$\mathcal{L}_\infty$-Norm Computation for Large-Scale Descriptor Systems Using Structured Iterative Eigensolvers

Peter Benner* Ryan Lowe† Matthias Voigt‡

July 5, 2016

Abstract

In this article, we discuss a method for computing the $\mathcal{L}_\infty$-norm for transfer functions of descriptor systems using structured iterative eigensolvers. In particular, the algorithm computes some desired imaginary eigenvalues of an even matrix pencil and uses them to determine an upper and lower bound to the $\mathcal{L}_\infty$-norm. Finally, we compare our method to a previously developed algorithm using structured pseudopole sets. Numerical examples demonstrate the reliability and accuracy of the new method along with a significant drop in the runtime.

1 Introduction

Descriptor systems (or singular systems/differential-algebraic control systems) are a type of linear dynamical system that arises in a large number of fields, from certain constrained multi-body dynamics problems to circuit simulation [7, 22]. The analysis and control of such systems often requires the use of a system norm. One common norm is the $\mathcal{L}_\infty$-norm, which has important applications in model order reduction as an error measure [22], and in robust control as a robustness measure for a dynamical systems [19, 14, 23]. In this paper, we will calculate such norms for descriptor systems based on an algorithm detailed in [7]. Here, we will adapt it for large-scale problems by employing structured iterative eigensolvers. We will present this method, give implementation details and compare the results to another calculation method using optimization over pseudopole sets.

Consider the following equations describing continuous-time linear time-invariant (LTI) systems

$$\tfrac{\mathrm{d}}{\mathrm{d}t}Ex(t) = Ax(t) + Bu(t), \qquad y(t) = Cx(t) + Du(t), \tag{1}$$

with E, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times m}$. Furthermore, $x : \mathbb{R} \to \mathbb{R}^n$, $u : \mathbb{R} \to \mathbb{R}^m$, and $y : \mathbb{R} \to \mathbb{R}^p$ are the (generalized) state, the control, and the output, respectively. We will always assume that the matrix pencil $sE - A$ is *regular*, meaning that $\det(sE - A) \not\equiv 0$, and that it does not have any finite eigenvalues on the imaginary axis. Furthermore, we will generally be dealing with matrices that are both large and sparse with $m, p \ll n$.

In many situations it is more convenient to work with the transfer function of the system instead of the system itself. The transfer function of (1) is given by

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

*Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstraße 1, 39106 Magdeburg, Germany, E-Mail: benner@mpi-magdeburg.mpg.de

†McGill University, Reasoning and Learning Laboratory, 3480 University Street, H3A 0E9 Montreal, Quebec, Canada, E-Mail: ryan.lowe@cs.mcgill.ca

‡Technische Universität Berlin, Institut für Mathematik, Straße des 17. Juni 136, 10623 Berlin, Germany, E-Mail: mvoigt@math.tu-berlin.de

If G is proper (i. e., bounded for $|s| \to \infty$) and has no poles on the imaginary axis (which is typically the case in control applications), then $G \in \mathcal{RL}_\infty^{p\times m}$, where $\mathcal{RL}_\infty^{p\times m}$ denotes the normed space of all rational $p \times m$ matrix-valued functions that are bounded on the imaginary axis. The induced norm for this space is the $\mathcal{L}_\infty$-norm, which is defined as

$$\|G\|_{\mathcal{L}_\infty} := \sup_{\omega\in\mathbb{R}} \sigma_{\max}(G(\mathrm{i}\omega)),$$

where $\sigma_{\max}(\cdot)$ denotes the maximum singular value [31]. For the sake of simplicity we will assume that $G \in \mathcal{RL}_\infty^{p\times m}$. The algorithm presented here will not verify whether $G \in \mathcal{RL}_\infty^{p\times m}$, although methods to do so have already been devised (for example, in [7]). Note that for stable transfer functions, i. e., all poles are in open left half-pane, the $\mathcal{L}_\infty$-norm is equivalent to the well-known $\mathcal{H}_\infty$-norm.

The structure of the remainder of this paper is as follows. We will summarize a method for computing the $\mathcal{L}_\infty$-norm for transfer functions of descriptor systems in Section 2. This algorithm, first developed for LTI systems in [12, 11, 13], has been generalized to descriptor systems in [7, 8] by using structure-preserving algorithms for computing eigenvalues of skew-Hamiltonian/Hamiltonian matrix pencils. An alternative method for calculating the $\mathcal{H}_\infty$-norm for large-scale descriptor systems was derived in [18, 24] and [10, 30] using an optimization procedure over spectral value sets and pseudopole sets, respectively. Recently, a third approach using the implicit determinant method has been developed [16] which has also the potential for being applied to large-scale systems.

For comparison we also briefly describe the method of [10] in this section. In Section 3, we will reveal the nature of the structured iterative eigensolvers used. These will allow for a more efficient calculation of the $\mathcal{L}_\infty$-norm using the matrix pencil method, as long as the input matrices E and A from (1) are sparse. In Section 4, we will give details on the implementation and use numerical examples to compare the results of this calculation method with the aforementioned pseudopole method. Finally, we will summarize and present our conclusions in Section 5.

2 Computation of the $\mathcal{L}_\infty$-Norm

In this section, we detail the pseudopole and structured matrix pencil methods for calculating the $\mathcal{L}_\infty$-norm.

2.1 Pseudopole Method

The pseudopole method deals with transfer functions $G \in \mathcal{RH}_\infty^{p\times m}$, and thus uses the $\mathcal{H}_\infty$-norm. A function fulfills $G \in \mathcal{RH}_\infty^{p\times m}$ if $G \in \mathcal{RL}_\infty^{p\times m}$ and all of the poles of G are in the open left half-plane – in other words, G must be stable. In addition, we assume for this approach that $D = 0$; if this is not the case, we must adjust the matrices A, B, C, and E accordingly, see [10] for details. We consider a perturbed transfer function G_Δ of the form

$$G_\Delta(s) = C(sE - (A + B\Delta C))^{-1}B$$

with $\Delta \in \mathbb{C}^{m\times p}$. Roughly speaking, we determine how large we can make Δ such that the perturbed transfer function G_Δ is guaranteed to remain in $\mathcal{RH}_\infty^{p\times m}$. More specifically, we make the following definition.

Definition 1. *The $\mathcal{H}_\infty$-radius $r(G)$ is the smallest $\varepsilon > 0$ such that there exists a $\Delta \in \mathbb{C}^{m\times p}$ with $\|\Delta\|_2 < \varepsilon$ and $G_\Delta \notin \mathcal{RH}_\infty^{p\times m}$.*

The $\mathcal{H}_\infty$-radius has the following connection to the $\mathcal{H}_\infty$-norm [10], given by

$$r(G) = \begin{cases} \|G\|_{\mathcal{H}_\infty}^{-1}, & \text{if } G \not\equiv 0, \\ \infty, & \text{if } G \equiv 0. \end{cases}$$

Thus, if the $\mathcal{H}_\infty$-radius is found, the $\mathcal{H}_\infty$-norm immediately follows. It is important to note that there exists an explicit, optimal way to construct Δ such that $G_\Delta \notin \mathcal{RH}_\infty^{p\times m}$. This is done by considering rank-1 perturbations of the system such that one of the poles of the perturbed transfer function G_Δ converges to the rightmost ε-pseudopole of the transfer function G, i. e., the rightmost pole of a perturbed transfer functions G_Δ with $\|\Delta\|_2 < \varepsilon$, see [10, 30].

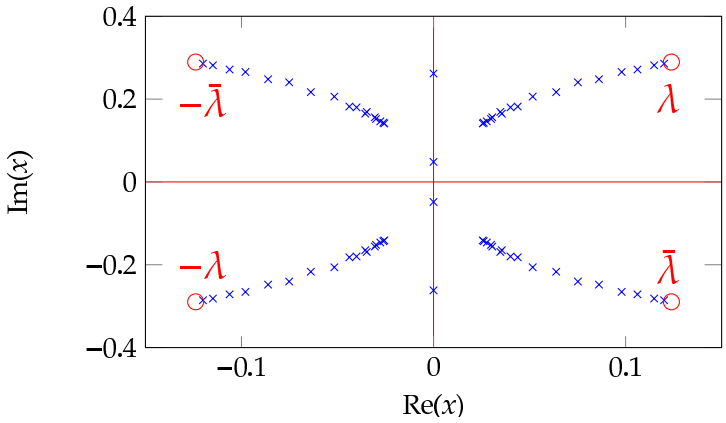

Figure 1: Eigenvalues of the matrix pencil $\mathcal{H}_\gamma(s)$

2.2 Structured Matrix Pencil Method

In order to compute the $\mathcal{L}_\infty$-norm, we will use the matrix pencil [7, 8]

$$\mathcal{H}_\gamma(s) = s\mathcal{N} - \mathcal{M}_\gamma := \left[\begin{array}{cc|cc} 0 & -sE^\mathsf{T} - A^\mathsf{T} & -C^\mathsf{T} & 0 \\ sE - A & 0 & 0 & -B \\ \hline -C & 0 & \gamma I_p & -D \\ 0 & -B^\mathsf{T} & -D^\mathsf{T} & \gamma I_m \end{array}\right], \tag{2}$$

where I_n represents the $n \times n$ identity matrix. It is important to note that this is an *even* matrix pencil, meaning that $\mathcal{N} = -\mathcal{N}^\mathsf{T}$ and $\mathcal{M}_\gamma = \mathcal{M}_\gamma^\mathsf{T}$. An important property of the eigenvalues of the matrix pencil $\mathcal{H}_\gamma(s)$ is the Hamiltonian spectral symmetry. As is evident from the graph of the sample spectrum in Figure 2.2, this property refers to the symmetry of the spectrum across the real and imaginary axes. If $\mathcal{H}_\gamma(s)$ has a real or imaginary eigenvalue λ, then it also has the eigenvalue $-\lambda$, while if the eigenvalue λ is complex, then $-\lambda, \bar{\lambda}$, and $-\bar{\lambda}$ are also eigenvalues.

Standard methods [7, 8] to calculate the eigenvalues of $\mathcal{H}_\gamma(s)$ use a related skew-Hamiltonian/Hamiltonian matrix pencil [2, 3]. In [7, 8], such a pencil is completely factorized using the structure-preserving method in [2, 9], resulting in a cubic algorithm complexity. While this approach is very reliable and accurate, the computations become prohibitively expensive for larger pencils. The structured iterative eigensolvers presented here instead use a projection approach by exploiting the even structure of the matrix pencil in (2) which makes the algorithm applicable in the large-scale setting.

We now present two theorems that form the basis for the algorithm to calculate the $\mathcal{L}_\infty$-norm. The first relates the singular values of $G(\mathrm{i}\omega)$ with the finite, purely imaginary eigenvalues of $\mathcal{H}_\gamma(s)$, see [7, 8].

Theorem 2. *Assume that $sE - A$ has no purely imaginary eigenvalues, $G \in \mathcal{RL}_\infty^{p\times m}$, $\gamma > 0$ and $\omega_0 \in \mathbb{R}$. Then, γ is a singular value of $G(\mathrm{i}\omega_0)$ if and only if $\mathcal{H}_\gamma(\mathrm{i}\omega_0)$ is singular.*

The following theorem is a direct consequence of Theorem 2, see [7].

Theorem 3. *Assume that $sE - A$ has no purely imaginary eigenvalues, $G \in \mathcal{RL}_\infty^{p\times m}$, and let $\gamma > \inf_{\omega\in\mathbb{R}} \sigma_{\max}(G(\mathrm{i}\omega))$. Then $\|G\|_{\mathcal{L}_\infty} \geq \gamma$ if and only if $\mathcal{H}_\gamma(s)$ has finite, purely imaginary eigenvalues.*

Using the theorems from above we are able to state an iterative method, as presented in [7, 8], to calculate the $\mathcal{L}_\infty$-norm. First, an initial γ is calculated that is less than the $\mathcal{L}_\infty$-norm. In each step, a check is performed to verify whether the matrix pencil $\mathcal{H}_\gamma(s)$ has finite, purely imaginary eigenvalues. If such eigenvalues are found, γ is incremented and the process is repeated. Finally, when no imaginary eigenvalues are found, γ serves as an upper bound of $\|G\|_{\mathcal{L}_\infty}$. By increasing γ in a certain way in each iteration, we can compute the $\mathcal{L}_\infty$-norm up to a desired accuracy.

Algorithm 1 is the generalization of the method presented in [11, 13] to the descriptor system case. It converges monotonically at a quadratic rate, with a relative error of at most the desired tolerance ε, as long as the arithmetic is exact.

Algorithm 1 Computation of the $\mathcal{L}_\infty$-Norm

Input: Continuous-time linear time-invariant descriptor system $(\lambda E - A, B, C, D)$ with transfer function $G \in \mathcal{RL}_\infty^{p\times m}$, tolerance ε.

Output: $\|G\|_{\mathcal{L}_\infty}$, optimizing frequency ω_{opt}.

1: Compute an initial value $\gamma_{\mathrm{lb}} < \|G\|_{\mathcal{L}_\infty}$ using the dominant pole algorithm (see Subsection 3.1).
2: **repeat**
3: Set $\gamma := (1 + 2\varepsilon)\gamma_{\mathrm{lb}}$.
4: Compute some desired eigenvalues of the matrix pencil $\mathcal{H}_\gamma(s)$ using the even IRA algorithm (see Subsection 3.2).
5: **if** no finite, purely imaginary eigenvalues **then**
6: Set $\gamma_{\mathrm{ub}} = \gamma$.
7: Break.
8: **else**
9: Set $\{\mathrm{i}\omega_1, \ldots, \mathrm{i}\omega_k\}$ = finite, purely imaginary eigenvalues of $\mathcal{H}_\gamma(s)$, with $\omega_j \geq 0$ for $j = 1, \ldots, k$.
10: Set $m_j = \sqrt{\omega_j \omega_{j+1}}$, $j = 1, \ldots, k-1$.
11: Compute the largest singular value of $G(\mathrm{i}m_j)$ for $j = 1, \ldots, k-1$.
12: Set $\gamma_{\mathrm{lb}} = \max_{1\leq j\leq k-1} \sigma_{\max}(G(\mathrm{i}m_j))$.
13: Set $\omega_{\mathrm{opt}} = \operatorname{argmax}_{m_1\leq m_j\leq m_{k-1}} \sigma_{\max}(G(\mathrm{i}m_j))$.
14: **end if**
15: **until** break
16: Set $\|G\|_{\mathcal{L}_\infty} = \frac{1}{2}(\gamma_{\mathrm{lb}} + \gamma_{\mathrm{ub}})$.

As we are comparing the performance of this algorithm to the pseudopole method of calculating the $\mathcal{H}_\infty$-norm, it is clearly desirable to minimize the computational costs. In general, the primary factors that affect the complexity of the algorithm are the total number of iterations, the number of imaginary eigenvalues of the matrix pencil $\mathcal{H}_\gamma(s)$ in each iteration, and the costs to compute each of these imaginary eigenvalues. In particular, the initial choice of γ_{lb} can have a significant impact on both the total number of iterations and the number of imaginary eigenvalues. To this effect, we choose

$$\gamma_{\mathrm{lb}} := \max\{\sigma_{\max}(G(0)), \sigma_{\max}(G(\mathrm{i}\omega_{\mathrm{p}})), \sigma_{\max}(G(\infty))\},$$

where ω_{p} is a test frequency that gives the maximum singular value [7].

To further increase the initial γ_{lb}, we want to select the test frequencies such that they consistently produce large singular values. An appropriate choice for these frequencies that satisfies this criterion is the imaginary parts of the dominant poles of the transfer function. A motivation for these dominant poles, as well as the method used for computing them, called the *subspace accelerated MIMO dominant pole algorithm (SAMDP)* [25, 26, 27], will be explained in Subsection 3.1.

Another factor affecting the computing time is the method used to calculate the eigenvalues. An efficient calculation method, the *even IRA algorithm*, was proposed in [21] that exploits the even structure of the matrix pencil $\mathcal{H}_\gamma(s)$. A summary of this algorithm is presented in Subsection 3.2. Alternatively, one can use shift-and-invert variants of the Hamiltonian Lanczos algorithm [5, 6]. In particular, the method discussed in [4] can be adapted here. The latter approach, based on the rational Krylov idea of Ruhe [28], would allow to change the shift on the fly (without restarting the Lanczos recursion) and thus, to search along the imaginary axis by incrementing the shift from 0 to an upper bound based on an (estimated) spectral bound of the shift-and-invert operator. This approach needs a careful implementation regarding the choice of the shift increment, based on deciding that no purely imaginary eigenvalue in a radius around the current shift exists. This is not currently available, but might be an interesting topic of future research.

3 Structured Iterative Eigensolvers

In this section, we detail the selection of test frequencies used to calculate the initial γ_{lb}, in addition to the algorithm for calculating the eigenvalues of the even pencils $\mathcal{H}_\gamma(s)$.

3.1 Choice of Test Frequencies for γ_{lb}

As previously mentioned, we want to find a series of test frequencies ω_j such that $\sigma_{\max}(G(\mathrm{i}\omega_p))$ is as large as possible. The dominant poles of the transfer function fulfill such a purpose, as they are the poles which have the highest influence on the frequency response of the transfer function G. This can be seen from the following consideration taken from [10]. Assume that $sE - A$ has only simple eigenvalues λ_k with left and right eigenvectors y_k and x_k, normalized such that $y_k^{\mathsf{H}} E x_k = 1$. Then we have

$$G(s) = \sum_{k=1}^{n} \frac{R_k}{s - \lambda_k} + R_\infty \tag{3}$$

with the *residues*

$$R_k = C x_k y_k^{\mathsf{H}} B \quad \text{and} \quad R_\infty = \lim_{\omega \to \infty} G(\mathrm{i}\omega).$$

We observe that, if λ_j is close to the imaginary axis and $\|R_j\|_2$ is large, then for $\omega \approx \mathrm{Im}(\lambda_j)$ we have

$$G(\mathrm{i}\omega) \approx \frac{R_j}{-\mathrm{Re}(\lambda_j)} + \sum_{\substack{k=1 \\ k \neq j}}^{n} \frac{R_k}{\mathrm{i}\omega - \lambda_k} + R_\infty,$$

and therefore, $\|G(\mathrm{i}\omega)\|_2$ is large as well. With this in mind, we define an eigenvalue $\lambda_j \in \Lambda(E, A)$ to be a *dominant pole* of $G(s)$, if

$$\frac{\|R_k\|_2}{|\mathrm{Re}(\lambda_k)|} < \frac{\|R_j\|_2}{|\mathrm{Re}(\lambda_j)|}, \quad k = 1, \ldots, n, \quad k \neq j.$$

The SAMDP algorithm can determine the most dominant poles, and is locally superlinearly convergent [27]. Therefore, if λ_j, $j = 1, \ldots, \ell$, are the computed dominant poles, we define the test frequency ω_j as

$$\omega_j = \mathrm{Im}(\lambda_j),$$

and the test frequency that gives the maximum singular value is thus

$$\omega_{\mathrm{p}} = \mathrm{argmax}_{\omega_1 \leq \omega_j \leq \omega_\ell} \sigma_{\max}(G(\mathrm{i}\omega_j)).$$

Multiple dominant poles are used to potentially increase the value of γ_{lb}.

3.2 The Eigenvalue Computation Method

The calculation of the eigenvalues of the even pencil $\mathcal{H}_\gamma(s)$ is another important part of the algorithm. Since the algorithm for computing the eigenvalues of the related skew-Hamiltonian/Hamiltonian pencils from [7, 8] relies on dense matrix algebra, it is not applicable in our setting.

Instead, we propose to use the even IRA algorithm, developed by Mehrmann, Schröder, and Simoncini [21] which can compute several eigenvalues *close to a prespecified shift* (and, if desired, the associated eigenvectors) of large, sparse even pencils.

In order to solve problems of this type, we limit ourselves to the case where the structure of $\mathcal{N}$ and $\mathcal{M}_\gamma$ allows for the use of sparse direct LU-factorizations of $\mathcal{M}_\gamma - \sigma\mathcal{N}$ for some shift σ. In particular, the even IRA algorithm is a structure-preserving method based on Krylov subspaces with implicit restarts [21]. A prespecified number of eigenvalues is calculated in a neighborhood of the shift σ, which is allowed to be either real or purely imaginary. The algorithm actually solves a related eigenvalue problem of the form $\mathcal{K}x = \theta x$, with

$$\mathcal{K} := (\mathcal{M}_\gamma + \sigma\mathcal{N})^{-1}\mathcal{N}(\mathcal{M}_\gamma - \sigma\mathcal{N})^{-1}\mathcal{N}.$$

Then, an eigenvalue pair $(\lambda, -\lambda)$ can be easily extracted from θ by a simple transformation. Since it is desirable for the shifts to be as near as possible to the calculated eigenvalues, the ones used in this paper are the midpoints of the imaginary parts of the eigenvalues of $\mathcal{H}_\gamma(s)$ calculated in the previous iteration, with a slight offset which is explained in Subsection 4.1. In the case of the first iteration, when such eigenvalues have yet to be calculated, the imaginary parts of some of the dominant poles are used

instead. In order to improve upon the accuracy of the eigenvalue calculation method, and to ensure that all desired purely imaginary eigenvalues of the matrix pencil are found, often multiple shifts are used in a loop.

As already mentioned before, in the future we might use an alternative approach making use of a structured rational Krylov method like the rational SHIRA method suggested in [4]. When implemented with a criterion deciding whether or not there exists a purely imaginary eigenvalue in a neighborhood of a given target shift, one could cover the imaginary axis by a certain distribution of shifts and associated neighborhoods including an interval from zero to an upper bound on the eigenvalue of largest magnitude of the structured matrix pencil. Finding a safe criterion to test for the existence of a purely imaginary eigenvalue in a given neighborhood of the target shift is an open problem, though. Therefore, we do not pursue this approach here any further and leave this to future work.

4 Implementation and Numerical Examples

In this section, we outline some of the details regarding the implementation of the structured matrix pencil algorithm, and compare its computed $\mathcal{L}_\infty$-norm and runtime to the pseudopole method. Furthermore, we address some of the limitations of the current algorithm.

4.1 Implementation Details

In this subsection we give details on the implementation of the method in MATLAB. The interface of our routine is given by

```
[ normval, fopt, info ] = linorm( E, A, B, C, D, options ),
```

where `E`, `A`, `B`, `C`, and `D` are the matrices defining the descriptor system, `normval` is the computed norm value, `fopt` is the optimal frequency, and `info` is a struct that contains information about the execution of the code like an error indicator or the runtime. Furthermore, `options` is a struct that defines design parameters and options of the routine. The most important parameters contained in `options` are listed in Table 1.

A parameter of note is `npolinit`, which determines the number of dominant poles calculated by the SAMDP solver. Since the most dominant pole does not always correlate to the test frequency that provides the highest singular value, this parameter is not trivially 1. Furthermore, for some examples this number must be much higher before the pole with the highest dominance is actually found; as such, the default value has been set to 20, although this must be increased for some examples.

Table 1: Most important `linorm` design parameters contained in `options` struct

parameter	description	default value
`tol`	desired relative accuracy of $\mathcal{L}_\infty$-norm value	1e-06
`npolinit`	number of dominant poles computed in initial stage	20
`domtol`	relative cutoff value for the dominance of the poles to determine the initial eigenvalues	0.5
`shifttol`	minimum relative distance between two subsequent shifts	0.01
`mmax`	maximum search space dimension even IRA	8
`maxcycles`	maximum number of restarts for even IRA	30
`nwanted`	number of eigenvalues calculated per shift in even IRA	4
`eigtol`	tolerance on the eigenvalue residual for even IRA	1e-06
`sweeptol`	relative eigenvalue distance for frequency sweep	1e-05
`shiftmv`	relative shift displacement	5e-04

To compute the purely imaginary eigenvalues of $\mathcal{H}_\gamma(s)$ in the initial iteration we use the imaginary parts of the dominant poles as shifts. When we take a large number of shifts we observe that some

eigenvalues occur repeatedly. Therefore, we only use a limited amount of them, determined by the parameter `domtol`. Let λ_j, $j = 1, \ldots, \ell$ be the dominant poles of G. With R_j as in (3) we define the *dominance* by

$$\operatorname{dom}(\lambda_j) = \left\|R_j\right\|_2 / \operatorname{Re}(\lambda_j).$$

Then we determine the shifts only by those dominant poles λ_j that satisfy

$$\operatorname{dom}(\lambda_k) > \texttt{domtol} \cdot \max_{1 \le j \le \ell} \operatorname{dom}(\lambda_j)$$

Moreover, we reduce the number of shifts, if some of them are close together. Therefore, let the shifts $\mathrm{i}\omega_1, \ldots, \mathrm{i}\omega_\ell \in \mathrm{i}\mathbb{R}$ be given in increasing order. If

$$|\omega_{j+1} - \omega_j| < \texttt{shifttol} \cdot \omega_\ell,$$

then we replace the shifts $\mathrm{i}\omega_j$ and $\mathrm{i}\omega_{j+1}$ be a single shift. In the code this is also done for multiple close-by shifts.

The even IRA solver, while providing a fast and relatively robust method for the computation of the eigenvalues of even matrix pencils, can return inaccurate eigenvalues under certain circumstances. Additional security measures pointed out below are put into the code presented here in an attempt to alleviate these issues.

A problem with the even IRA solver arises in the case of close proximity of the given shift to one of the eigenvalues of the matrix pencil to be calculated. In such a case, the eigenvalue close to the shift is calculated correctly, however all *other* eigenvalues have a high degree of inaccuracy. This is thought to be caused by resulting ill-conditioned matrices used in the even IRA solver [29]. This problem occurs fairly often, when taking the eigenvalues from the previous iteration as shifts. This is the origin of the slight offset by the parameter `shiftmv` – if the shifts are displaced by a small relative amount, all inaccuracies in the eigenvalue calculation associated with this problem are eliminated.

Another possible scenario involves a control system where the slope of a singular value plot around the optimal frequency ω_{opt} is very steep – in other words, the $\mathcal{L}_\infty$-norm is achieved at the tip of a very thin spike on the singular value vs. frequency graph, as shown for example in Figure 2. In this case, the eigenvalues of the matrix pencil $\mathcal{H}_\gamma(s)$ will be very close together in the later stages of computation, when the value of γ_{lb} is just outside the tolerance of the actual $\mathcal{L}_\infty$-norm value. The even IRA solver is not able to accurately calculate eigenvalues that are extremely close together which can result in a decreasing γ value in the next steps of the iteration. To prevent this, if the previously calculated eigenvalues have a relative distance less than `sweeptol`, a frequency sweep is performed around the imaginary parts of the eigenvalues to find the $\mathcal{L}_\infty$-norm. Since this process is relatively computationally inexpensive, many points can be evaluated over a large range (relative to the distance between the eigenvalues), so the final norm is usually very accurate. The drawback to this approach is, of course, that the accuracy of the resulting $\mathcal{L}_\infty$-norm is not *guaranteed*. However, since the eigenvalues around which the frequency sweep is performed are so close to the optimal frequency, this method usually still gives a good approximation.

Finally, in some rare occasions, the even IRA solver does not find any purely imaginary eigenvalues of the matrix pencil, even if the provided γ is smaller than the known $\mathcal{L}_\infty$-norm calculated using other methods. Often, the reason for this problem is that the shifts used for the even IRA solver are too far from the imaginary eigenvalues, for instance if the range in which shifts are averaged, is too big. In such a case, adjusting the `domtol` and `shifttol` parameters such that more shifts are obtained, usually solves the problem. Another possible reason is that the tolerance of the eigenvalues given by `eigtol` is too big. Then, the code of even IRA does not return the inaccurate eigenvalues. To obtain a correct result, `eigtol` must be further decreased.

The main disadvantage of our method is the fact that the user must supply good values for the parameters in the `options` struct. Even if the default values work well for most examples, for a few tests we have to change them to get good results.

4.2 Test Setup

The numerical tests are performed on a 2.6.32-23-generic-pae Kubuntu machine with Intel® Core™2 Duo CPU with 3.00GHz and 2GB RAM. The implementation is tested with MATLAB 7.14.0.739

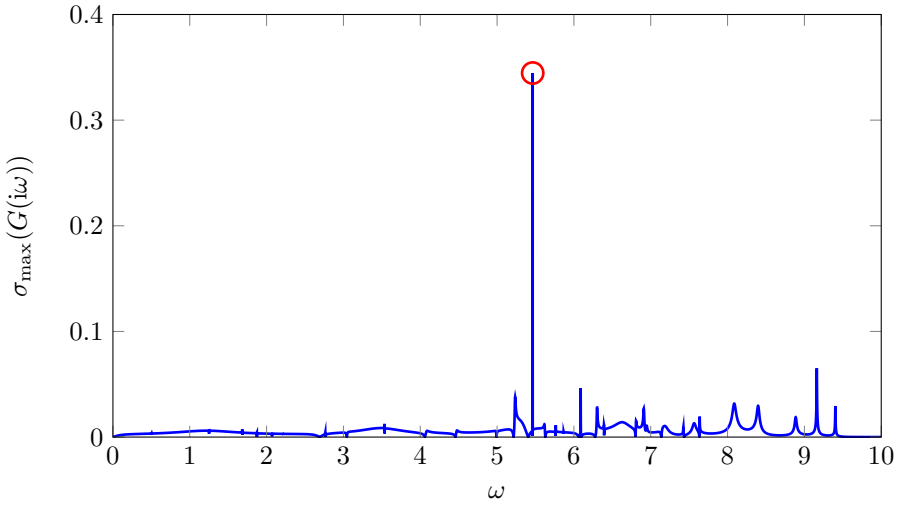

Figure 2: Maximum singular value plot for `peec` example. Note that not all peaks are captured due to plotting resolution.

(R2012a). The data for the numerical examples was taken from [26, 20, 17, 15], and is available on the website of Joost Rommes* and on the SLICOT page†. Furthermore, we use the MATLAB implementation for even IRA from Christian Schröder‡. All examples are stable, so we indeed compute the $\mathcal{H}_\infty$-norm in all tests. For all examples we use the default value for the options as above, except for `beam` (`eigtol = 1e-5`), `peec` (`npolinit = 40`), `bips07_1693` (`npolinit = 30`), and `bips07_1998` (`domtol = 0.2`) which are particularly difficult.

4.3 Test Results and Comparison to Pseudopole Method

In this subsection we analyze computational results of the new approach and compare them with those obtained by the method from [10]. The results for 33 test examples are summarized in Table 2. Using the matrix pencil approach, the correct value of $\|G\|_{\mathcal{H}_\infty}$ was found for all cases. For a few examples (`build`, `beam`, `M80PI_n1`, `M80PI_n`, `mimo8x8_system`), the result differed by only 1e-06 for the different methods, which is in the range of desired accuracy. Only for the `peec` example, the difference between the calculated $\mathcal{H}_\infty$-norm values is significant. The reason is the very thin spike at which the $\mathcal{L}_\infty$-norm is attained, see also Figure 2. In the run using the matrix pencil approach we actually did not find purely imaginary eigenvalues in the first iteration and therefore, we only take $\|G\|_{\mathcal{H}_\infty} = \sigma_{\max}(G(\mathrm{i}\omega_{\mathrm{p}}))$ with the optimal test frequency ω_{p} as defined above. When decreasing `eigtol`, we will get purely imaginary eigenvalues, which are so close together such that a frequency sweep is performed. This leads to a similar result.

In terms of the runtime, the matrix pencil method performs remarkably better than the pseudopole method. In fact, every single numerical example resulted in a decreased runtime. For almost all tests the runtime was reduced to 50% or less, especially for some larger examples we can have a speedup by a factor of 5. Only in the `beam` example, where the dominant pole calculation is the limiting factor, the speedup is much smaller. To demonstrate the behavior of the method, Figure 3 shows the MATLAB output of the function `linorm` for the `bips07_3078` example, along with the intermediate gamma values and imaginary eigenvalues. This printout is illustrated in Figure 4, which shows the intermediate γ values with dotted red lines and the purely imaginary eigenvalues with black crosses. It is easy to see in this illustration how quickly the algorithm converges – in fact, all 33 examples took 3 iterations or fewer.

*`http://sites.google.com/site/rommes/software`
†`http://slicot.org/20-site/126-benchmark-examples-for-model-reduction`
‡`http://www.math.tu-berlin.de/fachgebiete_ag_modnumdiff/fg_numerische_mathematik/v-menue/mitarbeiter/christianschroeder/software/even_ira/`

Table 2: Numerical results for 33 test examples

#	example	n	m	p	computed $\mathcal{H}_\infty$-norm		optimal frequency ω_{opt}		time in s	
					pseudopoles	matrix pencil	pseudopoles	matrix pencil	pseudopoles	matrix pencil
1	build	48	1	1	5.27633e-03	5.27634e-03	5.20608e+00	5.20584e+00	1.54	0.54
2	pde	84	1	1	1.08358e+01	1.08358e+01	0.00000e+00	0.00000e+00	2.08	0.61
3	CDplayer	120	2	2	2.31982e+06	2.31982e+06	2.25682e+01	2.25682e+01	2.70	0.54
4	iss	270	3	3	1.15887e-01	1.15887e-01	7.75093e-01	7.75089e-01	2.69	0.61
5	beam	348	1	1	4.55487e+03	4.55488e+03	1.04575e-01	1.04573e-01	50.22	38.15
6	S10PI_n1	528	1	1	3.97454e+00	3.97454e+00	7.53151e+03	7.53152e+03	1.78	0.79
7	S20PI_n1	1028	1	1	3.44317e+00	3.44317e+00	7.61831e+03	7.61835e+03	4.22	1.33
8	S40PI_n1	2028	1	1	3.34732e+00	3.34732e+00	6.95875e+03	6.95873e+03	4.77	2.12
9	S80PI_n1	4028	1	1	3.37016e+00	3.37016e+00	6.96149e+03	6.96147e+03	10.50	4.80
10	M10PI_n1	528	3	3	4.05662e+00	4.05662e+00	7.53181e+03	7.53182e+03	3.08	1.29
11	M20PI_n1	1028	3	3	3.87260e+00	3.87260e+00	5.06412e+03	5.06412e+03	3.63	1.62
12	M40PI_n1	2028	3	3	3.81767e+00	3.81767e+00	5.07107e+03	5.07107e+03	5.83	2.83
13	M80PI_n1	4028	3	3	3.80375e+00	3.80376e+00	5.07279e+03	5.07279e+03	10.88	5.23
14	peec	480	1	1	3.52651e-01	3.52541e-01	5.46349e+00	5.46349e+00	20.80	9.31
15	S10PI_n	682	1	1	3.97454e+00	3.97454e+00	7.53151e+03	7.53152e+03	2.29	1.03
16	S20PI_n	1182	1	1	3.44317e+00	3.44317e+00	7.61831e+03	7.61835e+03	4.67	1.53
17	S40PI_n	2182	1	1	3.34732e+00	3.34732e+00	6.95875e+03	6.95873e+03	5.35	2.47
18	S80PI_n	4182	1	1	3.37016e+00	3.37016e+00	6.96149e+03	6.96147e+03	10.41	4.51
19	M10PI_n	682	3	3	4.05662e+00	4.05662e+00	7.53181e+03	7.53182e+03	3.56	1.36
20	M20PI_n	1182	3	3	3.87260e+00	3.87260e+00	5.06412e+03	5.06412e+03	4.03	1.73
21	M40PI_n	2182	3	3	3.81767e+00	3.81767e+00	5.07107e+03	5.07107e+03	6.11	2.85
22	M80PI_n	4182	3	3	3.80375e+00	3.80376e+00	5.07279e+03	5.07279e+03	11.28	5.19
23	bips98_606	7135	4	4	2.01956e+02	2.01956e+02	3.81763e+00	3.81763e+00	35.43	14.43
24	bips98_1142	9735	4	4	1.60427e+02	1.60427e+02	4.93005e+00	4.93005e+00	69.65	16.37
25	bips98_1450	11305	4	4	1.97389e+02	1.97389e+02	5.64575e+00	5.64650e+00	61.65	17.91
26	bips07_1693	13275	4	4	2.04168e+02	2.04168e+02	5.53766e+00	5.53767e+00	167.10	31.98
27	bips07_1998	15066	4	4	1.97064e+02	1.97064e+02	6.39968e+00	6.39879e+00	102.11	29.99
28	bips07_2476	16861	4	4	1.89579e+02	1.89579e+02	5.88971e+00	5.89023e+00	146.18	31.62
29	bips07_3078	21128	4	4	2.09445e+02	2.09445e+02	5.55792e+00	5.55839e+00	91.05	34.73
30	xingo_afonso_itaipu	13250	1	1	4.05605e+00	4.05605e+00	1.09165e+00	1.09165e+00	39.24	16.80
31	mimo8x8_system	13309	8	8	5.34292e-02	5.34293e-02	1.03313e+00	1.03308e+00	78.47	23.25
32	mimo28x28_system	13251	28	28	1.18618e-01	1.18618e-01	1.07935e+00	1.07935e+00	85.36	35.45
33	mimo46x46_system	13250	46	46	2.05631e+02	2.05631e+02	1.07908e+00	1.07908e+00	115.91	49.13

```
============================================================
Example 29: bips07_3078
============================================================
The 1 most dominant poles with associated dominance values are:
-0.718 + 5.340i with dominance 192.110186

The 1 most dominant poles are used to calculate the initial gamma value.

The initial lower bound of gamma is 202.322627 at fopt = 5.340484.

For cycle 1:
The positive imaginary eigenvalues for shift = 5.337813287i are:
lambda = 5.340489445i
lambda = 5.850435899i
There are 1 shifts, and 1 shifts that produce imaginary eigenvalues.
There are 4 eigenvalues, and 4 imaginary eigenvalues.
The lower bound of gamma is 209.321519 at fopt = 5.589650.

For cycle 2:
The positive imaginary eigenvalues for shift = 5.592445187i are:
lambda = 5.527349583i
lambda = 5.589595369i
There are 1 shifts, and 1 shifts that produce imaginary eigenvalues.
There are 4 eigenvalues, and 4 imaginary eigenvalues.
The lower bound of gamma is 209.444736 at fopt = 5.558385.

For cycle 3:
There are 1 shifts, and 0 shifts that produce imaginary eigenvalues.
There are 4 eigenvalues, and 0 imaginary eigenvalues.

The L-infinity-norm is 209.445 at fopt = 5.55839.
The runtime is 34.7283 seconds.
```

Figure 3: `linorm` output for the `bips07_3078` example

5 Conclusions and Outlook

In this article, we have implemented a method for computing $\mathcal{L}_\infty$-norms introduced in [7] for transfer functions of descriptor systems using structured iterative eigensolvers. The algorithm computes the imaginary eigenvalues of an even matrix pencil and uses them to determine an upper and lower bound to the $\mathcal{L}_\infty$-norm. In particular, it uses the even IRA algorithm to calculate the eigenvalues of the matrix pencil, and employs the SAMDP algorithm to find the dominant poles of the transfer function, which are used to increase the speed and reliability of the algorithm. The calculated $\mathcal{L}_\infty$-norm values were identical (up to the desired tolerance) or very close to those calculated using a related pseudopole method, and were accompanied by a significant drop in runtime.

An important point of future research is to improve the reliability of the algorithm by finding, e. g., a way to ensure that really *all* purely imaginary eigenvalues of the even pencils are found in order to make the method more independent from the dominant poles and the design parameters.

At the moment we are also working on a subspace projection idea for computing the $\mathcal{H}_\infty$-norm for transfer functions of large-scale systems [1] which we will also compare to the method presented here. Furthermore, while this algorithm can already be used to solve continuous-time problems, it should also be generalized to the discrete-time case.

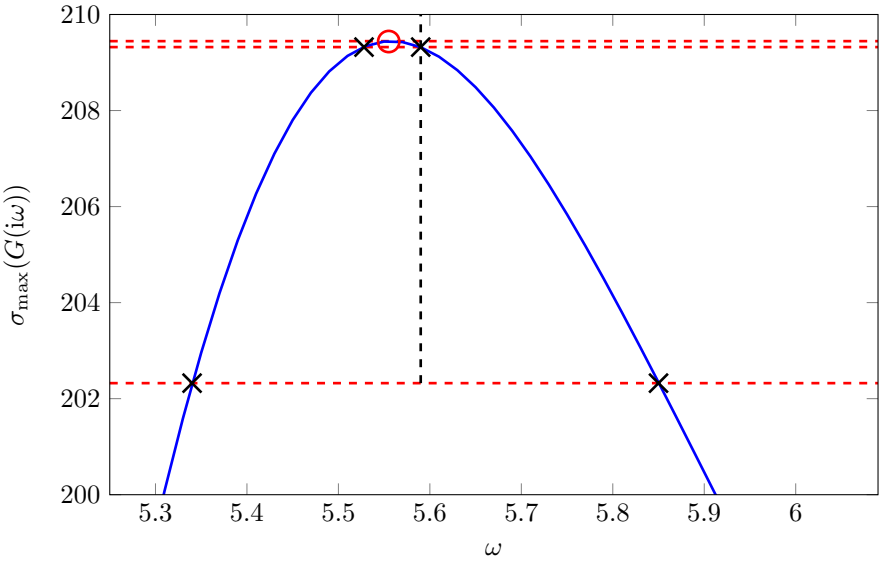

Figure 4: Illustration of the convergence of the `bips07_3078` example

Acknowledgment

The main parts of this work were developed while the second author visited the second author as an intern with the RISE program of the *Deutscher Akademischer Austauschdienst (DAAD)* at the Max Planck Institute for Dynamics of Complex Technical Systems in Magdeburg in summer 2013. We gratefully thank the MPI Magdeburg and the DAAD for their financial support. Furthermore, we thank Valeria Simoncini and Christian Schröder for the helpful discussions and providing the code of even IRA to us.

References

[1] N. ALiyev, P. Benner, E. Mengi, and M. Voigt. Large-scale computation of $\mathcal{H}_\infty$-norms by a greedy subspace method, 2016. In preparation.

[2] P. Benner, R. Byers, P. Losse, V. Mehrmann, and H. Xu. Numerical solution of real skew-Hamiltonian/Hamiltonian eigenproblems, November 2007. Unpublished report.

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

[4] P. Benner and C. Effenberger. A rational SHIRA method for the Hamiltonian eigenvalue problem. *Taiwanese J. Math.*, 14(3):805–823, 2010.

[5] P. Benner and H. Faßbender. An implicitly restarted symplectic Lanczos method for the Hamiltonian eigenvalue problem. *Linear Algebra Appl.*, 263:75–111, 1997.

[6] P. Benner, H. Faßbender, and M. Stoll. A Hamiltonian Krylov-Schur-type method based on the symplectic Lanczos process. *Linear Algebra Appl.*, 435:578–600, 2011.

[7] P. Benner, V. Sima, and M. Voigt. $\mathcal{L}_\infty$-norm computation for continuous-time descriptor systems using structured matrix pencils. *IEEE Trans. Automat. Control*, 57(1):233–238, 2012.

[8] P. Benner, V. Sima, and M. Voigt. Robust and efficient algorithms for $\mathcal{L}_\infty$-norm computation for descriptor systems. In *Proc. 7th IFAC Symposium on Robust Control Design*, pages 195–200, Aalborg, Denmark, 2012. IFAC.

[9] P. Benner, V. Sima, and M. Voigt. Algorithm 961 — Fortran 77 subroutines for the solution of skew-Hamiltonian/Hamiltonian eigenproblems. *ACM Trans. Math. Software*, 42(3), 2016. Paper 24.

[10] P. Benner and M. Voigt. A structured pseudospectral method for $\mathcal{H}_\infty$-norm computation of large-scale descriptor systems. *Math. Control Signals Systems*, 26(2):303–338, 2014.

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

[12] S. Boyd, V. Balakrishnan, and P. Kabamba. A bisection method for computing the H_∞ norm of a transfer matrix and related problems. *Math. Control Signals Systems*, 2(3):207–219, 1989.

[13] N. A. Bruinsma and M. Steinbuch. A fast algorithm to compute the $\mathcal{H}_\infty$-norm of a transfer function matrix. *Systems Control Lett.*, 14(4):287–293, 1990.

[14] J. V. Burke, D. Henrion, A. S. Lewis, and M. L. Overton. HIFOO – A MATLAB package for fixed-order controller design and H_∞ optimization. In *Proc. 5th IFAC Syposium on Robust Control Design*, Toulouse, France, Jul. 2006.

[15] Y. Chahlaoui and P. Van Dooren. A collection of benchmark examples for model reduction of linear time invariant dynamical systems. Technical Report 2002–2, February 2002. Available from `http://www.slicot.org/index.php?site=benchmodred`.

[16] M. A. Freitag, A. Spence, and P. Van Dooren. Calculating the H_∞-norm using the implicit determinant method. *SIAM J. Matrix Anal. Appl.*, 35(2):619–634, 2014.

[17] F. Freitas, J. Rommes, and N. Martins. Gramian-based reduction method applied to large sparse power system descriptor models. *IEEE Trans. Power Syst.*, 23(3):1258–1270, 2008.

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

[19] P. Losse, V. Mehrmann, L. Poppe, and T. Reis. The modified optimal $\mathcal{H}_\infty$ control problem for descriptor systems. *SIAM J. Control Optim.*, 47(6):2795–2811, 2008.

[20] N. Martins, P. C. Pellanda, and J. Rommes. Computation of transfer function dominant zeros with applications to oscillation damping control of large power systems. *IEEE Trans. Power Syst.*, 22(4):1657–1664, 2007.

[21] V. Mehrmann, C. Schröder, and V. Simoncini. An implicitly-restarted Krylov subspace method for real symmetric/skew-symmetric eigenproblems. *Linear Algebra Appl.*, 436:4070–4087, 2012.

[22] V. Mehrmann and T. Stykel. Balanced truncation model reduction for large-scale systems in descriptor form. In P. Benner, V. Mehrmann, and D. Sorensen, editors, *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes Comput. Sci. Eng.*, chapter 3, pages 89–116. Springer-Verlag, Berlin, Heidelberg, New York, 2005.

[23] T. Mitchell and M. L. Overton. Fixed low-order controller design and H_∞ optimization for large-scale dynamical systems. In *Proc. 8th IFAC Symposium on Robust Control Design*, pages 25–30, Bratislava, Slovakia, Jul. 2015.

[24] T. Mitchell and M. L. Overton. Hybrid expansion-contraction: a robust scaleable method for approxiating the H_∞ norm. *IMA J. Numer. Anal.*, 2015. In press.

[25] J. Rommes. Arnoldi and Jacobi-Davidson methods for generalized eigenvalue problems $Ax = \lambda Bx$ with singular B. *Math. Comp.*, 77:995–1015, 2008.

[26] J. Rommes and N. Martins. Efficient computation of multivariable transfer function dominant poles using subspace acceleration. *IEEE Trans. Power Syst.*, 21(4):1471–1487, 2006.

[27] J. Rommes and G. L. G. Sleijpen. Convergence of the dominant pole algorithm and Rayleigh quotient iteration. *SIAM J. Matrix Anal. Appl.*, 30(1):346–363, 2008.

[28] A. Ruhe. Rational Krylov algorithms for nonsymmetric eigenvalue problems. In G. Golub, A. Greenbaum, and M. Luskin, editors, *Recent Advances in Iterative Methods*, volume 60 of *IMA Vol. Math. Appl.*, pages 149–164. Springer-Verlag, New York, 1994.

[29] C. Schröder. Private communication, 2013.

[30] M. Voigt. *On Linear-Quadratic Optimal Control and Robustness of Differential-Algebraic Systems*. Logos-Verlag, Berlin, 2015. Also as Dissertation, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik, 2015.

[31] K. Zhou and J. C. Doyle. *Essentials of Robust Control*. Hemel Hempstead: Prentice Hall, 1997.