Technical Sciences
Academy of Romania
www.jesi.astr.ro

Journal of Engineering Sciences and Innovation
Volume 7, Issue 4 / 2022, p. 463 - 472

F. Electrical, Electronics Engineering, Computer Sciences and Engineering

Received ***5 August 2022*** *Accepted* ***20 December 2022***
Received in revised form ***24 October 2022***

Efficient computation of the L-infinity norm of descriptor systems

VASILE SIMA*

Technical Sciences Academy of Romania, Bucharest, Romania

Abstract. Many applications from industry and technology use models formulated by systems of differential equations. Often, some equations describe additional algebraic constraints. Such systems, referred to as descriptor systems (or singular systems), naturally arise, e.g., in electrical circuit simulation, in multibody dynamics with constraints, or by semidiscretization of certain partial differential equations. A very important characteristic value for a descriptor system is the L_∞-norm of its corresponding transfer function. The computation of this norm is essential in robust control, model order reduction, and other applications. The paper summarizes efficient and reliable algorithms for finding L_∞-norm, for continuous- and discrete-time descriptor systems, which exploit the underlying Hamiltonian or symplectic structure, respectively. An improved solver has been developed and will be made available in the SLICOT Library. Numerical results and comparisons illustrate the good performance and effectiveness of this solver.

Keywords: descriptor system, linear multivariable systems, numerical methods, robust control, software.

1. Introduction

System norms have an important role for the analysis and design of linear dynamical systems. One of the most useful norm is the $\mathcal{L}_\infty$-norm. The computation of this norm is essential in robust control, model order reduction, and other applications. For instance, the $\mathcal{L}_\infty$-norm is used as a robustness measure in the robust control field [1], [2], or as an error measure for model and controller order reduction applications, see [3] and the references therein.
Consider a linear time-invariant (LTI) system

$$E\lambda\big(x(t)\big) = Ax(t) + Bu(t), \quad y(t) = Cx(t) + Du(t), \tag{1}$$

*Correspondence address: vasilesima@ymail.com

where $A, E \in \boldsymbol{R}^{n\times n}$, $B \in \boldsymbol{R}^{n\times m}$, $C \in \boldsymbol{R}^{p\times n}$, $D \in \boldsymbol{R}^{p\times m}$, $x(t) \in \boldsymbol{R}^n$ is the state vector, $y(t) \in \boldsymbol{R}^p$ is the output vector, $u(t) \in \boldsymbol{R}^m$ is the control vector, and $\lambda(x(t))$ is the differential operator, $\mathrm{d}x(t)/\mathrm{d}t$, or the advance difference operator, $\lambda(x(t)) = x(t+1)$, for continuous- and discrete-time case, respectively. The matrix E can be singular. This means that the model (1) may include algebraic constraints, besides differential equations. Such systems, referred to as *descriptor systems* (or *singular systems*), naturally arise, e.g., in electrical circuit simulation, in multibody dynamics with constraints, or by semidiscretization of certain partial differential equations. It will be, however, assumed in the sequel that the matrix pencil $\lambda E - A$ is regular, that is, $\det(\lambda E - A) \not\equiv 0$. The *transfer function* matrix of the system (1) is defined by

$$G(\lambda) = C(\lambda E - A)^{-1}B + D. \tag{2}$$

Let $RL_\infty^{p\times m}(i\omega)$ and $RL_\infty^{p\times m}(e^{i\omega})$ be the rational subspaces of Banach spaces of all $p \times m$ matrix-valued functions that are bounded on the imaginary axis, or the unit circle, for continuous- and discrete-time systems, respectively. Each $G \in RL_\infty^{p\times m}(i\omega)$ or $G \in RL_\infty^{p\times m}(e^{i\omega})$ has a realization of the form (1). The $\mathcal{L}_\infty$-norm is defined by

$$\|G\|_{L_\infty} := sup_{\omega\in\boldsymbol{R}}\sigma_{max}(G(i\omega)), \tag{3}$$

for $G \in RL_\infty^{p\times m}(i\omega)$, and by

$$\|G\|_{L_\infty} := sup_{\omega\in[-\pi,\pi)}\sigma_{max}(G(e^{i\omega})), \tag{4}$$

for $G \in RL_\infty^{p\times m}(e^{i\omega})$, where $\sigma_{max}(.)$ denotes the maximum singular value. As a convention, $\|G\|_{L_\infty} = \infty$ if G is not in the corresponding subspace. For continuous-time systems, this happens when G has purely imaginary poles, or when it is improper, that is, $\lim_{\omega\to\infty} G(i\omega) = \infty$. For discrete-time systems, G is not in $RL_\infty^{p\times m}(e^{i\omega})$ when G has *unitary* poles, that is, poles on the unit circle. The poles of G are the controllable and observable eigenvalues of the matrix pencil $\lambda E - A$. For stable systems, the $\mathcal{L}_\infty$-norm coincides with the $\mathcal{H}_\infty$-norm.

There is a connection between the singular values of $G(i\omega)$ or $G(e^{i\omega})$ and the finite, purely imaginary or unitary eigenvalues, respectively, of some structured matrix pencils [4]. For continuous-time systems, such a pencil is

$$H_c(\gamma) = \begin{bmatrix} \lambda E - A & 0 \\ 0 & \lambda E^T + A^T \end{bmatrix} - \begin{bmatrix} B & 0 \\ 0 & -C^T \end{bmatrix} \begin{bmatrix} -D & \gamma I_p \\ \gamma I_m & -D^T \end{bmatrix}^{-1} \begin{bmatrix} C & 0 \\ 0 & B^T \end{bmatrix}, \tag{5}$$

where γ is a parameter and I_q denotes the identity matrix of order q. The pencil corresponding to discrete-time systems, $H_d(\gamma)$, has a similar formula, but the (2,2) elements of the first two matrices in the right-hand side are $\lambda A^T - E^T$ and $-\lambda C^T$, respectively. (See, for instance, [5] and the references therein.) The following *theorem* is proven in [6]: Assume that $G \in RL_\infty^{p\times m}(i\omega)$, $\gamma > 0$ is not a singular value of D and $\omega_0 \in \boldsymbol{R}$. Then, γ is a singular value of $G(i\omega_0)$ if and only if $H_c(\gamma)$ has the eigenvalue $i\omega_0$. For discrete-time systems, the result is: Assume that $G \in RL_\infty^{p\times m}(e^{i\omega})$, $\gamma > 0$ is not a singular value of D and $\omega_0 \in [-\pi, \pi)$. Then, γ is a singular value of $G(e^{i\omega_0})$ if and only if $H_d(\gamma)$ has the eigenvalue $e^{i\omega_0}$.

The results above have the following consequences [5]: Assume that $G \in RL_\infty^{p\times m}(i\omega)$ and let $\gamma > \min_{\omega\in\boldsymbol{R}} \sigma_{max}(G(i\omega))$ be not a singular value of D. Then, $\|G\|_{L_\infty} \geq \gamma$ if and only if $H_c(\gamma)$ has finite, purely imaginary eigenvalues. Similarly, assume that $G \in RL_\infty^{p\times m}(e^{i\omega})$

and let $\gamma > \min\limits_{\omega\in[-\pi,\pi)} \sigma_{max}(G(e^{i\omega}))$ be not a singular value of D. Then, $\|G\|_{L_\infty} \geq \gamma$ if and only if $H_d(\gamma)$ has unitary eigenvalues. The first part has been proven in [6].

These results allow to extend to descriptor systems the quadratically convergent method in [7], [8] for the computation of the $\mathcal{L}_\infty$-norm.

Conceptual algorithms are presented in [5], [6]. They start with an initial lower bound, γ_l, for the $\mathcal{L}_\infty$-norm; γ_l is found by evaluating $\sigma_{max}(G(i\omega))$ or $\sigma_{max}(G(e^{i\omega}))$ on the boundaries of the frequency intervals $[0, \infty)$ or $[0,\pi)$, respectively, and on further well-chosen inner test frequencies. At each iteration of the algorithm, a value γ is set as $\gamma = (1+\varepsilon)\gamma_l$, where ε is a given tolerance. Then, the finite, purely imaginary or unitary eigenvalues, $i\omega_j$ or $e^{i\omega_j}$, $j = 1, \ldots, k$, of $H_c(\gamma)$ or $H_d(\gamma)$, respectively, are used, via a bisection technique, to improve the approximation of the lower bound. Specifically, the *midpoints*, $m_j = \sqrt{\omega_j\omega_{j+1}}$ or $m_j = (\omega_j+\omega_{j+1})/2$, respectively, $j = 1, \ldots, k-1$, are obtained and the maximum over j of $\sigma_{max}(G(im_j))$ or $\sigma_{max}(G(e^{im_j}))$ is computed. The maximum of all these singular values is then used as the new value of γ_l. When no purely imaginary or unitary eigenvalues are found, the $\mathcal{L}_\infty$-norm is set to as $\|G\|_{L_\infty} = \gamma_l$.

The pencils $H_c(\gamma)$ and $H_d(\gamma)$ have a special structure: $H_c(\gamma)$ is a *skew-Hamiltonian/Hamiltonian* pencil, and $H_d(\gamma)$ is a generalization of a *symplectic* pencil. This implies eigenvalue symmetry with respect to both real and imaginary axes or with respect to the unit circle in the complex space, respectively. Clearly, finding reliable and accurate eigenvalues of these pencils is of paramount importance for avoiding failures and increasing the rate of convergence of the $\mathcal{L}_\infty$-norm computational algorithms. In order to make this possible, the pencils $H_c(\gamma)$ and $H_d(\gamma)$ are transformed to some equivalent *even* matrix pencils, and then to *skew-Hamiltonian/Hamiltonian* pencils, whose spectra can be computed with structure-preserving algorithms [7], [8].

2. Robust computation of eigenvalues

Even the best general, numerically stable algorithms for eigenvalue computation may deliver very inaccurate results for the matrix pencils $H_c(\gamma)$ and $H_d(\gamma)$. Directly evaluating the matrices of these pencils can produce wrong outcomes. The matrix to be inverted in (5) is very ill-conditioned if γ is close to a singular value of D. This loss of accuracy can be avoided by using the *Schur complement*. Specifically, $H_c(\gamma)$ and $H_d(\gamma)$ are replaced by the extended matrix pencils [5], of order $\hat{n} = 2n + p + m$,

$$\widehat{H}_c(\gamma) = \begin{bmatrix} \lambda E - A & 0 & -B & 0 \\ 0 & \lambda E^T + A^T & 0 & C^T \\ -C & 0 & -D & \gamma I_p \\ 0 & -B^T & \gamma I_m & -D^T \end{bmatrix}$$

$$\widehat{H}_d(\gamma) = \begin{bmatrix} \lambda E - A & 0 & -B & 0 \\ 0 & \lambda A^T - E^T & 0 & \lambda C^T \\ -C & 0 & -D & \gamma I_p \\ 0 & -B^T & \gamma I_m & -D^T \end{bmatrix}, \quad (6)$$

which have the same finite eigenvalues as the original pencils, but only use the given data. By doing some block-permutations (and transposing $\widehat{H}_d(\gamma)$), the following pencils are obtained

$$\widetilde{H}_c(\gamma) = \begin{bmatrix} 0 & \lambda E - A & 0 & -B \\ -\lambda E^T - A^T & 0 & -C^T & 0 \\ 0 & -C & \gamma I_p & -D \\ -B^T & 0 & -D^T & \gamma I_m \end{bmatrix},$$

$$\breve{H}_d(\gamma) = \begin{bmatrix} 0 & A^T - \lambda E^T & C^T & 0 \\ \lambda A - E & 0 & 0 & -B \\ \lambda C & 0 & \gamma I_p & -D \\ 0 & -B^T & -D^T & \gamma I_m \end{bmatrix}. \tag{7}$$

The pencil $\widetilde{H}_c(\gamma)$ is *even*, that is, $\tilde{S} = -\tilde{S}^T$ and $\tilde{H} = \tilde{H}^T$, where $\widetilde{H}_c(\gamma) =: \lambda\tilde{S} - \tilde{H}$, while $\breve{H}_d(\gamma)$ is a *D-type pencil* [9] and it has a symplectic eigenstructure. By applying the *generalized Cayley transform* and an additional *drop/add transformation* [9] to $\breve{H}_d(\gamma)$, an even pencil is also obtained

$$\widetilde{H}_d(\gamma) = \lambda \begin{bmatrix} 0 & -A^T - E^T & -C^T & 0 \\ A + E & 0 & 0 & 0 \\ C & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & E^T - A^T & -C^T & 0 \\ E - A & 0 & 0 & B \\ -C & 0 & -\gamma I_p & D \\ 0 & B^T & D^T & -\gamma I_m \end{bmatrix}. \tag{8}$$

If $\bar{n} = \hat{n}$ is even, then both $\widetilde{H}_c(\gamma)$ and $\widetilde{H}_d(\gamma)$ can be transformed to *skew-Hamiltonian/Hamiltonian* pencils, via the transformation $\overline{H}_c(\gamma) = \widetilde{H}_c(\gamma)J$, $\overline{H}_d(\gamma) = \widetilde{H}_d(\gamma)J$, where J is defined in (9). If $\hat{n}$ is odd, then $\widetilde{H}_c(\gamma)$ and $\widetilde{H}_d(\gamma)$ are further extended by a row and column, with all elements zero, but the last one set to 1; the corresponding $\overline{H}_c(\gamma)$ and $\overline{H}_d(\gamma)$ will therefore have an even order, $\bar{n} = \hat{n} + 1$. $\overline{H}_c(\gamma)$ has the same finite eigenvalues as $H_c(\gamma)$. The finite eigenvalues of $H_d(\gamma)$ can be obtained from those of $\overline{H}_d(\gamma)$ using the inverse Cayley transformation.

There are special algorithms which exploit the structure of the pencils $\overline{H}_c(\gamma)$ or $\overline{H}_c(\gamma)$ and ensure the needed symmetry of the spectra [7], [8]. These algorithms reduce a regular real skew-Hamiltonian/Hamiltonian matrix pencil of order $2\bar{n}$, $\lambda S - H$, where S is skew-Hamiltonian and H is Hamiltonian, to the following form

$$Q_1^T S J Q_1 J^T = \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{11}^T \end{bmatrix}, \quad J Q_2^T J^T S Q_2 = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{11}^T \end{bmatrix} =: T,$$

$$Q_1^T H Q_2 = \begin{bmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{bmatrix}, \quad J := \begin{bmatrix} 0 & I_{\bar{n}} \\ -I_{\bar{n}} & 0 \end{bmatrix}, \tag{9}$$

where Q_1 and Q_2 are orthogonal matrices, S_{11}, T_{11}, H_{11} are upper triangular, H_{22}^T is upper quasi-triangular (that is, block upper-triangular with 1×1 and 2×2 diagonal blocks), and the *formal* matrix product $S_{11}^{-1}H_{11}T_{11}^{-1}H_{22}^T$ is in a real *periodic Schur form* [10]. The first two matrices in (9) are skew-Hamiltonian and the third one is Hamiltonian. The spectrum of $\lambda S - H$ is given by

$$\lambda(S, H) = \pm i\sqrt{\lambda(S_{11}^{-1}H_{11}T_{11}^{-1}H_{22}^T)}, \tag{10}$$

and it can be obtained by using the diagonal blocks. It follows that the finite, purely imaginary eigenvalues correspond to the 1×1 diagonal blocks of the formal matrix product. Consequently, there will be no error in the real parts, hence, a robust and reliable detection of the desired eigenvalues is achieved. Details are given in [8]. The submatrices S_{11} and T_{11} in (10) can be singular. The eigenvalues of the formal matrix product are found using the iterative *periodic QZ algorithm* (pQZ) [10]. To increase the convergence rate, implicitly defined shifts are used and applied via an embedding of the Wilkinson polynomial. But the implicit approach may not converge for some periodic eigenvalue problems, since the shifts involved may be indefinitely unsuitable. Several improvements have been proposed in [11]-[13] to avoid failures and reduce the number of iterations. For instance, in a semi-implicit approach [12] the shifts are chosen based on eigenvalues computed explicitly using a special pQZ algorithm for subproblems of order two. Moreover, it was found that alternating implicit and semi-implicit iterations offers the advantages of both approaches, improving the behavior of the pQZ algorithm [14].
The computation of spectra for skew-Hamiltonian/Hamiltonian matrix pencils, as well as for formal matrix products can be performed using subroutines from the SLICOT Library [15], available on GitHub, https://github.com/SLICOT/SLICOT-Reference.

3. Implementation issues

A preliminary version of the $\mathcal{L}_\infty$-norm solver has been developed several years ago [5], [6]. Very recently, a new version, linorms, has been prepared and exhaustively tested. This version has many options and high flexibility. The solver works on both continuous- and discrete-time, standard or descriptor systems, with or without a feedthrough matrix D. The given matrices A, E, B, and C can optionally be balanced, to make the rows and columns of the system pencil matrix as close in norm as possible. Additional scaling can be performed for matrices with too large or too small elements, to avoid overflows during computations. There are options to check the properness of the transfer function matrix of a continuous-time descriptor system, and to reduce the system order (before computing the $\mathcal{L}_\infty$-norm), by removing all uncontrollable and unobservable poles. It is possible to specify an estimate of the frequency where the gain of the frequency response would achieve its peak value. The tolerance ε used to set the accuracy in determining the $\mathcal{L}_\infty$-norm should be specified, but other tolerances have default values. The optimal sizes of the real and complex working arrays can be computed by the solver using a special call with those sizes set to -1; the two returned values can then be given as input arguments in a second solver call. The use of this feature could significantly reduce the computing time for systems with large order.
The latest versions of the implementations for solving skew-Hamiltonian/Hamiltonian eigenvalue problems, as well as for the pQZ algorithm have been used.

4. Numerical results

An extensive testing has been performed to evaluate the new solver. The computations have been done in double precision on an Intel Core i7-3820QM portable computer (2.7 GHz, 16 GB RAM). An executable MEX-file has been built using the new solver, SLICOT routines and MATLAB-provided optimized LAPACK and BLAS routines. Tests with randomly generated matrices (from a uniform distribution), as well as with LTI systems from the

COMPl$_e$ib collection [16], have been run. The results have been compared to those returned by the MATLAB function norm from Release 2021b.

Table 1 shows comparative results for a set of small order randomly generated systems, with $n = 0:10$, $m = 0:15$, $p = 0:15$, where $x = i:j$ means $x = i, i+1, \ldots, j$. Three cases for the matrix E have been tried: general nonsingular E, singular E with randomly chosen rank, and $E = I_n$. Each system has been considered as either continuous- or discrete-time. Moreover, both options for balancing, checking properness, and reducing the system order, have been activated using program loops. All these combinations generated 135168 calls of linorms and 33792 calls of norm (for which the last two options cannot be enforced). The table heading shows the desired tolerance values. The notation ε_M means the machine precision ($\varepsilon_M \approx 2.22\ 10^{-16}$). For the last column, the tolerance values 10^{-4} and $\sqrt{\varepsilon_M}$ have been used for norm and linorms, respectively. The first row after the heading row shows the maximum relative error between the results returned by norm and linorms, considering both the $\mathcal{L}_\infty$-norm value and the associated frequency. Clearly, the relative error agrees to the specified tolerance. The next three rows present the sum of the CPU times (in seconds) for all 33792 runs for norm and linorms, as well their ratios. (For linorms, the CPU time has only been recorded without activating the options for checking the properness and reducing the system order.) The new solver is globally over five times faster than norm.

Table 1. Comparative results for small order examples ($n = 0:10, m = 0:15, p = 0:15$).

tol (ε)	$\sqrt{\varepsilon_M}$	10^{-6}	10^{-4}	$10^{-4}/\sqrt{\varepsilon_M}$
max error	$1.48\ 10^{-8}$	$9.97\ 10^{-7}$	$9.99\ 10^{-5}$	$1.16\ 10^{-5}$
norm time	82.7	73.6	72.6	73.4
linorms time	14.2	13.0	12.0	14.0
time ratio	5.82	5.66	6.05	5.24

Table 2 compares similarly the performance of norm and linorms for randomly generated systems with larger order. The notation x/y in the heading row means that $n =$ x, $m =$ y, and $p =$ y. The values for m and p are chosen much smaller than n, since this is the usual case in practice. There are 48 calls of linorms and 48/4=12 calls of norm for each system size. The tolerance $\varepsilon = \sqrt{\varepsilon_M}$ has been used for all runs. Clearly, the maximum relative error between the linorms and norm results is smaller than $\sqrt{\varepsilon_M}$. The ratios between the total CPU time for norm and linorms (for 12 calls) are between 1.22 and 2.4, usually with larger values for larger sizes. While this speedup is not so impressive as for small order examples, there is a big gain in computation time (for instance, compare 466 with 1120 seconds).

Table 2. Comparative results for larger order examples ($n \le 800, m \le 50, p \le 50$).

size	100/5	200/10	300/20	400/20	500/50	600/50	800/50
max error	$1.59\ 10^{-9}$	$1.35\ 10^{-8}$	$3.46\ 10^{-10}$	$7.74\ 10^{-10}$	$3.57\ 10^{-11}$	$2.02\ 10^{-9}$	$7.48\ 10^{-9}$
norm time	1.15	5.19	19.7	66.9	150	301	1120
linorms time	0.71	3.67	16.2	39.1	111	173	466
time ratio	1.62	1.41	1.22	1.71	1.35	1.74	2.4

Many other numerical tests have been run for systems from [16]. This collection includes 124 standard continuous-time systems ($E = I_n$), with several variations. All but 16 problems (for systems of order larger than 2000, with matrices in sparse format) have been tried. The number of solved examples is 152. The tolerance $\sqrt{\varepsilon_M}$ has been used for all runs. The relative error between the results returned by norm and linorms have usually been of the order $\sqrt{\varepsilon_M}$, sometimes much smaller.

Figure 1 shows the CPU time in seconds for $\mathcal{L}_\infty$-norm computation for all 152 systems, assuming that $D = 0$, which is often the case for continuous-time systems. The maximum dimensions are $n = 1006$, $m = 11$, and $p = 30$. The sums of the total time needed by norm and linorms for all systems are 644.03s and 127.06s, respectively. The ratios between the CPU times for norm and linorms for all examples are displayed in Fig. 2 as a bar plot. The minimum, maximum, and mean values of these ratios are 1.78, 191.13, and 25.82, respectively.

Similarly, Fig. 3 and Fig. 4 display the performance results when the systems are considered as being discrete-time. The sums of the total time needed by norm and linorms for all systems are 591.07s and 337.50s, respectively. The minimum, maximum, and mean values of the CPU time ratios are 0.91, 74.47, and 7.59, respectively.

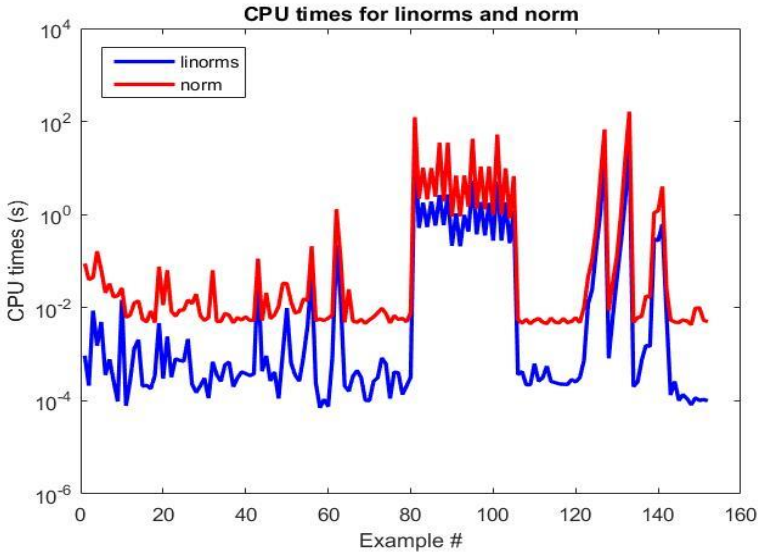

Fig. 1. CPU times in seconds for 152 COMPl$_e$ib examples ($D = 0$) using linorms and norm.

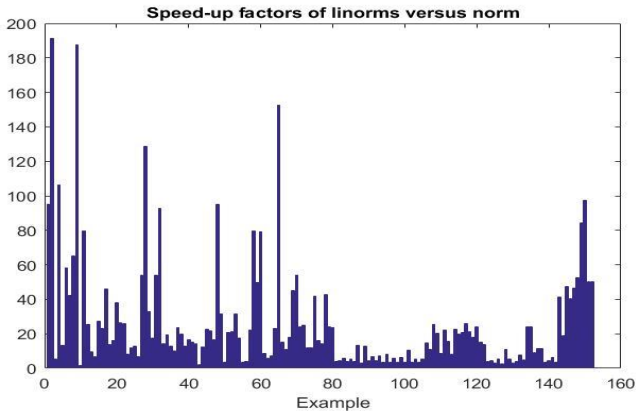

Fig. 2. Ratios of the CPU times in seconds for norm and linorms for 152 COMPl$_e$ib examples ($D = 0$).

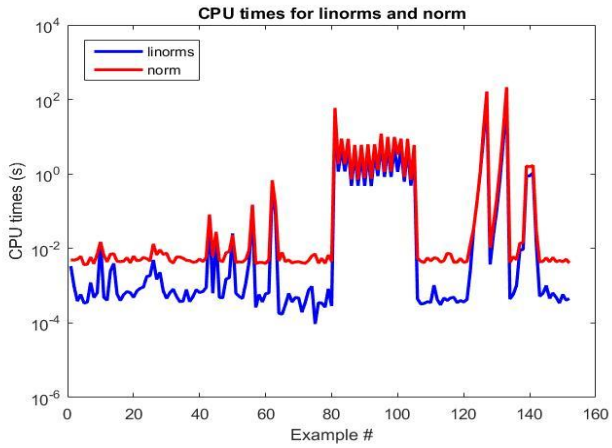

Fig. 3. CPU times in seconds for 152 COMPl$_e$ib examples, taken as discrete-time ($D = 0$), using linorms and norm.

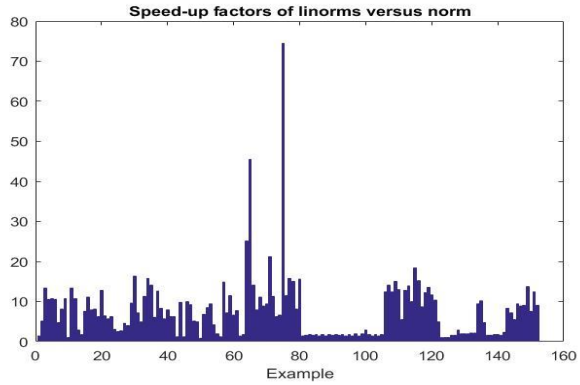

Fig. 4. Ratios of the CPU times in seconds for norm and linorms for 152 COMPl$_e$ib examples, taken as discrete-time ($D = 0$).

Figure 5 shows the CPU times for all 152 continuous-time systems, assuming that $D = D_1 := [\ D_{21}\ 0\ ]$, where D_{21} is defined in [16] and the zero submatrix has size (p, m). The maximum dimensions in this case are $n = 1006$, $m = 578$, and $p = 30$. The sums of the total time needed by norm and linorms for all systems are 656.36s and 184.97s, respectively. The minimum, maximum, and mean values of the CPU time ratios are 1.06, 3176.92, and 39.85, respectively. The maximum value of the ratios has been limited to 200 in Fig. 6. All 24 examples with abcissas in the range 82:105 have $m = n + 2 \geq 258$, and seven examples have $m > 500$. The computational effort of linorms for these examples is much higher than for the case $D = 0$, since the order $\bar{n}$ of the pencils $\bar{H}_c(\gamma)$, derived from (7), increases significantly.

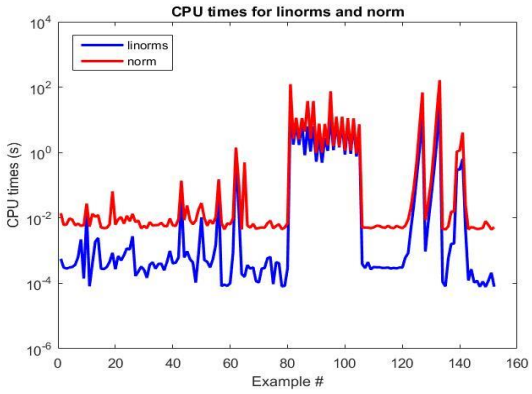

Fig. 5. CPU times in seconds for 152 COMPl$_e$ib examples ($D = D_1$), using linorms and norm.

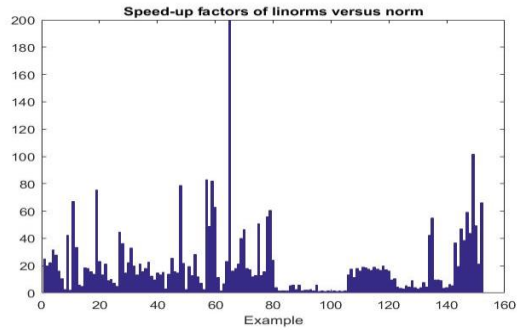

Fig.6. Ratios of the CPU times in seconds for norm and linorms for 152 COMPl$_e$ib examples ($D = D_1$).

5. Conclusions

A very important characteristic value for a descriptor system is the $\mathcal{L}_\infty$-norm of its corresponding transfer function matrix. The computation of this norm is essential in robust control, model order reduction, and other applications. Efficient and reliable algorithms for finding the $\mathcal{L}_\infty$-norm for continuous- and discrete-time descriptor systems have been briefly described. The underlying Hamiltonian or symplectic structure of the associated matrix pencils is exploited. The original pencils are transformed into skew-Hamiltonian/Hamiltonian pencils, conserving the finite eigenvalues for continuous-time systems; for discrete-time systems the eigenvalues are mapped by a Cayley transform. This

endeavor allowed the use of structure-exploiting algorithms for eigenvalue computation during the iterative process. An improved solver has been developed and will be made available in the SLICOT Library. Numerical results and comparisons with the state-of-the-art MATLAB function norm illustrate the good performance and effectiveness of this new solver.

References

[1] Zhou K., Doyle J.D., *Essentials of Robust Control*, Prentice Hall, 1st edition, 1998.

[2] Losse P., Mehrmann V., Poppe L., Reis T., *The modified optimal* H_∞ *control problem for descriptor systems*. SIAM J. Control Optim., **47**, 6, 2008, P. 2795-2811.

[3] Mehrmann V. and Stykel T., *Balanced truncation model reduction for large-scale systems in descriptor form*, Benner, P., Mehrmann, V., D. Sorensen (eds.), Dimension Reduction of Large-Scale Systems, vol. 45 of Lecture Notes in Computational Science and Engineering, ch. 3, p. 89-116, Springer-Verlag, Berlin, Heidelberg, New York, 2005.

[4] Genin Y., Van Dooren P., Vermaut V., *Convergence of the calculation of* H_∞*-norms and related questions*, Proceedings MTNS-98, 1998, p. 429-432.

[5] Benner P., Sima V. and Voigt M., *Robust and efficient algorithms for* L_∞*-norm computation for descriptor systems*, Preprints of the 7th IFAC Symposium on Robust Control Design (ROCOND'12), Aalborg, Denmark, June 20–22, 2012, p. 189-194.

[6] Benner P., Sima V. and Voigt M., L_∞*-norm computation for continuous-time descriptor systems using structured matrix pencils*, IEEE Trans. Automat. Control, 57, no 1 (2012), 233-238.

[7] Benner P., Byers R., Losse P., Mehrmann V., Xu H., *Numerical solution of real skew-Hamiltonian/Hamiltonian eigenproblems*, Techn. Rep., Nov. 2007.

[8] Benner P., Sima V., Voigt M., *Algorithm 961: FORTRAN 77 subroutines for the solution of skew-Hamiltonian/Hamiltonian eigenproblems*, ACM Trans. Math. Softw., **42**, 3, Article No. 24, 2016, p. 1-26.

[9] Xu H., *On equivalence of pencils from discrete-time and continuous-time control*, Lin. Alg. Appl., **414**, 1, 2006, p. 97-124.

[10] Bojanczyk A.W., Golub G., Van Dooren P., *The periodic Schur decomposition: Algorithms and applications*, Luk, F.T. (ed.) SPIE Conference Advanced Signal Processing Algorithms, Architectures, and Implementations III, vol. 1770, 1992, p. 31–42.

[11] Sima V., *Computation of initial transformation for implicit double step in the periodic QZ algorithm*, Precup, R.E. (ed.) 2019 23th International Conference on System Theory, Control and Computing, 2019, p. 7-12.

[12] Sima V., *A new semi-implicit approach for the periodic QZ algorithm*, 2020 24th International Conference on System Theory, Control and Computing, 2020, p. 190-195.

[13] Sima V., Gahinet P., *Improving the convergence of the periodic QZ algorithm*, Gusikhin, O., Madani, K., Zaytoon, J. (eds.), 16th International Conference on Informatics in Control, Automation and Robotics, vol. 1, 2019, p. 261-268.

[14] Sima V., Gahinet P., *Alternating implicit and semi-implicit iterations in the periodic QZ algorithm*, Gusikhin, O., Madani, K., Zaytoon (Eds.), Informatics in Control, Automation and Robotics, Lecture Notes in Electrical Engineering, vol. 793, Springer International Publishing, Cham, p. 47-71, 2022.

[15] Benner P., Mehrmann V., Sima V., Van Huffel S., Varga A., *SLICOT — A subroutine library in systems and control theory*, Applied and Computational Control, Signals, and Circuits, Datta B. N. (ed.), Birkhäuser, Boston, MA, 1, 10,1999, p. 499–539.

[16] Leibfritz F., Lipinski W., *Description of the benchmark examples in COMPl$_e$ib*, Technical Report, Department of Mathematics, University of Trier, D-54286 Trier, Germany, 2003.