubation matrix E with $\|E\|_{\mathrm{F}} = 1$ such that $M(\varepsilon E)$ has an eigenvalue in the right half-plane. We assume that E has rank at most two (see the discussion at the end of Section 2.1). See Section 7.2 for how an initial destabilizing perturbation $\varepsilon UV^{\mathsf{T}}$ can be found.

Let $U \in \mathbb{R}^{p\times 2}$ and $V \in \mathbb{R}^{m\times 2}$ be two matrices such that $\|UV^{\mathsf{T}}\|_{\mathrm{F}} = 1$. Consider the following matrix family *where U and V are fixed* and $0 < \varepsilon < \|D\|_2^{-1}$:

$$M_{UV}(\varepsilon) \coloneqq M(\varepsilon UV^{\mathsf{T}}) = A + B\varepsilon UV^{\mathsf{T}}(I - D\varepsilon UV^{\mathsf{T}})^{-1}C$$

and define the function

$$g_{UV}(\varepsilon) \coloneqq \alpha\left(M_{UV}(\varepsilon)\right), \tag{40}$$

the spectral abscissa of $M_{UV}(\varepsilon)$. Unlike g, this function is relatively easy to evaluate at a given ε, since all that is required is to compute the rightmost eigenvalue of the matrix $M_{UV}(\varepsilon)$, something that we assume can be done efficiently by an iterative method such as MATLAB's `eigs`, exploiting the equivalence (35). Now, as discussed above, suppose that $\varepsilon_{\mathrm{UB}}$ is known with $M_{UV}(\varepsilon_{\mathrm{UB}})$ having an eigenvalue in the right half-plane. There exists $\varepsilon_{\mathrm{c}} \in (0, \varepsilon_{\mathrm{UB}})$ such that $g_{UV}(\varepsilon_{\mathrm{c}}) = 0$ because g_{UV} is continuous, $g_{UV}(\varepsilon_{\mathrm{UB}}) > 0$ and $g_{UV}(0) < 0$ (as A is stable). The contraction phase of the hybrid expansion-contraction algorithm finds such an ε_{c} by a simple Newton-bisection method, using the derivative of $g_{UV}(\varepsilon)$ given in Section 5.2 below. Note that by definition of $\varepsilon_\star$, it must be the case that $\varepsilon_\star \le \varepsilon_{\mathrm{c}}$.

Once the contraction phase delivers ε_{c} with the rightmost eigenvalue of $M_{UV}(\varepsilon_{\mathrm{c}})$ on the imaginary axis, the expansion phase then "pushes" the rightmost eigenvalue of $M(\varepsilon UV^{\mathsf{T}})$ back into the right half-plane using Algorithm SVSA-RF, *with $\varepsilon = \varepsilon_{\mathrm{c}}$ fixed* and updating only the perturbation matrices U and V. The algorithm repeats this expansion-contraction process in a loop until SVSA-RF can no longer find a new perturbation that moves an eigenvalue off the imaginary axis into the right half-plane. Following [MO16], the method is formally defined in Algorithm HEC-RF. For an illustration of the main idea in the context of the complex stability radius, see [MO16, Fig. 4.1].

Convergence results for the original hybrid expansion-contraction algorithm developed for the complex stability radius were given in [MO16, Theorem 4.3]. The basic convergence result that, under suitable assumptions, the sequence $\{\varepsilon_k\}$ converges to some $\bar{\varepsilon} \ge \varepsilon_\star$ and the sequence $\{\mathrm{Re}\,(\lambda_k)\}$ converges to zero, can be extended to the real Frobenius-norm case without difficulty. However, the part that characterizes limit points of the sequence $\{\mathrm{Re}\,(\lambda_k)\}$ as stationary points or local maxima of the norm of the transfer function on the stability boundary does not immediately extend to the real Frobenius-norm case, because instead of $\|G(\mathbf{i}\omega)\|$, we would have to consider the potentially discontinuous function $\mu_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}(G(\mathbf{i}\omega))$.

Algorithm HEC-RF: (Hybrid Expansion-Contraction: Real Frobenius-norm)

Purpose: to approximate $r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$.

Input: $\varepsilon_0 \in (0, \|D\|_2^{-1})$ and matrices $U \in \mathbb{R}^{p\times 2}$ and $V \in \mathbb{R}^{m\times 2}$ with $\|UV^{\mathsf{T}}\|_{\mathrm{F}} = 1$ and $g_{UV}(\varepsilon_0) > 0$, along with λ_0, a rightmost eigenvalue of $M_{UV}(\varepsilon_0)$ in the right half-plane

Output: Final value of sequence $\{\varepsilon_k\}$ such that λ_k is a rightmost eigenvalue of $M(\varepsilon_k UV^{\mathsf{T}})$ sufficiently close to the imaginary axis but in the closed right half-plane, certifying that $\varepsilon_k \geq r_{\mathbb{R}}^{\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$

1: **for** $k = 0, 1, 2, \ldots$ **do**
2: **Contraction:** call a Newton-bisection zero-finding algorithm to compute $\varepsilon_{\mathrm{c}} \in (0, \varepsilon_k]$ so that $g_{UV}(\varepsilon_{\mathrm{c}}) = 0$, along with λ_{c}, a rightmost eigenvalue of $M_{UV}(\varepsilon_{\mathrm{c}})$ on the imaginary axis.
3: **Expansion:** call Algorithm SVSA-RF with input ε_{c}, U, V to compute U_{e}, V_{e} with $\|U_{\mathrm{e}}V_{\mathrm{e}}^{\mathsf{T}}\|_{\mathrm{F}} = 1$ and λ_{k+1}, a rightmost eigenvalue of $M(\varepsilon U_{\mathrm{e}}V_{\mathrm{e}}^{\mathsf{T}})$, satisfying $\mathrm{Re}\,(\lambda_{k+1}) \geq \mathrm{Re}\,(\lambda_{\mathrm{c}}) = 0$.
4: Set $\varepsilon_{k+1} \coloneqq \varepsilon_{\mathrm{c}}$, $U \coloneqq U_{\mathrm{e}}$, and $V \coloneqq V_{\mathrm{e}}$.
5: **end for**

NOTE: *In practice, we pass eigentriples computed by the contraction phase into the expansion phase and vice versa.*

5.2 The Derivatives of g_{UV} and g

The contraction phase of the algorithm needs the derivative of g_{UV} defined in (39) to implement the Newton-bisection method to find a root of g_{UV}. As we shall see, it is also of interest to relate this to the derivative of g defined in (40), although this is not actually used in the algorithm. The key tool for obtaining both is Lemma 3.1, which presented the derivative of $\Delta(t)\left(I - D\Delta(t)\right)^{-1}$ with respect to t. Here, the same matrix function depends on ε. We denote differentiation w.r.t. ε by $'$.

Theorem 5.1. *Let $O \subset (0, \|D\|_2^{-1})$ be open and suppose that, for all $\varepsilon \in O$, the rightmost eigenvalue $\lambda_{UV}(\varepsilon)$ of $M_{UV}(\varepsilon)$ in $H_{+=}$ is simple and unique. Then, for all $\varepsilon \in O$, g_{UV} is differentiable at ε with*

$$g'_{UV}(\varepsilon) = \frac{\mathrm{Re}\left((y_{UV}(\varepsilon)^* BU) \left[I + \varepsilon \left(V^{\mathsf{T}}DU\right)\left(I - \varepsilon V^{\mathsf{T}}DU\right)^{-1} \right]^2 \left(V^{\mathsf{T}}C x_{UV}(\varepsilon)\right) \right)}{y_{UV}(\varepsilon)^* x_{UV}(\varepsilon)} \tag{41}$$

where $(\lambda_{UV}(\varepsilon),\, x_{UV}(\varepsilon),\, y_{UV}(\varepsilon))$ is a rightmost RP-compatible eigentriple of $M_{UV}(\varepsilon)$.

Proof. Since $\lambda_{UV}(\varepsilon)$ is simple and unique, $x_{UV}(\varepsilon)$ and $y_{UV}(\varepsilon)$ are well defined (up to a unimodular scalar). Applying Lemma 3.1 with $\Delta(\varepsilon) \equiv \varepsilon UV^{\mathsf{T}}$, and using (33) – (34) and $\Xi \coloneqq I - \varepsilon V^{\mathsf{T}}DU$, we have

$$\begin{aligned} M'_{UV}(\varepsilon) &= B(I - \varepsilon UV^{\mathsf{T}}D)^{-1}UV^{\mathsf{T}}(I - \varepsilon DUV^{\mathsf{T}})^{-1}C \\ &= B(I + \varepsilon U\Xi^{-1}V^{\mathsf{T}}D)UV^{\mathsf{T}}(I + \varepsilon DU\Xi^{-1}V^{\mathsf{T}})C \\ &= BU(I + \varepsilon\Xi^{-1}V^{\mathsf{T}}DU)(I + \varepsilon V^{\mathsf{T}}DU\Xi^{-1})V^{\mathsf{T}}C \\ &= BU\left[I + \varepsilon \left(V^{\mathsf{T}}DU\right)\left(I - \varepsilon V^{\mathsf{T}}DU\right)^{-1} \right]^2 V^{\mathsf{T}}C \end{aligned} \tag{42}$$

noting that Ξ^{-1} and $V^{\mathsf{T}}DU$ commute. Using standard eigenvalue perturbation theory, as in the proof of Lemma 3.3, we have

$$g'_{UV}(\varepsilon) = \mathrm{Re}\,(\lambda'_{UV}(\varepsilon)) = \frac{\mathrm{Re}\,(y_{UV}(\varepsilon)^* M'_{UV}(\varepsilon) x_{UV}(\varepsilon))}{y_{UV}(\varepsilon)^* x_{UV}(\varepsilon)} \tag{43}$$

from which the result follows. □

Now we obtain the derivative of the function g defined in (39).

Theorem 5.2. *Let $O \subset (0, \|D\|_2^{-1})$ be open. Suppose that, for all $\varepsilon \in O$,*

1. *$\lambda(\varepsilon)$ is the unique rightmost point of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$ (considering only those in $H_{+=}$)*
2. *$E(\varepsilon)$, with $\|E(\varepsilon)\|_{\mathrm{F}} = 1$, is a smooth matrix function of ε such that $\lambda(\varepsilon)$ is the unique rightmost eigenvalue of $M(\varepsilon E(\varepsilon))$ (again considering only those in $H_{+=}$)*
3. *$\mathrm{Re}(u(\varepsilon)v(\varepsilon)^*) \neq 0$ where*

$$u(\varepsilon) = (I - \varepsilon E(\varepsilon)D)^{-\mathsf{T}} B^{\mathsf{T}} y(\varepsilon), \quad v(\varepsilon) = (I - \varepsilon DE(\varepsilon))^{-1} C^{\mathsf{T}} x(\varepsilon), \tag{44}$$

and $(\lambda(\varepsilon), x(\varepsilon), y(\varepsilon))$ is an RP-compatible eigentriple of $M(\varepsilon E(\varepsilon))$.

Then, for any $\varepsilon \in O$,

$$g'(\varepsilon) = \frac{\|\operatorname{Re}(u(\varepsilon)v(\varepsilon)^*)\|_{\mathrm{F}}}{y(\varepsilon)^* x(\varepsilon)}. \tag{45}$$

Proof. In this proof we again apply Lemma 3.1 but with $\Delta(\varepsilon) \equiv \varepsilon E(\varepsilon)$, obtaining

$$M'(\varepsilon) = B(I - \varepsilon E(\varepsilon)D)^{-1}(E(\varepsilon) + \varepsilon E'(\varepsilon))(I - \varepsilon DE(\varepsilon))^{-1}C.$$

Again using standard eigenvalue perturbation theory, we have

$$\begin{aligned} g'(\varepsilon) = \operatorname{Re}(\lambda'(\varepsilon)) &= \frac{\operatorname{Re}(y(\varepsilon)^* M'(\varepsilon) x(\varepsilon))}{y(\varepsilon)^* x(\varepsilon)} \\ &= \frac{\operatorname{Re}(u(\varepsilon)^* E(\varepsilon) v(\varepsilon)) + \varepsilon \operatorname{Re}(u(\varepsilon)^* E'(\varepsilon) v(\varepsilon))}{y(\varepsilon)^* x(\varepsilon)} \\ &= \frac{\langle E(\varepsilon), \operatorname{Re} u(\varepsilon)v(\varepsilon)^* \rangle + \varepsilon \langle E'(\varepsilon), \operatorname{Re} u(\varepsilon)v(\varepsilon)^* \rangle}{y(\varepsilon)^* x(\varepsilon)} \end{aligned} \tag{46}$$

The solution of the differential equation (21) for $t \geq 0$ with initial condition $E(\varepsilon)$ results in $\operatorname{Re}\dot{\lambda}(0) = 0$ as $\lambda(\varepsilon)$ is a rightmost point. Therefore, by Theorem 3.5, as the case $\mathrm{Re}(u(\varepsilon)v(\varepsilon)^*) = 0$ is ruled out by the assumptions, we have the identity

$$E(\varepsilon) = \frac{\operatorname{Re}(u(\varepsilon)v(\varepsilon)^*)}{\|\operatorname{Re}(u(\varepsilon)v(\varepsilon)^*)\|_{\mathrm{F}}}.$$

Plugging this identity into (46) and using the fact that $\langle E'(\varepsilon), E(\varepsilon)\rangle = \frac{1}{2}\frac{d\|E(\varepsilon)\|_F^2}{d\varepsilon} = 0$ we conclude that (45) holds. □

We now relate $g'_{UV}(\varepsilon)$ to $g'(\varepsilon)$.

Theorem 5.3. *Using the notation established above, suppose the assumptions of the two previous theorems apply for the same open interval O and that for some specific $\varepsilon \in O$,*

$$UV^{\mathsf{T}} = E(\varepsilon) = \frac{\operatorname{Re}(u(\varepsilon)v(\varepsilon)^*)}{\|\operatorname{Re}(u(\varepsilon)v(\varepsilon)^*)\|_{\mathrm{F}}}, \tag{47}$$

so that the matrices $M_{UV}(\varepsilon)$ and $M(\varepsilon E(\varepsilon))$ are the same and the eigentriples $(\lambda_{UV}(\varepsilon), x_{UV}(\varepsilon), y_{UV}(\varepsilon))$ and $(\lambda(\varepsilon), x(\varepsilon), y(\varepsilon))$ coincide, with $g_{UV}(\varepsilon) = g(\varepsilon)$. Then

$$g'_{UV}(\varepsilon) = g'(\varepsilon).$$

Proof. Using (43), (42) and (47) we have

$$g'_{UV}(\varepsilon) = \frac{\operatorname{Re}\left(y(\varepsilon)^*\left(B(I-\varepsilon E(\varepsilon)^{\mathsf{T}}D)^{-1}E(\varepsilon)(I-\varepsilon DE(\varepsilon))^{-1}C\right)x(\varepsilon)\right)}{y(\varepsilon)^*x(\varepsilon)}.$$

So, using (44) and (45), we obtain

$$g'_{UV}(\varepsilon) = \frac{\operatorname{Re}\left(u(\varepsilon)^*E(\varepsilon)v(\varepsilon)\right)}{y(\varepsilon)^*x(\varepsilon)} = \frac{\|\operatorname{Re}\left(u(\varepsilon)v(\varepsilon)^*\right)\|_{\mathrm{F}}}{y(\varepsilon)^*x(\varepsilon)} = g'(\varepsilon).$$

□

This result is important, because at the start of the contraction phase of Algorithm HEC-RF, assuming that the expansion phase has returned a locally rightmost point of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$, we have that (47) holds. Hence, the first Newton step of the contraction phase, namely a Newton step for finding a zero of g_{UV}, is *equivalent* to a Newton step for finding a zero of g, which is the ultimate goal. For this reason, under a suitable regularity condition, Algorithm HEC-RF is actually quadratically convergent. We omit the details here, but a convergence rate analysis similar to that given in [MO16, Theorem 4.4] for the complex stability radius hybrid expansion-contraction algorithm holds for Algorithm HEC-RF too.

6 Discrete-time systems

We now briefly summarize the changes to our results and algorithms that are needed to handle, instead of (1)–(2), the discrete-time system

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

where $k = 1, 2, \ldots$. The definitions of the transfer matrix function, spectral value sets and real-μ functions in Section 2.1 remain unchanged. In Section 2.2, the stability region is the open unit disk $\mathbb{D}_-$ instead of the open left half-plane $\mathbb{C}_-$, and the definition of the stability radius must be adjusted accordingly. In Section 2.3, instead of the spectral abscissa α and spectral value set abscissa α_ε, we require the spectral radius ρ and spectral value set radius $\rho_\varepsilon^{\mathbb{K},\|\cdot\|}$, which are defined by maximization of $|\lambda|$ instead of $\operatorname{Re}(\lambda)$ over the spectral value set.[2] Now, instead of "rightmost" points, we search for "outermost" points.

In Section 3, it is convenient to extend Definition 3.2 as follows: (λ, x, y) is an RP(z)-compatible eigentriple of M if λ is a simple eigenvalue of M, x and y are corresponding normalized right and left eigenvectors and y^*x is a real positive multiple of z. Then, instead of (3.3), we have, taking $(\lambda(t), x(t), y(t))$ to be an RP($\overline{\lambda}(t)$)-compatible eigentriple,

$$\frac{d}{dt}|\lambda(t)| = \frac{\operatorname{Re}\overline{\lambda}(t)\dot{\lambda}(t)}{|\lambda(t)|} = \frac{1}{|y(t)^*x(t)|}\operatorname{Re}(u(t)^*\dot{\Delta}(t)v(t)).$$

The ODE (21) then remains unchanged, except that the eigentriple $(\lambda(t), x(t), y(t))$ is an *outermost* RP($\overline{\lambda}(t)$)-compatible eigentriple of $M(\varepsilon E(t))$ instead of a rightmost RP-compatible eigentriple. Theorem 3.5 also holds as before, with the same change. In Section 4, we replace Algorithm SVSA-RF by Algorithm SVSR-RF (Spectral Value Set Radius: Real Frobenius-norm), whose purpose is to approximate $\rho_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$. The only change that is needed is to replace rightmost RP-compatible eigentriple (λ_k, x_k, y_k) by outermost RP($\overline{\lambda}_k$)-compatible eigentriple (λ_k, x_k, y_k). To ensure that

[2] Recall again the completely different usage of "radius" in these names, the stability radius referring to the data space and the spectral radius to the complex plane.

$|\lambda_{k+1}| \geq |\lambda_k|$, a line search can again be used, as explained in [Mit14, Sec. 6.3.3]. The algorithm produces λ_k certifying that $|\lambda_k| \leq \rho_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$.

In Section 5, since the stability region is now the open unit disk $\mathbb{D}_-$, instead of (39) we have $g(\varepsilon) = \rho_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D) - 1$, and instead of (40) we have $g_{UV}(\varepsilon) \coloneqq \rho(M_{UV}(\varepsilon)) - 1$. The derivatives of g_{UV} in (43) and g in (45) remain unchanged except for the RP-compatibility change and the replacement of y^*x by $|y^*x|$ in both denominators. Besides the RP-compatibility definition change, Algorithm HEC-RF is changed as follows: rightmost, right half-plane and imaginary axis are changed to outermost, $\mathbb{C}\backslash\mathbb{D}_-$ and unit circle respectively.

7 Implementation and Experiments

We implemented Algorithm HEC-RF by extending the open-source MATLAB code `getStabRadBound` [MO16, Section 7], which is the implementation of the original HEC algorithm for approximating the complex stability radius. Our new code supports approximating both the complex and the real Frobenius-norm bound stability radius, for both continuous-time and discrete-time systems, although for brevity, we continue to refer primarily only to the continuous-time case. We similarly adapted the related fast upper bound algorithm [MO16, Section 4.4], which aims to quickly find a destabilizing perturbation necessary for initializing Algorithm HEC-RF. This "greedy" strategy aims to take steps as large as possible towards a destabilizing perturbation by alternating between increasing ε and taking a single SVSA-RF update step of the perturbation matrices U and V. In the course of this work, we also significantly improved the convergence criteria of `getStabRadBound`. As these issues are crucial for implementing a practicable and reliable version of the HEC algorithm, but the discussion does not specifically pertain to the real stability radius case, we defer the details to Appendix A. Lastly, we also made several improvements to help accelerate the algorithm.

7.1 Acceleration features

First, we extended `getStabRadBound`'s feature for implicitly extrapolating the sequences of rank-1 perturbation matrices, produced in the expansion phases when approximating the complex stability radius, to also handle the corresponding rank-2 sequences that may occur when approximating the real stability radius. To be efficient, the procedure takes a history of U_k and V_k matrices from Algorithm SVSA-RF and then forms four vector sequences corresponding to two selected rows and two selected columns of the evolving $U_kV_k^T$ matrix. Vector extrapolation is then applied individually to these four vector sequences to obtain two rows r_1 and r_2 and two columns c_1 and c_2 from the extrapolation of the sequence $\{U_kV_k^T\}$, without ever explicitly forming these matrices. Using the resulting four vector extrapolations, a new pair $U_\star$ and $V_\star$ are computed such that $\|U_\star V_\star^T\|_F = 1$. For more details, see [Mit14, Section 6.3.5].

Second, for SVSA-RF, we note that even if a full update step satisfies monotonicity, and thus does not require a line search, it may happen that the line search could still sometimes produce a better update anyway, particularly if `getStabRadBound`'s interpolating quadratic or cubic-based line search option is enabled; see [Mit14, Section 4.3]. As such, even when a full SVSA-RF step satisfies monotonicity, our implementation will check whether an interpolated step might be even better, specifically, by considering the maximum of an interpolating quadratic line search model. If this quadratic interpolation-derived step is predicted to be at least 1.5 times better than the already computed full update step, the rightmost eigenvalue of the corresponding interpolated perturbation is computed and if it satisfies monotonicity, with respect to the full step, then it is accepted in lieu of the full step.

Finally, we modified the entire code to only compute left eigenvectors on demand, instead of always computing eigentriples. If the resulting rightmost eigenvalue of $M(\Delta)$ for some perturbation Δ encountered by the algorithm does not satisfy monotonicity, then computing the corresponding left eigenvector is unnecessary; left eigenvectors need only be computed at points accepted by the algorithm, since it is only at such points that derivatives of eigenvalues are used. This optimization

Small Dense Problems: Continuous-time								
	Iters		# Eig		Time (secs)		RSR Approximation	
Problem	v1	v2	v1	v2	v1	v2	$\min\{\varepsilon_1, \varepsilon_2\}$	$(\varepsilon_1 - \varepsilon_2)/\varepsilon_1$
CBM	3	3	122	79	8.646	5.577	$4.46769464697 \times 10^{0}$	-8.5×10^{-12}
CSE2	7	7	223	117	0.643	0.386	$4.91783643704 \times 10^{1}$	-
CM1	2	3	91	81	0.198	0.203	$1.22474487041 \times 10^{0}$	-
CM3	3	4	126	108	1.063	0.952	$1.22290355805 \times 10^{0}$	-
CM4	3	4	222	181	8.181	6.680	$6.30978638860 \times 10^{-1}$	-
HE6	11	9	20852	9972	13.828	8.305	$2.02865555290 \times 10^{-3}$	$+1.5 \times 10^{-10}$
HE7	4	6	492	248	0.406	0.322	$2.88575420548 \times 10^{-3}$	-3.2×10^{-12}
ROC1	2	3	93	78	0.127	0.150	$9.11416570667 \times 10^{-1}$	$+2.3 \times 10^{-12}$
ROC2	3	3	98	83	0.136	0.161	$7.49812117968 \times 10^{0}$	$+1.0 \times 10^{-10}$
ROC3	4	4	204	117	0.211	0.209	$7.68846259016 \times 10^{-5}$	-3.5×10^{-11}
ROC4	1	1	40	40	0.084	0.134	$3.47486815789 \times 10^{-3}$	-
ROC5	5.5	11	263	426	0.226	0.390	$1.02041223979 \times 10^{2}$	-8.0×10^{-9}
ROC6	4	4	149	80	0.174	0.182	$3.88148973329 \times 10^{-2}$	-
ROC7	3	3	142	107	0.165	0.163	$8.96564880558 \times 10^{-1}$	-
ROC8	3	4	160	114	0.183	0.194	$2.08497314619 \times 10^{-1}$	$+4.7 \times 10^{-7}$
ROC9	5	8	235	173	0.223	0.297	$4.20965764059 \times 10^{-1}$	-
ROC10	1	1	26	26	0.079	0.096	$1.01878607021 \times 10^{1}$	-

Table 1: The "Iters" columns show the number of HEC-RF iterations until termination for the "v1" and "v2" configurations of `getStabRadBound`; note that these can be fractional since the method may quit after either a contraction or expansion phase. The "# Eig" columns show the total number of eigensolves (the sum of the number of right and left eigenvectors computed) incurred while the "Time (secs)" columns show the elapsed wall-clock time in seconds per problem for both code variants. The left column under the "RSR Approximation" heading shows the better (smaller) of the two real stability radius approximations ε_1 and ε_2, respectively computed by "v1" and "v2" versions of the code. The rightmost column show the relative difference between these two approximations, with positive values indicating that the "v2" code produced a better approximation. Relative differences below the 10^{-12} optimality tolerances used for the code are not shown.

essentially halves the cost of all incurred line searches in both Algorithm SVSA-RF and step 3 of the fast upper bound procedure, while it also halves the cost of computing extrapolation and interpolation steps that end up being rejected.

7.2 Numerical evaluation of Algorithm HEC-RF

We tested our new version of `getStabRadBound` on the 34 small-scale and 14 large-scale linear dynamical systems used in the numerical experiments of [GGO13] and [MO16], noting that the system matrices (A, B, C, D) for these problems are all real-valued. We ran the code in two different configurations for each problem: once in its "pure" HEC form, which we call "v1" and which should converge quadratically, and a second time using an "accelerated" configuration, which we call "v2". This latter configuration enabled both of the interpolation and extrapolation features described above as well as the code's early contraction and expansion termination conditions, both of which aim to encourage the code to accept inexact and cheaply-acquired solutions to the subproblems when more accuracy is neither needed nor useful. The early expansion termination feature is described in [MO16, Section 4.3] and often greatly lessens the overall computational cost, despite the fact that it reduces the algorithm's theoretical convergence rate from quadratic to superlinear. We used 0.01 as the relative tolerances for governing these early contraction/expansion features. The expansion and contraction optimality tolerances were both set to 10^{-12}; these two tolerances act together to determine an overall tolerance for the HEC iteration. The "v2" configuration was set to attempt

Small Dense Problems: Discrete-time								
	Iters		# Eig		Time (secs)		RSR Approximation	
Problem	v1	v2	v1	v2	v1	v2	$\min\{\varepsilon_1, \varepsilon_2\}$	$(\varepsilon_1 - \varepsilon_2)/\varepsilon_1$
AC5	3	4	244	197	0.250	0.234	$2.01380141605 \times 10^{-2}$	$+6.3 \times 10^{-12}$
AC12	2	2	35	51	0.098	0.131	$9.33096040564 \times 10^{-2}$	-
AC15	5	6	143	94	0.178	0.179	$4.22159665084 \times 10^{-2}$	-
AC16	4	5	119	78	0.157	0.181	$7.75365184115 \times 10^{-2}$	-
AC17	5	5	222	150	0.201	0.219	$3.35508111043 \times 10^{-6}$	$+4.5 \times 10^{-10}$
REA1	2	2	77	66	0.116	0.167	$1.37498793652 \times 10^{-3}$	-
AC1	4	5	325	230	0.267	0.253	$7.99003318082 \times 10^{0}$	-
AC2	3	4	61	57	0.111	0.171	$3.36705685350 \times 10^{0}$	-
AC3	4	4	427	297	0.305	0.329	$7.43718998002 \times 10^{-2}$	-
AC6	5	9.5	253	366	0.215	0.356	$2.32030683553 \times 10^{-8}$	$+2.6 \times 10^{-1}$
AC11	5	3	198	113	0.213	0.167	$5.21908412146 \times 10^{-8}$	-2.5×10^{-8}
ROC3	4	5	204	187	0.209	0.264	$5.30806020326 \times 10^{-2}$	-
ROC5	6	5	280	176	0.246	0.306	$2.85628817204 \times 10^{-4}$	-1.7×10^{-10}
ROC6	5	7	324	111	0.269	0.239	$5.81391974240 \times 10^{-2}$	$+1.8 \times 10^{-1}$
ROC7	4	4	68	55	0.115	0.161	$9.01354011348 \times 10^{-1}$	-
ROC8	3	6	134	119	0.163	0.217	$2.08192687301 \times 10^{-5}$	$+1.6 \times 10^{-10}$
ROC9	3	4	137	101	0.160	0.177	$4.07812890254 \times 10^{-2}$	-

Table 2: See caption of Table 1 for the description of the columns.

Large Sparse Problems: Continuous-time (top), Discrete-time (bottom)								
	Iters		# Eig		Time (secs)		RSR Approximation	
Problem	v1	v2	v1	v2	v1	v2	$\min\{\varepsilon_1, \varepsilon_2\}$	$(\varepsilon_1 - \varepsilon_2)/\varepsilon_1$
NN18	1	2	27	37	1.833	2.430	$9.77424680376 \times 10^{-1}$	-
dwave	2	4	72	59	32.484	28.794	$2.63019715625 \times 10^{-5}$	-
markov	2	3	61	56	12.581	10.479	$1.61146532880 \times 10^{-4}$	-
pde	4	5	128	79	4.670	3.011	$2.71186478815 \times 10^{-3}$	-
rdbrusselator	2	3	50	45	4.517	4.402	$5.47132014748 \times 10^{-4}$	-
skewlap3d	2	2	103	78	115.004	90.605	$4.59992022215 \times 10^{-3}$	-
sparserandom	2	2	90	74	3.056	2.655	$7.04698184529 \times 10^{-6}$	-3.5×10^{-10}
dwave	2	4	34	33	14.776	14.397	$2.56235064981 \times 10^{-5}$	-
markov	3	3	73	64	15.740	14.056	$2.43146945130 \times 10^{-4}$	-
pde	2	2	46	35	1.713	1.450	$2.77295935785 \times 10^{-4}$	-
rdbrusselator	3	5	76	60	5.926	5.041	$2.56948942080 \times 10^{-4}$	-
skewlap3d	2	3	50	52	53.297	50.526	$3.40623440406 \times 10^{-5}$	-
sparserandom	2	2	21	19	1.035	0.880	$2.53298721605 \times 10^{-7}$	-3.5×10^{-8}
tolosa	3	3	98	53	10.097	6.251	$2.14966549184 \times 10^{-7}$	-6.5×10^{-12}

Table 3: See caption of Table 1 for the description of the columns.

extrapolation every fifth iteration, from the previous five iterates. For all other parameters, we used `getStabRadBound`'s default user options. All experiments were performed using MATLAB R2015a running on a Macbook Pro with an Intel i7-5557U dual-core CPU and 16GB of RAM, running Mac OS X v10.11.5.

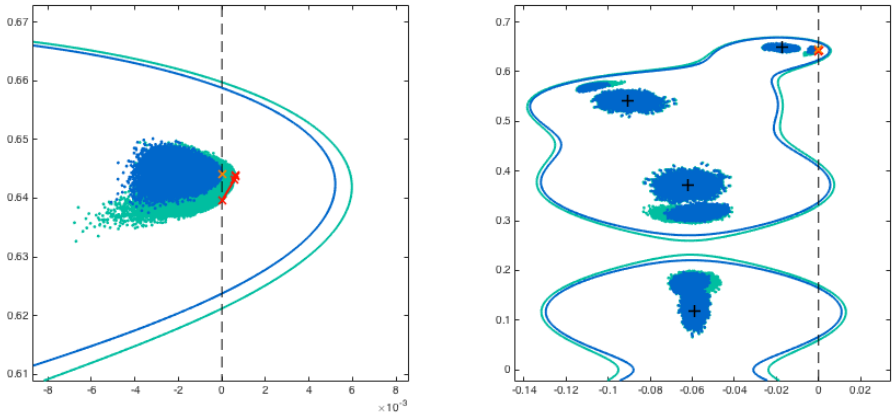

Figure 1: Test problem `ROC3`. Selected iterates of Algorithm HEC-RF, namely the first and last expansion phases (ε_1 and ε_4), are depicted as two sequences of x's connected by line segments, respectively in red and orange, in a close-up view (left) and in a wide view (right). The corresponding sets $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$ were realized by plotting points of $\sigma(M(\varepsilon U_r V_r^T))$, in green for $\varepsilon = \varepsilon_1$ and in blue for $\varepsilon = \varepsilon_4$, using many rank-1 and rank-2 "sample" matrices $U_r V_r^T$ with $\|U_r V_r^T\|_F = 1$. Specifically, for each value of ε, we used 100,000 randomly generated matrices $U_r V_r^T$ (using `randn()`), another 100,000 generated via quasi-random Sobol sequences, and 100,000 randomly perturbed versions of the expansion phases' sequences of matrices. The 200,000 random and quasi-random samples were unable to capture the region near the locally rightmost point found by Algorithm HEC-RF; the points from these samples only appear in the wider view on the right, in small regions about the eigenvalues of A (represented by the black +'s). The sample points shown in the close-up view on the left are all from the randomly perturbed versions of the expansion phases' matrix iterates, demonstrating Algorithm HEC-RF ability to efficiently find extremal rightmost values in real-valued spectral value sets. The solid curves depict the boundaries of the corresponding sets $\sigma_\varepsilon^{\mathbb{C},\|\cdot\|_2}(A,B,C,D)$ and were computed by MATLAB's `contour`. As can be readily seen, the iterates of Algorithm HEC-RF converged to a locally rightmost of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$, $\varepsilon = \varepsilon_4$, close to the imaginary axis (represented by the dashed vertical line) and in the interior of $\sigma_\varepsilon^{\mathbb{C},\|\cdot\|_2}(A,B,C,D)$.

In order to assess whether Algorithm SVSA-RF converges to locally rightmost points, we have relied upon plotting approximations of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$ in the complex plane using various random sampling and perturbation techniques (in contrast to the case of the complex stability radius, where the boundaries of $\sigma_\varepsilon^{\mathbb{C},\|\cdot\|_2}(A,B,C,D)$ can be plotted easily). For each of the 34 small-scale problems, we plotted the iterates of the expansion phases along with our $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$ approximations for the corresponding values of ε and examined them all by hand, observing that Algorithm SVSA-RF indeed does converge to locally rightmost points as intended, at least to the precision that can be assessed from such plots. See Figures 1 and 2 for two such plots.

As Algorithm HEC-RF is, to the best of our knowledge, the only available method to approximate the real Frobenius-norm bounded stability radius, we simply report the resulting upper bounds produced by our method to 12 digits for each test problem, along with statistics on the computational cost, in Tables 1-3. We observe that both variants of the code tend to produce approximations with high agreement, showing that there seems to be little to no numerical penalty for enabling the

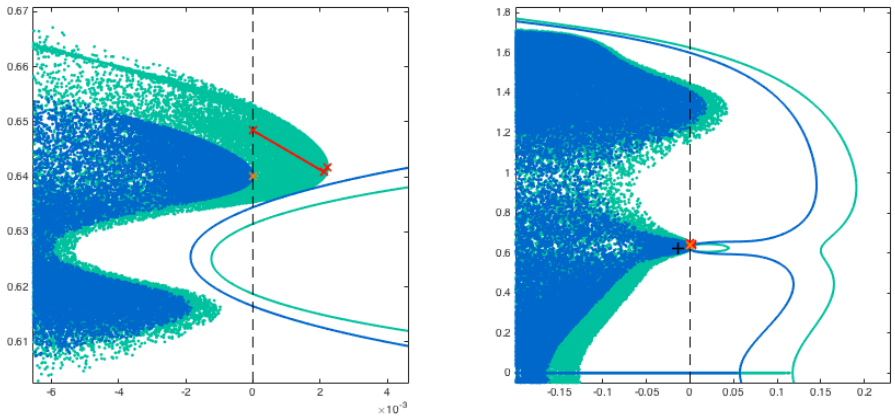

Figure 2: Test problem `ROC9`. Selected iterates of Algorithm HEC-RF, namely the first and last expansion phases (ε_1 and ε_5), are depicted as two sequences of x's connected by line segments, respectively in red and orange, in a close-up view (left) and in a wide view (right). The corresponding sets of point clouds for $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$ and set boundaries of $\sigma_\varepsilon^{\mathbb{C},\|\cdot\|_2}(A,B,C,D)$, in green for $\varepsilon=\varepsilon_1$ and in blue for $\varepsilon=\varepsilon_5$, were plotted in a similar manner as described in Figure 1. The black $+$ represents an eigenvalue of A. As can be seen, the iterates of Algorithm HEC-RF converged to a locally rightmost point of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$, with $\varepsilon=\varepsilon_5$, close to the imaginary axis (represented by the dashed vertical line), though in this case, it is clear that this is not a globally rightmost point. Interestingly, the corresponding sets $\sigma_\varepsilon^{\mathbb{C},\|\cdot\|_2}(A,B,C,D)$ have no locally rightmost points near the sequence of locally rightmost points of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$ found by Algorithm HEC-RF, highlighting the striking difference between real-valued and complex-valued spectral value sets. In fact, for $\varepsilon=\varepsilon_1$, it is seen that $\sigma_\varepsilon^{\mathbb{C},\|\cdot\|_2}(A,B,C,D)$ actually has a hole below and to the right of the locally rightmost point of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$ found by Algorithm HEC-RF; the hole is depicted by the small green ellipse in the right plot, a portion of which can be seen in the left plot.

acceleration features. In fact, on two examples (`AC6` and `ROC6`, both discrete-time systems), we see that the accelerated version of the code actually produced substantially better approximations, with improvement to their respective second-most significant digits. Furthermore, the accelerated "v2" configuration does appear to be effective in reducing the number of eigensolves incurred on most problems, though there are two notable exceptions to this: `ROC5` (continuous-time) and `ROC6` (discrete-time). It is worth noting that many of the small-scale test problems have such tiny dimensions that a reduction in eigensolves doesn't always correspond with a speedup in terms of wall-clock time (and can sometimes seemingly paradoxically have increased running times due to the inherent variability in collecting timing data). However, on only moderate-sized problems, such as `CBM`, `CSE2`, and `CM4` (all continuous-time), we start to see the correspondence between number of eigensolves and running time approaching a one-to-one relationship. This correspondence is readily apparent in the large and sparse examples in Table 3.

Though it is difficult to tease out the effects of the different acceleration options, since they interact with each other, we were able to determine that the early expansion termination feature was usually the dominant factor in reducing the number of eigensolves. However, extrapolation was crucial for the large gains observed on `HE6` (continuous-time) and `ROC6` (discrete-time). By comparison, in [MO16], when using HEC to approximate the complex stability radius, extrapolation tended to be much more frequently beneficial while usually providing greater gains as well. Part of this disparity may be because of the greatly increased number of eigensolves we observed when running `getStabRoundBound` to approximate the complex stability radius as opposed to the real

stability radius; on the 34 small-scale problems, the complex stability radius variant incurred 1226 more eigensolves per problem on average, with the median being 233 more. In our real stability radius experiments, `HE6` notwithstanding, Algorithm SVSA-RF simply did not seem to incur slow convergence as often nor to the same severity as its rank-1 counterpart for complex-valued spectral value sets. We note that our ODE-based approach for updating real rank-2 Frobenius-norm bounded perturbations underlying Algorithm SVSA-RF also provides a new expansion iteration for complex spectral value sets; in Appendix B, we evaluate the performance of this new variant when approximating the complex stability radius.

7.3 New challenges for the real stability radius case

Over the test set, only a handful of examples triggered our new rank-2 extrapolation routine: continuous-time problems `ROC1` and `ROC3` and discrete-time problems `AC5`, `AC1`, `AC3`, `ROC3`, and `ROC5`. Of these seven, the code only produced a successful extrapolation for `ROC1`, which is seemingly not a promising result for the rank-2 extrapolation procedure. However, perhaps none of these problems ended up being particularly good candidates for evaluating the rank-2 extrapolation procedure; their respective total number of perturbation updates to $U_k V_k^{\mathsf{T}}$, with all acceleration features disabled, was at most 160 updates (`AC3`), with the average only being 70.7. Simply put, these problems provided little opportunity for any extrapolation, yet alone need.

As an alternative to recovering the aforementioned normalized $U_\star$ and $V_\star$ matrices by the direct procedure described in Section 7.1, we also considered specifying it as a constrained optimization problem. For notational convenience, we assume that r_1, r_2 and c_1, c_2 are respectively the first and second rows and columns of the implicitly extrapolated matrix of $\{U_k V_k^{\mathsf{T}}\}$. To recover $U_\star$ and $V_\star$ from the extrapolated rows r_1, r_2 and columns c_1, c_2, we instead solve the following constrained optimization problem

$$\min_{\|UV^{\mathsf{T}}\|_F=1} \left\| \begin{bmatrix} c_1 & c_2 \end{bmatrix} - UV(1:2,:)^{\mathsf{T}} \\ \begin{bmatrix} r_1(3:\texttt{end}) \\ r_2(3:\texttt{end}) \end{bmatrix} - U(1:2,:)V(3:\texttt{end},:)^{\mathsf{T}} \right\|_2 . \tag{48}$$

We derived the gradients of the above objective function and equality constraint and used them with both MATLAB's `fmincon` and GRANSO: GRadient-based Algorithm for Non-Smooth Optimization [CMO17], to solve (48). We observed that the vectors obtained by our direct procedure frequently made excellent starting points for solving (48), often greatly reducing the iteration numbers incurred by `fmincon` and GRANSO compared to initializing these codes from randomly generated starting points. Regardless of the starting points employed, the optimization routines typically only found solutions that reduced the objective function of (48) by at most an order of magnitude compared to the solutions obtained by the direct procedure, and the resulting extrapolations were generally no better in terms of acceptance rate than the ones produced by our direct procedure. This seems to either confirm that these particular problems are poorly suited for evaluating extrapolation, as suggested above, or indicate that it is perhaps the quality, or lack thereof, of the vector extrapolations themselves, rows r_1, r_2 and columns c_1, c_2, that is causing the difficulties in producing good rank-2 extrapolations. We examined the vector sequences used to create r_1, r_2 and c_1, c_2 and did see significant oscillation, which appears to be a property of the iterates of Algorithm SVSA-RF itself, at least for these particular problems; this oscillation may be an additional difficulty to overcome for improved rank-2 extrapolation performance, but we leave such investigation for future work.

Lastly, we noticed that the fast upper bound procedure had some difficulty before it was able to find a destabilizing perturbation for problem `CM4` (continuous-time). Generally, we have found that the upper bound procedure can find a destabilizing perturbation within a handful of iterations, but on `CM4`, it took 23 iterations. In Figure 3, we show plots of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_{\mathrm{F}}}(A,B,C,D)$ for the largest value of ε obtained in the upper bound procedure, along with selected iterates of the routine corresponding to that value of ε. As is apparent from the plots, part of the difficulty in finding an upper bound is due to the highly nonconvex "horseshoe" shapes that create locally rightmost points in the left

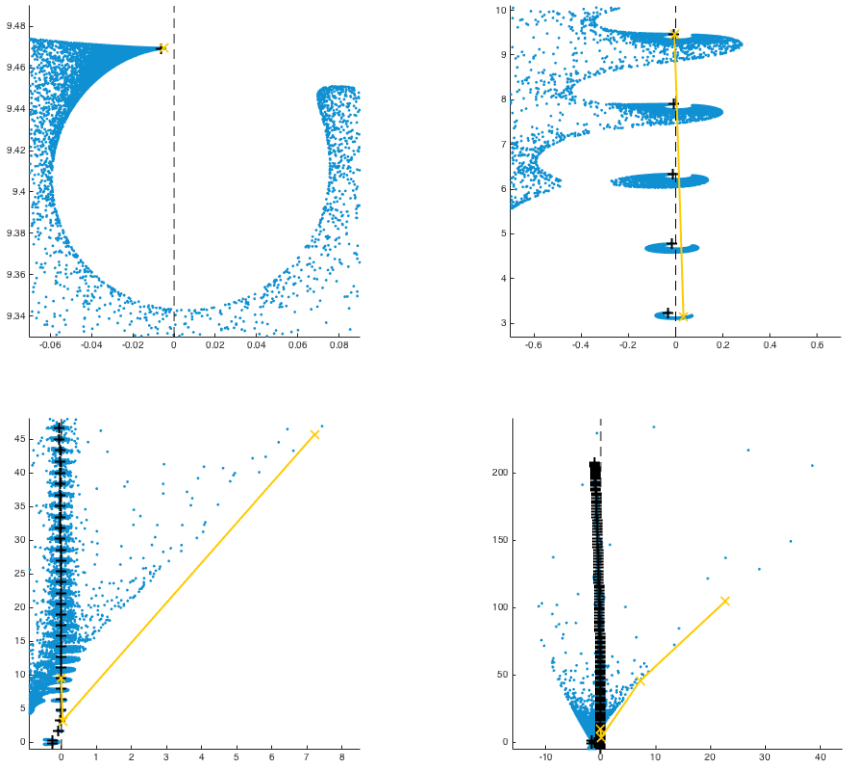

Figure 3: Test problem `CM4`. Successively wider views (left-right, top-to-bottom) of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$ (realized by the blue dot samples and generated in a similar manner as described in Figure 1) showing selected iterates of the upper bound procedure (yellow x's connected by line segments) with ε near its limit of $\|D\|^{-1}$. The black +'s are eigenvalues of A while the black dashed line is the imaginary axis. Top left: the expansion phase of the upper bound procedure has nearly converged to a locally rightmost point that is just to the right of an eigenvalue of A but this region of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$ is always contained in the left half-plane due to the limit $\varepsilon < \|D\|^{-1}$. Top right: this highly nonconvex "horseshoe" structure is repeated in multiple places; on the next expansion step, the routine was able to "jump" to a different region of $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$ that is in the right half-plane. Bottom left: on the next step after that, the expansion phase again jumps out of one region to another, this time significantly farther to the right; though technically an upper bound had already been found, the routine continued to iterate to better locate where a minimal destabilizing perturbation may lie. Bottom right: zooming out further, we see that $\sigma_\varepsilon^{\mathbb{R},\|\cdot\|_F}(A,B,C,D)$ is much larger than was initially apparent.

half-plane for values of ε near its upper bound $\|D\|^{-1}$. The expansion routine had converged to such a point and then iteratively increased epsilon to be near its upper bound in vain. However, and surprisingly, on the 23rd iteration of the upper bound procedure, the expansion phase was actually able to jump out of this region and land in the right half-plane to find a destabilizing perturbation and thus an upper bound. Even though the routine had essentially already converged to a perturbation corresponding to this locally rightmost point in the left half-plane, the routine still produced another update step to try, but this update step was nearly identical to the current perturbation, because further rightward continuous progress was not possible. The full update step failed to satisfy monotonicity so the line search was invoked with an initial interpolation of $t = 0.5$

and as a result, the resulting unnormalized interpolation of the current perturbation and the full update step perturbation nearly annihilated each other. When that interpolation was renormalized back to have unit Frobenius norm, as is necessary, the resulting perturbation was very different than the current perturbation (as well as the full update step), which thus allowed the algorithm to jump to an entirely different disconnected region of the spectral value set. We note that Algorithm SVSA-RF can also "jump" when a new perturbation just happens to result in a second eigenvalue of A being taken farther to the right than the eigenvalue it had intended to push rightward.

8 Conclusion

We have presented an algorithm that, to our knowledge, is the first method available to approximate the real stability radius of a linear dynamical system with inputs and outputs defined using Frobenius-norm bounded perturbations. It is efficient even in the large scale case, and since it generates destabilizing perturbations explicitly, it produces guaranteed upper bounds on the real stability radius. The hybrid expansion-contraction method works by alternating between (a) iterating over a sequence of destabilizing perturbations of fixed norm ε to push an eigenvalue of the corresponding perturbed system matrix as far to the right in the complex plane as possible and (b) contracting ε to bring the rightmost eigenvalue back to the imaginary axis. The final computed eigenvalue is very close to the imaginary axis and is typically at least a locally rightmost point of the corresponding ε-spectral value set. The method is supported by our theoretical results for the underlying ODE that motivates the method, and our computational results are validated by extensive random sampling techniques. The method has been implemented in our open-source MATLAB code `getStabRadBound`.

References

[BB90] 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(1):1–7, 1990.

[BB01] N.A. Bobylev and A.V. Bulatov. A bound on the real stability radius of continuous-time linear infinite-dimensional systems. *Computational Mathematics and Modeling*, 12(4):359–368, 2001.

[BBD01] N.A. Bobylev, A.V. Bulatov, and Ph. Diamond. Estimates of the real structured radius of stability of linear dynamic systems. *Automation and Remote Control*, 62(4):505–512, 2001.

[Bob99] N.A. Bobylev. An easily computable estimate for the real unstructured F-stability radius. *International Journal of Control*, 72(6):493–500, 1999.

[BS90] N.A. Bruinsma and M. Steinbuch. A fast algorithm to compute the H^∞-norm of a transfer function matrix. *Systems Control Letters*, 14:287–293, 1990.

[BS98] E.K. Boukas and P. Shi. Stochastic stability and guaranteed cost control of discrete-time uncertain systems with Markovian jumping parameters. *International Journal of Robust and Nonlinear Control*, 8(13):1155–1167, 1998.

[BS99] E.K. Boukas and P. Shi. H_∞ control for discrete-time linear systems with Frobenius norm-bounded uncertainties. *Automatica*, 35(9):1625 – 1631, 1999.

[BV14] P. Benner and M. Voigt. A structured pseudospectral method for H_∞-norm computation of large-scale descriptor systems. *Math. Control Signals Systems*, 26(2):303–338, 2014.

[CMO17] F.E. Curtis, T. Mitchell, and M.L. Overton. A BFGS-SQP method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles. *Optimization Methods and Software*, 32(1):148–181, 2017.

[FSVD14] M.A. Freitag, A. Spence, and P. Van Dooren. Calculating the H_∞-norm using the implicit determinant method. *SIAM J. Matrix Anal. Appl.*, 35(2):619–635, 2014.

[GGO13] N. Guglielmi, M. Gürbüzbalaban, and M.L. Overton. Fast approximation of the H_∞ norm via optimization over spectral value sets. *SIAM Journal on Matrix Analysis and Applications*, 34(2):709–737, 2013.

[GL13] N. Guglielmi and C. Lubich. Low-rank dynamics for computing extremal points of real pseudospectra. *SIAM J. Matrix Anal. Appl.*, 34(1):40–66, 2013.

[GM15] N. Guglielmi and M. Manetta. Approximating real stability radii. *IMA J. Numer. Anal.*, 35(3):1402–1425, 2015.

[GO11] N. Guglielmi and M.L. Overton. Fast algorithms for the approximation of the pseudospectral abscissa and pseudospectral radius of a matrix. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1166–1192, 2011.

[Gug16] N. Guglielmi. On the method by Rostami for computing the real stability radius of large and sparse matrices. *SIAM J. Sci. Comput.*, 38(3):A1662–A1681, 2016.

[GV83] G.H. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1983.

[HJ90] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1990. Corrected reprint of the 1985 original.

[HK94] D. Hinrichsen and B. Kelb. Stability radii and spectral value sets for real matrix perturbations. *Systems and Networks: Mathematical Theory and Applications*, 2:217–220, 1994.

[HP90a] D. Hinrichsen and A.J. Pritchard. A note on some differences between real and complex stability radii. *Systems Control Lett.*, 14(5):401–408, 1990.

[HP90b] D. Hinrichsen and A.J. Pritchard. Real and complex stability radii: a survey. In *Control of uncertain systems (Bremen, 1989)*, volume 6 of *Progr. Systems Control Theory*, pages 119–162. Birkhäuser Boston, Boston, MA, 1990.

[HP05] D. Hinrichsen and A.J. Pritchard. *Mathematical Systems Theory I: Modelling, State Space Analysis, Stability and Robustness*. Springer, Berlin, Heidelberg and New York, 2005.

[Kar03] M. Karow. *Geometry of Spectral Value Sets*. PhD thesis, Universität Bremen, 2003. http://page.math.tu-berlin.de/ karow/papers/dissUNDERSCOREkarow.pdf.

[KV14] D. Kressner and B. Vandereycken. Subspace methods for computing the pseudospectral abscissa and the stability radius. *SIAM J. Matrix Anal. Appl.*, 35(1):292–313, 2014.

[LKL96] J.H. Lee, W.H. Kwon, and J.W. Lee. Quadratic stability and stabilization of linear systems with Frobenius norm-bounded uncertainties. *Automatic Control, IEEE Transactions on*, 41(3):453–456, Mar 1996.

[LR14] H. Logemann and E.P. Ryan. *Ordinary Differential Equations. Analysis, Qualitative Theory and Control*. Springer, London, 2014. Springer Undergraduate Mathematics Series.

[Mit14] T. Mitchell. *Robust and efficient methods for approximation and optimization of stability measures*. PhD thesis, New York University, 2014.

[MO16] T. Mitchell and M.L. Overton. Hybrid expansion-contraction: a robust scaleable method for approximating the H_∞ norm. *IMA J. Numer. Anal.*, 36(3):985–1014, 2016.

[QBR+95] L. Qiu, B. Bernhardsson, A. Rantzer, E.J. Davison, P.M. Young, and J.C. Doyle. A formula for computation of the real stability radius. *Automatica J. IFAC*, 31(6):879–890, 1995.

[Ros15] M.W. Rostami. New algorithms for computing the real structured pseudospectral abscissa and the real stability radius of large and sparse matrices. *SIAM J. Sci. Comput.*, 37(5):S447–S471, 2015.

[SVDT96] J. Sreedhar, P. Van Dooren, and A.L. Tits. A fast algorithm to compute the real structured stability radius. In *Stability Theory*, pages 219–230. Springer, 1996.

[ZGD95] K. Zhou, K. Glover, and J. Doyle. *Robust and Optimal Control*. Prentice Hall, 1995.

A Notes on implementing HEC

We use HEC here to refer to the original HEC algorithm [MO16] and Algorithm HEC-RF since the issues outlined in this section apply equal to both methods. In theory, HEC converges once a locally rightmost point λ with $\mathrm{Re}\,(\lambda) = 0$ has been found. In practice however, it is not so straightforward. Indeed, we have discovered that the HEC convergence criteria described in [MO16, Section 7.1], using the expansion phase stopping condition proposed in [GGO13, Section 5], can sometimes be inadequate. We briefly recap these conditions and then present improved criteria, which we have used in our new version of `getStabRadBound` and all experiments in this paper.

To ensure that iterates always remain in the right half-plane (which is necessary for provable convergence of HEC), the Newton-bisection based contraction phase is set to find a point on the line $x = \tau_\varepsilon/2$, where $\tau_\varepsilon > 0$ is the contraction tolerance. The contraction is said to have converged if it finds a point $\lambda_c \in [0, \tau_\varepsilon)$. This shift permits either right or left-sided convergence; for the unshifted problem, i.e. $x = 0$, convergence from the left would almost always fail to satisfy the requirement that HEC points remain in the right half-plane. The expansion phase then pushes rightward from $\lambda_0 \coloneqq \lambda_c$ with iterates λ_k, where $\mathrm{Re}\,(\lambda_{k+1}) > \mathrm{Re}\,(\lambda_k)$ holds for all $k \geq 0$ (due to the line search ensuring monotonicity). In [GGO13], it was proposed that the expansion phase should be halted once $\mathrm{Re}\,(\lambda_{k+1}) - \mathrm{Re}\,(\lambda_k) < \tau_{uv} \max(1, \mathrm{Re}\,(\lambda_k))$ is satisfied.[3] In [MO16], HEC was then said to have converged if either (a) the expansion phase returned a point λ such that $\mathrm{Re}\,(\lambda) < \tau_\varepsilon + \tau_{uv}$, where $\tau_{uv} > 0$ is the expansion tolerance, or (b) if the expansion and contraction phases failed consecutively, in either order. We now discuss the inadequacies and propose improved conditions.

First, we make the change that HEC is said to have converged if $\mathrm{Re}\,(\lambda) \in [0, 2\tau_\varepsilon)$, where λ is a locally rightmost point of the associated spectral value set encountered by HEC. Unfortunately, detecting whether λ is locally rightmost is somewhat problematic due to the lack of an efficiently computable optimality measure. Furthermore, the expansion phase stopping condition described above may cause expansion to halt prematurely, which in turn can cause HEC to return an unnecessarily higher value of ε, that is, a worse approximation. Part of this is due to the fact that when the expansion iterates have real part less than one, the condition only measures an absolute difference between consecutive steps, which is often a rather poor optimality measure. Furthermore, by only measuring the difference between the real parts, it fails to capture change in the imaginary parts, which if present, would strongly indicate that a locally rightmost point has not yet been reached. To address both concerns, we instead propose to measure the relative change *in the complex plane*

[3] Note that compared to [GGO13], we have dropped the absolute value signs here, since in the context of HEC, all points lie in the right half-plane.

Simplified list of expansion/contraction termination possibilities
Contraction Phase:
0: maxit reached though some contraction achieved
1: point with real part in $[0, \tau_\varepsilon)$ attained
2: significant contraction achieved in right half-plane, halted early
3: desired point is bracketed by two consecutive floating point numbers (no convergence criteria satisfied)
Expansion Phase:
0: maxit reached
1: relative difference, in $\mathbb{C}$, between λ_{k+1} and λ_k is less than τ_{uv}
2: step length $\mathrm{Re}(\lambda_{k+1}) - \mathrm{Re}(\lambda_k)$ has significantly shortened, halted early
3: line search failed to produce a monotonic step

Table 4: For both phases, return code 1 is used to indicate that the respective desired convergence has been achieved, while return code 2 indicates that the early termination features were invoked. For the contraction phase, return code 3 indicates that the precision limits of the hardware precludes satisfying the convergence criteria, i.e., it is a hard failure. For the expansion phase, return code 3 can be interpreted as a sign of convergence, as further rightward progress is apparently no longer possible.

between consecutive iterates, that is, $|\lambda_{k+1} - \lambda_k|/|\lambda_k|$, and to halt the expansion phase when this value falls below τ_{uv}. Doing so however creates two additional wrinkles that must now be addressed. The condition is not meaningful when either point is exactly zero. Therefore, we simply skip this particular check if either is indeed zero, noting that it can, at most, only add two additional steps to be taken (provided no other termination condition is triggered). Also, we must prevent "false" oscillation in imaginary parts from being measured due to complex conjugacy; when matrices A, B, C, and D are all real, we simply ensure that distance is measured between eigenvalues in the upper half-plane only, by flipping signs of the imaginary parts as necessary.

Another major issue is that, in practice, both the contraction and expansion phases may terminate in a multitude of ways without satisfying their respective convergence criteria, including our improved expansion-halting condition above. In Table 4, we give a simplified list of these termination possibilities. We now describe how to interpret the combination of these possibilities that can occur and how their consequences should be handled and present a concise pseudocode for the resulting practical implementation of HEC.

We have designed the contraction procedure so that reaching its maximum allowed iteration count will only cause it to actually halt iterating if it has also achieved some amount of contraction, that is, it has encountered at least one point $\tilde{\lambda}$ such that $0 \leq \mathrm{Re}(\tilde{\lambda}) \leq \lambda$, where λ is the initial point. This seemingly unconventional behavior has the two benefits that the only case when it doesn't make any progress is when it is impossible to do so (i.e. when it exhausts the machine's precision) and a sufficiently large maximum iteration limit to find a first contraction step no longer needs to be known a priori, which is generally not possible. If the contraction routine has made progress (no matter the termination condition), then additional expansion is always potentially possible and so the next expansion phase must be attempted. Furthermore, even if the contraction phase failed to make any progress, but if the previous expansion did not converge, then the next expansion phase should attempt to *resume* it since further expansion is apparently possible and may enable the subsequent contraction phase to finally make progress. The only case remaining is when the contraction phase failed to make any progress (by reaching the limits of the hardware) *after* having had the preceding expansion phase converge (meaning it would not be able make further progress if it were to be rerun with the same value of ε). In this situation, HEC can no longer make any progress and must quit. However, even though the contraction phase failed to meet its convergence criteria, $\mathrm{Re}(\lambda_c) \in [0, 2\tau_\varepsilon)$ may still hold, so HEC may sometimes terminate successfully in this case. If not, HEC has stagnated, which is likely an indication that tolerances are too tight for the

Pertinent pseudocode for a practical HEC implementation

Input: $\varepsilon_0 > 0$, rightmost $\lambda_0 \in \sigma(M(\varepsilon_0 UV^*))$ such that $\mathrm{Re}\,(\lambda_0) > 0$ with $\|UV^*\| = 1$ for the chosen norm and $U \in \mathbb{C}^p$, $V \in \mathbb{C}^m$ or $U \in \mathbb{R}^{p\times 2}$, $V \in \mathbb{R}^{m\times 2}$ for respectively approximating the complex or real stability radius radius, initial contraction bracket given by $\varepsilon_{\mathrm{LB}} \coloneqq 0$ and $\varepsilon_{\mathrm{UB}} \coloneqq \varepsilon_0$, boolean variable `expand_converged` $\coloneqq$ `false`, ...

Output: Final value of sequence $\{\varepsilon_k\}$.

for $k = 0, 1, 2, \ldots$ **do**
  $[\varepsilon_{\mathrm{c}}, \lambda_{\mathrm{c}}, \varepsilon_{\mathrm{LB}}, \varepsilon_{\mathrm{UB}}, \texttt{ret\_con}] = \texttt{contract}(\varepsilon_{\mathrm{LB}}, \varepsilon_{\mathrm{UB}}, \lambda_k, \ldots)$ // *where:*
  // $0 < \hat{\varepsilon} \leq \varepsilon$ *is the possibly contracted value of* ε
  // $0 \leq \mathrm{Re}\,(\lambda_{\mathrm{c}}) \leq \mathrm{Re}\,(\lambda_k)$ *is the possibly contracted eigenvalue*
  // *possibly updated* $\varepsilon_{\mathrm{LB}}$ *and* $\varepsilon_{\mathrm{UB}}$ *giving the tightest bracket encountered*
  // `ret_con` *is the contraction's return code from Table 4*
  // *Check if no contraction was possible (precision of hardware exhausted)*
  if `ret_con` $== 3$ **and** `expand_converged` **then**
    if $\mathrm{Re}\,(\lambda_c) < 2 \cdot \tau_\varepsilon$ **then**
      return // *HEC converged to tolerance*
    else
      return // *HEC stagnated*
    end if
  end if
  $\varepsilon_{k+1} \coloneqq \varepsilon_{\mathrm{c}}$
  $[\lambda_{k+1}, \texttt{ret\_exp}] = \texttt{expand}(\varepsilon_{k+1}, \lambda_{\mathrm{c}}, \ldots)$ // *where:*
  // $\mathrm{Re}\,(\lambda_{k+1}) \geq \mathrm{Re}\,(\lambda_{\mathrm{c}})$
  // `ret_exp` *is the expansion's return code from Table 4*
  `expand_converged` $\coloneqq$ `(ret_exp` $== 1$ `or ret_exp` $== 3$`)`
  if `expand_converged` **and** $\mathrm{Re}\,(\lambda_{k+1}) < 2 \cdot \tau_\varepsilon$ **then**
    return // *HEC converged to tolerance*
  else if $\mathrm{Re}\,(\lambda_{k+1} - \lambda_{\mathrm{c}}) > 0$ **then**
    $\varepsilon_{\mathrm{LB}} \coloneqq 0$, $\varepsilon_{\mathrm{UB}} \coloneqq \varepsilon_{k+1}$ // *Expansion made some progress; do new contraction*
  else if `ret_con` $== 3$ **then**
    return // *HEC stagnated*
  end if // *Else contraction will be resumed/restarted from where it last left off*
end for

available precision of the hardware on the particular problem (or possibly that a subroutine has failed in practice).

For each expansion phase, we consider it to have converged once it can no longer make any meaningful rightward progress. Our new stopping criteria attempt to capture precisely that, and do so more accurately than the previous scheme. Furthermore, if the line search fails to produce a monotonic step, then the expansion routine is, by default, unable to make further progress. We have observed that the line search failing is generally a good sign, often implying that a stationary point has already been found. We thus consider the expansion phase to have converged if either our new stopping condition is met or if the line search fails. Otherwise, further expansion is potentially possible. After an expansion phase, HEC should first check if the expansion phase converged and whether $\mathrm{Re}\,(\lambda) \in [0, 2\tau_\varepsilon)$ holds, as the two conditions together indicate HEC has converged and can halt with success. However, if the expansion phase has made progress, then, since it has not converged, HEC should continue by starting a new contraction phase. Otherwise, we know that the expansion phase has not made any progress and is thus considered converged. If the previous

contraction phase exhausted the precision of the machine, then the HEC iteration can no longer continue and it has stagnated before meeting its convergence criteria for tolerances that are likely too tight. The only remaining possibility is that the contraction phase achieved some amount of contraction but did not yet converge. In this last case, the contraction phase should be restarted from its most recent bracket to see if it can make further progress, which might enable a subsequent expansion to succeed.

The above design also causes the respective maximum iteration limits of the expansion and contraction phases to act as additional early termination features early within the HEC iteration, without ever comprising the final numerical accuracy.

B A new iteration for the complex stability radius

Though we have developed Algorithm SVSA-RF specifically to iterate over real-valued perturbations with rank at most two, it also permits a natural extension to a complex-valued rank-1 expansion iteration as well, essentially by replacing $\mathrm{Re}\,(uv^*)$ with uv^* in (26). In Tables 5-7, we compare the original HEC algorithm with an alternative variant which employs this ODE-based expansion iteration for approximating the complex stability radius. Overall there doesn't seem to be a clear answer as to which version performs better, as they perform roughly the same on many problems. However, it is worth noting that ODE-based version had outstanding performance on continuous-time problems `ROC1` and `ROC2`, respectively requiring only 3.3% and 16.6% of the eigensolves as compared to the unmodified HEC algorithm. Furthermore, on the latter example, the ODE-based version also returns a significantly better approximation.

Small Dense Problems: Continuous-time								
	Iters		# Eig		Time (secs)		CSR Approximation	
Problem	v1	v2	v1	v2	v1	v2	$\min\{\varepsilon_1, \varepsilon_2\}$	$(\varepsilon_1 - \varepsilon_2)/\varepsilon_1$
`CBM`	4	4	169	242	29.136	39.991	$3.80262077280 \times 10^{0}$	-1.5×10^{-11}
`CSE2`	6	6	1298	1038	5.344	4.327	$4.91778669279 \times 10^{1}$	$+3.0 \times 10^{-5}$
`CM1`	-	-	-	-	-	-	-	-
`CM3`	4	3	224	173	3.602	2.794	$1.21736892616 \times 10^{0}$	-9.8×10^{-3}
`CM4`	4	4	379	500	27.367	36.763	$6.29458676972 \times 10^{-1}$	$+1.2 \times 10^{-12}$
`HE6`	11	11	20459	20981	12.524	12.637	$2.02865555308 \times 10^{-3}$	-6.1×10^{-11}
`HE7`	4	4	610	478	0.462	0.389	$2.88575420548 \times 10^{-3}$	-
`ROC1`	3	3	4139	136	2.521	0.204	$8.21970266187 \times 10^{-1}$	-
`ROC2`	5	3	481	80	0.361	0.131	$7.49812117958 \times 10^{0}$	$+9.1 \times 10^{-2}$
`ROC3`	6	4	270	193	0.253	0.197	$5.80347782972 \times 10^{-5}$	-6.2×10^{-11}
`ROC4`	3	1	223	111	0.204	0.133	$3.38236009391 \times 10^{-3}$	-2.7×10^{-2}
`ROC5`	10	15.5	546	669	0.356	0.422	$1.02041169816 \times 10^{2}$	$+2.5 \times 10^{-7}$
`ROC6`	4	4	183	127	0.176	0.145	$3.88148973329 \times 10^{-2}$	-
`ROC7`	5	5	4439	4541	2.183	2.323	$8.91295691482 \times 10^{-1}$	-
`ROC8`	3	3	236	124	0.226	0.167	$1.51539044957 \times 10^{-1}$	-9.9×10^{-9}
`ROC9`	6	6	705	674	0.509	0.479	$3.03578083291 \times 10^{-1}$	-
`ROC10`	3	3	412	413	0.282	0.305	$9.85411638072 \times 10^{0}$	-

Table 5: The columns are the same as described in the caption of Table 1 except that here, we compare the HEC algorithm for approximating the complex stability radius (CSR) using its original expansion iteration, which we call "v1", and an ODE-based variant, which we call "v2". We tested both versions without any acceleration features enabled. Note that the ODE-based approach failed to find an upper bound for `CM1` and thus no data is reported for this example.

Small Dense Problems: Discrete-time								
	Iters		# Eig		Time (secs)		CSR Approximation	
Problem	v1	v2	v1	v2	v1	v2	$\min\{\varepsilon_1, \varepsilon_2\}$	$(\varepsilon_1 - \varepsilon_2)/\varepsilon_1$
AC5	3	3	177	173	0.165	0.163	$1.31122274593 \times 10^{-2}$	-
AC12	3	3	1258	1290	0.789	0.768	$9.24428279719 \times 10^{-2}$	-
AC15	5	5	143	143	0.156	0.142	$4.22159665084 \times 10^{-2}$	-
AC16	4	4	940	931	0.553	0.537	$5.46839166119 \times 10^{-2}$	-
AC17	4	7	237	313	0.198	0.268	$3.33193351568 \times 10^{-6}$	$+5.5 \times 10^{-10}$
REA1	3	3	766	772	0.488	0.546	$1.34439972514 \times 10^{-3}$	-
AC1	4	4	4787	4771	2.471	2.591	$6.65190471447 \times 10^{0}$	-
AC2	4	4	511	570	0.357	0.393	$3.27216087222 \times 10^{0}$	-
AC3	4	4	938	917	0.541	0.548	$5.25195687767 \times 10^{-2}$	-
AC6	11	13	875	911	0.552	0.621	$1.88905032255 \times 10^{-8}$	-9.6×10^{-9}
AC11	7.5	6.5	286	347	0.245	0.268	$4.57670218088 \times 10^{-8}$	$+1.3 \times 10^{-8}$
ROC3	4	4	1510	1485	1.022	1.027	$4.27872787193 \times 10^{-2}$	-
ROC5	6	8	358	432	0.292	0.329	$2.55709313478 \times 10^{-4}$	$+8.0 \times 10^{-11}$
ROC6	11	11	16832	16646	9.600	9.137	$5.81391331473 \times 10^{-2}$	-
ROC7	4	4	3509	3480	1.855	1.928	$9.01354009455 \times 10^{-1}$	-
ROC8	3	3	127	124	0.145	0.181	$1.59160474017 \times 10^{-5}$	-1.2×10^{-10}
ROC9	4	4	286	272	0.272	0.249	$3.49507190967 \times 10^{-2}$	-

Table 6: See caption of Table 5 for the description of the columns.

Large Sparse Problems: Continuous-time (top), Discrete-time (bottom)								
	Iters		# Eig		Time (secs)		CSR Approximation	
Problem	v1	v2	v1	v2	v1	v2	$\min\{\varepsilon_1, \varepsilon_2\}$	$(\varepsilon_1 - \varepsilon_2)/\varepsilon_1$
NN18	3	3	89	107	8.315	8.295	$9.77172733234 \times 10^{-1}$	-
dwave	2	2	79	70	36.342	33.786	$2.63019715625 \times 10^{-5}$	-
markov	2	2	57	65	12.179	13.279	$1.61146532880 \times 10^{-4}$	-
pde	4	3	251	202	23.441	18.918	$2.71186478815 \times 10^{-3}$	-4.4×10^{-9}
rdbrusselator	4	4	289	369	91.167	112.681	$5.35246333569 \times 10^{-4}$	-
skewlap3d	2	2	118	91	136.307	141.747	$4.59992022215 \times 10^{-3}$	-
sparserandom	2	2	74	65	3.106	2.618	$7.04698184526 \times 10^{-6}$	-
dwave	2	2	42	33	21.303	14.683	$2.56235064981 \times 10^{-5}$	-
markov	3	3	65	75	14.383	17.617	$2.43146945130 \times 10^{-4}$	-
pde	5	3	2839	2091	184.350	133.007	$2.68649594344 \times 10^{-4}$	-4.6×10^{-4}
rdbrusselator	3	3	51	69	5.081	6.892	$2.56948942080 \times 10^{-4}$	-
skewlap3d	2	2	47	56	58.705	49.745	$3.40623440406 \times 10^{-5}$	-
sparserandom	2	2	24	22	0.927	0.897	$2.53298721307 \times 10^{-7}$	-8.2×10^{-10}
tolosa	4	4	2314	2315	598.283	547.306	$1.76587075884 \times 10^{-7}$	-

Table 7: See caption of Table 5 for the description of the columns.