# Computation of Pseudospectral Abscissa for Large Scale Nonlinear Eigenvalue Problems

Karl Meerbergen[†] Emre Mengi[‡] Wim Michiels[†]
Roel Van Beeumen[†]

April 7, 2016

**Abstract**

We present an algorithm to compute the pseudospectral abscissa for a nonlinear eigenvalue problem. The algorithm relies on global under-estimator and over-estimator functions for the eigenvalue and singular value functions involved. These global models follow from eigenvalue perturbation theory. The algorithm has three particular features. First, it converges to the globally rightmost point of the pseudospectrum, and it is immune to nonsmoothness. The global convergence assertion is under the assumption that a global lower bound is available for the second derivative of a singular value function depending on one parameter. It may not be easy to deduce such a lower bound analytically, but assigning large negative values works robustly in practice. Second, it is applicable to large scale problems since the dominant cost per iteration stems from computing the smallest singular value and associated singular vectors, for which efficient iterative solvers can be used. Furthermore, a significant increase in computational efficiency can be obtained by subspace acceleration, i.e., by restricting the domains of the linear maps associated with the matrices involved to small but suitable subspaces, and solving the resulting reduced problems. Occasional restarts of these subspaces further enhance the efficiency for large scale problems. Finally, in contrast to existing iterative approaches based on constructing low rank perturbations and rightmost eigenvalue computations, the algorithm only relies on computing singular values of complex matrices. Hence, the algorithm does not require solutions of nonlinear eigenvalue problems, thereby further increasing efficiency and reliability. This work is accompanied by a robust implementation of the algorithm, that is publicly available.

*Keywords*: pseudospectra, nonlinear eigenvalue problem, eigenvalue perturbation theory, nonsmooth optimization, subspace methods, global optimization

*AMS subject classifications*: 65F15, 90C30, 65H20

## 1 Introduction

We consider the nonlinear eigenvalue problems of the form

$$F(\lambda)x = 0, \tag{1.1}$$

where $F : \Omega \to \mathbb{C}^{n\times n}$ is an analytic matrix-valued function on $\Omega \subseteq \mathbb{C}$. The scalar $\lambda \in \Omega$ satisfying the equation above for an $x \in \mathbb{C}^n \setminus \{0\}$ is called an eigenvalue, while $x$ is called the

[†]Department of Computer Science, KU Leuven, University of Leuven, 3001 Heverlee, Belgium (Roel.VanBeeumen@cs.kuleuven.be, Karl.Meerbergen@cs.kuleuven.be, Wim.Michiels@cs.kuleuven.be). The work of the authors were supported in part by OPTEC, the Optimization in Engineering Center of KU Leuven and the Research Foundation Flanders (project G.0712.11N ).

[†]Department of Mathematics, Koç University, Rumelifeneri Yolu, 34450 Sarıyer-İstanbul, Turkey (emengi@ku.edu.tr). The work of the author was supported in part by the European Commision grant PIRG-GA-268355, TUBITAK - FWO (Scientific and Technological Research Council of Turkey - Belgian Research Foundation, Flanders) joint grant 113T053, and by the BAGEP program of Turkish Academy of Science.

corresponding eigenvector. Such eigenvalue problems when $F$ is a matrix polynomial, especially the quadratic eigenvalue problem, arise from various engineering applications for instance from applications in structural design and fluid mechanics [35]. Nonpolynomial nonlinear eigenvalue problems are also of great interest: finite-element discretizations of boundary value problems, for instance in photonics, lead to eigenvalue problems of the form (1.1), where $F(\lambda)$ is a rational function of $\lambda$ [27, 11]; delay systems in control theory necessitate nonlinear eigenvalue problems where $F(\lambda)$ involves exponentials of $\lambda$ [33]. For recent progresses on the topic, we refer to the survey paper [27] and the thesis works [11, 39].

Stability of the continuous dynamical system associated with the nonlinear eigenvalue problem is a fundamental issue. In terms of the nonlinear eigenvalue problem (1.1), this amounts to the inclusion of all of the eigenvalues on the left half of the complex plane. However, the system is often subject to uncertainties, thus it is often desired that a system remains stable under small perturbations of parameters. This is reflected into the nonlinear eigenvalue problem (1.1) as the inclusion of the eigenvalues of the original problem as well as all perturbed problems in the left half of the complex plane. Moreover, a stable system can still exhibit transient behavior before reaching the equilibrium eventually. For instance, for the standard eigenvalue problem $Av = \lambda v$ and the associated dynamical system $x'(t) = Ax(t)$, this is explained by the Kreiss matrix theorem [37, Theorem 18.5]. A corollary of this theorem is that a stable system $x'(t) = Ax(t)$ becoming unstable under small perturbations of $A$ must exhibit transient growth.

For robustness against uncertainties and to assess the transient behavior of solutions of a stable system, a modern approach is the consideration of the $\epsilon$-pseudospectrum of $F$. This is the set in the complex plane to which the eigenvalues of $F$ can be shifted when perturbations at a distance $\epsilon$ or closer are taken into account. The $\epsilon$-pseudospectral abscissa, the supremum of the real parts of the elements of the $\epsilon$-pseudospectrum, constitutes a uniform bound on the asymptotic growth rate of the solutions for all perturbations at a distance $\epsilon$ or closer. Consequently, it assesses robust stability [7]. The $\epsilon$-pseudospectral abscissa is also closely related to the distance to instability [41] and the H-infinity norm of transfer functions defined appropriately (see [46] for relations between H-infinity norms and robust stability criteria).

A standard approach to solve polynomial eigenvalue problems is by reformulating this type of eigenvalue problems as standard generalized linear eigenvalue problems, so-called linearization, with the same eigenvalues as the original polynomial one [26, 1]. The obtained linear eigenvalue problem can next be solved by a standard method of choice. For solving general nonlinear eigenvalue problems (1.1), first a polynomial or rational approximation is constructed [39] or the nonlinear eigenvalue problem is transformed to an equivalent infinite dimensional operator eigenvalue problem, e.g., as for the delay eigenvalue problem [18]. Next, a linearization process follows to obtain a linear eigenvalue problem whose spectrum approximates the spectrum of the original nonlinear eigenvalue problem. For state of the art general nonlinear eigenvalue solvers we refer to [15, 40]. Here, we will not consider the unstructured pseudospectra of a particular type of linearization. Instead, as in [34, 31, 41], we will explicitly take the structure of the original nonlinear eigenvalue problem into account in the definition of its pseudospectra.

## 1.1 Formal Definition

Formally, the analytic matrix-valued function $F$ can always be expressed of the form

$$F(\lambda) = \sum_{j=0}^{m} f_j(\lambda) A_j \tag{1.2}$$

where $A_j \in \mathbb{C}^{n\times n}$, the scalar function $f_j : \Omega \to \mathbb{C}$ is analytic on its entire domain $\Omega$ for $j = 0, \ldots, m$, and $m \leq n^2 - 1$. The spectrum of $F$ given by

$$\Lambda(F) := \left\{ \lambda \in \mathbb{C} : \ \det\left( \sum_{j=0}^{m} f_j(\lambda) A_j \right) = 0 \right\}, \tag{1.3}$$

more specifically the *spectral abscissa*

$$\alpha(F) := \sup\{\Re\lambda : \ \lambda \in \Lambda(F)\} \tag{1.4}$$

is responsible for the asymptotic behavior of the associated dynamical system, yet it does not say much about the transient behavior by itself.

To take the uncertainties and transient behavior into account, we are interested in the perturbed eigenvalue problem

$$\left(\sum_{j=0}^{m} f_j(\lambda)(A_j + \delta A_j)\right) x = 0. \tag{1.5}$$

We quantify the distance between the original matrix-valued function (1.2), and the perturbed one in (1.5) by introducing the norm

$$\|\Delta\|_{\text{glob}} := \left\| \begin{bmatrix} w_0\|\delta A_0\|_2 \\ \vdots \\ w_m\|\delta A_m\|_2 \end{bmatrix} \right\|_\infty, \tag{1.6}$$

where $\Delta := (\delta A_0, \ldots, \delta A_m) \in \mathbb{C}^{n\times n\times(m+1)}$ for given non-negative real scalars $w_j$ (possibly $\infty$) for $j = 0, \ldots, m$, equivalently we equip the vector space of analytic matrix-valued functions of the form (1.2) with variable coefficient matrices $A_j$ but fixed scalar functions $f_j$ with a norm.

We then define the $\epsilon$-pseudospectrum of $F$ by

$$\Lambda_\epsilon(F) := \bigcup_{\|\Delta\|_{\text{glob}}\le\varepsilon} \left\{\lambda \in \mathbb{C} : \ \det\left(\sum_{j=0}^{m} f_j(\lambda)(A_j + \delta A_j)\right) = 0\right\}, \tag{1.7}$$

and the $\epsilon$-pseudospectral abscissa by

$$\alpha_\epsilon(F) := \sup\{\Re\lambda : \ \lambda \in \Lambda_\epsilon(F)\} \tag{1.8}$$

as an indicator of the robust stability of the dynamical system associated with (1.1). The following characterization of $\Lambda_\epsilon(F)$ was derived in [30].

**Proposition 1.1.**

$$\Lambda_\epsilon(F) = \left\{\lambda \in \mathbb{C} : \ \sigma_{\min}\left(\sum_{j=0}^{m} f_j(\lambda)A_j\right) \le \epsilon\|w(\lambda)\|_1\right\}, \tag{1.9}$$

*where $\sigma_{\min}(\cdot)$ denotes the smallest singular value of its matrix argument, and*

$$w(\lambda) := \begin{bmatrix} \dfrac{f_0(\lambda)}{w_0} & \cdots & \dfrac{f_m(\lambda)}{w_m} \end{bmatrix}^T. \tag{1.10}$$

Throughout the text, we assume that the portion of $\Lambda_\epsilon(F)$ to the right-hand side of each vertical line in the complex plane is bounded. Formally, letting

$$\mathbb{C}_{\ge\delta} := \{z \in \mathbb{C} \ : \ \Re z \ge \delta\}$$

for a given $\delta \in \mathbb{R}$, it is assumed that $\Lambda_\epsilon(F) \cap \mathbb{C}_{\ge\delta}$ is bounded for all $\delta \in \mathbb{R}$. This assumption ensures the well-posedness of $\alpha_\epsilon(F)$ defined by (1.8). For a thorough discussion on this condition, we refer to [31].

## 1.2 Literature

The $\epsilon$-pseudospectrum for matrices, that is when $F(\lambda) = \lambda I - A$, has been popularized by Trefethen in the last two decades [37]. This set has found various applications in the literature in connection with robust stability and transient behavior, for instance to analyze the cutoff phenomenon in Markov chains [19] and in stability analysis in hydrodynamics [38]. Its computation benefits from its singular value characterization. The most standard approaches [36, 43] are based on discretizing the complex plane, and relying on powerful tools of numerical linear algebra such as Lancoz method and Schur factorization. Some curve-tracing approaches have also been suggested [5, 2].

Extensions to the nonlinear eigenvalue problems have been considered throughout the last two decades. For polynomial eigenvalue problems, its connection with the backward error of an eigenvalue has been studied, singular value characterizations have been derived and numerical approaches have been proposed for its computation in [34]. This has been extended to rectangular matrix polynomials in homogeneous form in [17]. The boundary and components of the $\epsilon$-pseudospectrum have been studied, and a curve-tracing algorithm have been proposed in the matrix polynomial setting in [22]. More recent research concentrated on the nonpolynomial setting. In particular, [13] concerns the computation of the pseudospectra in the delay eigenvalue problem setting. In [30], the pseudospectra for a general analytic matrix-valued function have been formally introduced, and singular value characterizations have been derived. This has been extended to analytic matrix-valued functions subject to structured perturbations with multiplicative structure in [42].

Particular attention has been paid to the computation of the pseudospectral abscissa. For the pseudospectral abscissa of a matrix, the first globally convergent algorithm was proposed in [7]. Since every iteration of this algorithm requires computing all eigenvalues of a matrix of twice the dimensions of the original matrix, it is restricted to problems of moderate size. In [14], a locally convergent algorithm for large scale matrices is proposed, where every iteration relies on computing the rightmost eigenvalue of the original matrix plus a rank one perturbation (see also [20] for an improvement of this algorithm based on subspace acceleration). The algorithm of [14] has been extended to nonlinear eigenvalue problems in [31], and it has also been adopted to compute the distance to instability from a nonlinear dynamical system in [41]. On a different direction, an implicit determinant method is proposed to compute the distance to instability from a matrix [12]. This approach is based on solutions of certain linear systems only, and can be coupled with a bisection method to compute the pseudospectral abscissa. In the next subsection we situate the proposed algorithm with respect to these works.

## 1.3 Motivation and Outline

We present a globally convergent algorithm for the computation of $\alpha_\epsilon(F)$, particularly suitable for large scale problems, that is when $A_j$ are large matrices. Three main components of the algorithm are introduced in [28, 29, 20]. In [28], a locally convergent algorithm is presented for optimizing a linear function subject to a constraint on a smallest eigenvalue function. The algorithm is immune to the nonsmooth nature of the smallest eigenvalue function. We describe how this algorithm can be adopted to compute $\alpha_\epsilon(F)$ based on the characterization (1.9) of $\Lambda_\epsilon(F)$ in Section 2.

Unfortunately, this yields a locally rightmost point, which is possibly not rightmost globally. We overcome this by performing a vertical search by means of the algorithm introduced in [29] for the global optimization of a prescribed eigenvalue of a Hermitian and analytic matrix-valued function. We fix the real part $\alpha$ of the locally rightmost point, and perform the minimization of

$$\frac{\sigma_{\min}\left(\sum_{j=0}^{m} f_j(\alpha+i\omega)A_j\right)}{\|w(\alpha+i\omega)\|_1} \tag{1.11}$$

over all $\omega \in \mathbb{R}$ globally. This global minimization assumes the availability of a global lower bound on the second derivative of the function above with respect to $\alpha$. In practice, choosing a large negative value for this bound works robustly. If the globally minimal value is less than $\epsilon$,

then we repeat the local search starting from $\alpha + i\omega_*$ where $\omega_*$ is a global minimizer of (1.11). We refer to Figure 5 (in the numerical examples section towards the end of this text) for an illustration of the interplay between the local searches and vertical searches. In this illustration, local searches yield locally (but not globally) rightmost points twice. In each of these two cases, a vertical search provides a point strictly inside $\Lambda_\epsilon(F)$ whose real part is the same as the locally rightmost point. The vertical search idea is discussed in Section 3.

Due to the fact that the computational cost is dominated by computing the smallest singular value and corresponding singular vectors, for which fast iterative methods are amenable, the proposed algorithm is applicable to large scale problems. Moreover, a significant speed-up can be achieved by incorporating a subspace restriction, whose idea is originally proposed in [20] for the computation of the pseudospectral abscissa of a matrix. The remarkable low rank property observed and exploited in that paper still holds in this more general nonlinear eigenvalue setting. In particular, there exists a one dimensional subspace of $\mathbb{C}^n$ such that the $\epsilon$-pseudospectral abscissa of $F(\lambda)$ remains the same when the domain of the map $v \mapsto F(\lambda)v$ is restricted to this one dimensional subspace. The details of this subspace idea for nonlinear eigenvalue problems are worked out in Section 4. The overall idea is to restrict the domain of $v \mapsto F(\lambda)v$ to very low dimensional subspaces of $\mathbb{C}^n$, and compute the $\epsilon$-pseudospectral abscissa of the resulting smaller problems by means of the locally convergent algorithm in [28]. The vertical searches are performed on the original $F(\lambda)$. This is justified by the rare need for these vertical searches.

One genuine aspect of the algorithm is an occasional restart strategy for the subspaces as argued in Section 5. Since the essential task is to determine or capture a one dimensional subspace, the algorithm erases the old subspaces occasionally. Thus after a vertical search, if a further application of the local algorithm is deemed to be necessary, the algorithm starts with a one dimensional subspace from scratch. Moreover, when the dimension of the subspace becomes large enough (still considerably smaller than $n$), the algorithm keeps only the lastly added one dimensional subspace discarding the rest.

The overall framework is outlined in Algorithm 4. This algorithm features favorable properties over exisiting algorithms, for instance [31]. Specifially, **(i)** it converges globally rather than locally (provided a sufficiently small global lower bound for the second derivative of the singular value function in (1.11) is chosen); **(ii)** it is immune to nonsmoothness, that is, even if $\alpha_\epsilon(F)$ is attained at a point say $z_* \in \mathbb{C}$ where $\sigma_{\min}(F(z_*))$ is not simple, it still converges; **(iii)** it handles large scale problems well. The subspace method coupled with the restart strategy contributes to this largely, but the restarts would not be as effective without vertical searches that are performed globally; **(iv)** in contrast to the approach of [31], the algorithm does not rely on a nonlinear eigenvalue solver (provided it is initialized with the rightmost eigenvalue): instead of the rightmost eigenvalue of perturbed nonlinear eigenvalue problems, it is based on the repeated computation of the smallest singular value of complex matrices. Robust and efficient numerical algorithms are available, e.g., the implicitly restarted Arnoldi method of [25], for the smallest singular value.

## 2 Determination of Locally Rightmost Points

Due to Proposition 1.9, the $\epsilon$-pseudospectral abscissa of $F$ can be cast as the following constrained eigenvalue optimization problem:

$$\begin{aligned} &\text{maximize } \Re z \\ &\quad z \in \mathbb{C} \\ &\text{subject to } \lambda(\Re z, \Im z) := \lambda_{\min}\left[F(\Re z, \Im z)^* F(\Re z, \Im z)\right] - \epsilon^2 \|w(\Re z, \Im z)\|_1^2 \le 0, \end{aligned} \tag{2.1}$$

where we view the matrix-valued function in (1.2) as $F : \mathbb{R}^2 \to \mathbb{C}^{n\times n}$ and the weight function in (1.10) as $w : \mathbb{R}^2 \to \mathbb{R}$, by associating $\mathbb{R}^2$ with $\mathbb{C}$. Throughout the text, to ease the notation $F(\cdot)$, $\lambda(\cdot)$, $w(\cdot)$ and $f_j(\cdot)$, $j = 0, \ldots, m$ represent both the functions from $\mathbb{C}$ and the functions from $\mathbb{R}^2$. The functions with one parameter or supplied with a particular point in $\mathbb{C}$ as its

argument correspond to the ones with domain $\mathbb{C}$, and the functions with two parameters or supplied with a point in $\mathbb{R}^2$ as its argument have domain $\mathbb{R}^2$. Furthermore, in (2.1) and in what follows the notations $\lambda_{\min}[\cdot]$ and $\lambda_{\max}[\cdot]$ represent the smallest eigenvalue and the largest eigenvalue of the matrix argument, respectively.

An approach to maximize a linear objective subject to a smallest eigenvalue constraint was suggested in [28]. Below, we describe how this approach can be extended to deal with (2.1), in particular the additional nonsmoothness due to $\|w(\Re z, \Im z)\|_1^2$, which occurs whenever $f_j(\Re z, \Im z) = 0$ for some $j \in \{0, \ldots, m\}$. The extension relies on the global over-estimators for $\lambda(\cdot)$ of the form specified in Theorem 2.1 below. These global over-estimators are defined in terms of global bounds $\gamma_\lambda$ and $\gamma_w$ satisfying

$$\lambda_{\max}\left\{\nabla^2 \lambda_{\min}\left[F(\Re z, \Im z)^* F(\Re z, \Im z)\right]\right\} \leq \gamma_\lambda \tag{2.2}$$

for all $z \in \mathbb{C}$ where $\sigma_{\min}[F(\Re z, \Im z)]$ is simple, and

$$\|\nabla^2\left[\|w(\Re z, \Im z)\|_1^2\right]\|_2 \leq \gamma_w \tag{2.3}$$

for all $z \in \mathbb{C}$ where $f_j(\Re z, \Im z) \neq 0$ for each $j$, respectively, more specifically, in terms of $\gamma := \gamma_\lambda + \epsilon^2 \gamma_w$. Analytical deduction of such bounds is discussed in Section 2.1 below.

**Theorem 2.1.** *Suppose $z_k = (z_{k1}, z_{k2}) \in \mathbb{R}^2$ is a point such that $\sigma_{\min}[F(z_k)]$ is simple, and $f_j(z_k) \neq 0$ for each $j$. We have*

$$\lambda(\Re z, \Im z) \;\leq\; q_k(\Re z, \Im z) := \lambda_k + \nabla\lambda_k^T((\Re z, \Im z) - z_k) + \frac{\gamma}{2}\|(\Re z, \Im z) - z_k\|_2^2 \quad \forall z \in \mathbb{C}$$

*where $\lambda_k := \lambda(z_{k1}, z_{k2})$ and $\nabla\lambda_k := \nabla\lambda(z_{k1}, z_{k2})$.*

The proof of Theorem 2.1 is in essence identical to the proof of Theorem 2.2 in [28], but $\lambda(\cdot)$ here takes the role of $\lambda_{\min}(\cdot)$ over there. In this direction, we note the following:

- The function $\phi(\alpha) = \lambda((\Re z, \Im z) + \alpha p)$ for a given $p \in \mathbb{R}^2$ is the minimum of finitely many analytic functions, namely

  $$\phi_{j,s_0,\ldots,s_m}(\alpha) \;:=\; \phi_j(\alpha) - \epsilon^2\left\{s_0 f_0((\Re z, \Im z) + \alpha p) + \cdots + s_m f_m((\Re z, \Im z) + \alpha p)\right\}^2$$

  for $j = 1, \ldots, n$, $s_0 = -1, 1, \ldots, s_m = -1, 1$. Here, $\phi_1(\alpha), \ldots, \phi_n(\alpha)$ represent the eigenvalues of $F((\Re z, \Im z) + \alpha p)^* F((\Re z, \Im z) + \alpha p)$ ordered so that each $\phi_j(\alpha)$ is analytic;

- The left-derivative $\phi'_-(\alpha)$, the right-derivative $\phi'_+(\alpha)$ exist everywhere, and satisfy $\phi'_-(\alpha) \geq \phi'_+(\alpha)$ at all $\alpha$, since $\phi(\alpha)$ is the minimum of finitely many analytic functions;

- Furthermore, $\phi(\alpha)$ is analytic everywhere excluding finitely many points in a finite interval for the very same reason that it is the minimum of finitely many analytic functions;

- Finally, $\gamma$ is such that

  $$\begin{aligned}\lambda_{\max}\left[\nabla^2\lambda(\Re z, \Im z)\right] &\leq \lambda_{\max}\left\{\nabla^2\lambda_{\min}\left[F(\Re z, \Im z)^* F(\Re z, \Im z)\right]\right\} + \lambda_{\max}\left\{\nabla^2\left[-\epsilon^2\|w(\Re z, \Im z)\|_1^2\right]\right\} \\ &\leq \lambda_{\max}\left\{\nabla^2\lambda_{\min}\left[F(\Re z, \Im z)^* F(\Re z, \Im z)\right]\right\} + \epsilon^2\left\|\nabla^2\left[\|w(\Re z, \Im z)\|_1^2\right]\right\|_2 \\ &\leq\; \gamma_\lambda + \epsilon^2\gamma_w = \gamma\end{aligned}$$

  for all $z \in \mathbb{C}$ such that $\lambda(\Re z, \Im z)$ is twice differentiable.

Replacing the eigenvalue constraint in (2.1) with the over-estimator of Theorem 2.1 results in the following convex and smooth problem:

$$\begin{aligned}&\text{maximize } \alpha \\ &(\alpha, \beta) \in \mathbb{R}^2 \\ &\text{subject to } \lambda_k + \nabla\lambda_k^T((\alpha, \beta) - z_k) + \frac{\gamma}{2}\|(\alpha, \beta) - z_k\|_2^2 \leq 0.\end{aligned} \tag{2.4}$$

The algorithm generates a sequence $\{z_k\}$ in $\mathbb{R}^2$ such that $z_{k+1}$ is the maximizer of (2.4) given $z_k$. Since the feasible set of (2.4) (a disk) is a subset of the feasible set of the original problem (2.1), each $z_k$ remains feasible with respect to the original problem provided $z_0$ is feasible. By applying the first order optimality conditions to (2.4), two consecutive iterates in the sequence $\{z_k\}$ are tied by the recurrence

$$z_{k+1} = z_k + \frac{1}{\gamma}\left[\frac{1}{\mu_+}\cdot(1,0) - \nabla\lambda_k\right], \quad \text{where} \quad \mu_+ = \frac{1}{\sqrt{\|\nabla\lambda_k\|_2^2 - 2\gamma\lambda_k}}, \tag{2.5}$$

$$\nabla\lambda_k = \begin{bmatrix} \text{Real}\left(v_k^*\frac{\partial F(z_k)^*}{\partial \Re z}F(z_k)v_k + v_k^*F(z_k)^*\frac{\partial F(z_k)}{\partial \Re z}v_k\right) - 2\epsilon^2\|w(z_k)\|_1\frac{\partial\|w(z_k)\|_1}{\partial \Re z} \\ \text{Real}\left(v_k^*\frac{\partial F(z_k)^*}{\partial \Im z}F(z_k)v_k + v_k^*F(z_k)^*\frac{\partial F(z_k)}{\partial \Im z}v_k\right) - 2\epsilon^2\|w(z_k)\|_1\frac{\partial\|w(z_k)\|_1}{\partial \Im z} \end{bmatrix},$$

$$\frac{\partial\|w(z_k)\|_1}{\partial \Re z} = \sum_{j=0}^{m}\frac{1}{w_j}\frac{\partial\,|f_j(z_k)|}{\partial \Re z}, \qquad \frac{\partial\|w(z_k)\|_1}{\partial \Im z} = \sum_{j=0}^{m}\frac{1}{w_j}\frac{\partial\,|f_j(z_k)|}{\partial \Im z},$$

and $v_k \in \mathbb{C}^n$ is a unit right singular vector corresponding to $\sigma_{\min}[F(z_k)]$. Here, we benefit from the analytical formulas for the derivatives of eigenvalue functions [21], in particular to calculate the derivatives of $\lambda_{\min}[F(\Re z, \Im z)^* F(\Re z, \Im z)]$. Recurrence (2.5) holds under the assumption that $\nabla q_k(z_{k+1}) \neq 0$. The condition $\nabla q_k(z_{k+1}) = 0$ is rather unlikely, it occurs only if $\nabla\lambda_k = 0$ and $\lambda_k = 0$ (see Theorem 2.3 in [28]).

## 2.1 Upper Bounds on Second Derivatives

In this section, we present bounds $\gamma_\lambda$ and $\gamma_w$ satisfying (2.2) and (2.3), respectively. An application of Theorem 6.1 in [28] yields

$$\lambda_{\max}\left\{\nabla^2\lambda_{\min}[F(\Re z, \Im z)^* F(\Re z, \Im z)]\right\} \leq \lambda_{\max}\left\{\nabla^2[F(\Re z, \Im z)^* F(\Re z, \Im z)]\right\} \tag{2.6}$$

for $z \in \mathbb{C}$ such that $\sigma_{\min}[F(z)]$ is simple, where

$$\nabla^2[F(\Re z, \Im z)^* F(\Re z, \Im z)] := \begin{bmatrix} \frac{\partial^2[F(\Re z, \Im z)^* F(\Re z, \Im z)]}{\partial \Re z^2} & \frac{\partial^2[F(\Re z, \Im z)^* F(\Re z, \Im z)]}{\partial \Re z \cdot \partial \Im z} \\ \frac{\partial^2[F(\Re z, \Im z)^* F(\Re z, \Im z)]}{\partial \Im z \cdot \partial \Re z} & \frac{\partial^2[F(\Re z, \Im z)^* F(\Re z, \Im z)]}{\partial \Im z^2} \end{bmatrix}.$$

Above, $\partial^2[F(\Re z, \Im z)^* F(\Re z, \Im z)]/\partial \Re z^2$ denotes the second derivative of the matrix-valued function $F(\Re z, \Im z)^* F(\Re z, \Im z)$ with respect to $\Re z$, and is an $n \times n$ matrix obtained from $F(\Re z, \Im z)^* F(\Re z, \Im z)$ by differentiating each of its entry with respect to $\Re z$ twice. The other second derivatives of $F(\Re z, \Im z)^* F(\Re z, \Im z)$ above are defined similarly.

For the standard $\epsilon$-pseudospectral abscissa of a matrix $A$, i.e., when $F(z) = A - zI$, we have $\nabla^2[F(\Re z, \Im z)^* F(\Re z, \Im z)] = 2I$. Consequently, inequality (2.6) leads to the upper bound $\lambda_{\max}\left\{\nabla^2\lambda_{\min}[F(\Re z, \Im z)^* F(\Re z, \Im z)]\right\} \leq 2$ for all $z \in \mathbb{C}$ such that $\sigma_{\min}[F(z)]$ is simple. In the general nonlinear setting (1.2), routine calculations yield

$$\nabla^2[F(\Re z, \Im z)^* F(\Re z, \Im z)] = \sum_{k=0}^{m}\sum_{j=0}^{m}\left[\overline{F_{k,j}(\Re z, \Im z)} + F_{j,k}(\Re z, \Im z)\right] \otimes A_k^* A_j,$$

where

$$\begin{aligned} F_{j,k}(\Re z, \Im z) &:= \nabla^2\overline{f_k(\Re z, \Im z)}\cdot f_j(\Re z, \Im z) + \nabla\overline{f_k(\Re z, \Im z)}\cdot\nabla f_j(\Re z, \Im z)^T \\ &= \overline{f_k''(z)}f_j(z)\begin{bmatrix} 1 & -i \\ -i & -1 \end{bmatrix} + \overline{f_k'(z)}f_j'(z)\begin{bmatrix} 1 & i \\ -i & 1 \end{bmatrix}, \end{aligned} \tag{2.7}$$

and $\otimes$ denotes the Kronecker product. Above, $f_k'(z), f_k''(z)$ represent first, second complex derivatives of $f_k(z)$, thus functions from $\Omega \subseteq \mathbb{C}$ to $\mathbb{C}$. From here, by employing (2.6) and exploiting

$$\overline{F_{k,j}(\Re z, \Im z)} + F_{j,k}(\Re z, \Im z) = \left[\overline{F_{j,k}(\Re z, \Im z)} + F_{k,j}(\Re z, \Im z)\right]^*,$$

we deduce the following bound.

**Theorem 2.2.** *Suppose that $z \in \mathbb{C}$ is such that $\sigma_{\min}[F(z)]$ is simple. The following holds:*

$$\lambda_{\max}\left\{\nabla^2 \lambda_{\min}\left[F(\Re z, \Im z)^* F(\Re z, \Im z)\right]\right\} \;\leq\; 2 \cdot \sum_{k=0}^{m}\sum_{j=0}^{m} \|F_{k,j}(\Re z, \Im z)\|_2 \cdot \|A_k^* A_j\|_2.$$

**Example (Polynomial Eigenvalue Problem):** Consider $F(z) = \sum_{j=0}^{m} z^j A_j$ for given matrices $A_j \in \mathbb{C}^{n\times n}$ for $j = 0, \ldots, m$. Noting that $f_j(z) = z^j$, Theorem 2.2 combined with expression (2.7) for $F_{j,k}(\Re z, \Im z)$ would imply

$$\lambda_{\max}\left\{\nabla^2 \lambda_{\min}\left[F(\Re z, \Im z)^* F(\Re z, \Im z)\right]\right\} \leq \\ 4 \cdot \left[\sum_{k=0}^{m}\sum_{j=2}^{m} j\cdot(j-1)\cdot|z|^{j+k-2}\|A_k^* A_j\|_2 + \sum_{k=1}^{m}\sum_{j=1}^{m} k\cdot j\cdot|z|^{j+k-2}\|A_k^* A_j\|_2\right].$$

Assuming that the $\epsilon$-pseudospectrum of $F$ is bounded and contained inside a ball of radius $\delta$ in the complex plane and letting $a := \max_{j=0,\ldots,m}\|A_j\|_2$ , we could set

$$\gamma_\lambda \;:=\; 4a^2 \cdot \left[\left(\frac{\delta^{m+1}-1}{\delta-1}\right)\left(\frac{\delta^{m+1}-1}{\delta-1}\right)'' + \left(\left\{\frac{\delta^{m+1}-1}{\delta-1}\right\}'\right)^2\right],$$

where the derivatives are with respect to $\delta$. Similarly, bounds could be also derived for delay and rational eigenvalue problems based on Theorem 2.2.

The following bound is the consequence of rudimentary calculations.

**Theorem 2.3.** *Suppose $z \in \mathbb{C}$ is such that $f_j(z) \neq 0$ for $j = 0, \ldots, m$. The following holds:*

$$\left\|\nabla^2\left[\|w(\Re z, \Im z)\|_1^2\right]\right\|_2 \;\leq\; 2\cdot\left[\sum_{j=0}^{m}\frac{1}{w_j}|f_j'(z)|\right]^2 + 2\cdot\left[\sum_{j=0}^{m}\frac{1}{w_j}|f_j(z)|\right]\cdot\left[\sum_{j=0}^{m}\frac{1}{w_j}\left\{3\cdot\frac{|f_j'(z)|^2}{|f_j(z)|} + |f_j''(z)|\right\}\right]$$

For instance, for the matrix polynomial $F(z) = \sum_{j=0}^{m} z^j A_j$ with $f_j(z) = z^j$, Theorem 2.3 gives rise to the bound

$$\left\|\nabla^2\left[\|w(\Re z, \Im z)\|_1^2\right]\right\|_2 \;\leq\; 2\cdot\left[\sum_{j=1}^{m}\frac{j\cdot|z|^{j-1}}{w_j}\right]^2 + 2\cdot\left[\sum_{j=0}^{m}\frac{|z|^j}{w_j}\right]\cdot\left[\sum_{j=1}^{m}\frac{3j^2\cdot|z|^{j-2}}{w_j} + \sum_{j=2}^{m}\frac{j\cdot(j-1)\cdot|z|^{j-2}}{w_j}\right].$$

If the $\epsilon$-pseudospectral abscissa of $F$ is contained inside the ball of radius $\delta$ and $\underline{w} := \min_{j=0,\ldots,m} w_j$, we could choose

$$\gamma_w \;:=\; \frac{2}{\underline{w}^2}\cdot\left[\left(\left\{\frac{\delta^{m+1}-1}{\delta-1}\right\}'\right)^2 + \left(\frac{\delta^{m+1}-1}{\delta-1}\right)\left(\frac{3}{\delta} + 4\left\{\frac{\delta^{m+1}-1}{\delta-1}\right\}'' + 3\left\{\frac{\delta^{m+1}-1}{\delta-1}\right\}'\right)\right].$$

## 2.2 Convergence

Let us denote the components of $z_k \in \mathbb{R}^2$ with $z_{k1}$ and $z_{k2}$. We call $\mathcal{C}(z_k) := z_{k1} + iz_{k2}$ the *complexification* of $z_k$. The sequence $\{z_{k1}\}$ is monotone increasing. This is because $z_{k+1}$ is chosen among all $(\Re z, \Im z)$ satisfying $q_k(\Re z, \Im z) \leq 0$ such that $\Re z$ is as large as possible, and in particular $z_k$ satisfies $q_k(z_k) = 0$. Additionally, since it is assumed that $\Lambda_\epsilon(F) \cap \mathbb{C}_{\geq\delta}$ is bounded for all $\delta \in \mathbb{R}$, the sequence $\{z_{k1}\}$ is bounded above. This would imply the convergence of $\{z_{k1}\}$ as stated next.

**Theorem 2.4.** *Suppose that $\Lambda_\epsilon(F) \cap \mathbb{C}_{\geq\delta}$ is bounded for all $\delta \in \mathbb{R}$, $\sigma_{\min}[F(z_k)]$ is simple for each $k$, and $f_j(z_k) \neq 0$ for each $j, k$. Then the sequence $\{z_{k1}\}$ is convergent.*

The boundedness of $\Lambda_\epsilon(F) \cap \mathbb{C}_{\geq\delta}$ for all $\delta \in \mathbb{R}$ rather than the boundedness of $\Lambda_\epsilon(F)$ is also sufficient for the convergence of the sequence $\{z_k\}$ to a desired point, provided $\|\nabla\lambda_k\|_2$ remains bounded away from zero.

**Theorem 2.5** (Convergence). *Suppose that $\Lambda_\epsilon(F) \cap \mathbb{C}_{\geq\delta}$ is bounded for all $\delta \in \mathbb{R}$, $\sigma_{\min}[F(z_k)]$ is simple for each $k$, and $f_j(z_k) \neq 0$ for each $j, k$.*

**(i)** *If $\nabla\lambda_k \neq 0$ for each $k$ sufficiently large, then $\lambda_k \to 0$ as $k \to \infty$.*

**(ii)** *If there exists a real scalar $L > 0$ such that $\|\nabla\lambda_k\|_2 > L$ for each $k$ sufficiently large, then*

$$\frac{(1,0)\cdot\nabla\lambda_k}{\|\nabla\lambda_k\|_2} \to 1 \qquad \text{as } k \to \infty.$$

The proofs of parts **(i)** and **(ii)** are similar to the proofs of Lemma 3.5 and Theorem 3.6 in [28]. Part **(i)** means that $\mathcal{C}(z_k)$ approaches the boundary of $\Lambda_\epsilon(F)$ as $k \to \infty$. Moreover, part **(ii)** amounts to $\nabla\lambda(z_k)$ pointing in the direction of $(1,0)$ as $k \to \infty$. Thus, eventually $\mathcal{C}(z_k)$ becomes aligned with the points on the boundary of $\Lambda_\epsilon(F)$ with vertical tangent line. The assertions of Theorem 2.5 amounts to the satisfaction of the first order optimality conditions by the sequence $\{z_k\}$ in the smooth and nonsmooth sense, i.e., regardless of the multiplicity of $\sigma_{\min}[F(z_*)]$ whenever the limit $z_* = \lim_{k\to\infty} z_k$ exists. Formally, the first order optimality conditions for (2.1) are given by

$$\exists \mu > 0 \quad \text{s.t.} \quad (1,0) \in \mu \cdot \partial\lambda(\Re z, \Im z) \qquad \text{and} \qquad \lambda(\Re z, \Im z) = 0,$$

where the generalized gradient $\partial\lambda(\Re z, \Im z)$ is defined by [9, page 11]

$$\partial\lambda(\Re z, \Im z) := \mathrm{Co}\left\{ \lim_{k\to\infty} \nabla\lambda(\tilde{z}_k) \;\mid\; \tilde{z}_k \to (\Re z, \Im z), \tilde{z}_k \notin \Omega \right\}, \tag{2.8}$$

the set $\Omega$ is the subset of $\mathbb{R}^2$ (of measure zero) on which $\lambda$ is not differentiable, and $\mathrm{Co}(H)$ denotes the convex hull of the set $H$. The property $\lambda(z_*) = 0$ is immediate from part **(i)** of Theorem 2.5. Additionally, part **(ii)** of Theorem 2.5 implies

$$(1,0) = \lim_{k\to\infty} \frac{\nabla\lambda(z_k)}{\|\nabla\lambda(z_k)\|_2} \;\in\; \mu_* \cdot \partial\lambda(z_*)$$

where $\mu_* := 1/\lim_{k\to\infty} \|\nabla\lambda(z_k)\|_2$, amounting to the satisfaction of the first order optimality conditions.

The assumptions of Theorem 2.5, that $f_j(z_k) \neq 0$ and $\sigma_{\min}[F(z_k)]$ is simple, are satisfied generically. The set of points $z$ where $f_j(z)$ vanish, or $\sigma_{\min}[F(z)]$ is not simple is a subset of $\Omega$ of measure zero. The algorithm never encounters such points in practice. It can generate points close to such points of nonsmoothess, but this has no effect on the convergence of the algorithm. On the other hand, the nonsmoothness at an optimal point has a different nature. Smooth algorithms may not converge to such points. But the algorithm here converges to optimal points regardless of whether they are smooth or not.

We conclude this section with a description of the algorithm below. This description is given in the more general rectangular setting, when $F : \Omega \to \mathbb{C}^{n\times p}$ is analytic on $\Omega$. The $\epsilon$-pseudospectrum can be defined for a rectangular analytic matrix-valued function in a similar fashion by (1.9). The algorithm extends without any modification to this rectangular setting. Throughout this text, in the descriptions of the algorithms we state the termination criteria in exact terms to keep the descriptions neat. Obviously, numerical implementations would require the satisfaction of these conditions up to specified tolerances.

# 3 Vertical Search

It is essential that $\mathcal{C}(z_0) \in \Lambda_\epsilon(F)$ for the locally convergent algorithm of the previous section. In this section, we further impose $\mathcal{C}(z_0)$ to be the rightmost eigenvalue of $F(\lambda)$. This turns out to be essential for global convergence. The sequence $\{z_k\}$ defined by the update rule (2.5), when it converges, yields a point $z_* = (\alpha_*, \beta_*)$ such that

**Algorithm 1** Local Search

**Require:** A matrix-valued function $F : \Omega \to \mathbb{C}^{n\times p}$ analytic on $\Omega$ and a positive scalar $\epsilon \in \mathbb{R}$

$z_0 \leftarrow (\Re z_R, \Im z_R)$, where $z_R$ is any point in $\Lambda_\epsilon(F)$ and $k \leftarrow 0$.
Calculate $\sigma_0 := \sigma_{\min}[F(z_0)]$ and an associated unit right singular vector $v_0$.
Calculate $\lambda_0, \nabla\lambda_0$ using $\sigma_0, v_0, z_0$.
**while** $(\lambda_k \neq 0)$ or $(\nabla\lambda_k \neq c \cdot (1,0)\ \ \forall c \in \mathbb{R}^+)$ **do**
  Apply the recurrence (2.5) to find $z_{k+1}$ given $z_k, \lambda_k, \nabla\lambda_k$.
  Calculate $\sigma_{k+1} := \sigma_{\min}[F(z_{k+1})]$ and an associated unit right singular vector $v_{k+1}$.
  Calculate $\lambda_{k+1}, \nabla\lambda_{k+1}$ using $\sigma_{k+1}, v_{k+1}, z_{k+1}$.
  Increment $k$.
**end while**
**Output:** $z_k$.

**(1)** $\mathcal{C}(z_*)$ is on the boundary of $\Lambda_\epsilon(F)$ with vertical tangent line, or

**(2)** $0 \in \partial\lambda(z_*)$.

Recall that $\partial\lambda(z_*)$ denotes the generalized gradient of $\lambda$ at $z_*$ defined by (2.8). Case **(2)** can occur after finitely many iterations if it happens that $\lambda_k = 0$ and $\nabla\lambda_k = 0$ for some $k$. In this case, $z_\ell = z_k$ for each $\ell > k$ due to the fact that $z_{k+1}$ is the local maximizer of (2.4). In the more probable infinite convergence case, unless $0 \in \partial\lambda(z_*)$, the point $\mathcal{C}(z_*)$ must be on the boundary of $\Lambda_\epsilon(F)$ with a vertical tangent line by Theorem 2.5.

The point $\mathcal{C}(z_*)$ may or may not be a rightmost point globally in $\Lambda_\epsilon(F)$. To check whether $\mathcal{C}(z_*)$ is indeed a rightmost point globally in $\Lambda_\epsilon(F)$, we globally minimize

$$\sigma(\alpha_*, \omega) \ := \ \frac{\sigma_{\min}\left[F(\alpha_*, \omega)\right]}{\|w(\alpha_*, \omega)\|_1} \tag{3.1}$$

over all $\omega \in \mathbb{R}$, which we call a vertical search. This global minimization is achieved by means of the algorithm in [29] for the optimization of a prescribed eigenvalue of a Hermitian and analytic matrix-valued function. If the globally smallest value of $\sigma(\alpha_*, \omega)$ is $\epsilon$, then $\mathcal{C}(z_*)$ is indeed a rightmost point of $\Lambda_\epsilon(F)$ globally. We draw this conclusion based on the assumption that $z_0$ is the rightmost eigenvalue of $F$, and by the fact that each connected component of $\Lambda_\epsilon(F)$ must contain an eigenvalue. If the globally minimal value of $\sigma(\alpha_*, \omega)$ is strictly less than $\epsilon$, then we repeat the locally convergent algorithm of the previous section starting from $(\alpha_*, \omega_*)$, where $\omega_*$ is the computed global minimizer of $\sigma(\alpha_*, \omega)$. The point $\mathcal{C}(\alpha_*, \omega_*)$ lies strictly inside $\Lambda_\epsilon(F)$.

Figure 5 illustrates this vertical search idea combined with the local search of the previous section on a nonlinear delay eigenvalue problem. (Note that the approach in this figure also benefits from the subspace idea, described in detail in the next section.) In this example, the local search converges to a locally rightmost point $(\alpha_*, \omega_*)$ initially. A vertical search determines that there are points with real part equal to $\alpha_*$ that lie strictly inside the $\epsilon$-pseudospectrum. Such a point is given by $(\alpha_*, \omega_{**})$ where $\omega_{**}$ is the global minimizer of $\sigma(\alpha_*, \omega)$ over $\omega$. Thus, the local search is resumed from $(\alpha_*, \omega_{**})$. The local search again leads to a locally rightmost point, which is followed by another vertical search. A third application of the local search ends up at a point that is globally rightmost. This globally rightmost assertion is drawn by a final vertical search. It is determined in this final vertical search that the smallest value of the singular value function $\sigma(\cdot)$ along the dashed vertical line is $\epsilon$.

The algorithm in [29] to minimize $\sigma(\alpha_*, \omega)$ constructs piecewise quadratic functions of the form

$$Q_k(\omega) := \max_{\ell=0,\ldots,k} q_\ell(\omega) \quad \text{where} \quad q_\ell(\omega) := \sigma(\alpha_*, \omega_\ell) + \frac{\partial\sigma(\alpha_*, \omega_\ell)}{\partial\omega}(\omega - \omega_\ell) - \frac{\gamma}{2}(\omega - \omega_\ell)^2$$

and $\gamma$ is required to satisfy $\partial^2\sigma(\alpha_*, \omega)/\partial\omega^2 \geq \gamma$ for all $\omega$ such that $\sigma(\alpha_*, \omega)$ is differentiable. This function is constructed so as to satisfy $\sigma(\alpha_*, \omega) \geq Q_k(\omega)$ for all $\omega$ and $\sigma(\alpha_*, \omega_\ell) = Q_k(\omega_\ell)$ as well as $\partial\sigma(\alpha_*, \omega_\ell)/\partial\omega = Q_k'(\omega_\ell)$ for $\ell = 0, \ldots, k$. The overall algorithm generates the sequence

$\{\omega_k\}$ for a given $\omega_0$ such that $\omega_{k+1} := \arg\min_\omega Q_k(\omega)$. Every convergent subsequence of this sequence is shown to converge to a global minimizer of $\sigma(\alpha_*, \omega)$ in [29]. Thus the algorithm makes use of the derivative

$$\frac{\partial \sigma(\alpha_*,\omega)}{\partial \omega} = \frac{1}{\|w(\alpha_*,\omega)\|_1} \cdot \Re\left(u^* \frac{\partial F(\alpha_*,\omega)}{\partial \omega} v\right) - \frac{1}{\|w(\alpha_*,\omega)\|_1^2} \cdot \frac{\partial \|w(\alpha_*,\omega)\|_1}{\partial \omega} \cdot \sigma_{\min}\left[F(\alpha_*,\omega)\right]$$

where $u, v$ represent a consistent pair of unit left and unit right singular vectors associated with $\sigma_{\min}[F(\alpha_*,\omega)]$, whenever $\sigma_{\min}[F(\alpha_*,\omega)]$ is simple and $f_j(\alpha_*,\omega) \neq 0$ for each $j \in \{0,\ldots,m\}$. Unlike the previous section which offered analytical means to choose $\gamma_\lambda$ and $\gamma_w$, analytical determination of the lower bound $\gamma$ on the second derivatives of $\sigma(\alpha_*,\omega)$ does not seem easy. The additional difficulty is due to a lower bound sought, rather than an upper bound, on the second derivatives of a smallest eigenvalue function. In practice, assigning a large negative real value to $\gamma$ works robustly. A numerical example is given at the end of Section 6.3 to illustrate the effect of the choice of $\gamma$ on the number of iterations of this algorithm for vertical searches.

In special cases, it may be possible to adapt the level set approach proposed to minimize $\sigma_{\min}(A - i\omega I)$ over $\omega \in \mathbb{R}$ in [8] and its quadratically convergent variants [4, 6] for the minimization of $\sigma(\alpha_*,\omega)$ as in (3.1). But the applicability of these approaches depends on the particular form of $F(z)$, in particular the scalar functions $f_j(z)$. For the standard eigenvalue problem (i.e., when $F(z) = A - zI$) and the polynomial eigenvalue problem, such approaches would require the solution of the eigenvalue problems of the same kind but twice the size of the original problem. For the delay eigenvalue problem, this would be give rise to a nonlinear eigenvalue problem of twice the original dimension involving positive and negative powers of $\exp(\lambda)$, see [32] where the level set approach is fully worked out for the problem of the $\mathcal{H}_\infty$ norm computation.

A description of the vertical search combined with the local search is given in Algorithm 2 below. This yields a globally convergent algorithm to compute $\alpha_\epsilon(F)$. Vertical searches also apply regardless of whether $F$ is a square or a rectangular matrix-valued function.

**Algorithm 2** Computation of $\epsilon$-pseudospectral Abscissa for Matrix-Valued Functions

**Require:** A matrix-valued function $F : \Omega \to \mathbb{C}^{n\times p}$ analytic on $\Omega$ and a positive scalar $\epsilon \in \mathbb{R}$

1: $z_0 \leftarrow (\Re z_R, \Im z_R)$, where $z_R$ is a rightmost eigenvalue of $F$.
2: *Convergence* $\leftarrow$ False.
3: **while** $\neg$*Convergence* **do**
4: **Local Search:** Apply Algorithm 1 starting from $z_0$ to find $z_* = (\alpha_*, \beta_*)$ such that $\mathcal{C}(z_*) \in \partial\Lambda_\epsilon(F)$ with a vertical tangent line (or $0 \in \partial\lambda(z_*)$).
5: **Vertical Search:** $\omega_* \leftarrow \arg\min_{\omega\in\mathbb{R}} \sigma(\alpha_*,\omega)$ and $\sigma_* \leftarrow \sigma(\alpha_*,\omega_*)$.
6: **if** $\sigma_* = \epsilon$ **then**
7: *Convergence* $\leftarrow$ True.
8: **else**
9: $z_0 \leftarrow (\alpha_*, \omega_*)$.
10: **end if**
11: **end while**
12: **Output:** $z_*$.

# 4 Subspace Methods

To cope with large scale problems, we consider the map $v \mapsto F(\lambda)v$ when its domain is restricted to a subspace $\mathcal{S}$ of $\mathbb{C}^n$. Let $S$ be an isometry (i.e., $S$ is a matrix with more rows than columns satisfying $S^*S = I$) whose columns form an orthonormal basis for $\mathcal{S}$. The matrix representation of the linear map acting on $\mathcal{S}$ with respect to this basis becomes

$$F_S(\lambda) := F(\lambda)S = \sum_{j=0}^{m} f_j(\lambda) A_j S.$$

Such a subspace idea is introduced in [20] for the computation of the $\epsilon$-pseudospectral abscissa of a matrix, i.e., when $F(\lambda) = \lambda I - A$. Here we extend it to the general setting when $F(\lambda)$ is an analytic matrix-valued function of the form (1.2). In this section, we use the following definitions of the $\epsilon$-pseudospectrum and the $\epsilon$-pseudospectral abscissa of $F_S$:

$$\Lambda_\epsilon(F_S) := \left\{ \lambda \in \mathbb{C} : \ \sigma_{\min}\left( \sum_{j=0}^{m} f_j(\lambda) A_j S \right) \leq \epsilon \|w(\lambda)\|_1 \right\} \quad \text{and}$$
$$\alpha_\epsilon(F_S) := \sup\left\{ \Re\lambda : \ \lambda \in \Lambda_\epsilon(F_S) \right\}.$$

These terminologies and notations are slightly illusive. Indeed, the set $\Lambda_\epsilon(F_S)$ and the quantity $\alpha_\epsilon(F_S)$ are intrinsic to the underlying linear map acting on $\mathcal{S}$, and independent of the choice of the orthonormal basis (given by the columns of $S$) for $\mathcal{S}$. We pursue them in order to remain consistent with the previous sections.

Theorem 4.3 below shows the existence of small subspaces $\mathcal{S}$ such that $\alpha_\epsilon(F) = \alpha_\epsilon(F_S)$, where the columns of the matrix $S$ form an orthonormal basis for $\mathcal{S}$. The notation $\mathcal{V}(z)$ is used in this result for the set consisting of right singular vectors corresponding to $\sigma_{\min}[F(z)]$ for a given $z \in \mathbb{C}$. Additionally, $\mathrm{Col}(S)$ represents the column space of the matrix $S$. Lemma 4.1 below concerning the monotonicity of $\Lambda_\epsilon(F_S)$ with respect to $\mathrm{Col}(S)$ is a generalization of Lemma 3.1 in [20], which specifically addresses the matrix case. Theorem 4.3 generalizes Lemma 3.2 of [20], similarly, from the matrix setting to the nonlineat eigenvalue setting.

**Lemma 4.1** (Monotonicity)**.** *Two isometries $S_1, S_2$ such that $\mathrm{Col}(S_1) \subseteq \mathrm{Col}(S_2)$ satisfy*

***(1)*** $\Lambda_\epsilon(F_{S_1}) \subseteq \Lambda_\epsilon(F_{S_2})$ and ***(2)*** $\alpha_\epsilon(F_{S_1}) \leq \alpha_\epsilon(F_{S_2})$.

*Proof.* Let $\mathcal{S}_j := \mathrm{Col}(S_j)$ for $j = 1, 2$. Suppose $z \in \Lambda_\epsilon(F_{S_1})$, that is

$$\sigma_{\min}\left( \sum_{j=0}^{m} f_j(z) A_j S_1 \right) \leq \epsilon \|w(z)\|_1 \tag{4.1}$$

holds. Notice that Supposing

$$\sigma_{\min}\left( \sum_{j=0}^{m} f_j(z) A_j S_1 \right) = \min_{v \in \mathcal{S}_1, \|v\|_2 = 1} \left\| \sum_{j=0}^{m} f_j(z) A_j v \right\|_2 \geq$$
$$\min_{v \in \mathcal{S}_2, \|v\|_2 = 1} \left\| \sum_{j=0}^{m} f_j(z) A_j v \right\|_2 = \sigma_{\min}\left( \sum_{j=0}^{m} f_j(z) A_j S_2 \right),$$

where the inequality is due to $\mathcal{S}_1 \subseteq \mathcal{S}_2$. Combining this with inequality (4.1), we deduce that $z \in \Lambda_\epsilon(F_{S_2})$ proving **(1)**. Furthermore, **(2)** is an immediate consequence of **(1)**. □

**Lemma 4.2.** *For $z \in \Lambda_\epsilon(F)$ and a unit right singular vector $v$ associated with $\sigma_{\min}[F(z)]$, we have $z \in \Lambda_\epsilon(F_v)$.*

*Proof.* This is immediate from

$$\epsilon \|w(z)\|_1 \ \geq \ \sigma_{\min}\left( \sum_{j=0}^{m} f_j(z) A_j \right) = \left\| \sum_{j=0}^{m} f_j(z) A_j v \right\|_2 = \sigma_{\min}\left( \sum_{j=0}^{m} f_j(z) A_j v \right).$$

□

**Theorem 4.3** (Low Dimensionality)**.** *Suppose that $S$ is an isometry. Then $\alpha_\epsilon(F) = \alpha_\epsilon(F_S)$ if and only if $\mathcal{V}(z_*) \cap \mathrm{Col}(S) \neq \emptyset$ for some globally rightmost point $z_*$ of $\Lambda_\epsilon(F)$.*

*Proof.* First observe that $\alpha_\epsilon(F) = \alpha_\epsilon(F_{v_*})$ for any unit right singular vector $v_*$ associated with $\sigma_{\min}[F(z_*)]$. This is due to $\alpha_\epsilon(F) \geq \alpha_\epsilon(F_{v_*})$ by Lemma 4.1, as well as Lemma 4.2 which implies $z_* \in \Lambda_\epsilon(F_{v_*})$, so $\alpha_\epsilon(F) = \Re z_* \leq \alpha_\epsilon(F_{v_*})$.

Suppose $\mathcal{V}(z_*) \cap \mathrm{Col}(S) \neq \emptyset$ for some globally rightmost point $z_*$ of $\Lambda_\epsilon(F)$. Consider any $v_* \in \mathcal{V}(z_*) \cap \mathrm{Col}(S)$, which we assume a unit vector without loss of generality. We have $\mathrm{span}\{v_*\} \subseteq \mathrm{Col}(S)$, so by Lemma 4.1, we obtain

$$\alpha_\epsilon(F_{v_*}) \leq \alpha_\epsilon(F_S) \leq \alpha_\epsilon(F).$$

Thus the equality $\alpha_\epsilon(F_{v_*}) = \alpha_\epsilon(F)$ leads us to $\alpha_\epsilon(F_S) = \alpha_\epsilon(F)$.

To prove the converse, suppose $\alpha_\epsilon(F) = \alpha_\epsilon(F_S)$. Denote a globally rightmost point of $\Lambda_\epsilon(F_S)$ with $z_S$. Due to Lemma 4.1, we have $z_S \in \Lambda_\epsilon(F)$ and $\Re z_S = \alpha_\epsilon(F_S) = \alpha_\epsilon(F)$, so $z_S$ is also a rightmost point of $\Lambda_\epsilon(F)$ globally. Furthermore, letting $v_S$ be a unit right singular vector associated with $\sigma_{\min}[F_S(z_S)]$, observe

$$\left\| \sum_{j=0}^{m} f_j(z_S) A_j S v_S \right\|_2 = \sigma_{\min}[F_S(z_S)] = \sigma_{\min}[F(z_S)].$$

Thus $Sv_S$ is a unit right singular vector associated with $\sigma_{\min}[F(z_S)]$, that is $\mathcal{V}(z_S) \cap \mathrm{Col}(S) \neq \emptyset$, completing the proof. □

Theorem 4.3 gives the initiative to work on an $n \times p$ matrix-valued function $F_S$ for $p \ll n$ and for a properly chosen subspace $\mathcal{S} = \mathrm{Col}(S)$ (rather than working on the full $n \times n$ matrix-valued function $F$). A natural choice for $\mathcal{S}$ appears to be the span of right singular vectors of $\sigma_{\min}[F(z)]$ for various $z \in \mathbb{C}$ close to globally rightmost points of $\Lambda_\epsilon(F)$. The following observations lead us to this choice: **(i)** *the right singular vectors of* $\sigma_{\min}[F(z)]$ *are continuous w.r.t.* $z$ - if $z \approx z_*$, then $\mathcal{V}(z) \approx \mathcal{V}(z_*)$; **(ii)** $\alpha_\epsilon(F_S)$ *is continuous w.r.t.* $\mathrm{Col}(S)$ - if $\mathrm{Col}(S) \approx \mathrm{Col}(S_*)$ for any isometry $S_*$ such that $\mathrm{Col}(S_*) \cap \mathcal{V}(z_*) \neq \emptyset$, then $\alpha_\epsilon(F_S) \approx \alpha_\epsilon(F_{S_*}) = \alpha_\epsilon(F)$. Initially, we could consider the rightmost eigenvalue $z_R$ of $F$ as a good approximation for the globally rightmost point of $\Lambda_\epsilon(F)$ (this is especially true for $\epsilon \approx 0$). Setting $z_0 := z_R$, we could then generate a sequence $\{z_k\}$ in $\mathbb{C}$ such that

**Framework 1 (Subspace Selection Based on Smallest Singular Value)**

**(1)** $S_k$ is an isometry s.t. its columns form an orthonormal basis for $\mathcal{S}_k := \mathrm{span}\{v_0, \ldots, v_k\}$, where $v_j$ is a right singular vector corresponding to $\sigma_{\min}[F(z_j)]$,

**(2)** $z_{k+1}$ is a rightmost point of $\Lambda_\epsilon(F_{S_k})$

for $k \in \mathbb{N}$.

**Remark.** *There is a natural alternative to this way of choosing subspaces. Theorem 4.3 could be interpreted in terms of eigenvectors corresponding to rightmost eigenvalues of perturbed matrix-valued functions $F + \Delta$. In [31, Proposition 3.1], it is shown that each $z \in \mathbb{C}$ on the boundary of $\Lambda_\epsilon(F)$ is an eigenvalue of*

$$(F + \Delta_z)(\lambda) := \sum_{j=0}^{m} f_j(\lambda)(A_j + \delta A_{z,j}) \quad \text{where } \delta A_{z,j} := -\frac{\epsilon \cdot \overline{f_j(z)}}{w_j \cdot |f_j(z)|} uv^*,$$

*and $u, v$ are consistent unit left, unit right singular vectors corresponding to $\sigma_{\min}[F(z)]$. Indeed, $u, v$ are left, right eigenvectors associated with the eigenvalue $z$ of $F + \Delta_z$. Furthermore, if $z_*$ is a globally rightmost point of $\Lambda_\epsilon(F)$, then it is a rightmost eigenvalue of $F + \Delta_{z_*}$. Thus, $\mathcal{V}(z_*)$ in Theorem 4.3 could be interpreted as the set of all eigenvectors associated with a rightmost eigenvalue of $F + \Delta_{z_*}$. These observations suggest forming the subspace $\mathcal{S}$ from eigenvectors associated with a rightmost eigenvalue of $F + \Delta_z$ for various $z$ close to globally rightmost points of $\Lambda_\epsilon(F)$. Setting $z_0 := z_R$ the rightmost eigenvalue of $F$, an alternative sequence $\{z_k\}$ in $\mathbb{C}$ is defined by:*

***Alternative Framework for Subspace Selection***

***(1)*** *$S_k$ is an isometry s.t. its columns form an orthonormal basis for $\mathcal{S}_k := \operatorname{span}\{v_0, \ldots, v_k\}$, where $v_j$ is an eigenvector associated with a rightmost eigenvalue of $F + \Delta_{z_j}$,*

***(2)*** *$z_{k+1}$ is a rightmost point of $\Lambda_\epsilon(F_{S_k})$*

*for $k \in \mathbb{N}$. Note that the alternative framework above requires the solution of a nonlinear eigenvalue problem in every iteration, whereas Framework 1 only involves standard singular value problems. Furthermore, numerical estimation of the rightmost eigenvalue of a nonlinear eigenvalue problem is, in general, harder than that of the smallest singular value of a matrix. Thus, we abandon this alternative framework, rather we adopt Framework 1 based on standard smallest singular value computations.*

Below, we illustrate how the subspace idea can be put in use for large-scale $F$ by coupling Framework 1 above with Algorithm 1 of Section 2. This results in Algorithm 3 that returns a locally rightmost point of $\Lambda_\epsilon(F)$. Approaches other than Algorithm 1 of Section 2 can be employed at the local search stage on line 5 to determine a point on the boundary of $\Lambda_\epsilon(F_{S_k})$ with the vertical tangent line. For instance, it is possible to adopt the approach of [31], which would require the rightmost eigenvalues of rank one perturbations of the original matrix-valued function. As mentioned above, the numerical computation of a rightmost nonlinear eigenvalue is usually harder than the computation of a smallest singular value, on which Algorithm 1 is based. Consequently, we rely on Algorithm 1 for this local search. In the description of this algorithm, we adopt the notation $\sigma(\Re z, \Im z) := \sigma_{\min}\left[F(\Re z, \Im z)\right] / \|w(\Re z, \Im z)\|_1$. We disregard the possibility that the local search (on line 5) converges to a point where the generalized gradient of $\lambda$ (associated with $F_{S_k}$) contains zero, which is extremely unlikely. Such an unlikely case can be dealt with for instance by occasional vertical searches at additional cost. Furthermore, when $\sigma(z_{k+1}) = \epsilon$ holds, the condition $c \cdot (1,0) \in \partial\sigma(z_{k+1}) \;\; \exists c \in \mathbb{R}^+$ on line 6 (recall that $\partial\sigma(z_{k+1})$ represents the generalized gradient of $\sigma$ at $z_{k+1}$) amounts to having a vertical tangent line on the boundary of $\Lambda_\epsilon(F)$ at $z_{k+1}$ in the nonsmooth sense. In the smooth case, when $\sigma(z_{k+1})$ is simple, this condition reduces to $\nabla\sigma(z_{k+1}) = c \cdot (1,0) \;\; \exists c \in \mathbb{R}^+$.

**Algorithm 3** Large-Scale Local Search

**Require:** A matrix-valued function $F : \Omega \to \mathbb{C}^{n\times n}$ analytic on $\Omega$ and a positive scalar $\epsilon \in \mathbb{R}$

1: $z_0 \leftarrow$ a rightmost eigenvalue of $F$ and $k \leftarrow 0$.
2: $\mathcal{S}_0 \leftarrow \operatorname{span}\{v_0\}$, where $v_0$ is a right singular vector associated with $\sigma_{\min}\left[F(z_0)\right]$.
3: *Convergence* $\leftarrow$ False.
4: **while** $\neg$*Convergence* **do**
5: **Local Search:** Apply Algorithm 1 to find $z_{k+1} = (\alpha_*, \beta_*)$ such that $\mathcal{C}(z_{k+1}) \in \partial\Lambda_\epsilon(F_{S_k})$ with a vertical tangent line.
6: **if** $(\sigma(z_{k+1}) = \epsilon) \quad$ and $\quad (c \cdot (1,0) \in \partial\sigma(z_{k+1}) \;\; \exists c \in \mathbb{R}^+) \quad$ **then**
7: *Convergence* $\leftarrow$ True.
8: **else**
9: $\mathcal{S}_{k+1} \leftarrow \operatorname{span}\left(\mathcal{S}_k \cup \{v_{k+1}\}\right)$, where $v_{k+1}$ is a right singular vector associated with $\sigma_{\min}\left[F(z_{k+1})\right]$.
10: Increment $k$.
11: **end if**
12: **end while**
13: **Output:** $z_{k+1}$.

# 5 Restarts

In the subspace $\mathcal{S}_k = \operatorname{span}\{v_0, \ldots, v_k\}$ of Framework 1 the vectors added lately are more relevant to the set of optimal right singular vectors $\mathcal{V}(z_*)$ of Corollary 4.3. This is because the sequence

$\{\Re z_k\}$ is monotone increasing and later points in $\{z_k\}$ usually represent the optimal $z_*$ better. This brings up a subspace-restart idea: when the subspace $\mathcal{S}_k$ becomes of high dimension, erase the earlier vectors and keep the lastly added few vectors among $v_0, \ldots, v_k$, possibly only $v_k$. Thus redefine $\mathcal{S}_j := \text{span}\{v_{k-j}, \ldots, v_k\}$ and restart. Such restart strategies have already been employed in the context of large scale eigenvalue computation based on Krylov subspace methods [24], and incorporated into modern software, for instance ARPACK [25].

The vertical searches described in Section 3 can also benefit from this restart strategy. A vertical search, when it determines that a point is not globally rightmost, also provides a new point $z_0$ to start with. Then the subspace could be reset to $\text{span}\{v_0\}$ where $v_0$ is a right singular vector associated with $\sigma_{\min}[F(z_0)]$.

## 5.1 Quality of Subspace Approximations

To further motivate the restart strategy and discarding poor approximations in the subspace, below we relate the quality of a subspace $\mathcal{S}_k$ (specifically its proximity to $v_*$) with the quality of $\alpha_\epsilon(F_{S_k})$ (specifically its proximity to $\alpha_\epsilon(F)$). Here and throughout this subsection, $S_k$ is a matrix whose columns form an orthonormal basis for $\mathcal{S}_k$, and $z_*$ is as defined in Theorem 4.3. Furthermore, let $u_*, v_*$ be consistent unit left, right singular vectors associated with $\sigma_{\min}[F(z_*)]$, that is

$$F(z_*)v_* = \sigma_{\min}[F(z_*)]\,u_* \quad \text{and} \quad u_*^* F(z_*) = \sigma_{\min}[F(z_*)]\,v_*^*.$$

We measure the quality of $\mathcal{S}_k$ in terms of $\delta = v_* - v_{k*}$ where $v_{k*} := \arg\min_{v\in\mathcal{S}_k, \|v\|_2=1} \|v_* - v\|_2$, and focus on the error $\alpha_\epsilon(F) - \alpha_\epsilon(F_{S_k})$ as $\delta \to 0$. This issue of relating the quality of the restricted pseudospectral abscissa to the quality of the subspace has been addressed in the more simple matrix setting by [20, Theorem 3.3].

Our approach has two stages. In the first stage, we establish

$$\|F(z_*)v_{k*}\|_2 / \|w(z_*)\|_1 = \epsilon + O\left(\|\delta\|_2^2\right). \tag{5.1}$$

In the second stage, starting from this equality, we deduce the existence of a $z_{k*} \in \mathbb{C}$ satisfying

$$\|F(z_{k*})v_{k*}\|_2 / \|w(z_{k*})\|_1 = \epsilon \tag{5.2}$$

and $\Re z_* - \Re z_{k*} = O(\|\delta\|_2^2)$. The last equality implies that $z_{k*} \in \Lambda_\epsilon(F_{S_k})$, meaning $\Re z_{k*} \le \alpha_\epsilon(F_{S_k})$. The desired relation between the $\epsilon$-pseudospectral abscissa of $F$ and $F_{S_k}$ follows from $\alpha_\epsilon(F) - \alpha_\epsilon(F_{S_k}) \le \Re z_* - \Re z_{k*}$.

To prove equality (5.1), let us define the vector-valued function

$$v : \mathbb{R} \to \mathbb{R}^n, \quad v(t) := \left[v_* + \frac{v_{k*} - v_*}{\|v_{k*} - v_*\|_2} t\right] \Big/ \left\| v_* + \frac{v_{k*} - v_*}{\|v_{k*} - v_*\|_2} t \right\|_2,$$

and the scalar function

$$\mu : \mathbb{R} \to \mathbb{R}, \quad \mu(t) := \|F(z_*)v(t)\|_2,$$

which is real analytic near 0. We benefit from a Taylor expansion of $\mu(t)$ about 0 to obtain (5.1). Specifically, since $v(\|\delta\|_2) = v(\|v_{k*} - v_*\|_2) = v_{k*}$, we have

$$\begin{aligned} \frac{\|F(z_*)v_{k*}\|_2}{\|w(z_*)\|_1} &= \frac{\|F(z_*)v(\|\delta\|_2)\|_2}{\|w(z_*)\|_1} = \frac{\mu(\|\delta\|_2)}{\|w(z_*)\|_1} \\ &= \frac{\mu(0) + \mu'(0)\|\delta\|_2 + O(\|\delta\|_2^2)}{\|w(z_*)\|_1}. \end{aligned}$$

The desired equality (5.1) follows from the observations $\mu(0) = \sigma_{\min}[F(z_*)] = \epsilon\|w(z_*)\|_1$ (since $v(0) = v_*$), and

$$\mu'(0) = \Re\left(u_*^* F(z_*) v'(0)\right) = \Re\left(\sigma_{\min}[F(z_*)]\, v_*^* v'(0)\right) = \epsilon\|w(z_*)\|_1 \Re\left(v_*^* v'(0)\right) = 0,$$

where $\Re\left(v_*^* v'(0)\right) = 0$ due to $\|v(t)\|_2^2 = 1$ for all $t$.

To establish (5.2) for some $z_{k*} \in \mathbb{C}$ such that $\Re z_* - \Re z_{k*} = O(\|\delta\|_2^2)$, let us suppose that $\sigma_{\min}[F(z_*)]$ is simple, and $f_j(z_*) \neq 0$ for $j = 0, \ldots, m$. In this case, all of the functions

$$\sigma(\Re z, \Im z) := \frac{\sigma_{\min}[F(\Re z, \Im z)]}{\|w(\Re z, \Im z)\|_1}, \quad \hat{\sigma}(\Re z, \Im z) := \frac{\|F(\Re z, \Im z)v_*\|_2}{\|w(\Re z, \Im z)\|_1}$$
$$\text{and} \quad \widetilde{\sigma}(\Re z, \Im z) := \frac{\|F(\Re z, \Im z)v_{k*}\|_2}{\|w(\Re z, \Im z)\|_1}$$

are continuously differentiable at $(\Re z, \Im z) = (\Re z_*, \Im z_*)$. Target equality (5.2) can be written as $\widetilde{\sigma}(\Re z_{k*}, \Im z_{k*}) = \epsilon$, whereas (5.1) can be expressed as

$$\widetilde{\sigma}(\Re z_*, \Im z_*) = \epsilon + O\left(\|\delta\|_2^2\right). \tag{5.3}$$

Assuming that the gradient of $\sigma(\Re z, \Im z)$ does not vanish at $(\Re z_*, \Im z_*)$, the first order optimality conditions imply

$$\nabla\sigma(\Re z_*, \Im z_*) = \nabla\hat{\sigma}(\Re z_*, \Im z_*) = c \cdot (1, 0) \tag{5.4}$$

for some positive $c \in \mathbb{R}$. But then, by continuity and the second equality in (5.4), we have

$$\eta := p^T \nabla\widetilde{\sigma}(\Re z_*, \Im z_*) < 0, \qquad \text{where } p = -(1, 0). \tag{5.5}$$

Employing (5.3) and due to (5.5), there exists $\alpha > 0$ such that

$$\widetilde{\sigma}(\Re z_* - \alpha, \Im z_*) = \frac{\|F(\Re z_* - \alpha, \Im z_*)v_{k*}\|_2}{\|w(\Re z_* - \alpha, \Im z_*)\|_1} = \epsilon.$$

Defining $z_{k*}$ by $\Re z_{k*} := \Re z_* - \alpha$ and $\Im z_{k*} := \Im z_*$, we deduce (5.2). To quantify $\alpha = \Re z_* - \Re z_{k*}$, we expand $\widetilde{\sigma}(\Re z, \Im z)$ about $(\Re z_*, \Im z_*)$ only varying $\Re z$ (note that $\widetilde{\sigma}(\Re z, \Im z_*)$ is real analytic with respect to $\Re z$ near $\Re z_*$). This leads to

$$\widetilde{\sigma}(\Re z_* - \alpha, \Im z_*) = \widetilde{\sigma}(\Re z_*, \Im z_*) + \eta\alpha + O(\alpha^2) \quad \Longrightarrow \quad \alpha = O(\|\delta\|_2^2).$$

Finally, since $v_{k*} \in \mathcal{S}_k$, we have

$$\frac{\sigma_{\min}[F_{S_k}(z_{k*})]}{\|w(z_{k*})\|_1} \leq \frac{\|F(z_{k*})v_{k*}\|_2}{\|w(z_{k*})\|_1} = \epsilon$$

meaning $z_{k*} \in \Lambda_\epsilon(F_{S_k})$. From $\alpha_\epsilon(F) = \Re z_*$ and $\alpha_\epsilon(F_{S_k}) \geq \Re z_{k*}$, we obtain $\alpha_\epsilon(F) - \alpha_\epsilon(F_{S_k}) \leq \Re z_* - \Re z_{k*} = \alpha$. Hence,

$$\alpha_\epsilon(F) - \alpha_\epsilon(F_{S_k}) = O(\|\delta\|_2^2).$$

## 5.2 Overall Algorithm

We apply the subspace method, specifically Framework 1 in Section 4. Initially, $z_0$ is chosen as the rightmost eigenvalue of $F$, and $\mathcal{S}_0$ is the associated one-dimensional subspace. The subspace method requires the determination of a rightmost point of $\Lambda_\epsilon(F_{S_k})$ for several $k$, each of which we achieve by the local algorithm in Section 2. In practice, this results in convergence to a point $z_* = (z_{*1}, z_{*2}) \in \mathbb{R}^2$ such that $\mathcal{C}(z_*)$ is (up to a tolerance) on the boundary of $\Lambda_\epsilon(F)$ with a vertical tangent line for a rather small subspace $\mathcal{S}_k$. We do not allow $\mathcal{S}_k$ to expand arbitrarily and restart with a one-dimensional subspace once its dimension reaches a prescribed value. We perform the vertical search discussed in Section 3 along the line $\{z \in \mathbb{C} \mid \Re z = z_{*1}\}$. The vertical search is performed on the full matrix-valued function $F$ in order to ensure global convergence. Termination occurs if this vertical search yields $\epsilon$ as the smallest value of the singular value function involved (up to a specified tolerance in practice). Otherwise, we restart the subspace method from a global minimizer on the vertical line and with the associated one-dimensional subspace. A detailed description is given in Algorithm 4 below.

The computational work is usually dominated by the vertical search on line 7, which is needed only a few times in practice. Each vertical search requires the computation of the smallest singular value of the full matrix-valued function $F(z)$ at several $z$, and this constitutes the main computational burden. On the other hand, the local search on line 5 is performed quite a few times. But their computational cost is quite low, since they involve small matrix-valued functions.

**Algorithm 4** Computation of $\epsilon$-pseudospectral Abscissa for Large-Scale Matrix-Valued Functions

**Require:** A matrix-valued function $F : \Omega \to \mathbb{C}^{n\times n}$ analytic on $\Omega$, a positive scalar $\epsilon \in \mathbb{R}$ and the maximal subspace dimension allowed $k_{\max} \in \mathbb{Z}^+$

$z_0 \leftarrow$ a rightmost eigenvalue of $F$ and $k \leftarrow 0$.
$\mathcal{S}_0 \leftarrow \text{span}\{v_0\}$, where $v_0$ is a right singular vector associated with $\sigma_{\min}[F(z_0)]$.
*Convergence* $\leftarrow$ False.
**while** $\neg$*Convergence* **do**
  **Local Search:** Apply Algorithm 1 to find $z_{k+1} = (\alpha_*, \beta_*)$ such that $\mathcal{C}(z_{k+1}) \in \partial\Lambda_\epsilon(F_{S_k})$ with a vertical tangent line.
  **if** $(\sigma(z_{k+1}) = \epsilon)$ and $(c \cdot (1, 0) \in \partial\sigma(z_{k+1}) \ \exists c \in \mathbb{R}^+)$ **then**
    **Vertical Search:** $\omega_* \leftarrow \arg\ \min_{\omega\in\mathbb{R}} \sigma(\alpha_*, \omega)$ and $\sigma_* \leftarrow \sigma(\alpha_*, \omega_*)$.
    **if** $\sigma_* = \epsilon$ **then**
      *Convergence* $\leftarrow$ True.
    **else**
      $z_0 \leftarrow \mathcal{C}(\alpha_*, \omega_*)$ and $k \leftarrow 0$.
      $\mathcal{S}_0 \leftarrow \text{span}\{v_0\}$, where $v_0$ is a right singular vector associated with $\sigma_{\min}[F(z_0)]$.
    **end if**
  **else**
    **if** $k = k_{\max}$ **then**
      $z_0 \leftarrow z_{k+1}$ and $k \leftarrow 0$.
      $\mathcal{S}_0 \leftarrow \text{span}\{v_0\}$, where $v_0$ is a right singular vector associated with $\sigma_{\min}[F(z_0)]$.
    **else**
      $\mathcal{S}_{k+1} \leftarrow \text{span}(\mathcal{S}_k \cup \{v_{k+1}\})$, where $v_{k+1}$ is a right singular vector associated with $\sigma_{\min}[F(z_{k+1})]$.
      Increment $k$.
    **end if**
  **end if**
**end while**
**Output:** $z_{k+1}$.

# 6 Numerical Examples

A practical implementation of Algorithm 4 requires several parameters, most notably the upper and lower bounds for the second derivatives of the eigenvalue functions involved for local and vertical searches, as well as the maximal subspace dimension. In the examples in this section and for local searches, we set the upper bound $\gamma$ equal to 2 for the standard eigenvalue problem, and 40000 for other nonlinear eigenvalue problems. The former is an analytical upper bound, while the latter is a heuristic which works well in practice. For the latter case, alternatively, the bounds implied by Theorem 2.2 and 2.3 could be used. We set the lower bound $\gamma$ for vertical searches equal to -4, which again works well in our experience. The maximal subspace dimension is by default chosen as $n/4$ for an $n \times n$ matrix-valued function. This excludes the delay example in Section 6.3 for which the maximal subspace dimension is chosen as 10.

We always start the algorithm with the rightmost eigenvalue computed by Matlab for the standard and polynomial eigenvalue problems. This is computed by means of the QR algorithm (i.e., eig command in Matlab) for matrices of size up to 300, and by Arnoldi's method (i.e., eigs command in Matlab) for matrices of size beyond 300. We linearize the matrix polynomials and compute the eigenvalues of linearizations by means of the QR algorithm for small scale problems (i.e, polyeig command in Matlab). For large scale problems the current state of the art CORK algorithm [40] should be used. This algorithm fully exploits the Kronecker structure of the linearization pencils. For other nonlinear eigenvalue problems, we expect the user to specify the rightmost eigenvalue as an input parameter. This eigenvalue can for example be computed via the software NLEIGS [15] for generic nonlinear eigenvalue problems, or by the algorithm of [45] for the delay eigenvalue problem.

## 6.1 Standard Eigenvalue Problem

Algorithm 4 is especially suitable for the computation of the pseudospectral abscissa for large-scale matrices, that is when $F(z) = A - zI$ for a given large matrix $A$ and weights are given by $[1\ \infty]$. The *criss-cross algorithm* [7] is the most reliable choice for the computation of the pseudospectral abscissa of a matrix at the moment, but its use is limited to mainly small up to medium scale matrices. We compare Algorithm 4 with the criss-cross algorithm in this subsection.

In our numerical experiments, Algorithm 4 in practice is terminated after a vertical search, whenever it is determined that the globally smallest value of $\sigma_{\min}[A - (\alpha_* + \omega i)I]$ over $\omega$ (for an $\alpha_*$ converged by the local search algorithm) does not differ from $\epsilon$ by more than a tolerance, $10^{-6}\|A\|_2$ for the examples below. The criss-cross algorithm also performs vertical searches, but they are based on extracting all imaginary eigenvalues of $2n \times 2n$ Hamiltonian matrices from which the intersection points of a vertical line with the $\epsilon$-pseudospectrum boundary are inferred. It terminates either if a vertical search fails to find any intersection point or if two consecutive estimates for the $\epsilon$-pseudospectral abscissa are not increasing due to rounding errors.

We start with a random $50 \times 50$ matrix formed by typing `randn(50) + 0.7*i*randn(50)` in Matlab. Algorithm 4 applies the subspace iteration initially. Each subspace iteration amounts to a local search on a small problem. When the subspace becomes eight dimensional, it stops expanding as the local searches on seven and eight dimensional subspaces return nearly identical rightmost points. Instead, it performs a vertical search and terminates. In Figure 1, the progress of the subspace iteration on this example is shown for two, four and six dimensional subspaces. The results of Algorithm 4 and the criss-cross algorithm match up to 12 decimal digits.

The next three sets of examples illustrate the superiority of Algorithm 4 to the criss-cross algorithm for medium to large scale matrices. All these examples can be generated using EigTool [44]. Each one of the three test sets consists of four matrices of size 200, 400, 800 and 1200 chosen from a particular family. The matrices in the first set are Landau matrices arising from an integral equation in laser theory [23]. The matrices in the second set are Hatano-Nelson matrices, which are tridiagonal and arise from quantum mechanics [16]. The matrices in the third set are Davies matrices originating from a spectral method discretization of an anharmonic oscillator, i.e., a second order differential operator subject to boundary conditions in 1-dimension [10]. A comparison of running times of the algorithms are given in Table 1. In all cases, Algorithm 4 becomes superior in terms of the running times, as soon as $n$ (the size of the matrix) is slightly larger than 200. The gap grows quickly as $n$ increases. Furthermore, roughly the quadratic dependence of the running time for Algorithm 4 on the sizes of the matrices is apparent. This is due to the fact that the required smallest eigenvalues and singular values, and the corresponding eigenvectors and singular vectors are computed by means of ARPACK [25], which is based on Arnoldi's method. The computed results by the two algorithms differ by amounts on the order of the double machine precision in a relative sense. More precisely, denoting the results returned by Algorithm 4 and the criss-cross algorithm with $f_1$ and $f_2$, respectively, and the 2-norm of the input matrix with $\|A\|_2$, the quantity $|f_1 - f_2|/\|A\|_2$ does not exceed $10^{-14}$.

An alternative for the local searches (Algorithm 1) in this matrix setting is the approach suggested in [14], which computes repeatedly the rightmost eigenvalues of rank one perturbations of the matrix whose $\epsilon$-pseudospectral abscissa is sought. We have performed comparisons of Algorithm 1 and this approach on four family of matrices (without subspace restrictions and vertical searches). Two of these families are comprised of dense matrices, namely the Landau and Hatano matrices of Table 1. The other two families are the Poisson matrices, and Wathen matrices, which are sparse and available through Matlab gallery. The Poisson matrices arise from the application of the five-point finite difference formula to the 2-dimensional Poisson equation, whereas the Wathen matrices arise from a two dimensional finite element discretization. In our experiments, we have observed that the particular implementation* of the algorithm in [14] that we rely on has terminated with failure occasionally on the sparse Poisson and Wathen matrices, which we attribute to the Arnoldi solver (i.e., eigs in Matlab) to retrieve

*PSAPSR version 1.2 that is available at http://www.cs.nyu.edu/overton/software/psapsr/index.html

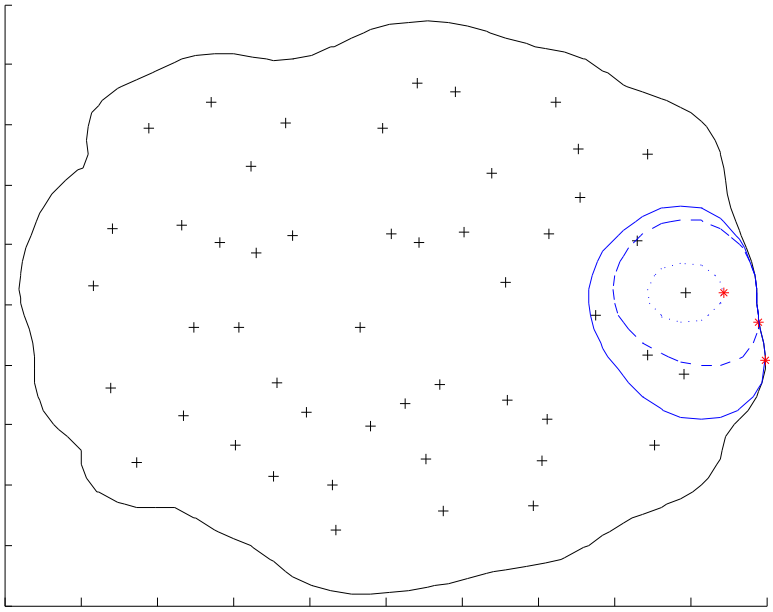


Figure 1: Subspace iteration on a $50 \times 50$ random matrix is displayed. The outermost black curve represents the boundary of the $\epsilon$-pseudospectrum of this matrix for $\epsilon = 1$, while each $+$ represents an eigenvalue. The $\epsilon$-pseudospectrum for the restricted problem $F_{S_k}$ is shown with dotted, dashed and solid blue curves for $k = 2, 4, 6$, respectively. The red asterisks mark the rightmost points for these restricted problems.

the rightmost eigenvalue. On successful attempts, the running times of the two approaches together with their number of iterations are listed in Table 2. The particular implementation of [14] that we depend on uses the QR algorithm (i.e., calls eig) for dense matrices. This is partly the reason for the excessive amount of time this algorithm takes on the Landau and Hatano matrices. But beyond the running time considerations, this algorithm also requires quite a few additional iterations compared to Algorithm 1 on these dense matrices. For the $200 \times 200$ Landau and Hatano matrices, first few iterations of Algorithm 1 and Algorithm in [14] are given in Table 3. Faster convergence of Algorithm 1 is apparent from this table. It requires only 6 and 5 iterations so that two consecutive iterations do not differ by more than $10^{-8}$ for the Landau and Hatano matrices, respectively. On the other hand, the algorithm in [14] performs 46 and 24 iterations to satisfy the same criterion. On the sparse Poisson and Wathen matrices, two algorithms exhibit similar convergence behavior, both require only a few iterations. For matrices on the order of ten thousands in Table 1, the algorithm in [14] runs faster than Algorithm 1. We attribute this to the fact that the estimation of a smallest singular value based on the Arnoldi iteration requires computation of a sparse Cholesky or a LU factorization followed by sparse forward and back substitutions, which may not be needed to compute a rightmost eigenvalue. But this comes at the expense of reliability. The particular implementation of [14] that we use terminates with failure in most of the attempts on the Poisson and Wathen matrices.

## 6.2 Polynomial Eigenvalue Problem

In this subsection, we experiment with several polynomial eigenvalue problems available in the collection [3]. In all of these experiments, Algorithm 4 is terminated in practice whenever a vertical search determines that the globally smallest value of $\sigma_{\min}\left[F(\alpha_*, \omega)\right] / \|w(\alpha_*, \omega)\|_1$ over $\omega$ does not differ from $\epsilon$ by more than $10^{-6}\|A_m\|_2$, where $A_m$ denotes the leading coefficient matrix of the matrix polynomial $F(\lambda) := \sum_{j=0}^{m} \lambda^j A_j$ whose $\epsilon$-pseudospectral abscissa is sought. Furthermore, in all experiments in this subsection, all weights are set equal to one, unless

| **Landau**, $\epsilon = 10^{-0.5}$ | 200 | 400 | 800 | 1200 |
|---|---|---|---|---|
| Algorithm 4 | 7 | 20 | 78 | 176 |
| Criss-Cross Algorithm | 7 | 43 | 223 | 662 |
| $\alpha_\epsilon$ | 1.3153 | 1.3161 | 1.3161 | 1.3161 |

| **Hatano**, $\epsilon = 1$ | 200 | 400 | 800 | 1200 |
|---|---|---|---|---|
| Algorithm 4 | 14 | 29 | 89 | 207 |
| Criss-Cross Algorithm | 7 | 46 | 2030 | |
| $\alpha_\epsilon$ | 4.0765 | 4.0903 | 4.0678 | 4.1474 |

| **Davies**, $\epsilon = 10^5$ | 200 | 400 | 800 | 1200 |
|---|---|---|---|---|
| Algorithm 4 | 8 | 9 | 19 | 40 |
| Criss-Cross Algorithm | 3 | 41 | 223 | |
| $\alpha_\epsilon$ | $4.0355 \cdot 10^5$ | $4.8867 \cdot 10^6$ | $7.6266 \cdot 10^7$ | $3.8504 \cdot 10^8$ |

Table 1: Running times for the algorithms in seconds with respect to the sizes of the matrices, and the computed pseudospectral abscissa; The running times of the criss-cross algorithm are omitted for Hatano and Davies matrices of size 1200, since its computations take excessive time.

| **Landau**, $\epsilon = 10^{-0.5}$ | 200 | 400 | 800 | 1200 |
|---|---|---|---|---|
| Algorithm 1 | 0.8, 5 | 2.3, 5 | 10.7, 5 | 21.2, 5 |
| Algorithm in [14] | 25.5, 46 | 509, 176 | | |

| **Hatano**, $\epsilon = 1$ | 200 | 400 | 800 | 1200 |
|---|---|---|---|---|
| Algorithm 1 | 0.5, 4 | 0.7, 4 | 2, 4 | 4.4, 4 |
| Algorithm in [14] | 11, 24 | 57.3, 29 | 225.6, 22 | |

| **Poisson**, $\epsilon = 10^2$ | 225 | 900 | 2500 | 10000 | 40000 |
|---|---|---|---|---|---|
| Algorithm 1 | 0.07, 2 | 0.4, 2 | 1, 2 | 5.4, 2 | 123, 2 |
| Algorithm in [14] | 0.16, 2 | 1, 2 | 1.8, 2 | 8.5, 2 | 77.5, 2 |

| **Wathen**, $\epsilon = 10^2$ | 341 | 560 | 1281 | 2821 | 14981 |
|---|---|---|---|---|---|
| Algorithm 1 | 0.1, 2 | 0.3, 2 | 0.4, 2 | 0.9, 2 | 6.3, 2 |
| Algorithm in [14] | 0.1, 2 | 0.5, 2 | 0.6, 2 | 1.3, 2 | 2.7, 2 |

Table 2: Comparison of the running times and number of iterations of Algorithm 1 and the algorithm [14] due to Guglielmi and Overton on four family of matrices: Landau and Hatano matrices are dense, whereas Poisson and Wathen matrices are sparse. In each table in the second and third rows, the numbers separated by a comma correspond to the running time in seconds and the number of iterations, respectively. These entries are omitted for the algorithm in [14], and for the $800 \times 800, 1200 \times 1200$ Landau matrices, as well as for the $1200 \times 1200$ Hatano matrices, since the computations take excessive amount of time.

| $k$ | Algorithm 1 | Algorithm in [14] |
|---|---|---|
| 1 | 0.998463500946451 | 0.998463500946449 |
| 2 | 1.314691266963286 | 1.314183947855148 |
| 3 | 1.315321099397434 | 1.314549337187003 |
| 4 | 1.315321120654863 | 1.314742547970619 |
| 5 | 1.315321120661177 | 1.314870338002996 |
| 6 | 1.315321120661177 | 1.314963416776238 |

| $k$ | Algorithm 1 | Algorithm in [14] |
|---|---|---|
| 1 | 2.975503186571098 | 2.975503186571121 |
| 2 | 3.975503186571100 | 3.958394806886049 |
| 3 | 4.076353694553352 | 4.044979289335425 |
| 4 | 4.076461835403100 | 4.065624227604959 |
| 5 | 4.076461835511951 | 4.072484100899472 |
| 6 | | 4.074749026199803 |

Table 3: A comparison of the iterations of Algorithm 1 and the Algorithm in [14] on two matrices: (Left) First six iterations of these algorithms on the $200 \times 200$ Landau matrix. For this example $\alpha_\epsilon(A) = 1.3153$ rounded to four decimal digits; (Right) First five and six iterations of Algorithm 1 and the Algorithm in [14] on the $200 \times 200$ Hatano matrix. Algorithm 1 converges after 5 iterations. In this case $\alpha_\epsilon(A) = 4.0765$ rounded to four decimal digits.

otherwise stated. All of the plots of the pseudospectra are generated by computing the singular value function $\sigma_{\min}\left[P(z)\right]/\|w(z)\|_1$ on a grid.

**Wing Example:** The first one arises from the analysis of oscillations of a wing of an airplane, leading to a $3 \times 3$ quadratic eigenvalue problem $Q(\lambda) = A_0 + \lambda A_1 + \lambda^2 A_2$ where

$$A_0 = \begin{bmatrix} 121 & 18.9 & 15.9 \\ 0 & 2.7 & 0.145 \\ 11.9 & 3.64 & 15.5 \end{bmatrix}, \quad A_1 = \begin{bmatrix} 7.66 & 2.45 & 2.1 \\ 0.23 & 1.04 & 0.223 \\ 0.60 & 0.756 & 0.658 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 17.6 & 1.28 & 2.89 \\ 1.28 & 0.824 & 0.413 \\ 2.89 & 0.413 & 0.725 \end{bmatrix}.$$

The progress of Algorithm 4 on this example with $\epsilon = 10^{-0.8}$ is illustrated in Figure 2. The algorithm starts with a rightmost eigenvalue $z_r = 0.0947 + 2.2529i$. However, this eigenvalue is considerably less sensitive as compared to the eigenvalues $-0.8848 \pm 8.4415i$. The first few iterations yield estimates in the component of $\Lambda_\epsilon(Q)$ containing $z_r$. When the rightmost point in this component is obtained, a vertical search is performed, and the algorithm jumps into the component of the eigenvalue $-0.8848 - 8.4415i$. A few more subspace iterations result in the convergence to a rightmost point globally. The computed value $\alpha_\epsilon(Q) = 9.25817665382$ matches the result reported in [31].

**Butterfly Example:** We next experiment on the butterfly example in [3]. This involves a $64 \times 64$ quartic polynomial $P(\lambda) = B_0 + \lambda B_1 + \lambda^2 B_2 + \lambda^3 B_3 + \lambda^4 B_4$ for which the computation of $\alpha_\epsilon(P)$ appears notoriously difficult. A particular application of Algorithm 4 for the computation of $\alpha_\epsilon(P)$ for $\epsilon = 0.08$ is illustrated in Figure 3. The algorithm converges to (nonglobal) local solutions twice. It escapes from these local solutions by means of vertical searches. Consequently, it generates iterates with imaginary parts about -2. But slow convergence occurs, and our numerical implementation applies another vertical search. This is an artifact of the numerical implementation; this vertical search is not essential for convergence to a globally rightmost point, as the pseudospectra is symmetric with respect to the real axis. But it speeds up the convergence. This leads to iterates with imaginary parts about 2, and eventually termination with $\alpha_\epsilon(P) = 1.3858189142$.

It has been proven in [20] that the subspace approach for the computation of a pseudospectral abscissa of a matrix converges superlinearly with respect to the subspace dimension. Even a faster quadratic convergence has been reported in practice in the same paper. In the more general nonlinear eigenvalue setting, we still observe quadratic convergence. For instance, for the butterfly example above, the first five iterations of the subspace approach before the first vertical search are given on the left in Table 4, where a quadratic convergence is apparent.

To illustrate the effect of the weights, the algorithm is applied to the butterfly example for $\epsilon = 0.2$ with weights [1 1 1 1 1], [1 1 1 1 $\infty$], [1 1 1 $\infty$ $\infty$] and [1 1 $\infty$ $\infty$ $\infty$] . The iterates generated on the associated $\epsilon$-pseudospectrum for each of these four cases are provided in Figure 4. The computed $\alpha_\epsilon(P)$ are 3.6758307326, 1.4144528011, 1.2006081257 and 1.1221784200, respectively. The decrease in the $\epsilon$-pseudospectral abscissa is dramatic when the perturbations

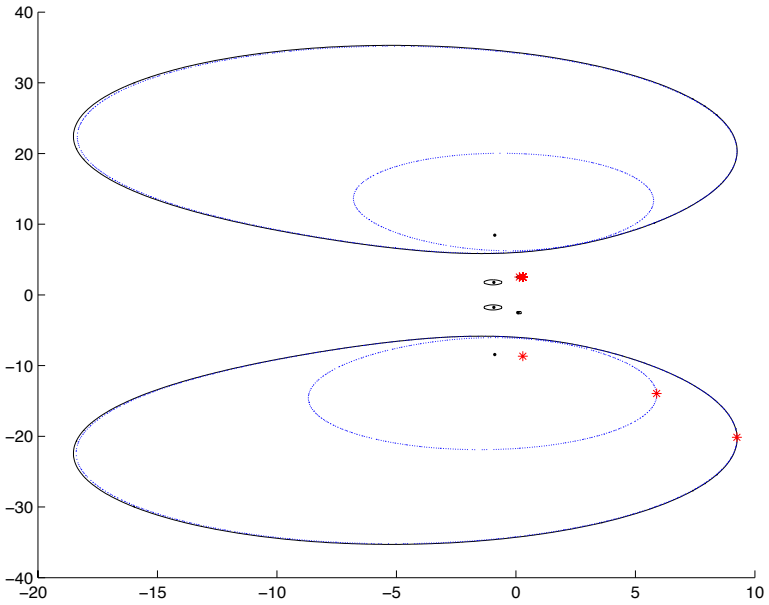



Figure 2: The progress of the algorithm on the wing example $Q$ for $\epsilon = 10^{-0.8}$ is shown. The eigenvalues are marked with black points, and the iterates of the algorithm are marked with red asterisks. The outermost solid curve corresponds to the boundary of $\Lambda_\epsilon(Q)$ for $\epsilon = 10^{-0.8}$, whereas the inner dotted curves represent the boundary of this $\epsilon$-pseudospectrum when the domain of the map $v \mapsto Qv$ is restricted to one and two dimensional subspaces.

| $k$ | $\alpha_\epsilon(P_{S_k})$ |
|---|---|
| 1 | 1.13113137749520 |
| 2 | 1.17557525555857 |
| 3 | 1.17810581840297 |
| 4 | 1.17824505975687 |
| 5 | 1.17824509591110 |
| 6 | 1.17824509591110 |

| $k$ | $\alpha_\epsilon(D_{S_k})$ |
|---|---|
| 1 | 14.77293966129789 |
| 2 | 15.54331045219255 |
| 3 | 15.58945602178569 |
| 4 | 15.58946152761596 |
| 5 | 15.58946153015292 |
| 6 | 15.58946153015296 |

Table 4: The real parts of the rightmost points are listed with respect to the subspace dimension $k$ for the following examples: (left) the butterfly example illustrated in Figure 3 before the first vertical search with all weights equal to one and $\epsilon = 0.08$; (right) the delay example of Figure 5 after the first vertical search with weights $[\infty\ 1\ 1\ 1]$ and $\epsilon = 0.45$.

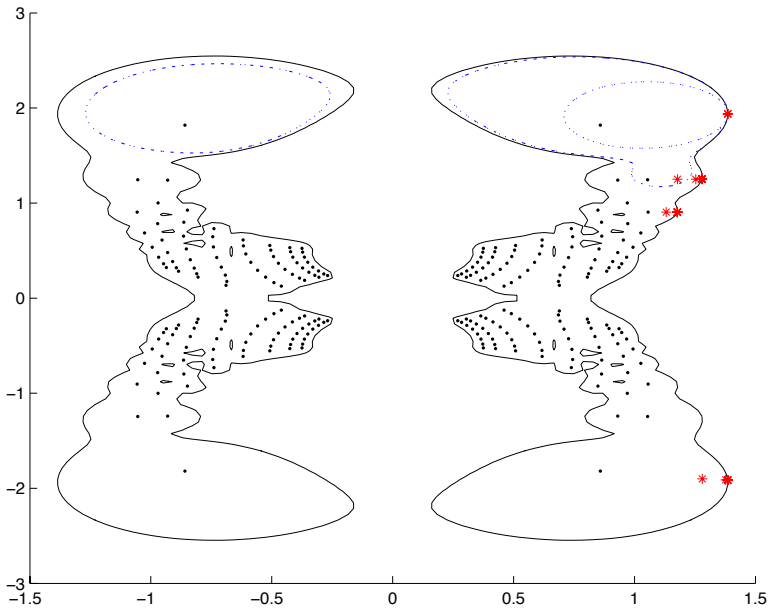



Figure 3: The algorithm is depicted on the butterfly example with $\epsilon = 0.08$. Once again, the eigenvalues and the iterates of the algorithm are marked with black points and red asterisks. The outermost solid curve corresponds to the boundary of $\epsilon$-pseudospectrum. The dotted curve and dashed dotted curve represent the $\epsilon$-pseudospectrum when the domain is restricted to a one dimensional subspace and a three dimensional subspace just a few iterations before termination.

of the leading coefficient are not allowed. The algorithm does not require any vertical searches on the top left figure, it performs vertical searches on the other three figures to avoid locally rightmost points.

**2D Acoustic Wave Example:** This concerns a quadratic matrix polynomial $W(\lambda) = K_0 + \lambda K_1 + \lambda^2 K_2$ arising from a finite element discretization of a 2-dimensional harmonic wave equation over the unit square $[0,1] \times [0,1]$. The size of the matrix polynomial $W$ depends on the coarseness of the finite element grid. Running times of Algorithm 4 to compute $\alpha_\epsilon(W)$ for $\epsilon = 0.01$ with respect to the size of $W$ are listed in Table 5.

Majority of the running time is consumed by the vertical searches for the examples in Table 5. Algorithm 3 without the vertical searches is applicable to large scale matrix polynomials. This is illustrated in Table 6 on the matrix polynomials of size $1100 \times 1100$, $2100 \times 2100$ and $4200 \times 4200$ arising from the acoustic wave equation. Here, only the subspace approach combined with the local searches is applied, in a way such that it terminates when the two consecutive estimates for the $\epsilon$-pseudospectral abscissa differ by less than $10^{-12}$. Thus, the computed values for $\alpha_\epsilon(W)$ can possibly correspond to the real parts of points in $\Lambda_\epsilon(W)$ that are locally rightmost. The rightmost eigenvalues are computed a priori, so the time consumed for their computation is not included in the overall running time. The running times in Table 6 are remarkable; given this rightmost eigenvalue, the subspace approach requires less than a minute to compute a locally rightmost point in $\Lambda_\epsilon(W)$. Majority of the running time for the subspace approach is taken by the smallest singular value computations on the full matrix polynomial. Recall that the right singular vectors associated with these singular values are needed to form the subspaces.

## 6.3 Delay Eigenvalue Problem

We test Algorithm 4 on the following delay eigenvalue problem with weights $[\infty \ 1 \ 1 \ 1]$:

$$D(\lambda) = \lambda I - D_0 - D_1 e^{-\lambda} - D_2 e^{-3\lambda}. \tag{6.1}$$

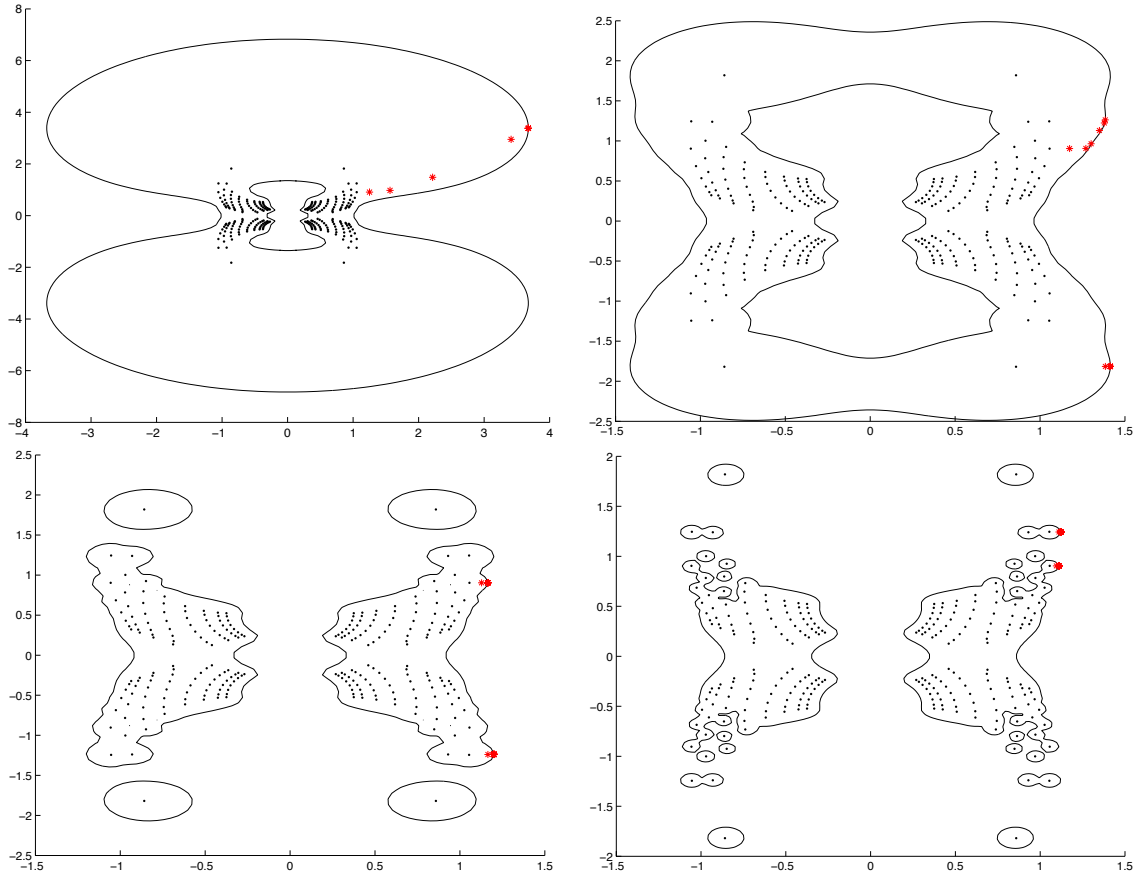


Figure 4: The algorithm on the butterfly example with $\epsilon = 0.2$ is depicted for various choices of weights. The eigenvalues and the iterates are marked with dots and asterisks, respectively. The solid curve is the boundary of the $\epsilon$-pseudospectrum. Weights are as follows: **(top left)** [1 1 1 1 1]; **(top right)** [1 1 1 1 $\infty$]; **(bottom left)** [1 1 1 $\infty$ $\infty$]; **(bottom right)** [1 1 $\infty$ $\infty$ $\infty$].

| **2D Acoustic Wave**, $\epsilon = 0.01$ | 110 | 210 | 420 |
|---|---|---|---|
| Running Time | 11 | 138 | 584 |
| $\alpha_\epsilon$ | 4.99778 | 6.95044 | $1.00718 \cdot 10^1$ |

Table 5: Running time for Algorithm 4 in seconds on the 2-dimensional acoustic wave equation with respect to the size of the quadratic matrix polynomial involved, and the computed pseudospectral abscissa

| **2D Acoustic Wave**, $\epsilon = 0.0001$ | 1100 | 2100 | 4200 |
|---|---|---|---|
| Running Time | 3 | 7 | 57 |
| $\alpha$ | $3.50140 \cdot 10^2$ | $6.68451 \cdot 10^2$ | $1.33690 \cdot 10^3$ |
| $\alpha_\epsilon$ | $3.50631 \cdot 10^2$ | $6.70238 \cdot 10^2$ | $1.34408 \cdot 10^3$ |

Table 6: Running time for Algorithm 3 (without the vertical searches) in seconds on the 2-dimensional acoustic wave equation with respect to the size of the quadratic matrix polynomial involved, the real part of the rightmost eigenvalue $\alpha$, and the computed pseudospectral abscissa $\alpha_\epsilon$

| $\gamma$ | -4 | -8 | -16 | -32 | -100 | -200 |
|---|---|---|---|---|---|---|
| First vertical search | 58 | 87 | 113 | 171 | 285 | 404 |
| Last vertical search | 46 | 72 | 103 | 144 | 244 | 342 |

Table 7: The number of iterations during the first and last vertical searches with respect to $\gamma$ for the delay example of Section 6.3

The coefficient matrices $D_0, D_1, D_2$ are obtained by typing `randn(100) + 1.2*randn(100)*i`, `randn(100)` and `gallery('poisson',10)` in Matlab[†]. Thus $D_0$ and $D_1$ are complex and real random matrices, respectively, whereas $D_2$ comes from the five-point finite difference discretization of the 2-dimensional Poisson equation. For $\epsilon = 0.45$ the pseudospectrum $\Lambda_\epsilon(D)$ is shown in Figure 5, for which the boundary is generated by calculating $\sigma_{\min}(D(z))/\|w(z)\|_1$ on a grid. The rightmost eigenvalue is given by $\lambda = 16.3300 - 1.9812i$. Initiating Algorithm 4 with this value, only one local search is needed to locate the globally rightmost point, leading to $\alpha_\epsilon(D) = 17.1899477706$.

A nice, didactic illustration of the interplay between the two components of the Algorithm 4 (searching for a locally rightmost point, and switching by vertical searches), is obtained by initializing the algorithm with the origin instead. Then the algorithm ends up at locally rightmost points twice, see Figure 5. Each time this happens, a vertical search provides a better estimate strictly inside the $\epsilon$-pseudospectrum and well-away from the boundary. All together 53 subspace iterations are needed to retrieve the pseudospectral abscissa. The subspace dimension is never allowed to exceed ten; whenever the subspace dimension becomes ten, it is reset to a one dimensional subspace based on the latest iterate. The algorithm concludes with convergence after an eventual vertical search, when it is found out that the globally smallest value of $\sigma_{\min}\left[D(\alpha_*,\omega)\right]/\|w(\alpha_*,\omega)\|_1$ for fixed $\alpha_* = 17.1899477706.$ over all $\omega$ does not differ from $\epsilon$ by more than $10^{-6}\|D_2\|_2$. The iterates of the subspace approach on this example after the first vertical search are listed on the right-hand side of Table 4, where we again observe a quadratic rate of convergence with respect to the subspace dimension. Note that by starting from an arbitrary point $\lambda_0$ satisfying $\sigma_{\min}\left[D(\lambda_0)\right] < \epsilon\|w(\lambda_0)\|_1$ does by itself not guarantee to find a globally rightmost point of the pseudospectrum, since, for instance, the existence of an isolated component of the pseudospectrum to the right of the dashed line in Figure 5 is not excluded. Such a situation is avoided by initiating the algorithm with the rightmost eigenvalue.

In this example, the global lower bound $\gamma$ for the second derivatives of the singular value function minimized during the vertical searches is set equal to -4. As this lower bound is chosen smaller, the number of iterations required by the vertical searches to satisfy the prescribed error tolerance $10^{-6}\|D_2\|_2$ increases. But the increase in the number of iterations is typically sublinear with respect to $\gamma$. This is illustrated in Table 7 for the first and the last vertical searches for the particular delay example.

# 7 Software

Algorithm 4 is implemented in Matlab. This Matlab software is available on the web publicly[‡]. For a nonpolynomial nonlinear eigenvalue problem, the user is expected to write down a routine calculating the functions $f_j(z)$ as in (1.2) and its first derivatives at a given $z \in \mathbb{C}$. The user must provide the name of this routine and a right-most eigenvalue to the software as input parameters.

[†]The precise data is available on the web at http://home.ku.edu.tr/∼emengi/software/delay.mat

[‡]http://home.ku.edu.tr/∼emengi/software/nonlinear_pseudoabscissa.tar

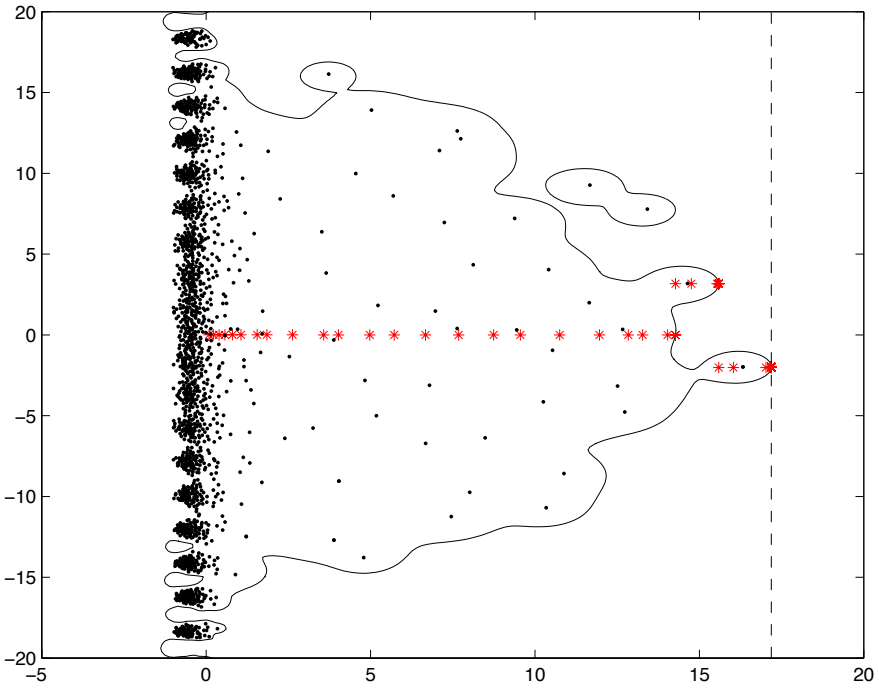



Figure 5: The progress of the algorithm on delay example (6.1) for $\epsilon = 0.45$ is shown, when initiating at the origin. The iterates are marked with red asterisks, the eigenvalues are indicated by black dots, whereas the solid curve corresponds to the boundary of the $\epsilon$-pseudospectrum. The dashed vertical line represents the points with real part equal to the computed $\epsilon$-pseudospectral abscissa.

# 8 Conclusion

An algorithm is proposed for the computation of the $\epsilon$-pseudospectral abscissa of an analytic matrix-valued function $F(\lambda)$ depending on one complex parameter. The algorithm is capable of handling large scale problems. This is made possible by an adaptation of the subspace iteration [20] for the nonlinear eigenvalue problem setting. Each subspace iteration involves the computation of the $\epsilon$-pseudospectral abscissa when the domain of the map $v \mapsto F(\lambda)v$ is restricted to a small subspace. This computation is realized locally, but in a robust way against nonsmoothness, by adapting the support based algorithm of [28] for optimization subject to eigenvalue constraints. Repeated applications of the subspace iteration result in a point on the boundary of the $\epsilon$-pseudospectrum with a vertical tangent line. Vertical searches are performed to check whether these converged points are globally right-most in the $\epsilon$-pseudospectrum. These vertical searches are realized by means of the support based algorithm of [29], which determines the globally smallest value of a prescribed eigenvalue of a Hermitian and analytic matrix-valued function. They depend on the availability of a global lower bound $\gamma$ for the second derivative of a certain singular value function. Assigning a large, negative value to $\gamma$ works robustly in practice. A restarting strategy for the subspaces further enhances the efficiency of the algorithm.

The algorithm is both globally convergent and well-suited for large scale problems. The accompanying software that is publicly available aims for large scale standard, polynomial and more general nonlinear eigenvalue problems.

**Acknowledgements.** The authors are grateful to two anonymous referees, and Francoise Tisseur for their valuable remarks on an initial version of this paper.

# References

[1] A. Amiraslani, R. M. Corless, and P. Lancaster. Linearization of matrix polynomials expressed in polynomial bases. *IMA J. Numer. Anal.*, 29(1):141–157, 2009.

[2] C. Bekas and E. Gallopoulos. Cobra: Parallel path following for computing the matrix pseudospectrum. *Parallel Comput.*, 27:1879–1896, 2001.

[3] T. Betcke, N. J. Higham, Mehrmann V., Schroder C., and Tisseur F. NLEVP: A collection of nonlinear eigenvalue problems. Technical Report Eprint 2010.98, Manchester Institute for Mathematical Sciences, 2010.

[4] S. Boyd and V. Balakrishnan. A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its $L_\infty$-norm. *Systems and Control Letters*, 15:1–7, 1990.

[5] M. Brühl. Pseudospectra of rectangular matrices. *BIT*, 36:441–454, 1996.

[6] N. A. Bruinsma and M. Steinbuch. A fast algorithm to compute the $H_\infty$-norm of a transfer function matrix. *Systems and Control Letters*, 14:287–293, 1990.

[7] 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.

[8] R. Byers. A bisection method for measuring the distance of a stable matrix to the unstable matrices. *SIAM J. Sci. Stat. Comp.*, 9:875–881, 1988.

[9] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1990.

[10] E. B. Davies. Pseudo-spectra, the harmonic oscillator and complex resonances. *Proc. Roy. Soc. Lond. Ser. A.*, pages 585–599, 1999.

[11] C. Effenberg. *Robust Solution Methods for Nonlinear Eigenvalue Problems*. Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland, 2013. Ph.D. Thesis.

[12] M. A. Freitag and A. Spence. A newton-based method for the calculation of the distance to instability. *Linear Algebra Appl.*, 435(12):3189–3205, 2011.

[13] K. Green and T. Wagenknecht. Pseudospectra and delay differential equations. *J. Comput. Appl. Math.*, 196(2):567–578, 2006.

[14] N. Guglielmi and M. L. Overton. Fast algorithms for the approximation of the pseudospectral abscissa and pseudospectral radius of a matrix. *SIAM J. Matrix Anal. Appl.*, 32(4):1166–1192, 2011.

[15] S. Güttel, R. Van Beeumen, K. Meerbergen, and W. Michiels. NLEIGS: A class of fully rational krylov methods for nonlinear eigenvalue problems. *SIAM J. Sci. Comput.*, 36(6):A2842–A2864, 2014.

[16] N. Hatano and D. R. Nelson. Localization transitions in non-Hermitian quantum mechanics. *Phys. Rev. Lett.*, 77:570–573, 1996.

[17] N. J. Higham and F. Tisseur. More on pseudospectra for polynomial eigenvalue problems and applications in control theory. *Linear Algebra Appl.*, 351-352(1):435 – 453, 2002.

[18] E. Jarlebring, K. Meerbergen, and W. Michiels. A Krylov method for the delay eigenvalue problem. *SIAM J. Sci. Comput.*, 32(6):3278–3300, 2010.

[19] G. F. Jónsson and L. N. Trefethen. A numerical analyst looks at the "cutoff phenomenon" in card shuffling and other markov chains. In D. F. Griffiths, D. J. Higham, and G. A. Watson, editors, *Numerical Analysis 1997*, pages 150–178. Addison Wesley Longman, Harlow, Essex, UK, 1998.

[20] 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.

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

[22] P. Lancaster and P. Psarrakos. On the pseudospectra of matrix polynomials. *SIAM J. Matrix Anal. Appl.*, 27(1):115–129, 2005.

[23] H. J. Landau. The notion of approximate eignevalues applied to an integral equation of laser theory. *Quart. Appl. Math.*, 35:165 – 172, 1977.

[24] R. B. Lehoucq and D. C. Sorensen. Deflation techniques for an implicitly restarted Arnoldi iteration. *SIAM J. Matrix Anal. Appl.*, 17(4):789–821, 1996.

[25] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK users' guide: Solution of large scale eigenvalue problems with implicitly restarted Arnoldi methods.* SIAM, Philadelphia, PA, USA, 1998.

[26] D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann. Vector spaces of linearizations for matrix polynomials. *SIAM J. Matrix Anal. Appl.*, 28(4):971–1004, 2006.

[27] V. Mehrmann and H. Voss. Nonlinear eigenvalue problems: a challenge for modern eigenvalue methods. *GAMM Mitt. Ges. Angew Math. Mech.,*, 27(2):121–152, 2004.

[28] E. Mengi. A support function based algorithm for optimization with eigenvalue constraints. *SIAM J. Optim.*, 2014. Accepted subject to revision.

[29] E. Mengi, E. A. Yildirim, and M. Kilic. Numerical optimization of eigenvalues of Hermitian matrix functions. *SIAM J. Matrix Anal. Appl.*, 35(2):699–724, 2014.

[30] W. Michiels, K. Green, T. Wagenknecht, and S. I. Niculescu. Pseudospectra and stability radii for analytic matrix functions with application to time-delay systems. *Linear Algebra Appl.*, 418(1):315 – 335, 2006.

[31] W Michiels and N. Guglielmi. An iterative method for computing the pseudospectral abscissa for a class of nonlinear eigenvalue problems. *SIAM J. Sci. Comput.*, 34(4):A2366–A2393, 2012.

[32] W. Michiels and S. Gumussoy. Characterization and computation of H-infinity norms of time-delay systems. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2093–2115, 2010.

[33] W. Michiels and S. I. Niculescu. *Stability and Stabilization of Time-Delay Systems (Advances in Design & Control) (Advances in Design and Control)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2007.

[34] F. Tisseur and N. J. Higham. Structured pseudospectra for polynomial eigenvalue problems, with applications. *SIAM J. Matrix Anal. Appl.*, 23(1):187–208, 2001.

[35] F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43(2):235–286, 2001.

[36] L. N. Trefethen. Computation of pseudospectra. *Acta Numerica*, 8:247–295, 1999.

[37] L. N. Trefethen and M. Embree. *Spectra and Pseudospectra. The Behavior of Nonnormal Matrices and Operators*. Princeton University Press, Princeton, NJ, USA, 2005.

[38] L. N. Trefethen, A. E. Trefethen, S. C. Reddy, and T. A. Driscoll. Hydrodynamic stability without eigenvalues. *Science*, 261:578–584, 1993.

[39] R. Van Beeumen. *Rational Krylov Methods for Nonlinear Eigenvalue Problems*. KU Leuven, Leuven, Belgium, 2015. Ph.D. Thesis.

[40] R. Van Beeumen, K. Meerbergen, and W. Michiels. Compact rational krylov methods for nonlinear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 36(2):820–838, 2015.

[41] D. Verhees, R. Van Beeumen, K. Meerbergen, N. Guglielmi, and W. Michiels. Fast algorithms for computing the distance to instability of nonlinear eigenvalue problems, with applications to time-delay systems. *Int. J. Dynam. Control*, 2(4):133 – 142, 2014.

[42] T. Wagenknecht, W. Michiels, and K. Green. Structured pseudospectra for nonlinear eigenvalue problems. *J. Comput. Appl. Math.*, 212(2):245 – 259, 2008.

[43] T. G. Wright and L. N. Trefethen. Pseudospectra of rectangular matrices. *IMA J. Numer. Anal.*, 22:501–519, 2002.

[44] T.G. Wright. Eigtool: a graphical tool for nonsymmetric eigenproblems. Oxford University Computing Laboratory. `https://github.com/eigtool/eigtool`.

[45] Z. Wu and W. Michiels. Reliably computing all characteristic roots of delay differential equations in a given right half plane. *Journal of Computational and Applied Mathematics*, 236:2499–2514, 2012.

[46] K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.