Computing the Kreiss Constant of a Matrix

Tim Mitchell*

July 15th, 2019

Abstract

Inspired by algorithms for computing the distance to uncontrollability, we present the first algorithm for computing the Kreiss constant of a matrix, for both continuous-time and discrete-time systems. Our method combines fast optimization and globality certificate computations to converge to the Kreiss constant of a matrix. Furthermore, locally-optimal approximations to Kreiss constants of large-scale matrices can be efficiently obtained via the optimization techniques we employ. We also present a new result on the numerical accuracy of the trisection method for computing the distance to uncontrollability and investigate the viability of adapting trisection for also computing Kreiss constants.

1 Introduction

Given a matrix $A \in \mathbb{C}^{n\times n}$, the ordinary difference equation

$$x_{k+1} = Ax_k \tag{1.1}$$

is asymptotically stable if A is Schur stable, i.e., if $\rho(A) < 1$, where ρ denotes the spectral radius. While $\rho(A)$ tells one about the asymptotic behavior of (1.1), it does not convey information about its transient behavior. For that, we can look at the Kreiss Matrix Theorem, which says for any matrix $A \in \mathbb{C}^{n\times n}$ [TE05, Eq. 18.2]

$$\mathcal{K}(A) \leq \sup_{k\geq 0} \|A^k\| \leq en\mathcal{K}(A), \tag{1.2}$$

where $\|\cdot\|$ is the 2-norm and the *Kreiss constant* $\mathcal{K}(A)$ is given by [TE05, p. 143]

$$\mathcal{K}(A) = \sup_{z\in\mathbb{C},|z|>1} (|z|-1)\|(zI-A)^{-1}\|. \tag{1.3}$$

As noted in [TE05], $\mathcal{K}(A)$ is also equivalent to

$$\mathcal{K}(A) = \sup_{\varepsilon>0} \frac{\rho_\varepsilon(A)-1}{\varepsilon}, \tag{1.4}$$

where the *ε-pseudospectral radius* ρ_ε is defined by

$$\rho_\varepsilon(A) = \max\{|z| : z \in \Lambda(A+\Delta), \|\Delta\| \leq \varepsilon\} \tag{1.5a}$$

$$= \max\{|z| : z \in \mathbb{C}, \|(zI-A)^{-1}\| \geq \varepsilon^{-1}\}. \tag{1.5b}$$

From (1.2), it is clear that that $\mathcal{K}(A) \geq 1$ (since k can be zero) and $\mathcal{K}(A)$ may be arbitrarily large. As is well known, a matrix A is *power-bounded*, i.e., $\mathcal{K}(A) < \infty$, if and only if $\rho(A) \leq 1$

*Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, 39106 Germany mitchell@mpi-magdeburg.mpg.de.

and all eigenvalues of A with modulus 1 are nondefective. If A is normal and $\rho(A) \leq 1$, then $\mathcal{K}(A) = 1$.

As discussed on [TE05, p. 177], the original statement by Kreiss in 1962 [Kre62] actually had a far looser upper bound than (1.2): approximately $c^{n^n}\mathcal{K}(A)$. The reduction of the constant factor to its current form in fact occurred over nearly thirty years in at least nine separate steps, with Spijker proving the conjecture of [LT84, p. 590] to finally obtain the (in a certain sense) tight factor of en in 1991 [Spi91].

The Kreiss Matrix Theorem also comes in a continuous-time variant for an ordinary differential equation

$$\dot{x} = Ax, \tag{1.6}$$

which is asymptotically stable if A is Hurwitz stable, i.e., if $\alpha(A) < 0$, where α denotes the spectral abscissa. In this case, the Kreiss Matrix Theorem states [TE05, Eq. 18.8]

$$\mathcal{K}(A) \leq \sup_{t \geq 0} \|\mathrm{e}^{tA}\| \leq en\mathcal{K}(A) \tag{1.7}$$

where by [TE05, Eq. 14.7], $\mathcal{K}(A)$ is now equivalently given by either

$$\mathcal{K}(A) = \sup_{z \in \mathbb{C}, \operatorname{Re} z > 0} (\operatorname{Re} z)\|(zI - A)^{-1}\| \tag{1.8}$$

or

$$\mathcal{K}(A) = \sup_{\varepsilon > 0} \frac{\alpha_\varepsilon(A)}{\varepsilon}, \tag{1.9}$$

and where the *ε-pseudospectral abscissa* α_ε is defined by

$$\alpha_\varepsilon(A) = \max\{\operatorname{Re} z : z \in \Lambda(A + \Delta), \|\Delta\| \leq \varepsilon\} \tag{1.10a}$$

$$= \max\{\operatorname{Re} z : z \in \mathbb{C}, \|(zI - A)^{-1}\| \geq \varepsilon^{-1}\}. \tag{1.10b}$$

Like the discrete-time case, $\mathcal{K}(A) \geq 1$ (take $t = 0$) and can be arbitrary large. If A is normal and $\alpha(A) \leq 0$, then $\mathcal{K}(A) = 1$.

Despite the wealth of work done over decades towards making the upper bound of the Kreiss Matrix Theorem now tight, to the best of our knowledge there has been no algorithm given to actually compute $\mathcal{K}(A)$. For example, in the literature, $\mathcal{K}(A)$ has simply been approximated by either plotting (1.4) or (1.9) and simply taking the maximum of the curve; e.g., see [EK17] and [Men06, Chapter 3.4.1]. In this paper, we address this deficiency by proposing the first algorithm to compute $\mathcal{K}(A)$ to arbitrary accuracy, under some assumptions regarding its subcomputations.

As it turns out, the task of computing the Kreiss constant of a matrix has some similarity to computing the *distance to uncontrollability*. Given matrices $A \in \mathbb{C}^{n \times n}$ and $B \in \mathbb{C}^{n \times m}$, consider the linear control system

$$\dot{x} = Ax + Bu, \tag{1.11}$$

where $x \in \mathbb{C}^n$ and $u \in \mathbb{C}^p$, the *control input*, are both dependent on time. A system is *controllable* if given respective initial and final states $x(0)$ and $x(T)$ there exists a control $u(\cdot)$ that realizes some trajectory $x(\cdot)$ with endpoints $x(0)$ and $x(T)$. The distance to uncontrollability, which we denote as $\tau(A, B)$, can be computed via solving the nonconvex optimization problem [Eis84]

$$\tau(A, B) = \min_{z \in \mathbb{C}} \sigma_{\min}([A - zI, B]), \tag{1.12}$$

where $\sigma_{\min}(\cdot)$ denotes the smallest singular value.

The first practical algorithm to address computing $\tau(A, B)$ is due to Gu [Gu00], who proposed a bisection method to compute $\tau(A, B)$ *within a factor of two*. This was done by devising a novel test that, given a guess $\gamma \geq 0$ for $\tau(A, B)$ and a second parameter $\eta \geq 0$, asserts that either $\tau(A, B) \leq \gamma$ or $\tau(A, B) > \gamma - \frac{\eta}{2}$ must hold (but the test only verifies one of these two conditions,

even if both hold). As this test is both expensive ($\mathcal{O}(n^6)$) and numerically challenging for small values of η, Burke, Lewis, and Overton [BLO04] followed up by proposing two improved iterations: a trisection algorithm and a hybrid optimization-with-restarts method, both of which make use of this same novel test of [Gu00, Section 3.2]. While it is theoretically possible to extend the original bisection method to compute $\tau(A, B)$ to higher precision, doing so then requires that the test be performed at crucial points for very small values of η, where the test is more likely to return an incorrect result numerically; hence, bisection may fail in practice. To postpone this numerical issue as long as possible, the trisection algorithm instead always reduces the value of parameter η by a factor of two-thirds every iteration, so that it only goes to zero in the limit as the method converges linearly to $\tau(A, B)$. Nevertheless, the authors advocated their hybrid optimization-with-restarts algorithm as preferable. This works by using much cheaper ($\mathcal{O}(n^3)$) optimization techniques to find local minimizers of (1.12) and then uses the expensive test as a *globality certificate*, i.e., it checks whether the minimizer is in fact a global minimizer, at which point the method can terminate. If not, the expensive test provides one or more new starting points from which optimization can be restarted with the guarantee that a better (lower) minimum of (1.12) will be found and so the process continues in a loop until the test asserts that a global minimizer has been attained. Compared to trisection, the hybrid optimization-with-restarts method is generally much faster since the number of expensive tests is greatly reduced. Finally, in [GMO$^+$06], a divide-and-conquer strategy was proposed that exploited additional structure of the expensive test so that it can instead be done in $\mathcal{O}(n^5)$ work in the worst case and $\mathcal{O}(n^4)$ on average.

Remark 1.1. *Regarding the aforementioned work complexity results, these are all with respect to treating computations of singular values, eigenvalues, solutions of Sylvester equations, etc., as* atomic operations *with cubic costs in the dimensions of the associated matrices. Following [BLO04], we adopt the same convention here but additionally assume that when using sparse methods to compute singular values or eigenvalues, the costs are then linear in the dimension of matrices.*

Inspired by the ideas of [Gu00, BLO04, GMO$^+$06], we propose a new method for obtaining the Kreiss constant of a matrix, for both the continuous- and discrete-time cases. This algorithm will be based on finding local maximizers of (1.3) or (1.8), as appropriate, using fast local optimization techniques, and then performing globality certificate computations to check for global convergence or to restart local optimization. Although we could alternatively consider using optimization to find maximizers of either (1.4) or (1.9), which has the benefit of working with only one optimization parameter instead of two, these objective functions are simply much more expensive to compute than (1.3) or (1.8); the quadratically-convergent criss-cross algorithms of [BLO03, MO05] to compute $\rho_\varepsilon(A)$ or $\alpha_\varepsilon(A)$, as well as the faster algorithms of [BM17], all require computing all eigenvalues of $2n \times 2n$ matrices, often several times.

Finally, we also derive a new result regarding the accuracy of the trisection algorithm for the distance to controllability and consider the viability of trisection iterations for computing $\mathcal{K}(A)$.

The paper is organized as follows. We present the existing bisection and trisection algorithms for computing the distance to uncontrollability in more detail in §2. In §3, we describe our algorithm using local optimization and globality certifcates for the computing the continuous-time version of the Kreiss constant for a given matrix; the optimization component can also be used by itself to obtain locally optimal approximations to the Kreiss constant of a large-scale matrix. We discuss alternative trisection iterations based on globality certificate computations and their downsides in §4. In §5, we adapt all of the components of the Kreiss constant algorithm given in §3 to the discrete-time case. Some numerical examples and concluding remarks are respectively given in §6 and §7.

2 Algorithms for the distance to uncontrollability

Definition 2.1. *Given a domain $\mathcal{D} \subseteq \mathbb{R}$, a function $f : \mathcal{D} \to \mathbb{R}$ has a* global Lipschitz constant (GLC) *of $c \geq 0$ if $|f(x) - f(y)| \leq c|x - y|$ for all $x, y \in \mathcal{D}$.*

The aforementioned algorithms for computing the distance to uncontrollability [Gu00, BLO04, GMO+06] are all built upon the following key result of [Gu00]:

Theorem 2.2. *Let $\gamma, \eta \geq 0$ be given. If $\tau(A, B) \leq \gamma$ and $\eta \in [0, 2(\gamma - \tau(A, B))]$, then there exists a pair $x, y \in \mathbb{R}$ such that*

$$\sigma_{\min}([A - (x + \mathbf{i}y)I, B]) = \sigma_{\min}([A - (x + \eta + \mathbf{i}y)I, B]) = \gamma. \tag{2.1}$$

Corollary 2.3. *Let $\gamma, \eta \geq 0$ be given. If there do not exist any pairs $x, y \in \mathbb{R}$ such that* (2.1) *holds, then*

$$\tau(A, B) > \gamma - \tfrac{\eta}{2}. \tag{2.2}$$

The proof of Theorem 2.2 relies on the fact that $\sigma_{\min}([A-(x+\mathbf{i}y)I, B])$ has a GLC of 1 with respect to either x or y.

What [Gu00, Section 3.2] additionally devised was a sequence of computations that verifies whether either (2.1) or (2.2) holds. The main idea is that given a guess $\gamma \geq 0$ for $\tau(A, B)$, the verification test checks if there are any points (x, y) such that (2.1) holds for $\eta = \gamma$. If there are such points, the test returns these points and so $\tau(A, B) \leq \gamma$ is verified. Otherwise, the test returns no points but then (2.2) must hold. The test was then used to estimate $\tau(A, B)$ with a factor of two via bisection. For initialization, γ is set to $\sigma_{\min}([A, B])$ and $\eta = \gamma$. If the test verifies $\tau(A, B) \leq \gamma$, then γ and η are both halved (so $\eta = \gamma$ still holds) and the test is redone. Otherwise, (2.2) holds and so γ and $\tau(A, B)$ must be within a factor of 2 of each other. Thus, halving γ and η is done in a loop until the test verifies that (2.2) holds.

As noted in [BLO04, p. 358], it is tempting to try to obtain $\tau(A, B)$ to higher precision by a bisection method, which would work by updating upper and lower bounds (in contrast, the bisection method of [Gu00] only updates an upper bound). Let $L = 0$ be an initial (trivial) lower bound for $\tau(A, B)$, $U = \sigma_{\min}([A, B])$ be an initial upper bound, $\gamma = \frac{L+U}{2}$, and $\eta > 0$. If the verification test returns any points satisfying (2.1) for the current values of γ and η, then the upper bound is updated by setting $U = \gamma$. Otherwise, (2.2) holds but this is not enough to assert whether $\tau(A, B) > \gamma$ holds. To do that, one can instead repeat the verification test for a very small value of η. If the test still returns no points, then it is reasonable to assume that $\tau(A, B) > \gamma$ does indeed hold and so the lower bound is updated by setting $L = \gamma$ and bisection continues. The main problem with this strategy is that in the presence of rounding errors, the verification test is likely to fail for the very small values of η that are needed, i.e., the test may fail to detect points which would be detected in exact arithmetic. If this happens, the lower bound will be updated erroneously and so bisection will not converge to $\tau(A, B)$.

The trisection algorithm of [BLO04, Algorithm 5.2] balances how much the lower bound is updated with how quickly the value of η is decreased, precisely to postpone this difficult numerical issue as long as possible. The trisection algorithm works as follows. Again let $L = 0$ and $U = \sigma_{\min}([A, B])$ be initial lower and upper bounds, respectively. Then on the kth iteration, $\eta_k = \frac{2}{3}(U - L)$ and $\gamma_k = L + \eta_k$ are set as the current values of η and γ for the verification test. If the test finds points satisfying (2.1), then the upper bound is updated by setting $U = \gamma_k$. Otherwise, (2.2) holds so we know that $\tau(A, B) \geq \gamma_k - \frac{\eta_k}{2} = L + \frac{\eta_k}{2}$ and so now the lower bound can be updated by setting $L = L + \frac{\eta_k}{2}$. Hence, the new interval has length $\frac{2}{3}(U - L)$.

3 The continuous-time case for $\mathcal{K}(A)$

To compute $\mathcal{K}(A)$, we will work with the inverse of (1.8), identifying $\mathbb{C}$ with $\mathbb{R}^2$, and equivalently solve

$$\mathcal{K}(A)^{-1} = \inf_{x>0, y\in\mathbb{R}} \sigma_{\min}\left(\frac{(x+\mathbf{i}y)I - A}{x}\right). \tag{3.1}$$

It will be helpful to have the following definition

$$f(x,y) = \sigma_{\min}(F(x,y)) \quad \text{where} \quad F(x,y) = \frac{(x+\mathbf{i}y)I - A}{x}. \tag{3.2}$$

The algorithm we propose will work as follows. We assume that $\alpha(A) \leq 0$ (since otherwise $\mathcal{K}(A) = \infty$) and that A is a nonnormal matrix (as otherwise $\mathcal{K}(A) = 1$). To begin, an optimization solver is used to search for a local or possibly global minimizer (x_*, y_*) of (3.2) with $x_* > 0$ and $f(x_*, y_*) = \gamma < 1$, though it may be that (x_*, y_*) is merely stationary. We assume for simplicity that the solver finds an exact stationary point. Then a globality certificate is computed to assert whether (x_*, y_*) is a global minimizer, in which case $\gamma = \mathcal{K}(A)^{-1}$ and the algorithm terminates. Otherwise, the certificate computation provides a point $(\tilde{x}, \tilde{y})$ on the same (or lower) level set as (x_*, y_*), i.e., $f(\tilde{x}, \tilde{y}) \leq \gamma$, from which optimization can be restarted to obtain a better (lower) minimizer of (3.2). This entire process continues until $\gamma = \mathcal{K}(A)^{-1}$.

As we will discuss in §3.1, the local optimization phase when using dense SVD methods is $\mathcal{O}(n^3)$ work, with a superlinear or quadratic rate of convergence, but the superlinear variant is also obtainable when using a scalable sparse method for computing smallest singular values. Inspired by the verification test of [Gu00, Section 3.2] for the case of the distance to uncontrollability, in §3.2 we derive a globality certificate computation for (3.1), which requires $\mathcal{O}(n^6)$ work if done directly. Then in §3.3, by adapting the divide-and-conquer technique from [GMO+06] for the distance to uncontrollability, we show how this globality certificate can be computed in $\mathcal{O}(n^4)$ work on average and $\mathcal{O}(n^5)$ in the worst case. However, as we will see in §6.3, for the case of computing Kreiss constants, this appears to be more of a theoretical result than a practical method.

At a high level, our algorithm is similar to other hybrid optimize and restart strategies, such as the aforementioned [BLO04, Algorithm 5.3] for the distance to uncontrollability as well as the more recent method of [BM18, Algorithm 2] for the $\mathcal{H}_\infty$ norm. The shared idea is that instead of an iteration based solely on certificate computations, e.g. bisection and trisection algorithms, optimization can be used so that these expensive certificate computations are only done to assert whether or not the obtained stationary points are global optimizers. This strategy generally provides meaningful speedups of runtimes because applying optimization is much cheaper than doing the certificate computations and still provides fast local convergence. Furthermore, the speedups can be particularly dramatic when only a handful of restarts are necessary to find a global optimizer.

3.1 Local optimization

To find minimizers of (3.2) in the right half-plane using quasi-Newton (or Newton) methods, we will use the gradient (and the Hessian) of $f(x,y)$. Although singular values can vary nonsmoothly with respect to matrix entries, they are nevertheless locally Lipschitz, and so this nonsmoothness is confined to a set of measure zero. We need the first partial derivatives of $F(x,y)$ for $x \neq 0$:

$$\frac{\partial F(x,y)}{\partial x} = \frac{xI - ((x+\mathbf{i})I - A)}{x^2} = \frac{A - \mathbf{i}yI}{x^2} \quad \text{and} \quad \frac{\partial F(x,y)}{\partial y} = \frac{\mathbf{i}I}{x}. \tag{3.3}$$

Let $(\hat{x}, \hat{y})$ be a point where the minimum singular value of $F(x,y)$ is both simple and nonzero, with associated left and right singular vectors u and v. By standard perturbation theory for

singular values it follows that

$$\nabla f(\hat{x},\hat{y}) = \operatorname{Re}\begin{bmatrix} u^* \frac{\partial F(x,y)}{\partial x} v \\ u^* \frac{\partial F(x,y)}{\partial y} v \end{bmatrix}. \tag{3.4}$$

To compute $\nabla^2 f(\hat{x},\hat{y})$, we will need the following result for the second partial derivatives of eigenvalues, which can be found in various forms, e.g. [Lan64, OW95].

Theorem 3.1. *Let $H(x,y)$ be a twice-differentiable $n \times n$ Hermitian matrix family for $x, y \in \mathbb{R}$. Given a point $(\hat{x},\hat{y})$, if $\lambda_1 \geq \ldots \geq \lambda_n$ are the eigenvalues of $H(\hat{x},\hat{y})$ with associated unit-norm eigenvectors $q_1, \ldots, q_n$, then*

$$\frac{\partial^2}{\partial x \partial y}\lambda_j \bigg|_{x=\hat{x},y=\hat{y}} = q_j^* \frac{\partial^2 H(\hat{x},\hat{y})}{\partial x \partial y} q_j + 2 \sum_{k \neq j} \frac{q_j^* \frac{\partial H(\hat{x},\hat{y})}{\partial x} q_k \cdot q_j^* \frac{\partial H(\hat{x},\hat{y})}{\partial y} q_k}{\lambda_j - \lambda_k}.$$

Since the minimum singular value of $F(x,y)$ is also the nth eigenvalue (in descending order) of the $2n \times 2n$ Hermitian matrix

$$H(x,y) = \begin{bmatrix} 0 & F(x,y) \\ F(x,y)^* & 0 \end{bmatrix},$$

$\nabla^2 f(\hat{x},\hat{y})$ can be computed by applying Theorem 3.1 to $H(x,y)$. Computationally, the necessary first and second partial derivatives of $H(x,y)$ can be obtained via the first partials given in (3.3) and the following second partial derivatives:

$$\frac{\partial^2 F(x,y)}{\partial x^2} = \frac{-2(A - \mathbf{i}yI)}{x^3}, \quad \frac{\partial^2 F(x,y)}{\partial y^2} = 0, \quad \text{and} \quad \frac{\partial^2 F(x,y)}{\partial x \partial y} = \frac{-\mathbf{i}I}{x^2}. \tag{3.5}$$

Although the full eigendecomposition of $H(\hat{x},\hat{y})$ is also necessary, it can actually be constructed more or less for free given the full SVD of $F(\hat{x},\hat{y})$. For brevity, we refer the reader to [BM17, Section 2.2] for these implementation details.

The cost of obtaining $f(\hat{x},\hat{y})$, its gradient, and its Hessian is $\mathcal{O}(n^3)$ since they can all be computed given the full SVD of $F(\hat{x},\hat{y})$. Provided $f(x,y)$ is sufficiently smooth about its stationary points, one can expect local quadratic convergence when using a Newton-based optimization method and superlinear convergence with a quasi-Newton method (forgoing the use of the Hessian). Note that scalable methods for computing smallest singular values, e.g. PROPACK [Lar], can also be used to compute $f(\hat{x},\hat{y})$ and its associated pair of left and right singular vectors in order to obtain $\nabla f(\hat{x},\hat{y})$. By using such a sparse solver in conjunction with a quasi-Newton method to solve (3.1), locally-optimal approximations to the Kreiss constants of large-scale matrices can also be efficiently obtained.

Although (3.1) is technically a constrained optimization problem, it suffices to simply return the value of $f(x,y)$ as ∞ whenever $x \leq 0$ in order to guarantee that an optimization solver always returns a point in the open right half-plane.

3.2 A globality certificate and restarting

Following [Gu00, Section 3.2], we now derive our globality certificate computation for testing whether a point (x,y) is a global minimizer of (3.2). As the "horizontal orientation" of looking for points (x,y) and $(x+\eta,y)$ to satisfy (2.1) for the distance to uncontrollability was arbitrary, here we will look for pairs of points that lie on lines of a given orientation. Specifically, given angle $\theta \in (-\frac{\pi}{2},\frac{\pi}{2}]$, we will look for pairs of points $(\hat{x},\hat{y})$ and $(\hat{x}+\eta\cos\theta, \hat{x}+\eta\sin\theta)$ such that $f(x,y) = \gamma$ holds at both of them and $\hat{x} > 0$.

Suppose γ is a singular value of both $F(x,y)$ and $F(x+\eta\cos\theta, y+\eta\sin\theta)$, with respective left and right singular vectors pairs u,v and $\hat{u}$,$\hat{v}$. Then

$$\left(I + \frac{\mathbf{i}y}{x}I - \frac{1}{x}A\right)v = \gamma u \qquad \left(I + \frac{\mathbf{i}(y+\eta\sin\theta)}{x+\eta\cos\theta}I - \frac{1}{x+\eta\cos\theta}A\right)\hat{v} = \gamma\hat{u} \tag{3.6a}$$

$$\left(I - \frac{\mathbf{i}y}{x}I - \frac{1}{x}A^*\right)u = \gamma v \qquad \left(I - \frac{\mathbf{i}(y+\eta\sin\theta)}{x+\eta\cos\theta}I - \frac{1}{x+\eta\cos\theta}A^*\right)\hat{u} = \gamma\hat{v}. \tag{3.6b}$$

Multiplying the left and right pair of equations respectively by x and $x+\eta\cos\theta$ and then rearranging terms yields:

$$(A - xI)\,v + \gamma x u = \mathbf{i}yv \qquad \left(A - (x+\eta\mathrm{e}^{\mathbf{i}\theta})I\right)\hat{v} + \gamma(x+\eta\cos\theta)\hat{u} = \mathbf{i}y\hat{v} \tag{3.7a}$$

$$(xI - A^*)\,u - \gamma x v = \mathbf{i}yu \qquad \left((x+\eta\mathrm{e}^{-\mathbf{i}\theta})I - A^*\right)\hat{u} - \gamma(x+\eta\cos\theta)\hat{v} = \mathbf{i}y\hat{u}. \tag{3.7b}$$

The two equations on the left can then be transformed into a standard eigenvalue problem, as can the two on the right, respectively:

$$\begin{bmatrix} xI - A^* & -\gamma xI \\ \gamma xI & A - xI \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{i}y\begin{bmatrix} u \\ v \end{bmatrix} \tag{3.8a}$$

$$\begin{bmatrix} (x+\eta\mathrm{e}^{-\mathbf{i}\theta})I - A^* & -\gamma(x+\eta\cos\theta)I \\ \gamma(x+\eta\cos\theta)I & A - (x+\eta\mathrm{e}^{\mathbf{i}\theta})I \end{bmatrix}\begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = \mathbf{i}y\begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix}. \tag{3.8b}$$

Let A_1 and A_2 denote the two matrices above and $W = \left[\begin{smallmatrix} u \\ v \end{smallmatrix}\right]\left[\begin{smallmatrix} \hat{u}^* & \hat{v}^* \end{smallmatrix}\right]$ so that we have $A_1 W = \mathbf{i}yW$ and $A_2 W^* = \mathbf{i}yW^*$. Taking the conjugate transpose of the second and then adding the two together yields the Sylvester equation:

$$\begin{bmatrix} xI - A^* & -\gamma xI \\ \gamma xI & A - xI \end{bmatrix} W + W \begin{bmatrix} (x+\eta\mathrm{e}^{\mathbf{i}\theta})I - A & \gamma(x+\eta\cos\theta)I \\ -\gamma(x+\eta\cos\theta)I & A^* - (x+\eta\mathrm{e}^{-\mathbf{i}\theta})I \end{bmatrix} = 0. \tag{3.9}$$

Hence the two eigenvalue problems in (3.8) share a common eigenvalue if (3.9) has a nonzero solution $W \in \mathbb{R}^{2n\times 2n}$. We now separate out all terms involving x to get

$$\left(\begin{bmatrix} -A^* & 0 \\ 0 & A \end{bmatrix} W + W \begin{bmatrix} \eta\mathrm{e}^{\mathbf{i}\theta}I - A & \gamma\eta\cos\theta I \\ -\gamma\eta\cos\theta I & A^* - \eta\mathrm{e}^{-\mathbf{i}\theta}I \end{bmatrix}\right) - x\left(\begin{bmatrix} -I & \gamma I \\ -\gamma I & I \end{bmatrix} W + W \begin{bmatrix} -I & -\gamma I \\ \gamma I & I \end{bmatrix}\right) = 0. \tag{3.10}$$

Rewriting both Sylvester forms using the vectorize operator, and letting $w = \mathrm{vec}(W)$, results in the generalized eigenvalue problem

$$\mathcal{A}_1 w = x\mathcal{A}_2 w, \tag{3.11}$$

where

$$\mathcal{A}_1 = I \otimes \begin{bmatrix} -A^* & 0 \\ 0 & A \end{bmatrix} + \begin{bmatrix} \eta\mathrm{e}^{\mathbf{i}\theta}I - A^{\mathsf{T}} & -\gamma\eta\cos\theta I \\ \gamma\eta\cos\theta I & \overline{A} - \eta\mathrm{e}^{-\mathbf{i}\theta}I \end{bmatrix} \otimes I \tag{3.12a}$$

$$\mathcal{A}_2 = I \otimes \begin{bmatrix} -I & \gamma I \\ -\gamma I & I \end{bmatrix} + \begin{bmatrix} -I & \gamma I \\ -\gamma I & I \end{bmatrix} \otimes I. \tag{3.12b}$$

Hence, all points $(\hat{x},\hat{y})$ such that $f(\hat{x},\hat{y}) = f(\hat{x}+\eta\cos\theta, \hat{y}+\eta\sin\theta) = \gamma$ can be computed as follows. First, all real and positive eigenvalues of the large $4n^2 \times 4n^2$ generalized eigenvalue problem in (3.11) must be computed. This first step is the dominant cost of the entire globality certificate and has $\mathcal{O}(n^6)$ work with a notably large constant if the explicit matrices $\mathcal{A}_1$ and $\mathcal{A}_2$ are simply passed to a solver such as the `eig` routine from MATLAB. Then, for each real and positive eigenvalue $\hat{x}$ of (3.11), the imaginary eigenvalues of the two $2n \times 2n$ standard eigenvalue

problems in (3.8) must be computed, to check that there is a match and in order to obtain the corresponding $\hat{y}$ value for that $\hat{x}$. Finally, the singular values of $F(\hat{x}, \hat{y})$ must be computed to check whether γ is the minimum singular value of $F(\hat{x}, \hat{y})$. If γ is indeed the minimum singular value of $F(\hat{x}, \hat{y})$, then another point on the same level has been obtained. Otherwise, a point on a lower level set has been computed.

Remark 3.2. *Unlike the $4n^2 \times 4n^2$ generalized eigenvalue problem* (3.11) *we have derived here, for the distance to uncontrollability Gu derived a smaller $2n^2 \times 2n^2$ generalized eigenvalue problem [Gu00, (3.13)]. In [GMO+06, §3.1], this was then simplified further to a computationally easier $2n^2 \times 2n^2$ simple eigenvalue problem [GMO+06, (3.7)]. However, due to the presence of the nonzero off-diagonal $\pm\gamma I$ blocks in $\mathcal{A}_2$, these additional simplifications do not appear to extend to the case of Kreiss constants. If one attempts to similarly partition W into four $n \times n$ blocks, multiplying out* (3.10) *for each block of W results in four equations that all have nonzero terms involving x. In the case of the distance to uncontrollability, the additional reductions are achieved by taking advantage of the fact that the eigenvalue x (α in the notation of [Gu00, GMO+06]) does not appear in the two equations for the off-diagonal blocks, since it ends up being multiplied by zero; this can be readily seen in [GMO+06, (3.5)]. Consequently, computing the eigenvalues of* (3.11) *is not only more expensive (by a constant factor), it is likely more numerically challenging since $\mathcal{A}_2$ is also singular.*

The computed globality certificate works as follows. Assuming one or more points $(\hat{x} + \eta\cos\theta, \hat{y} + \eta\sin\theta)$ is obtained, whether or not γ is the minimum singular value of $F(\hat{x}, \hat{y})$, it is clear that $\mathcal{K}(A)^{-1} \leq \gamma$. Furthermore, the test provides new points from from which an optimization solver can be restarted to find better (lower) minimizers. If no such points are returned, then the entire test is simply repeated for smaller and smaller values of η, until either a) it succeeds and so optimization is restarted or b) η becomes sufficiently small to indicate convergence to a global minimizer. As we will see in §4.2, if $\mathcal{K}(A)^{-1} < \gamma$, then there exists some $\eta_{\max} > 0$ such that the test must return points for any $\eta \in [0, \eta_{\max}]$.

3.3 Faster computation of the real eigenvalues of $\mathcal{A}_1 w = x\mathcal{A}_2 w$

We now adapt the divide-and-conquer approach from [GMO+06, Section 3.3.2] for computing the distance to uncontrollability to our setting of Kreiss constants. Under some assumptions about the computations, this faster approach obtains the real eigenvalues of (3.11) in $\mathcal{O}(n^4)$ work on average and $\mathcal{O}(n^5)$ in the worst case. The key point is that the large matrices defined in (3.12) arose from vectorizations of the two corresponding $2n \times 2n$ Sylvester forms in (3.10). As such, applying $\mathcal{A}_1$ and $\mathcal{A}_2$, or their inverses, can actually be done with just $\mathcal{O}(n^3)$ work, even though they are $4n^2 \times 4n^2$ in size. In turn, this means that for any $\nu \in \mathbb{C}$, $(\mathcal{A}_1 - \nu\mathcal{A}_2)^{-1}$ can be applied to a vector with $\mathcal{O}(n^3)$ work and so a shift-and-invert eigenvalue solver, such as the MATLAB routine `eigs`, can be employed to find the eigenvalues of (3.11) closest to ν also in $\mathcal{O}(n^3)$. Before we discuss the specifics of efficiently applying $(\mathcal{A}_1 - \nu\mathcal{A}_2)^{-1}$, we first give a high-level overview of the divide-and-conquer method adapted to computing $\mathcal{K}(A)$ and refer to [GMO+06, Algorithm 4] for the full details in the context of the distance to uncontrollability.

Let $D > 0$ be an upper bound on the real parts of all eigenvalues of (3.11). Then $L = 0$ and $U = D$ are respectively lower and upper bounds on all real-valued positive eigenvalues $\hat{x}$ of (3.11). The idea of divide-and-conquer is to recursively explore the interval $[L, U]$ on the real axis using a shift-and-invert eigensolver with real-valued shifts, in order to obtain all eigenvalues near this section of the real axis, and thus all the positive, real-valued eigenvalues of (3.11). The technique works as follows. Consider the real-valued shift $\nu = \frac{L+U}{2}$ and let λ be an eigenvalue of (3.11) that is closest to the chosen shift ν, with distance $\mu = |\lambda - \nu| \geq 0$. Assuming that a shift-and-invert eigenvalue solver indeed finds λ, which is very reasonable in practice, then $(\nu - \mu, \nu + \mu)$ contains no eigenvalues of (3.11). If $\operatorname{Im}\lambda = 0$, then a real-valued eigenvalue of (3.11) has been found, namely λ. In any case, a 2μ length section of the real axis has now

been "explored" and the same procedure can be applied recursively to the remaining unexplored portions of $[L, U]$ to the left and right of shift ν, respectively $[L, \nu - \mu]$ and $[\nu + \mu, U]$. In the special case that $\lambda = \nu$, the amount of the real axis searched is technically not reduced, as $\mu = 0$, but this is not an issue for the divide-and-conquer scheme as it eventually will search all of $[L, U]$ for real-valued eigenvalues of (3.11).

In [GMO+06, Theorem 3.2], it was proven that this divide-and-conquer method requires at most $2q+1$ "closest eigenvalue" computations, where q is the dimension of the eigenvalue problem, which thus gives a worst case complexity analysis of $\mathcal{O}(n^5)$. However, [GMO+06] also proves that on average the divide-and-conquer method only requires $\mathcal{O}(n^4)$ work, assuming that the eigenvalues are distributed uniformly. They also show that the cost of choosing D unnecessarily large only results in about four extra closest eigenvalue computations [GMO+06, p. 490].

For now, we assume that we have an a priori suitable initial upper bound $D > 0$ necessary to begin divide-and-conquer. We will also need to be able to use $\mathcal{A}_1$ and $\mathcal{A}_2$ for matrix-vector products efficiently. For $\mathcal{A}_2$, this is immediate by just storing the matrix in a sparse format, since it only has $10n^2$ nonzero entries. Given a vector $u \in \mathbb{C}^{4n^2}$, computing $\mathcal{A}_1 u$ can be done efficiently via

$$\operatorname{vec}\left(\begin{bmatrix} -A^* & 0 \\ 0 & A \end{bmatrix} U + U \begin{bmatrix} \eta \mathrm{e}^{\mathbf{i}\theta} I - A & \gamma\eta\cos\theta I \\ -\gamma\eta\cos\theta I & A^* - \eta \mathrm{e}^{-\mathbf{i}\theta} I \end{bmatrix}\right), \tag{3.13}$$

where $U \in \mathbb{C}^{2n\times 2n}$ and $u = \operatorname{vec}(U)$. In this form, the dominant cost in obtaining $\mathcal{A}_1 u$ is the two matrix multiplies of the $2n \times 2n$ matrices above, hence it is $\mathcal{O}(n^3)$ work.

To obtain $v = (\mathcal{A}_1 - \nu\mathcal{A}_2)^{-1} u$ efficiently, consider $(\mathcal{A}_1 - \nu\mathcal{A}_2)v = u$, which can be "unvectorized" into:

$$\begin{aligned}\left(\begin{bmatrix} -A^* & 0 \\ 0 & A \end{bmatrix} V + V \begin{bmatrix} \eta \mathrm{e}^{\mathbf{i}\theta} I - A & \gamma\eta\cos\theta I \\ -\gamma\eta\cos\theta I & A^* - \eta \mathrm{e}^{-\mathbf{i}\theta} I \end{bmatrix}\right) \\ - \nu\left(\begin{bmatrix} -I & \gamma I \\ -\gamma I & I \end{bmatrix} V + V \begin{bmatrix} -I & -\gamma I \\ \gamma I & I \end{bmatrix}\right) = U,\end{aligned} \tag{3.14}$$

where $U, V \in \mathbb{C}^{2n\times 2n}$ and $u = \operatorname{vec}(U)$ and $v = \operatorname{vec}(V)$. Combining terms to simplify yields this Sylvester equation:

$$\begin{bmatrix} -A^* + \nu I & -\gamma\nu I \\ \gamma\nu I & A - \nu I \end{bmatrix} V + V \begin{bmatrix} \eta \mathrm{e}^{\mathbf{i}\theta} I - A + \nu I & \gamma(\eta\cos\theta + \nu) I \\ -\gamma(\eta\cos\theta + \nu) I & A^* - \eta \mathrm{e}^{-\mathbf{i}\theta} I - \nu I \end{bmatrix} = U. \tag{3.15}$$

Thus $v = (\mathcal{A}_1 - \nu\mathcal{A}_2)^{-1} u$ can also be computed in $\mathcal{O}(n^3)$ time, by applying the vectorization of the solution V to the $2n \times 2n$ Sylvester equation above.

4 Discussion of alternative iterations and numerical accuracy

Naturally the globality certificate computation of §3.2 also immediately leads to a bisection algorithm for computing $\mathcal{K}(A)$ but this would not be of much practical use, since it would incur a far greater number of expensive globality certificate computations than our proposed method using optimization from §3. However, both are still susceptible to aforementioned numerical difficulties when performing the certificate tests for small values of η. This raises the questions of whether a) a trisection iteration is worth developing for computing $\mathcal{K}(A)$, or is even possible, and b) given fixed values of γ and η, if a meaningful (positively valued) lower bound similar to (2.2) can be derived for $\mathcal{K}(A)^{-1}$ when the globality certificate does not find any points satisfying its conditions.

4.1 Numerical accuracy of trisection for $\tau(A, B)$

We first present new results regarding the numerical accuracy of the trisection algorithm.

Theorem 4.1. *Suppose the trisection algorithm for the distance to uncontrollability terminates at the kth iterate with $|\tau(A,B) - \gamma_k| \leq \psi\tau(A,B)$ holding for some given relative error tolerance $\psi > 0$. Then $\eta_k \leq (1+\psi)\tau(A,B)$.*

Proof. By construction, $\gamma_k = L + \eta_k$ and $L \geq 0$ so $\eta_k \leq \gamma_k$ always holds. If $\gamma_k \geq \tau(A,B)$, then

$$\psi\tau(A,B) \geq |\tau(A,B) - \gamma_k| = \gamma_k - \tau(A,B) \quad \Longrightarrow \quad \eta_k \leq (1+\psi)\tau(A,B).$$

Otherwise, $\gamma_k < \tau(A,B)$ holds and

$$\psi\tau(A,B) \geq |\tau(A,B) - \gamma_k| = \tau(A,B) - \gamma_k > \gamma_k - \tau(A,B),$$

so the result holds. □

Corollary 4.2. *Given a fixed $\psi > 0$, if $\eta_k > (1+\psi)\tau(A,B)$, then the relative error at the kth iteration is bounded below by ψ, specifically $\frac{|\tau(A,B)-\gamma_k|}{\tau(A,B)} > \psi$.*

Though perhaps somewhat simple, Lemma 4.1 and Corollary 4.2 are rather illuminating about the numerical limitations of the trisection algorithm in practice. If $\tau(A,B)$ is close to zero, η_k will also have to be commensurately small in order to achieve any relative accuracy whatsoever, e.g. even if only one digit of accuracy is desired ($\psi = 0.1$). Thus it may be difficult to accurately compute $\tau(A,B)$ when it is small, as the globality certificate computations may be inaccurate for such small values of η. Furthermore, Lemma 4.1 and Corollary 4.2 would also apply to a would-be version of the trisection algorithm for computing Kreiss constants. In fact, such an algorithm would likely encounter numerical problems more frequently since $\mathcal{K}(A)^{-1} \ll 1$ often holds.

4.2 A lower bound for the globality certificate test for $\mathcal{K}(A)^{-1}$

We now give analogues of Theorem 2.2 and Corollary 2.3 adapted to the case of (3.2). However, instead of considering pairs of horizontally-oriented points, for (3.2) it will be simpler to consider pairs of points that are oriented vertically ($\theta = \frac{\pi}{2}$). Also, since the proof of Theorem 2.2 relies on the fact that (2.1) has a GLC of 1, while the GLC of (3.2) is not 1, the corresponding results we give here will be weaker. We first need the following fundamental result.

Lemma 4.3. *Let domain $\mathcal{D} \subseteq \mathbb{R}$ and $f : \mathcal{D} \to \mathbb{R}$ be continuous with a GLC of $c \geq 0$. If $f(\hat{x}) = f(\hat{x}+\eta) = \gamma$ holds for $\hat{x} \in \mathcal{D}$ and $\eta \geq 0$ so that $(\hat{x}+\eta) \in \mathcal{D}$, then*

$$\min_{x \in [\hat{x}, \hat{x}+\eta]} f(x) \geq \gamma - \tfrac{c\eta}{2}.$$

Proof. For $\eta = 0$ the statement is trivial so suppose $\eta > 0$ and $\exists \tilde{x} \in [\hat{x}, \hat{x}+\eta]$ such that $f(\tilde{x}) < \gamma - \frac{c\eta}{2}$ (otherwise the proof is complete). Thus $|f(\tilde{x}) - f(\hat{x})| > \frac{c\eta}{2}$. Without loss of generality, assume $\tilde{x} \in [\hat{x}, \hat{x} + \frac{\eta}{2}]$, so $|\tilde{x} - \hat{x}| \leq \frac{\eta}{2}$. Hence f has a local Lipschitz constant strictly greater than c, hence the GLC of f must also be strictly greater than c, a contradiction. □

Theorem 4.4. *Let $\gamma \in [0,1)$, $\eta \geq 0$ be given and $f(x,y)$ be as defined in (3.2) with $(x_\star, y_\star)$ being a global minimizer. If $\mathcal{K}(A)^{-1} \leq \gamma$ and $\eta \in [0, 2x_\star(\gamma - \mathcal{K}(A)^{-1})]$, then there exists a pair $x, y \in \mathbb{R}$ such that*

$$f(x,y) = f(x, y+\eta) = \gamma. \tag{4.1}$$

Corollary 4.5. *Let $\gamma \in [0,1)$, $\eta \geq 0$ be given and $f(x,y)$ be as defined in (3.2) with $(x_\star, y_\star)$ being a global minimizer. If there do not exist any pairs $x, y \in \mathbb{R}$ such that (4.1) holds, then*

$$\mathcal{K}(A)^{-1} > \gamma - \tfrac{\eta}{2x_\star}. \tag{4.2}$$

As the proof of Theorem 4.4 more or less follows the proof of Theorem 2.2, which is given in full detail in [Gu00, Theorem 3.1], we only sketch out the necessary modifications here.

First, note that for a fixed $y \in \mathbb{R}$, $\lim_{x\to\infty} f(x,y) = 1$ and for a fixed $x > 0$, $\lim_{y\to\pm\infty} f(x,y) = \infty$. Thus, since $\mathcal{K}(A)^{-1} \leq \gamma$ and crucially $\gamma < 1$, it follows that the level set for $f(x,y) = \gamma$ is comprised of a finite number of continuous closed algebraic curves in the right half-plane and that the global minimizer $(x_\star, y_\star)$ is in the interior of one of these curves. Since there may be more than one such curve around $(x_\star, y_\star)$, let $\mathcal{G}$ be the one enclosing the smallest area. By continuity, there exist two points $\mathcal{P}_1 = (x_\star, y_\star - \eta_1)$ and $\mathcal{P}_2 = (x_\star, y_\star + \eta_2)$ both on $\mathcal{G}$ with $\eta_1 > 0$ and $\eta_2 > 0$ and so

$$f(x_\star, y_\star - \eta_1) = f(x_\star, y_\star + \eta_2) = \gamma. \tag{4.3}$$

Furthermore, we can assume that $\mathcal{P}_1$ and $\mathcal{P}_2$ are the closest two points on $\mathcal{G}$ to $(x_\star, y_\star)$.

Second, since the numerator of $f(x,y)$ has a GLC of 1, it follows by standard differentiation that $|\frac{\partial}{\partial y} f(x,y)| \leq \frac{1}{x}$. Hence for $x_\star$, we have that $f(x_\star, y)$ with respect to y has a GLC of $\frac{1}{x_\star}$. Then it follows that

$$\eta_1 \geq x_\star(\gamma - \mathcal{K}(A)^{-1}) \quad \text{and} \quad \eta_2 \geq x_\star(\gamma - \mathcal{K}(A)^{-1}).$$

Hence, choose any $\eta \in [0, 2x_\star(\gamma - \mathcal{K}(A)^{-1})]$ and consider all the points on $\mathcal{G}$ shifted downward by η; we will call this shifted curve $\widehat{\mathcal{G}}$. Then point $\widehat{\mathcal{P}}_2 = (x_\star, y_\star + \eta_2 - \eta)$ must be on the line segment connecting $\mathcal{P}_1$ and $\mathcal{P}_2$ and hence $\mathcal{G}$ and $\widehat{\mathcal{G}}$ must intersect somewhere. Letting $(\tilde{x}, \tilde{y})$ be such an intersection point, then the pair $(\tilde{x}, \tilde{y})$ and $(\tilde{x}, \tilde{y} + \eta)$ satisfies (4.3). This completes the needed modifications to the proof.

Theorem 4.4 tells us that $\eta_{\max}$ from §3.2 is in fact equal to $2x_\star(\gamma - \mathcal{K}(A)^{-1})$ and is always positive for $\mathcal{K}(A)^{-1} < \gamma$. Hence, we know that the certification procedure must eventually return points as η gets smaller, unless $\gamma = \mathcal{K}(A)^{-1}$.

Unfortunately, the lower bound (4.2) depends on $x_\star$, which is unknown, and is only informative (positively valued) when $x_\star > \frac{\eta}{2\gamma}$. If $\mathcal{K}(A)^{-1}$ is close to zero, small values of γ will need to be tested but then the bound will likely be meaningless for a large portion of the right half-plane (as $\frac{\eta}{2\gamma}$ could be very large). Consequently, the certificate test cannot be used to maintain and update a lower bound when the test returns no points, which is in stark contrast to the trisection algorithm for the distance to uncontrollability.

Although a bit more complicated, a similar result to Corollary 4.5 can also be derived for $\theta = 0$ to show that the global certificate test when looking for horizontally-oriented pairs is also not compatible with trisection.

4.3 An alternative trisection-compatible certificate test for $\mathcal{K}(A)^{-1}$

The crux of why the certificate computation of §3.2 is not compatible with trisection is that when it fails to find any satisfying pairs of points, it does *not* imply that a lower bound of $\mathcal{K}(A)^{-1} > \gamma - \frac{\eta}{2}$ holds. However, it is possible to construct such a modified certificate computation, at least theoretically. The key idea is to cancel out the lower bound's dependence on $x_\star$ in (4.2); the certificate test should not detect pairs of points that are a constant distance apart, namely η, but rather vary the distance between vertically-oriented pairs with respect to x. Thus, for each given $\hat{x}$, looking for pairs of points that are $\hat{x}\eta$ apart causes the $x_\star$ in (4.2) to cancel out, which in turns gives the needed $\mathcal{K}(A)^{-1} > \gamma - \frac{\eta}{2}$ lower bound. The modified certificate procedure is as follows.

Suppose γ is both a singular value of $F(x,y)$ and $F(x, y + x\eta)$, with respective left and right singular vectors pairs u,v and $\hat{u}$,$\hat{v}$. Then

$$\left(I + \frac{\mathbf{i}y}{x} I - \frac{1}{x} A\right) v = \gamma u \qquad \left(I + \frac{\mathbf{i}(y + x\eta)}{x} I - \frac{1}{x} A\right) \hat{v} = \gamma \hat{u} \tag{4.4a}$$

$$\left(I - \frac{\mathbf{i}y}{x} I - \frac{1}{x} A^*\right) u = \gamma v \qquad \left(I - \frac{\mathbf{i}(y + x\eta)}{x} I - \frac{1}{x} A^*\right) \hat{u} = \gamma \hat{v}. \tag{4.4b}$$

Multiplying both pairs of equations by x and rearranging terms yields:

$$(A - xI)\, v + \gamma x u = \mathbf{i}y v \qquad (A - x(1+\mathbf{i}\eta)I)\, \hat{v} + \gamma x \hat{u} = \mathbf{i}y \hat{v} \tag{4.5a}$$

$$(xI - A^*)\, u - \gamma x v = \mathbf{i}y u \qquad (x(1-\mathbf{i}\eta)I - A^*)\, \hat{u} - \gamma x \hat{v} = \mathbf{i}y \hat{u}. \tag{4.5b}$$

The two equations on the left can then be transformed into a standard eigenvalue problem, as can the two on the right, respectively:

$$\begin{bmatrix} xI - A^* & -\gamma x I \\ \gamma x I & A - xI \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{i}y \begin{bmatrix} u \\ v \end{bmatrix} \tag{4.6a}$$

$$\begin{bmatrix} x(1-\mathbf{i}\eta)I - A^* & -\gamma x I \\ \gamma x I & A - x(1+\mathbf{i}\eta)I \end{bmatrix} \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = \mathbf{i}y \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix}. \tag{4.6b}$$

These two eigenvalue problems share a common eigenvalue if

$$\begin{bmatrix} xI - A^* & -\gamma x I \\ \gamma x I & A - xI \end{bmatrix} W + W \begin{bmatrix} x(1+\mathbf{i}\eta)I - A & \gamma x I \\ -\gamma x I & A^* - x(1-\mathbf{i}\eta)I \end{bmatrix} = 0$$

has a nonzero solution $W \in \mathbb{R}^{2n \times 2n}$. Separating out the terms involving x, we have

$$\left(\begin{bmatrix} -A^* & 0 \\ 0 & A \end{bmatrix} W + W \begin{bmatrix} -A & 0 \\ 0 & A^* \end{bmatrix} \right) - x \left(\begin{bmatrix} -I & \gamma I \\ -\gamma I & I \end{bmatrix} W + W \begin{bmatrix} -(1+\mathbf{i}\eta)I & -\gamma I \\ \gamma I & (1-\mathbf{i}\eta)I \end{bmatrix} \right) = 0. \tag{4.7}$$

Rewriting both Sylvester forms using the vectorize operator, and letting $w = \mathrm{vec}(W)$, we have the following generalized eigenvalue problem

$$\widetilde{\mathcal{A}}_1 w = x \widetilde{\mathcal{A}}_2 w, \tag{4.8}$$

where

$$\widetilde{\mathcal{A}}_1 = I \otimes \begin{bmatrix} -A^* & 0 \\ 0 & A \end{bmatrix} + \begin{bmatrix} -A^{\mathsf{T}} & 0 \\ 0 & A \end{bmatrix} \otimes I \tag{4.9a}$$

$$\widetilde{\mathcal{A}}_2 = I \otimes \begin{bmatrix} -I & \gamma I \\ -\gamma I & I \end{bmatrix} + \begin{bmatrix} -(1+\mathbf{i}\eta)I & \gamma I \\ -\gamma I & (1-\mathbf{i}\eta)I \end{bmatrix} \otimes I. \tag{4.9b}$$

The computation would then proceed by first computing the real-valued eigenvalues of (4.8), and then for each of those, computing the imaginary eigenvalues of the pair of eigenvalue problems in (4.8), and lastly computing the singular values of $F(x, y)$ to check whether γ is its minimum singular value.

Unfortunately, actually computing the eigenvalues of (4.8) seems to be much more difficult numerically than those of (3.11). We suspect this is because when η is small, as it will need to be per Lemma 4.1 and Corollary 4.2 for trisection (particularly if $\mathcal{K}(A)$ is large), (4.8) is very close to a singular matrix pencil and so its eigenvalues cannot be reliably obtained with any accuracy, which has been our experience in experiments. Given these practical limitations, we do not pursue this further. We now turn to adapting our algorithm and its components from §3 to compute the discrete-time variant of $\mathcal{K}(A)$.

5 The discrete-time case for $\mathcal{K}(A)$

To compute the discrete-time version of the Kreiss constant of a matrix, we will analogously work with the inverse of (1.3) and solve

$$\mathcal{K}(A)^{-1} = \inf_{r>1, \theta \in [0, 2\pi)} \sigma_{\min} \left(\frac{r\mathrm{e}^{\mathbf{i}\theta} I - A}{r - 1} \right), \tag{5.1}$$

now using polar coordinates. Correspondingly, for this section, we will consider

$$f(r,\theta) = \sigma_{\min}(F(r,\theta)) \quad \text{where} \quad F(r,\theta) = \frac{re^{\mathbf{i}\theta}I - A}{r-1}. \tag{5.2}$$

We assume that $\rho(A) \leq 1$ (since otherwise $\mathcal{K}(A) = \infty$) and that A is a nonnormal matrix (as otherwise $\mathcal{K}(A) = 1$). At a high-level, our algorithm from §3 will remain the same but the local optimization procedure and the globality certificate computations will both need to be adapted for the discrete-time case. It is also possible to obtain analogues of Theorem 4.4 and Corollary 4.5, by considering that $\lim_{r\to\infty} f(r,\theta) = 1$ for all $\theta \in [0, 2\pi)$, so a corresponding $\eta_{\max} > 0$ must also exist in the discrete-time case. Similarly, the initial optimization phase must find a point such that $f(r,\theta) = \gamma < 1$. Though the asymptotic work complexities will remain as before, as we will soon see, computing the globality certificate ends up involving a higher constant factor.

5.1 Adapting the gradient and Hessian for local optimization

To instead find minimizers of (5.2), outside of the unit disk, we will need the analogous gradients and Hessian of $f(r,\theta)$. For brevity, we just provide the first and second partial derivatives of $F(r,\theta)$ for $r \neq 1$ here, which are respectively

$$\frac{\partial F(r,\theta)}{\partial r} = \frac{(r-1)e^{\mathbf{i}\theta}I - (re^{\mathbf{i}\theta}I - A)}{(r-1)^2} = \frac{A - e^{\mathbf{i}\theta}I}{(r-1)^2} \quad \text{and} \quad \frac{\partial F(r,\theta)}{\partial \theta} = \frac{\mathbf{i}re^{\mathbf{i}\theta}I}{r-1}, \tag{5.3}$$

and

$$\frac{\partial^2 F(r,\theta)}{\partial r^2} = \frac{-2(A - e^{\mathbf{i}\theta}I)}{(r-1)^3}, \quad \frac{\partial^2 F(r,\theta)}{\partial \theta^2} = \frac{-re^{\mathbf{i}\theta}I}{r-1}, \quad \text{and} \quad \frac{\partial^2 F(r,\theta)}{\partial r \partial \theta} = \frac{-\mathbf{i}e^{\mathbf{i}\theta}I}{(r-1)^2}. \tag{5.4}$$

The costs to compute $f(r,\theta)$ along with its gradient and Hessian are no different in the discrete-time case and a scalable method for smallest singular values can be used to obtain $f(r,\theta)$ and its gradient, in order to employ a quasi-Newton method for computing the Kreiss constants of large-scale discrete-time matrices. To guarantee finding solutions to (5.1) that are outside the unit disk, we simply return the value of $f(r,\theta)$ as ∞ whenever $r \leq 1$.

5.2 A globality certificate using radial segments

In the discrete-time case, we will again look for pairs of points a distance $\eta \geq 0$ apart but now along rays from the origin (as opposed to lines of fixed angular orientation). Given $\gamma \in [0, 1)$, we wish to find all pairs of $\hat{r} > 1$ and $\hat{\theta} \in [0, 2\pi)$ such that $f(\hat{r}, \hat{\theta}) = f(\hat{r} + \eta, e^{\mathbf{i}\hat{\theta}}) = \gamma$.

Suppose γ is a singular value of $F(r + \eta, \theta)$ with respective left and right singular vectors $\hat{u}$ and $\hat{v}$. Then

$$\left(\frac{(r+\eta)e^{\mathbf{i}\theta}I - A}{r+\eta-1}\right)\hat{v} = \gamma\hat{u} \tag{5.5a}$$

$$\left(\frac{(r+\eta)e^{-\mathbf{i}\theta}I - A^*}{r+\eta-1}\right)\hat{u} = \gamma\hat{v}. \tag{5.5b}$$

First multiplying both equations by $r + \eta - 1$ and the bottom one also by $e^{\mathbf{i}\theta}$, a rearrangement of terms yields:

$$\gamma(r+\eta-1)\hat{u} + A\hat{v} = e^{\mathbf{i}\theta}(r+\eta)\hat{v} \tag{5.6a}$$

$$(r+\eta)\hat{u} = e^{\mathbf{i}\theta}\left(A^*\hat{u} + \gamma(r+\eta-1)\hat{v}\right). \tag{5.6b}$$

Thus, we have the following generalized eigenvalue problem:

$$\begin{bmatrix} \gamma(r+\eta-1)I & A \\ (r+\eta)I & 0 \end{bmatrix} \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = \mathrm{e}^{\mathbf{i}\theta} \begin{bmatrix} 0 & (r+\eta)I \\ A^* & \gamma(r+\eta-1)I \end{bmatrix} \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix}, \tag{5.7}$$

where we will denote the matrix on the left by $\widetilde{M}$ and the matrix on the right by $\widetilde{N}$. For $\eta = 0$, similarly supposing that γ is a singular value of $F(r,\theta)$ with right and left singular vectors u and v leads to this second generalized eigenvalue problem:

$$\begin{bmatrix} \gamma(r-1)I & A \\ rI & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \mathrm{e}^{\mathbf{i}\theta} \begin{bmatrix} 0 & rI \\ A^* & \gamma(r-1)I \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}, \tag{5.8}$$

where we will denote the matrix on the left by M and the matrix on the right by N.

Now define $X = \left[\begin{smallmatrix} u \\ v \end{smallmatrix}\right]\left[\begin{smallmatrix} \hat{u}^* & \hat{v}^* \end{smallmatrix}\right]$. Multiplying the two equations above from the right side, respectively by $\left[\begin{smallmatrix} u^* & v^* \end{smallmatrix}\right]N^*$ and $\left[\begin{smallmatrix} \hat{u}^* & \hat{v}^* \end{smallmatrix}\right]\widetilde{M}^*$, yields

$$MX\widetilde{M}^* = \mathrm{e}^{\mathbf{i}\theta} NX\widetilde{M}^* \tag{5.9}$$

and

$$\widetilde{M}X^*N^* = \mathrm{e}^{\mathbf{i}\theta}\widetilde{N}X^*N^*. \tag{5.10}$$

If we then take the conjugate transpose of (5.10) and multiply it by $\mathrm{e}^{\mathbf{i}\theta}$, we obtain

$$NX\widetilde{N}^* = \mathrm{e}^{\mathbf{i}\theta} NX\widetilde{M}^*. \tag{5.11}$$

Thus if there exists a nonzero X that is in neither the right null space of N nor the left null space of $\widetilde{M}^*$ and solves

$$MX\widetilde{M}^* - NX\widetilde{N}^* = 0, \tag{5.12}$$

the two generalized eigenvalue problems must share an eigenvalue $\mathrm{e}^{\mathbf{i}\theta}$.

We now want to separate out the r terms of $MX\widetilde{M}^*$ and $NX\widetilde{N}^*$. First, we have:

$$\widetilde{M}^* = \begin{bmatrix} \gamma(r+\eta-1)I & (r+\eta)I \\ A^* & 0 \end{bmatrix} \quad \text{and} \quad \widetilde{N}^* = \begin{bmatrix} 0 & A \\ (r+\eta)I & \gamma(r+\eta-1)I \end{bmatrix}. \tag{5.13}$$

Then $MX\widetilde{M}^*$ is

$$\left(\begin{bmatrix} -\gamma I & A \\ 0 & 0 \end{bmatrix} + r \begin{bmatrix} \gamma I & 0 \\ I & 0 \end{bmatrix} \right) X \left(\begin{bmatrix} \gamma(\eta-1)I & \eta I \\ A^* & 0 \end{bmatrix} + r \begin{bmatrix} \gamma I & I \\ 0 & 0 \end{bmatrix} \right), \tag{5.14}$$

which is equal to

$$\begin{aligned} &\left[\begin{smallmatrix} -\gamma I & A \\ 0 & 0 \end{smallmatrix}\right] X \left[\begin{smallmatrix} \gamma(\eta-1)I & \eta I \\ A^* & 0 \end{smallmatrix}\right] + \\ &\qquad r\left(\left[\begin{smallmatrix} -\gamma I & A \\ 0 & 0 \end{smallmatrix}\right] X \left[\begin{smallmatrix} \gamma I & I \\ 0 & 0 \end{smallmatrix}\right] + \left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] X \left[\begin{smallmatrix} \gamma(\eta-1)I & \eta I \\ A^* & 0 \end{smallmatrix}\right] \right) + r^2 \left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] X \left[\begin{smallmatrix} \gamma I & I \\ 0 & 0 \end{smallmatrix}\right]. \end{aligned} \tag{5.15}$$

Vectorizing the above equation, with $x = \mathrm{vec}(X)$, yields

$$\begin{aligned} &\left[\begin{smallmatrix} \gamma(\eta-1)I & \overline{A} \\ \eta I & 0 \end{smallmatrix}\right] \otimes \left[\begin{smallmatrix} -\gamma I & A \\ 0 & 0 \end{smallmatrix}\right] x + \\ &\qquad r\left(\left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] \otimes \left[\begin{smallmatrix} -\gamma I & A \\ 0 & 0 \end{smallmatrix}\right] + \left[\begin{smallmatrix} \gamma(\eta-1)I & \overline{A} \\ \eta I & 0 \end{smallmatrix}\right] \otimes \left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] \right) x + r^2 \left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] \otimes \left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] x, \end{aligned} \tag{5.16}$$

which we will abbreviate as

$$\mathcal{M}_0 x + r\mathcal{M}_1 x + r^2 \mathcal{M}_2 x. \tag{5.17}$$

Likewise, $NX\widetilde{N}^*$ is

$$\left(\begin{bmatrix} 0 & 0 \\ A^* & -\gamma I \end{bmatrix} + r \begin{bmatrix} 0 & I \\ 0 & \gamma I \end{bmatrix}\right) X \left(\begin{bmatrix} 0 & A \\ \eta I & \gamma(\eta-1)I \end{bmatrix} + r \begin{bmatrix} 0 & 0 \\ I & \gamma I \end{bmatrix}\right), \tag{5.18}$$

which is equal to

$$\begin{bmatrix} 0 & 0 \\ A^* & -\gamma I \end{bmatrix} X \begin{bmatrix} 0 & A \\ \eta I & \gamma(\eta-1)I \end{bmatrix} + r\left(\begin{bmatrix} 0 & 0 \\ A^* & -\gamma I \end{bmatrix} X \begin{bmatrix} 0 & 0 \\ I & \gamma I \end{bmatrix} + \begin{bmatrix} 0 & I \\ 0 & \gamma I \end{bmatrix} X \begin{bmatrix} 0 & A \\ \eta I & \gamma(\eta-1)I \end{bmatrix}\right) + r^2 \begin{bmatrix} 0 & I \\ 0 & \gamma I \end{bmatrix} X \begin{bmatrix} 0 & 0 \\ I & \gamma I \end{bmatrix}. \tag{5.19}$$

Similarly vectorizing this gives

$$\begin{bmatrix} 0 & \eta I \\ A^{\mathsf{T}} & \gamma(\eta-1)I \end{bmatrix} \otimes \begin{bmatrix} 0 & 0 \\ A^* & -\gamma I \end{bmatrix} x + r\left(\begin{bmatrix} 0 & I \\ 0 & \gamma I \end{bmatrix} \otimes \begin{bmatrix} 0 & 0 \\ A^* & -\gamma I \end{bmatrix} + \begin{bmatrix} 0 & \eta I \\ A^{\mathsf{T}} & \gamma(\eta-1)I \end{bmatrix} \otimes \begin{bmatrix} 0 & I \\ 0 & \gamma I \end{bmatrix}\right) x + r^2 \begin{bmatrix} 0 & I \\ 0 & \gamma I \end{bmatrix} \otimes \begin{bmatrix} 0 & I \\ 0 & \gamma I \end{bmatrix} x, \tag{5.20}$$

which we will abbreviate as

$$\mathcal{N}_0 x + r\mathcal{N}_1 x + r^2 \mathcal{N}_2 x. \tag{5.21}$$

Thus, we finally have the following quadratic eigenvalue problem:

$$(\mathcal{M}_0 - \mathcal{N}_0)\, x + r\,(\mathcal{M}_1 - \mathcal{N}_1)\, x + r^2\,(\mathcal{M}_2 - \mathcal{N}_2)\, x = 0, \tag{5.22}$$

which we will abbreviate as:

$$\mathcal{P}_0 x + r\mathcal{P}_1 x + r^2 \mathcal{P}_2 x = 0. \tag{5.23}$$

Hence, to find all points $(\hat{r}, \hat{\theta})$ such that $f(\hat{r}, \hat{\theta}) = f(\hat{r} + \eta, \hat{\theta}) = \gamma$, we first compute all the real-valued eigenvalues r with $r > 1$ of the quadratic eigenvalue problem given in (5.23). While negative radii with $r < -(1 + \eta)$ are also permissible, they can be ignored since the associated points are also associated with positive radii $\hat{r} = |r| - \eta$ in the opposite direction ($\hat{\theta} = \theta + \pi$). Then, for each candidate radius $\hat{r} > 1$, the eigenvalues of the two generalized eigenvalue problems respectively given in (5.7) and (5.8) must be computed in order to check if they share a common unimodular eigenvalue $\mathrm{e}^{\mathbf{i}\hat{\theta}}$. Finally, the singular values of $F(\hat{r}, \hat{\theta})$ must be computed, to check whether γ is the minimum singular value of $F(\hat{r}, \hat{\theta})$. Since there also exists a corresponding $\eta_{\max} > 0$ for the discrete-time case, the certificate computation is repeated for smaller and smaller values of η until either it returns points so optimization can be restarted or η is sufficiently small to indicate global convergence.

The asymptotic work complexity for directly computing the discrete-time certificate is also $\mathcal{O}(n^6)$, e.g. if the explicit matrices of (5.23) are just passed to a eigensolver like `polyeig` from MATLAB. Since (5.23) is quadratic, the hidden constant factor is now even higher than in the continuous-time version.

Remark 5.1. *It is also possible to derive a globality certificate based on arcs instead of radial segments, where we try to find pairs of points (r, θ) and $(r, \theta + \eta)$. By a similar derivation as shown here, this too results in a quadratic eigenvalue problem, but the crucial difference is that then neither of the corresponding $\mathcal{P}_0$ and $\mathcal{P}_2$ matrices have full rank. As a result, this alternative quadratic eigenvalue problem is not well-posed and cannot be reliably solved in practice. Fortunately, this issue does not occur in the above globality certificate procedure using radial segments.*

5.3 Faster computation of the real eigenvalues of $\mathcal{P}_0 x + r\mathcal{P}_1 x + r^2 \mathcal{P}_2 x = 0$

We now show how the real eigenvalues of (5.23) may be computed using divide-and-conquer. As we will first linearize this quadratic eigenvalue problem, the hidden constant factor in the work

complexity will be larger; we make use of the companion linearization of (5.23):

$$\begin{bmatrix} \mathcal{P}_1 & \mathcal{P}_0 \\ -I & 0 \end{bmatrix} z = r \begin{bmatrix} -\mathcal{P}_2 & 0 \\ 0 & -I \end{bmatrix} z, \tag{5.24}$$

where $z = \left[\begin{smallmatrix} rx \\ x \end{smallmatrix}\right]$. Assuming an a priori upper bound $D > 0$ is known for for all real eigenvalues of (5.23), divide-and-conquer will sweep the interval $[1, D]$ to find all the real-valued eigenvalues in this range. We now detail how matrix-vector products and applying shift-and-invert with the matrices in (5.24) can all be done in at most $\mathcal{O}(n^3)$ work.

Consider doing matrix-vector products with either matrix in (5.24) to a partitioned vector $u = \left[\begin{smallmatrix} u_1 \\ u_2 \end{smallmatrix}\right]$. The nontrivial parts of these products are:

$$\mathcal{P}_0 u_2 = \mathcal{M}_0 u_2 - \mathcal{N}_0 u_2 \tag{5.25a}$$

$$\mathcal{P}_1 u_1 = \mathcal{M}_1 u_1 - \mathcal{N}_1 u_1 \tag{5.25b}$$

$$\mathcal{P}_2 u_1 = \mathcal{M}_2 u_1 - \mathcal{N}_2 u_1, \tag{5.25c}$$

which are equal to the following respective vectorizations:

$$\mathcal{P}_0 u_2 = \operatorname{vec}\left(\left[\begin{smallmatrix} -\gamma I & A \\ 0 & 0 \end{smallmatrix}\right] U_2 \left[\begin{smallmatrix} \gamma(\eta-1)I & \eta I \\ A^* & 0 \end{smallmatrix}\right] - \left[\begin{smallmatrix} 0 & 0 \\ A^* & -\gamma I \end{smallmatrix}\right] U_2 \left[\begin{smallmatrix} 0 & A \\ \eta I & \gamma(\eta-1)I \end{smallmatrix}\right] \right) \tag{5.26a}$$

$$\begin{aligned} \mathcal{P}_1 u_1 = \operatorname{vec}\Big(&\left[\begin{smallmatrix} -\gamma I & A \\ 0 & 0 \end{smallmatrix}\right] U_1 \left[\begin{smallmatrix} \gamma I & I \\ 0 & 0 \end{smallmatrix}\right] + \left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] U_1 \left[\begin{smallmatrix} \gamma(\eta-1)I & \eta I \\ A^* & 0 \end{smallmatrix}\right] \\ &- \left[\begin{smallmatrix} 0 & 0 \\ A^* & -\gamma I \end{smallmatrix}\right] U_1 \left[\begin{smallmatrix} 0 & 0 \\ I & \gamma I \end{smallmatrix}\right] - \left[\begin{smallmatrix} 0 & I \\ 0 & \gamma I \end{smallmatrix}\right] U_1 \left[\begin{smallmatrix} 0 & A \\ \eta I & \gamma(\eta-1)I \end{smallmatrix}\right] \Big) \end{aligned} \tag{5.26b}$$

$$\mathcal{P}_2 u_1 = \operatorname{vec}\left(\left[\begin{smallmatrix} \gamma I & 0 \\ I & 0 \end{smallmatrix}\right] U_1 \left[\begin{smallmatrix} \gamma I & I \\ 0 & 0 \end{smallmatrix}\right] - \left[\begin{smallmatrix} 0 & I \\ 0 & \gamma I \end{smallmatrix}\right] U_1 \left[\begin{smallmatrix} 0 & 0 \\ I & \gamma I \end{smallmatrix}\right] \right), \tag{5.26c}$$

where $u_1 = \operatorname{vec}(U_1)$ and $u_2 = \operatorname{vec}(U_2)$. The first two of these can be obtained in $\mathcal{O}(n^3)$ work since they only involve matrix-matrix products with $2n \times 2n$ matrices. The third can be obtained in $\mathcal{O}(n^2)$ as the number of nonzero entries in $\mathcal{P}_2$ is simply $8n^2$. Hence, this particular matrix-vector product can be done efficiently just by storing $\mathcal{P}_2$ in a sparse format. This is also fortunate since for applying shift-and-invert to a generalized eigenvalue problem $Ax = \lambda Bx$, `eigs` currently only allows A to be provided implicitly as a function handle while B must be provided explicitly.

Given a shift $\nu \in \mathbb{C}$ and a partitioned vector $u = \left[\begin{smallmatrix} u_1 \\ u_2 \end{smallmatrix}\right]$, we now focus on efficiently obtaining

$$v = \left(\begin{bmatrix} \mathcal{P}_1 & \mathcal{P}_0 \\ -I & 0 \end{bmatrix} - \nu \begin{bmatrix} -\mathcal{P}_2 & 0 \\ 0 & -I \end{bmatrix} \right)^{-1} u. \tag{5.27}$$

Also partitioning $v = \left[\begin{smallmatrix} v_1 \\ v_2 \end{smallmatrix}\right]$ and then grouping terms, we have that

$$\left(\begin{bmatrix} \mathcal{P}_1 & \mathcal{P}_0 \\ -I & 0 \end{bmatrix} - \nu \begin{bmatrix} -\mathcal{P}_2 & 0 \\ 0 & -I \end{bmatrix} \right) \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \mathcal{P}_1 + \nu\mathcal{P}_2 & \mathcal{P}_0 \\ -I & \nu I \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{5.28}$$

and so the bottom block row provides:

$$v_2 = \frac{u_2 + v_1}{\nu}. \tag{5.29}$$

Substituting (5.29) into the top block row of (5.28) and then multiplying by ν, we get

$$\mathcal{P}_0 v_1 + \nu\mathcal{P}_1 v_1 + \nu^2 \mathcal{P}_2 v_1 = \nu u_1 - \mathcal{P}_0 u_2. \tag{5.30}$$

We can obtain $\hat{u} = \nu u_1 - \mathcal{P}_0 u_2$ efficiently by using (5.26a) to do the matrix-vector product $\mathcal{P}_0 u_2$. Thus v_1 can be obtained by solving

$$\mathcal{P}_0 v_1 + \nu\mathcal{P}_1 v_1 + \nu^2 \mathcal{P}_2 v_1 = \hat{u}, \tag{5.31}$$

which in turn can be solved via the matrix equation

$$M V_1 \widetilde{M}^* - N V_1 \widetilde{N}^* = \widehat{U}, \tag{5.32}$$

where $v_1 = \operatorname{vec}(V_1)$, $\hat{u} = \operatorname{vec}(\widehat{U})$, and the matrix pairs M, N and $\widetilde{M}^*$, $\widetilde{N}^*$ are respectively given in (5.8) and (5.13), all with r set to ν. As (5.32) is a generalized continuous-time algebraic Sylvester equation, it can be solved in $\mathcal{O}(n^3)$ work [KW89].

6 Numerical experiments

To validate our methods for computing Kreiss constants, we have done some basic numerical evaluations by implementing proof-of-concept codes in MATLAB. Note that these prototypes have been written and tuned not for efficiency but to instead better demonstrate the globality certificates and restarts in action. We plan to add "production-ready" implementations, optimized for performance, to a future release of the open-source software package ROSTAPACK: RObust STAbility PACKage [Mit], which is implemented in MATLAB and licensed under the AGPL.

All experiments were done in MATLAB R2017b on a laptop with an Intel i7-6567U dual-core CPU laptop, 16GB of RAM, and macOS v10.14. Our setup for the continuous-time case was as follows. For finding minimizers of (3.2), we used `fminunc` from Optimization Toolbox in MATLAB, providing it with both the gradients and Hessians derived in §3.1. For asserting globality or obtaining new points from which to restart `fminunc`, our code solved all instances of the large $4n^2 \times 4n^2$ eigenvalue problem (3.11) by giving the explicit matrices to `eig`; the fixed parameter θ appearing in the $\mathcal{A}_1$ matrix was set to zero. We also used `eig` to find the imaginary eigenvalues of the two $2n \times 2n$ eigenvalue problems in (3.8), but we note that it would be more robust to compute these imaginary eigenvalues using the structure-preserving eigenvalue solvers of [BBMX02], which are available in SLICOT. When restarting optimization, i.e., when the globality certificate computation returns new starting points, we ran `fminunc` from these points one at a time until a minimizer was found such that the relative improvement from the previous best estimate for $\mathcal{K}(A)^{-1}$ was at least 10^{-6}. This choice was made solely to increase the likelihood of showing multiple restarts working in practice. As checking globality is by far the most expensive operation, it would likely be more efficient to run optimization from all of the new points in order to increase the chance of finding a global minimizer on every iteration. Similarly, while it would probably be more efficient to initialize the method from multiple points, we only used a single starting point intentionally chosen so that `fminunc` would not find a global minimizer and so at least one restart would be required.

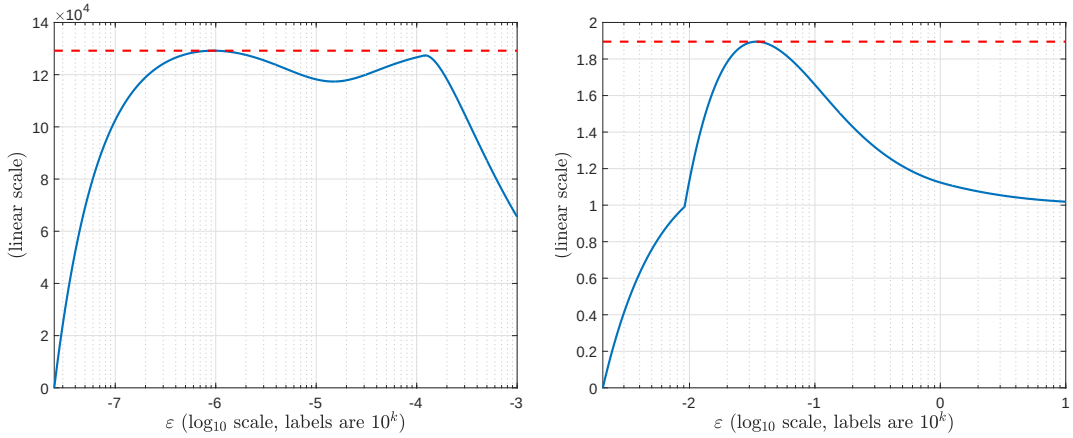
We used this same general configuration and (intentionally poor) initialization for the discrete-time case, except for the necessary changes outlined in §5. We again used `fminunc` with gradient and Hessian information to find minimizers, now for (5.2). For the discrete-time globality certificate, the eigenvalues of the $4n^2 \times 4n^2$ quadratic eigenvalue problem given in (5.23) were computed by providing the explicit matrices to `polyeig`.

We will address using divide-and-conquer to obtain the positive real eigenvalues of the large $4n^2 \times 4n^2$ eigenvalue problems separately in §6.3.

6.1 A continuous-time example

We begin with a continuous-time example based on `companion_demo` from EigTool [Wri02]. Letting $B =$ `companion_demo(10)`, which is not a stable matrix, we used $A = B - \kappa I$ as a 10×10 test example, where $\kappa = 1.001\alpha(B)$ and thus A is stable. This matrix has a large Kreiss constant, and the one-dimensional maximization problem given in (1.9) for the value of $\mathcal{K}(A)$ has two local maximizers; see Figure 1a. This plot was produced with Chebfun [DHT14] and the `specValSet` routine from ROSTAPACK v2.1 to compute $\alpha_\varepsilon(A)$ using the improved criss-cross method of [BM17]. Creating the Chebfun interpolant on this interval required about 62.7 seconds. However, in practice, there is generally additional cost in finding an interval which contains the global maximizer in the first place and there is no guarantee one will actually find such an interval. If one creates a surface plot (not shown here) of (3.2), it can be seen that the two-dimensional minimization problem given in (3.1) for the value of $\mathcal{K}(A)^{-1}$ has two local minimizers in the closed upper half-place.

In Figure 2, a contour plot of the level sets of (3.2) is given; the iterates for each run of optimization are also depicted. With our initial point choice of $(x, y) = (6, 6)$, i.e., the complex



(a) $\frac{\alpha_\varepsilon(A)}{\varepsilon}$ (b) $\frac{\rho_\varepsilon(A)-1}{\varepsilon}$

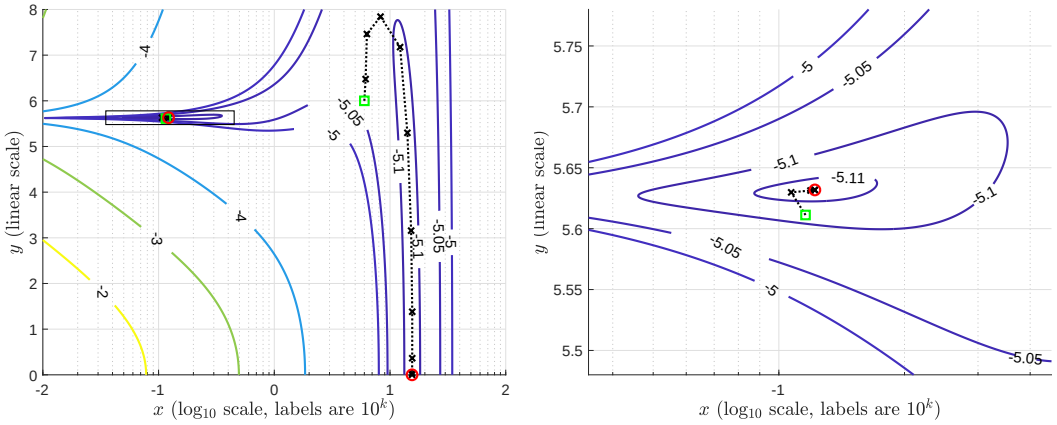
Figure 1: The solid curves plot $\frac{\alpha_\varepsilon(A)}{\varepsilon}$ and $\frac{\rho_\varepsilon(A)-1}{\varepsilon}$ as they vary with ε for our continuous- and discrete-time test examples, respectively. The dashed horizontal lines show the respective values of $\mathcal{K}(A)$ computed by our new method.

number $6+6\mathbf{i}$, optimization first finds a local minimizer on the real axis. A single certificate computation provides a new starting point near the global minimizer such that after this single restart, optimization finds the global minimizer. In total, these two runs of optimization amounted to 24 evaluations of (3.2). The two certification computations, one for the restart and another to assert that the second minimizer was indeed a global minimizer, resulted in the eigenvalues of 14 different instances of (3.11) being computed. Recall that when the certificate test returns no points, the computation is repeated for decreasing values of η, until it either does return one or more new starting points or η falls a below a tolerance, indicating the method has found a global minimizer of (3.2) and thus (3.1) has been computed. The total running time of our code was approximately 1.67 seconds. Since our code is only a prototype and was set to encourage restarts for the purpose of demonstration, a code developed and tuned for general use is likely to be faster.

Our code returned $1.291867070207492\times10^5$ for the value of $\mathcal{K}(A)$, which had a high agreement with the value produced by taking the max of the Chebfun interpolant used to created Figure 1a; the relative error compared to the Chebfun-derived value was 1.15×10^{-10}, with our method returning the higher value for $\mathcal{K}(A)$. While our method was many times faster than using Chebfun on this 10×10 example, the $\mathcal{O}(n^6)$ work required to assert globality (when using `eig`) means that Chebfun may actually be the more efficient choice for larger problems, since computing the pseudospectral abscissa is $\mathcal{O}(n^3)$ work. On the other hand, our method to compute $\mathcal{K}(A)$ is an unsupervised one with guaranteed convergence, while computing $\mathcal{K}(A)$ using Chebfun is generally an interactive process guided by a user with no guarantee that $\mathcal{K}(A)$ will be found.

6.2 A discrete-time example

For a discrete-time test example, we chose $A=\frac{1}{13}B+\frac{11}{10}I$, where $B=$ `convdiff_demo(11)` is a 10×10 matrix, also from EigTool. We picked this particular stable matrix A in order to illustrate our method incurring more than a single restart, even though it has a relatively small Kreiss constant. By construction, matrix A has several eigenvalues very close to the unit circle. Consequently, the two-dimensional minimization problem given in (5.1) for the value of $\mathcal{K}(A)^{-1}$ has several minimizers; this can be easily seen if one makes a surface plot (not shown) of (5.2). In contrast, the one-dimensional maximization problem given in (1.4) for the value of $\mathcal{K}(A)$ only



(a) Iterates (b) Iterates enlarged

Figure 2: The left plot shows a contour plot of the level sets (in $\log_{10}$ scale) of (3.2) for our continuous-time example; note that the x-axis is also in $\log_{10}$ scale. As this example is a real matrix, we only show the upper half of the complex plane. The iterates of optimization are shown connected by black dotted lines, with the initial, intermediate, and final points respectively denoted by a green square, black x's, and a red circle. Optimization begins at $(x, y) = (6, 6)$ and finds a local minimizer of (3.2) on the real axis. The certificate test provides a new starting point in the upper left quadrant of the plot, from which optimization then finds the global maximizer of (3.2). The right plot shows a close-up view of the denoted rectangular region surrounding the global maximizer.

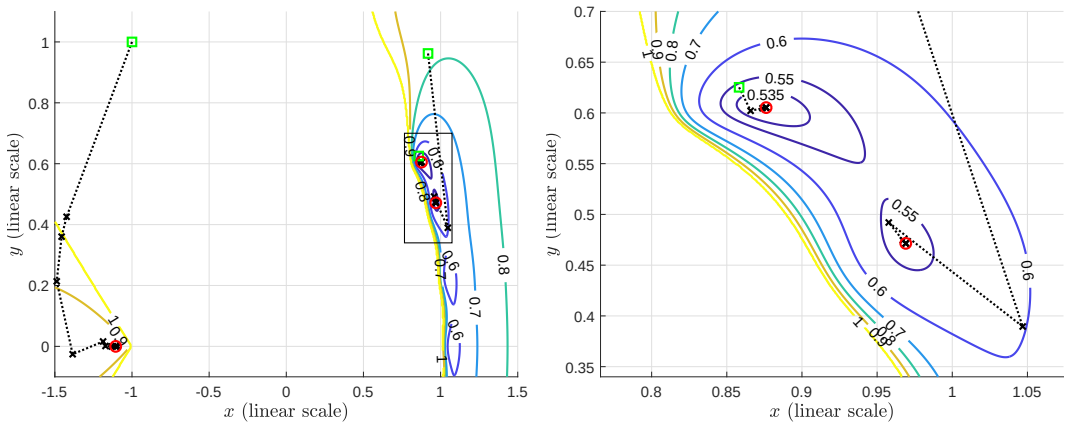
has a single maximizer and appears to be nonsmooth (though not at the maximizer), as shown in Figure 1b. Generating the Chebfun interpolant on the interval for this plot took about 50.2 seconds; we again used `specValSet` since it can also be configured to compute $\rho_\varepsilon(A)$ using the improved pseudospectral radius method of [BM17].

A contour plot of the level sets of (5.2) and the optimization iterates are shown in Figure 3. Initialized from $(r, \theta) = (\sqrt{2}, \frac{3}{4}\pi)$, i.e., the complex number $-1+1\mathbf{i}$, `fminunc` converged to a local minimizer instead of a global one. On the first restart, optimization then found a better (lower) but still local minimizer. However, this was only because of our "demonstration" configuration for the code; if optimization had been allowed to run from all the starting points returned by the first certificate computation, `fminunc` would have found the global minimizer on this first restart. A second certificate computation was then done, which produced new initial points for yet another round of optimization. With this second restart, optimization then found the global minimizer of (5.2), thus computing (5.1), and the final certificate computations were done to assert globality. The two restarts plus the final globality check entailed computing the eigenvalues of 15 different instances of the $4n^2 \times 4n^2$ quadratic eigenvalue problem given in (5.23). The code ran `fminunc` from three different starting points, including our initial one, which amounted to a total of 33 evaluations of (5.2). The total running time was approximately 6.91 seconds.

Our code returned 1.895013390905803 as the value of $\mathcal{K}(A)$ for our discrete-time example; compared to taking the max of the Chebfun used to create Figure 1b, the relative error was 2.67×10^{-14}, with the Chebfun-derived value being slightly higher.

6.3 Divide-and-conquer

We also attempted to run our test examples using the theoretically faster divide-and-conquer approach for computing the globality certificates. Unfortunately, in practice divide-and-conquer



(a) Iterates (b) Iterates enlarged

Figure 3: The left plot shows a contour plot of the level sets (now in linear scale) of (5.2), for our discrete-time example. Although (5.2) is defined using polar coordinates, for visualization purposes, it is easier to show the contour plot in the complex plane using Cartesian coordinates, as is done in Figure 2. This example is also a real matrix so we only need to show the upper half of the complex plane, but we also include a small area underneath it so that all optimization iterates can be seen. The optimization iterates are depicted using the same convention as in Figure 2. Optimization begins at $(r, \theta) = (\sqrt{2}, \frac{3}{4}\pi)$, i.e., $(-1, 1)$ in Cartesian coordinates, and finds a local minimizer of (5.2) on the real axis near -1. The first restart provides a new starting point (near the top right corner of the plot) from which optimization finds a better but still not global minimizer (of the several different minimizers of (5.2) in the right half-plane). The second and final restart provides another starting point much closer to the global minimizer, which is now found by optimization. The right plot shows a close-up view of the denoted rectangular region surrounding the global maximizer as well as another local minimizer just below and to the right of it, which was found after the first restart.

does not appear to work well in our setting of Kreiss constants. First, divide-and-conquer requires an upper bound D on the real parts of all eigenvalues of (3.11). One might consider using `eigs` to compute eigenvalues of either the largest modulus or largest real part in order to obtain a satisfactory value of D, but this does not work in practice; `eigs` generally only returns `inf`'s when computing eigenvalues of (3.11) because half of its eigenvalues are ∞ (since the rank of $\mathcal{A}_2$ is half its dimension). Alternatively selecting an arbitrarily large value for D can be problematic as well, since the conditioning of $\mathcal{A}_1 - \nu\mathcal{A}_2$ generally gets worse as the shift ν increases, and $\mathcal{A}_1$ can be highly ill-conditioned itself. Even though we do not solve linear systems with $\mathcal{A}_1 - \nu\mathcal{A}_2$ directly, in practice we still observe numerical problems with the faster approach done by solving $2n \times 2n$ Sylvester equations. The second problem is that divide-and-conquer is built upon the assumption that a shift-and-invert solver will return the closest eigenvalue to any given shift. However, for the eigenvalue problems arising when computing Kreiss constants, we have observed that `eigs` frequently fails to find these closest eigenvalues. Interestingly, this problematic phenomenom can even be observed in the original paper proposing divide-and-conquer for faster computation of the distance to uncontrollability [GMO$^+$06, top left plot of Fig. 3.1].

While our testing is still limited, in practice we have observed that the divide-and-conquer method frequently fails to accurately capture the positive real eigenvalues of (3.11). Often it misses these real eigenvalues completely. Furthermore, it can even fail to cover the entire requested region of the real axis because for large shifts, `eigs` often returns false values that are exceptionally close to the requested shifts, meaning the area of the real axis considered "searched"

gets smaller and smaller as ν gets larger and larger. This generally results in the routine getting stuck in a loop, as it is unable to entirely cover the real axis to reach the upper bound. We have also observed these difficulties when applying divide-and-conquer to (5.23) for the discrete-time case. We suspect that the reason, perhaps in part, that divide-and-conquer appears to be much less reliable for our $4n^2 \times 4n^2$ eigenvalue problems than for the $2n^2 \times 2n^2$ problems for the distance to uncontrollability is related to the fact that the additional simplifications and reductions discussed in Remark 3.2 do not appear to extend here. We have also seen that (3.11) can sometimes have very close conjugate pair eigenvalues; the presence of these may also be why divide-and-conquer struggles to reliably find its real eigenvalues in practice.

7 Concluding remarks

By adapting ideas for computing the distance to uncontrollability, we have presented the first algorithm for computing continuous- and discrete-time Kreiss constants of matrices. We have also shown that attempting to compute Kreiss constants via trisection would not only be much slower but also seems to be fraught with numerical issues. The alternative trisection-compatible certificate test is exceptionally challenging numerically speaking, and even if it were not, our new result on the numerical accuracy of trisection nevertheless shows that it is likely to have little to no accuracy for matrices with large Kreiss constants.

Acknowledgments

Many thanks to Mark Embree for suggesting to look at the problem of computing Kreiss constants, and especially to Michael L. Overton, for not only pointing out the possible connection to computing the distance to uncontrollability but also hosting the author at the Courant Institute in New York for several visits, where much of this work was conducted.

References

[BBMX02] P. Benner, R. Byers, V. Mehrmann, and H. Xu. Numerical computation of deflating subspaces of skew-Hamiltonian/Hamiltonian pencils. *SIAM J. Matrix Anal. Appl.*, 24(1):165–190, 2002.

[BLO03] J. V. Burke, A. S. Lewis, and M. L. Overton. Robust stability and a criss-cross algorithm for pseudospectra. *IMA J. Numer. Anal.*, 23(3):359–375, 2003.

[BLO04] J. V. Burke, A. S. Lewis, and M. L. Overton. Pseudospectral components and the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 26(2):350–361, 2004.

[BM17] P. Benner and T. Mitchell. Extended and improved criss-cross algorithms for computing the spectral value set abscissa and radius. e-print arXiv:1712.10067, arXiv, December 2017. math.OC.

[BM18] P. Benner and T. Mitchell. Faster and more accurate computation of the $\mathcal{H}_\infty$ norm via optimization. *SIAM J. Sci. Comput.*, 40(5):A3609–A3635, October 2018.

[DHT14] T. A Driscoll, N. Hale, and L. N. Trefethen. *Chebfun Guide*. Pafnuty Publications, 2014. `http://www.chebfun.org/docs/guide/`.

[Eis84] R. Eising. Between controllable and uncontrollable. *Syst. Cont. Lett.*, 4(5):263–264, 1984.

[EK17] M. Embree and B. Keeler. Pseudospectra of matrix pencils for transient analysis of differential-algebraic equations. *SIAM J. Matrix Anal. Appl.*, 38(3):1028–1054, 2017.

[GMO+06] M. Gu, E. Mengi, M. L. Overton, J. Xia, and J. Zhu. Fast methods for estimating the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 28(2):477–502, 2006.

[Gu00] M. Gu. New methods for estimating the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 21(3):989–1003, 2000.

[Kre62] H.-O. Kreiss. Über die Stabilitätsdefinition für Differenzengleichungen die partielle Differentialgleichungen approximieren. *BIT Numerical Mathematics*, 2(3):153–181, 1962.

[KW89] B. Kågström and L. Westin. Generalized Schur methods with condition estimators for solving the generalized Sylvester equation. *IEEE Trans. Autom. Control*, 34(7):745–751, July 1989.

[Lan64] P. Lancaster. On eigenvalues of matrices dependent on a parameter. *Numer. Math.*, 6:377–387, 1964.

[Lar] R. M. Larsen. PROPACK - Software for large and sparse SVD calculations. `http://sun.stanford.edu/~rmunk/PROPACK`.

[LT84] R. J. LeVeque and L. N. Trefethen. On the resolvent condition in the Kreiss matrix theorem. *BIT Numerical Mathematics*, 24(4):584–591, 1984.

[Men06] E. Mengi. *Measures for Robust Stability and Controllability*. PhD thesis, New York University, New York, NY 10003, USA, September 2006.

[Mit] T. Mitchell. ROSTAPACK: RObust STAbility PACKage. `http://timmitchell.com/software/ROSTAPACK`.

[MO05] E. Mengi and M. L. Overton. Algorithms for the computation of the pseudospectral radius and the numerical radius of a matrix. *IMA J. Numer. Anal.*, 25(4):648–669, 2005.

[OW95] M. L. Overton and R. S. Womersley. Second derivatives for optimizing eigenvalues of symmetric matrices. *SIAM J. Matrix Anal. Appl.*, 16(3):697–718, 1995.

[Spi91] M. N. Spijker. On a conjecture by LeVeque and Trefethen related to the Kreiss matrix theorem. *BIT Numerical Mathematics*, 31(3):551–555, 1991.

[TE05] L. N. Trefethen and M. Embree. *Spectra and pseudospectra: The behavior of nonnormal matrices and operators*. Princeton University Press, Princeton, NJ, 2005.

[Wri02] T. G. Wright. EigTool. `http://www.comlab.ox.ac.uk/pseudospectra/eigtool/`, 2002.