Computation of the $\mathcal{L}_\infty$-Norm Using Rational Interpolation

Paul Schwerdtner[1] **Matthias Voigt[2]**

[1]Mitsubishi Electric Research Laboratories
Cambridge, MA, USA

[2]Technische Universität Berlin
Institut für Mathematik
Berlin, Germany

Joint 9th IFAC Symposium on Robust Control Design and 2nd IFAC Workshop on Linear Parameter Varying Systems
Section: $\mathcal{H}_2$ and $\mathcal{H}_\infty$ Control
Florianópolis, Brazil
September 03–05, 2018

Technische Universität Berlin

$\mathcal{L}_\infty$-Functions

Consider a function

$$H : \Omega \to \mathbb{C}^{p \times m}$$

where Ω is assumed to be an open subset of the complex plane enclosing the imaginary axis $\mathrm{i}\mathbb{R}$.

$\mathcal{L}_\infty$-Functions

Consider a function

$$H : \Omega \to \mathbb{C}^{p \times m}$$

where Ω is assumed to be an open subset of the complex plane enclosing the imaginary axis $\mathrm{i}\mathbb{R}$.

Typical examples: Transfer functions of

a) differential-algebraic control systems/descriptor systems:

$$H(s) = C(sE - A)^{-1}B,$$

b) higher-order systems:

$$H(s) = C\left(s^2M + sD + K\right)^{-1}B,$$

c) delay differential-algebraic equations:

$$H(s) = C\left(sE - A_0 - \mathrm{e}^{-s\tau_1}A_1 - \mathrm{e}^{-s\tau_2}A_2\right)^{-1}B,$$

d) systems with input/output delays:

$$H(s) = C(sE - A)^{-1}B\mathrm{e}^{-s\tau}.$$

$\mathcal{L}_\infty$-Functions

Consider the space

$$\mathcal{L}_\infty^{p\times m} := \left\{ H|_{\mathrm{i}\mathbb{R}} \;\middle|\; H : \Omega \to \mathbb{C}^{p\times m} \text{ is analytic for an open } \Omega \subseteq \mathbb{C} \text{ with } \mathrm{i}\mathbb{R} \subset \Omega \text{ and } \sup_{\omega\in\mathbb{R}} \|H(\mathrm{i}\omega)\|_2 < \infty \right\}$$

with the induced norm

$$\|H\|_{\mathcal{L}_\infty} := \sup_{\omega\in\mathbb{R}} \|H(\mathrm{i}\omega)\|_2 .$$

Main assumption: $H \in \mathcal{L}_\infty^{p\times m}$ (we write this instead of $H|_{\mathrm{i}\mathbb{R}} \in \mathcal{L}_\infty^{p\times m}$).
Goal of this talk: Computation of the $\mathcal{L}_\infty$-norm for this general class of functions.

Literature Review

- Methods for rational functions $H(s) = C(sE - A)^{-1}B$:
  - first algorithms based on Hamiltonian eigenvalue problems: [Byers '88], [Boyd, Balakrishnan '90]
  - extension to transfer functions of descriptor systems: [Benner, Sima, V. '12]
  - extensions to large-scale problems:
    - methods based on pseudospectra: [Guglielmi, Gürbüzbalaban, Overton '13], [Benner, V. '14], [Mitchell, Overton '16]
    - using the implicit determinant method: [Freitag, Spence, Van Dooren '14]
    - using iterative eigensolvers: [Benner, Lowe, V. '18]

Literature Review

- Methods for rational functions $H(s) = C(sE - A)^{-1}B$:
  - first algorithms based on Hamiltonian eigenvalue problems: [Byers '88], [Boyd, Balakrishnan '90]
  - extension to transfer functions of descriptor systems: [Benner, Sima, V. '12]
  - extensions to large-scale problems:
    - methods based on pseudospectra: [Guglielmi, Gürbüzbalaban, Overton '13], [Benner, V. '14], [Mitchell, Overton '16]
    - using the implicit determinant method: [Freitag, Spence, Van Dooren '14]
    - using iterative eigensolvers: [Benner, Lowe, V. '18]
- Methods for (possibly) irrational functions $H(s)$:
  - extension of the Hamiltonian method for transfer functions of delay systems: [Michiels, Gummussoy '10]
  - methods based on eigenvalue optimization algorithms: [Mengi, Yildirim, Kiliç '10]
  - methods for large-scale systems based on model reduction techniques: [Aliyev, Benner, Mengi, Schwerdtner, V. '17]

Algorithm from [Aliyev, Benner, Mengi, Schwerdtner, V. '17]

We have

$$H(s) = C(s)D(s)^{-1}B(s). \qquad \big((p \times n) \cdot (n \times n) \cdot (n \times m)\big)$$

where $n \gg p$, m, i. e., the large-scale part of the problem is the middle factor. Do model reduction to make the middle factor small, i. e., determine $V,\ W \in \mathbb{R}^{n \times k}$ such that

$$\widetilde{H}(s) = \widetilde{C}(s)\widetilde{D}(s)^{-1}\widetilde{B}(s),$$

with

$$\widetilde{C}(s) = C(s)V, \quad \widetilde{D}(s) = W^{\mathsf{H}}D(s)V, \quad \widetilde{B}(s) = W^{\mathsf{H}}B(s).$$

Algorithm from [Aliyev, Benner, Mengi, Schwerdtner, V. '17]

We have

$$H(s) = C(s)D(s)^{-1}B(s). \qquad \big((p \times n) \cdot (n \times n) \cdot (n \times m)\big)$$

where $n \gg p$, m, i. e., the large-scale part of the problem is the middle factor.
Do model reduction to make the middle factor small, i. e., determine V, $W \in \mathbb{R}^{n \times k}$ such that

$$\widetilde{H}(s) = \widetilde{C}(s)\widetilde{D}(s)^{-1}\widetilde{B}(s),$$

with

$$\widetilde{C}(s) = C(s)V, \quad \widetilde{D}(s) = W^{\mathsf{H}}D(s)V, \quad \widetilde{B}(s) = W^{\mathsf{H}}B(s).$$

Question: How to choose the projection matrices V and W?

Choice of the projection matrices

Choose a set of interpolation points $\mathbb{W} = \{\mathrm{i}\omega_1, \ldots, \mathrm{i}\omega_r\} \subset \mathrm{i}\mathbb{R}$. Under the assumption that $m = p$ choose

$$\mathcal{V} = \operatorname{span}\left[D(\mathrm{i}\omega_1)^{-1}B(\mathrm{i}\omega_1) \quad \ldots \quad D(\mathrm{i}\omega_r)^{-1}B(\mathrm{i}\omega_r)\right],$$
$$\mathcal{W} = \operatorname{span}\left[D(\mathrm{i}\omega_1)^{-\mathsf{H}}C(\mathrm{i}\omega_1)^{\mathsf{H}} \quad \ldots \quad D(\mathrm{i}\omega_r)^{-\mathsf{H}}C(\mathrm{i}\omega_r)^{\mathsf{H}}\right]$$

and let V, W be matrices with orthonormal columns spanning $\mathcal{V}$ and $\mathcal{W}$, respectively.

Choice of the projection matrices

Choose a set of interpolation points $\mathbb{W} = \{\mathrm{i}\omega_1, \ldots, \mathrm{i}\omega_r\} \subset \mathrm{i}\mathbb{R}$. Under the assumption that $m = p$ choose

$$\mathcal{V} = \operatorname{span}\left[D(\mathrm{i}\omega_1)^{-1}B(\mathrm{i}\omega_1) \quad \ldots \quad D(\mathrm{i}\omega_r)^{-1}B(\mathrm{i}\omega_r)\right],$$
$$\mathcal{W} = \operatorname{span}\left[D(\mathrm{i}\omega_1)^{-\mathsf{H}}C(\mathrm{i}\omega_1)^{\mathsf{H}} \quad \ldots \quad D(\mathrm{i}\omega_r)^{-\mathsf{H}}C(\mathrm{i}\omega_r)^{\mathsf{H}}\right]$$

and let V, W be matrices with orthonormal columns spanning $\mathcal{V}$ and $\mathcal{W}$, respectively.
With the reduced function

$$\widetilde{H}(s) = C(s)V\left(W^{\mathsf{H}}D(s)V\right)^{-1}W^{\mathsf{H}}B(s),$$

we obtain the Hermite interpolation conditions

$$H(\mathrm{i}\omega_j) = \widetilde{H}(\mathrm{i}\omega_j), \quad H'(\mathrm{i}\omega_j) = \widetilde{H}'(\mathrm{i}\omega_j), \quad j = 1, \ldots, r,$$

and under some weak condition

$$\sigma_{\max}(H(\mathrm{i}\omega_j)) = \sigma_{\max}\left(\widetilde{H}(\mathrm{i}\omega_j)\right), \quad \sigma'_{\max}(H(\mathrm{i}\omega_j)) = \sigma'_{\max}\left(\widetilde{H}(\mathrm{i}\omega_j)\right), \quad j = 1, \ldots, r.$$

Algorithm

Main steps of the algorithm:

a) create an initial reduced-order model with transfer function $\widetilde{H}$ and projection matrices V, W,

b) compute $\sup_{\omega \in \mathbb{R}} \left\| \widetilde{H}(\mathrm{i}\omega) \right\|_2$ with global optimizer ω_*,

c) update V and W with subspaces at $\mathrm{i}\omega_*$ (assume $m = p$):
   - set $\widetilde{V} := D(\mathrm{i}\omega_*)^{-1} B(\mathrm{i}\omega_*)$, $\quad \widetilde{W} := \left(C(\mathrm{i}\omega_*) D(\mathrm{i}\omega_*)^{-1} \right)^{\mathsf{H}}$,
   - set $V := \begin{bmatrix} V & \widetilde{V} \end{bmatrix}$ and $W := \begin{bmatrix} W & \widetilde{W} \end{bmatrix}$ and update $\widetilde{H}$,

d) if not converged, repeat b), c), and d).

Algorithm

Main steps of the algorithm:

a) create an initial reduced-order model with transfer function $\widetilde{H}$ and projection matrices V, W,

b) compute $\sup_{\omega\in\mathbb{R}} \left\|\widetilde{H}(\mathrm{i}\omega)\right\|_2$ with global optimizer ω_*,

c) update V and W with subspaces at $\mathrm{i}\omega_*$ (assume $m = p$):
   - set $\widetilde{V} := D(\mathrm{i}\omega_*)^{-1}B(\mathrm{i}\omega_*)$, $\quad \widetilde{W} := \left(C(\mathrm{i}\omega_*)D(\mathrm{i}\omega_*)^{-1}\right)^{\mathsf{H}}$,
   - set $V := \begin{bmatrix} V & \widetilde{V} \end{bmatrix}$ and $W := \begin{bmatrix} W & \widetilde{W} \end{bmatrix}$ and update $\widetilde{H}$,

d) if not converged, repeat b), c), and d).

Remarks:

- In case of convergence, one can prove a local superlinear rate of convergence using the Hermite interpolation properties.
- In Step 2, $\widetilde{H}$ has a low-dimensional representation, so use well-established methods for computing its $\mathcal{L}_\infty$-norm:
  - Boyd-Balakrishnan algorithm for linear systems
    [Boyd, Balakrishnan '90], [Benner, Sima, V. '12]
  - eigenvalue optimization methods for the general problem (`eigopt`)
    [Mengi, Yildirim, Kılıç '14]

Problems with `eigopt`

For irrational transfer functions we use `eigopt`, which is a method for global optimization of eigenvalues of analytic Hermitian functions. It takes quadratic support functions to build an upper bound of the eigenvalue function. This method is very reliable, **but**:

- needs additional parameters that the user must supply, for example
  - an interval $[\omega_{\mathrm{lb}}, \omega_{\mathrm{ub}}]$ in which the $\mathcal{L}_\infty$-norm is attained,
  - a global lower bound $\gamma < \min_{\omega \in [\omega_{\mathrm{lb}}, \omega_{\mathrm{ub}}]}(-\partial^2 \sigma_{\max}(H(\mathrm{i}\omega))/\partial\omega^2)$,
- the lower γ and the larger $[\omega_{\mathrm{lb}}, \omega_{\mathrm{ub}}]$, the more quadratic support functions have to be computed which deteriorates performance.

Problems with `eigopt`

For irrational transfer functions we use `eigopt`, which is a method for global optimization of eigenvalues of analytic Hermitian functions. It takes quadratic support functions to build an upper bound of the eigenvalue function. This method is very reliable, **but**:

- needs additional parameters that the user must supply, for example
  - an interval $[\omega_{\mathrm{lb}}, \omega_{\mathrm{ub}}]$ in which the $\mathcal{L}_\infty$-norm is attained,
  - a global lower bound $\gamma < \min_{\omega \in [\omega_{\mathrm{lb}}, \omega_{\mathrm{ub}}]}(-\partial^2 \sigma_{\max}(H(\mathrm{i}\omega))/\partial\omega^2)$,
- the lower γ and the larger $[\omega_{\mathrm{lb}}, \omega_{\mathrm{ub}}]$, the more quadratic support functions have to be computed which deteriorates performance.

Solution: Construct a sequence of rational reduced functions $\rightsquigarrow$ allows to use the Boyd-Balakrishnan algorithm and needs no previous knowledge.

Rational Approximation using the Loewner Framework

With interpolation points $\mathbb{W} := \{\mathrm{i}\omega_1, \ldots, \mathrm{i}\omega_r\} \subset \mathrm{i}\mathbb{R}$, construct the matrices

$$\left[\mathbf{E}\right]_{ij} := \begin{cases} \frac{1}{\mathrm{i}\omega_i - \mathrm{i}\omega_j}(H(\mathrm{i}\omega_i) - H(\mathrm{i}\omega_j)), & \text{if } i \neq j, \\ H'(\mathrm{i}\omega_i), & \text{else,} \end{cases}$$

$$\left[\mathbf{A}\right]_{ij} := \begin{cases} \frac{1}{\omega_i - \omega_j}(\omega_i H(\mathrm{i}\omega_i) - \omega_j H(\mathrm{i}\omega_j)), & \text{if } i \neq j, \\ H(\mathrm{i}\omega_i) + \mathrm{i}\omega_i H'(\mathrm{i}\omega_i), & \text{else,} \end{cases}$$

$$\mathbf{B} := \begin{bmatrix} H(\mathrm{i}\omega_1) \\ \vdots \\ H(\mathrm{i}\omega_r) \end{bmatrix}, \quad \mathbf{C} := \begin{bmatrix} H(\mathrm{i}\omega_1) & \ldots & H(\mathrm{i}\omega_r) \end{bmatrix}.$$

If $s\mathbf{E} - \mathbf{A}$ is a regular pencil, then our rational approximation is

$$\widetilde{H}(s) := -\mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}.$$

In particular, we still have the Hermite interpolation properties

$$\sigma_{\max}(H(\mathrm{i}\omega_j)) = \sigma_{\max}\left(\widetilde{H}(\mathrm{i}\omega_j)\right), \quad \sigma'_{\max}(H(\mathrm{i}\omega_j)) = \sigma'_{\max}\left(\widetilde{H}(\mathrm{i}\omega_j)\right), \quad j = 1, \ldots, r.$$

Regularization

Often the pencil $s\mathbf{E} - \mathbf{A}$ is singular. If for all $\mu \in \{\mathrm{i}\omega_1, \ldots, \mathrm{i}\omega_r\}$ we have

$$\operatorname{rank}(\mu\mathbf{E} - \mathbf{A}) = \operatorname{rank}\begin{bmatrix}\mathbf{E} & \mathbf{A}\end{bmatrix} = \operatorname{rank}\begin{bmatrix}\mathbf{E} \\ \mathbf{A}\end{bmatrix},$$

then we perform economic SVDs

$$\begin{bmatrix}\mathbf{E} & \mathbf{A}\end{bmatrix} = Y\Sigma_{\mathrm{l}}\widetilde{X}^{\mathsf{H}}, \quad \begin{bmatrix}\mathbf{E} \\ \mathbf{A}\end{bmatrix} = \widetilde{Y}\Sigma_{\mathrm{r}}X^{\mathsf{H}}$$

and obtain a regularized system

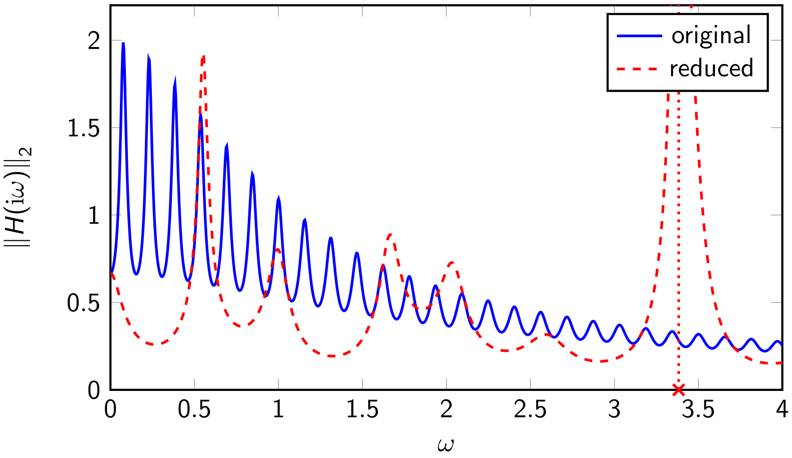
$$\widetilde{E} = -Y^{\mathsf{H}}\mathbf{E}X, \quad \widetilde{A} = -Y^{\mathsf{H}}\mathbf{A}X, \quad \widetilde{B} = Y^{\mathsf{H}}\mathbf{B}, \quad \widetilde{C} = \mathbf{C}X,$$

which gives the reduced function

$$\widetilde{H}(s) := \widetilde{C}\left(s\widetilde{E} - \widetilde{A}\right)^{-1}\widetilde{B}.$$

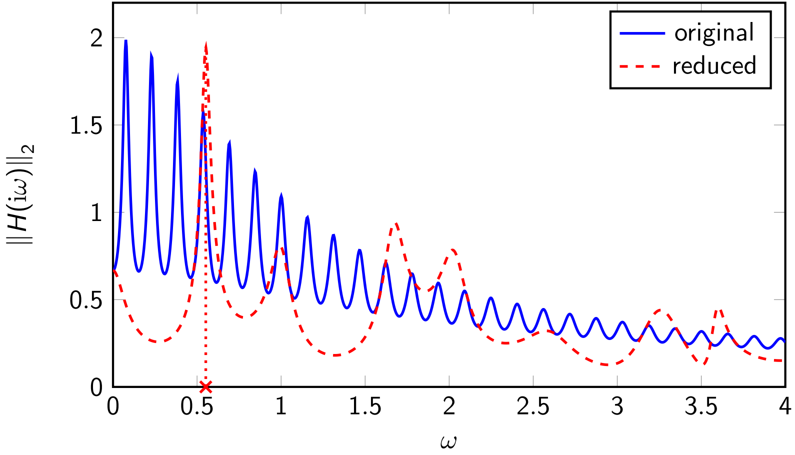
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



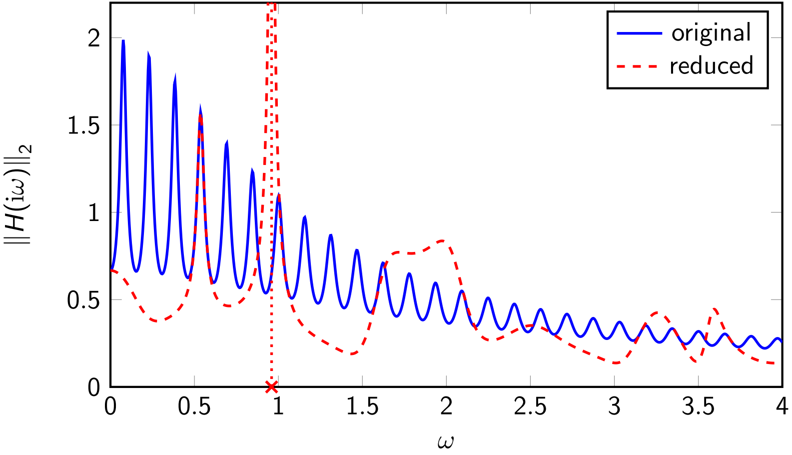
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



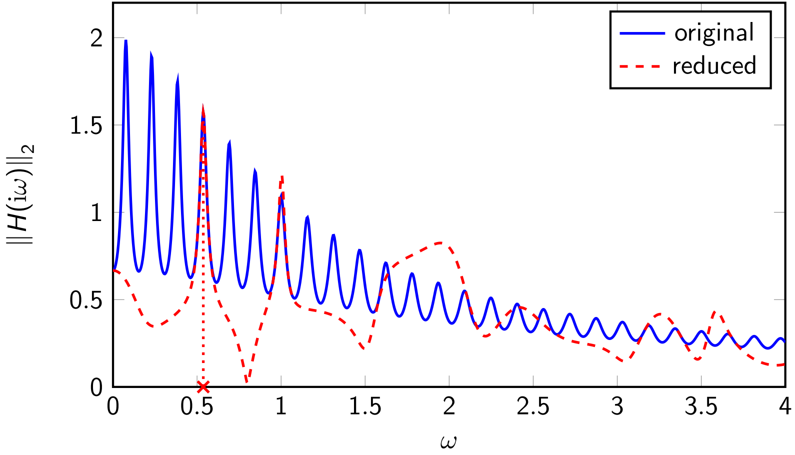
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



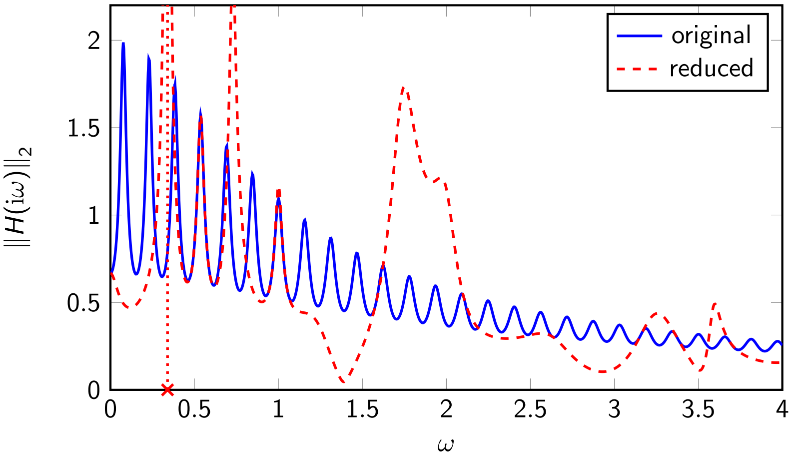
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



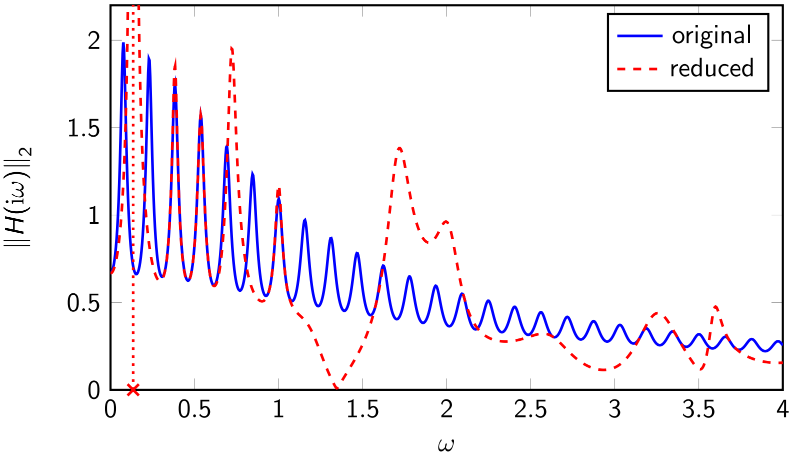
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



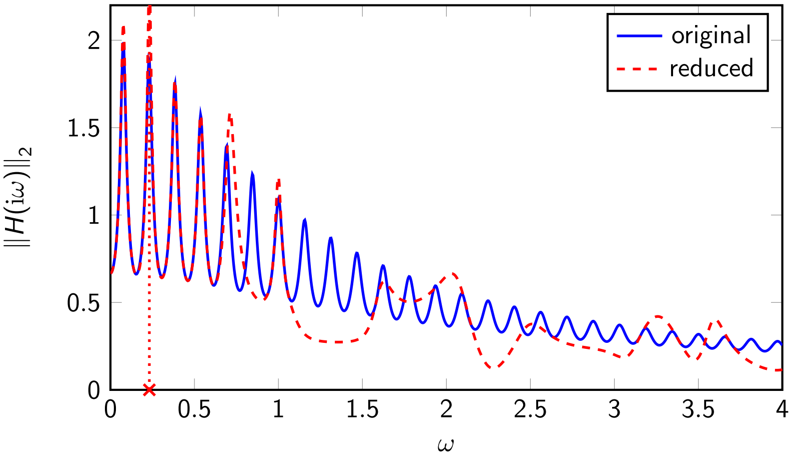
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



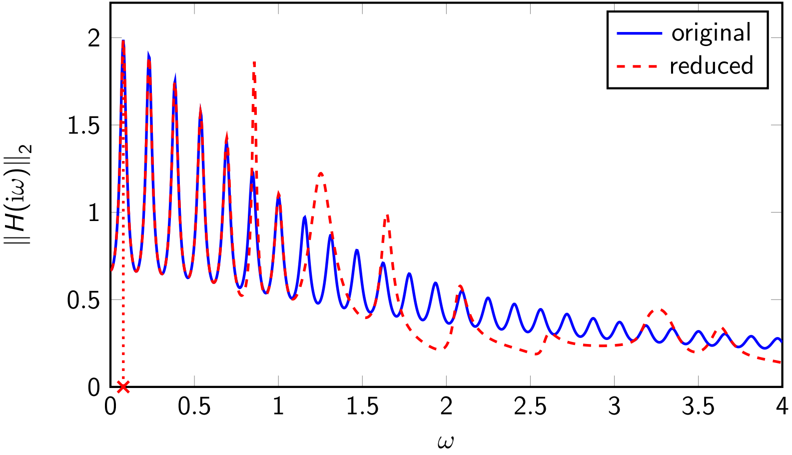
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



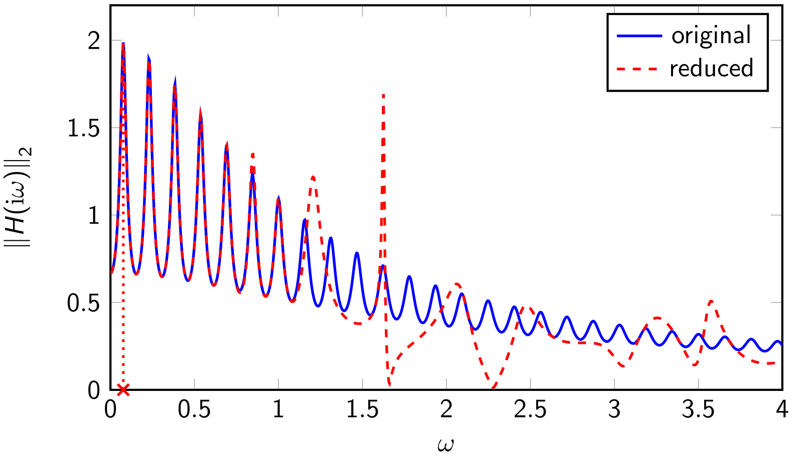
Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



Example: `hinfn_ex4`

Delay model with $n = 1$, $m = p = 1$.



Timings: $0.157s$ (`eigopt`: $0.128s$)

Comparison with eigopt

#	example	n	m	p	n_{iter}	time in seconds: new algor.	time in seconds: eigopt	ratio
1	delay_model	500	1	1	4	0.028	0.821	29.51
2	butterfly	64	2	1	3	0.034	0.017	0.51
3	dirac	80	2	1	2	0.033	0.225	6.75
4	gen_hyper2	15	2	1	3	0.025	0.112	4.52
5	gen_tantipal2	16	2	1	4	0.024	0.032	1.33
6	gen_tpal2	16	2	1	3	0.026	0.036	1.40
7	hadeler	8	2	1	2	0.011	0.023	2.05
8	loaded_string	20	2	1	2	0.011	0.023	2.03
9	sleeper	10	2	1	2	0.018	0.026	1.43
10	spring	5	2	1	2	0.012	0.012	0.95
11	spring_dashpot	10	2	1	3	0.020	0.149	7.36
12	wiresaw2	10	2	1	3	0.025	0.048	1.97
13	hinfn_ex1	3	1	3	2	0.009	0.011	1.26

Comparison with [Aliyev, Benner, Mengi, Schwerdtner, V. '17]

We use a scalable delay model from [Beattie, Gugercin '08].

	time in seconds		
n	new algor.	[ABMSV' 17]	ratio
100	0.023	1.289	56.208
300	0.025	1.280	51.812
1000	0.029	1.293	43.867
3000	0.046	1.283	28.128
10000	0.093	1.305	13.997
30000	0.262	1.433	5.470
100000	0.872	1.866	2.138
300000	2.509	3.096	1.234
1000000	8.606	7.511	0.873

Comparison with [Michiels, Gummussoy '10]

#	example	time in seconds new algor.	hinfn	ratio
1	hinfn_ex1	0.024	0.063	2.645
2	hinfn_ex2	0.015	0.121	8.284
3	hinfn_ex3	0.011	0.029	2.665
4	hinfn_ex4	0.147	0.045	0.309
5	hinfn_ex5	0.039	0.306	7.895
6	hinfn_ex6	0.048	0.229	4.721
7	hinfn_ex7	0.143	0.230	1.610
8	hinfn_ex8	0.462	1.975	4.271
9	hinfn_ex9	0.098	40.219	408.498
10	hinfn_ex10	0.064	233.989	3657.742
11	hinfn_ex11	0.051	0.128	2.483
12	hinfn_ex12	0.047	–	–

Concluding Remarks

Concluding Remarks:

- Rational interpolation is an efficient way to compute the $\mathcal{L}_\infty$-norm of possibly irrational functions.
- Only needs little a priori knowledge of the function under consideration.
- No realizations are needed, only function evaluations!!
- But:
  - The algorithm does not necessarily converge (counter-example: periodic functions).
  - Convergence is only local (convergence to a non-global maximizer of $\sigma_{\max}(H(\mathrm{i}\cdot))$ happens only rarely), need an efficient check for global convergence.

References

Paper with all references:

- P. Schwerdtner and M. Voigt. Computation of the $\mathcal{L}_\infty$-norm using rational interpolation, IFAC-PapersOnLine, 2018.

Code:

- `linorm_subsp v1.2`, available from `http://www.math.tu-berlin.de/index.php?id=186267&L=1`.

References

Paper with all references:

- P. Schwerdtner and M. Voigt. Computation of the $\mathcal{L}_\infty$-norm using rational interpolation, IFAC-PapersOnLine, 2018.

Code:

- `linorm_subsp v1.2`, available from `http://www.math.tu-berlin.de/index.php?id=186267&L=1`.

Thank you for your Attention!