Robust rational approximations of nonlinear eigenvalue problems

Güttel, Stefan and Negri Porzio,
Gian Maria and Tisseur, Francoise

2020

MIMS EPrint: **2020.24**

Manchester Institute for Mathematical Sciences
School of Mathematics

The University of Manchester

Reports available from: http://eprints.maths.manchester.ac.uk/
And by contacting: The MIMS Secretary
School of Mathematics
The University of Manchester
Manchester, M13 9PL, UK

ISSN 1749-9097

ROBUST RATIONAL APPROXIMATIONS OF NONLINEAR EIGENVALUE PROBLEMS*

STEFAN GÜTTEL†, GIAN MARIA NEGRI PORZIO†, AND FRANÇOISE TISSEUR†

Abstract. We develop algorithms that construct robust (i.e., reliable for a given tolerance, and scaling independent) rational approximants of matrix-valued functions on a given subset of the complex plane. We consider matrix-valued functions provided in both split form (i.e., as a sum of scalar functions times constant coefficient matrices) and in black box form. We develop a new error analysis and use it for the construction of stopping criteria, one for each form. Our criterion for split forms adds weights chosen relative to the importance of each scalar function, leading to the weighted AAA algorithm, a variant of the set-valued AAA algorithm that can guarantee to return a rational approximant with a user-chosen accuracy. We propose two-phase approaches for black box matrix-valued functions that construct a surrogate AAA approximation in phase one and refine it in phase two, leading to the surrogate AAA algorithm with exact search and the surrogate AAA algorithm with cyclic Leja–Bagby refinement. The stopping criterion for black box matrix-valued functions is updated at each step of phase two to include information from the previous step. When convergence occurs, our two-phase approaches return rational approximants with a user-chosen accuracy. We select problems from the NLEVP collection that represent a variety of matrix-valued functions of different sizes and properties and use them to benchmark our algorithms.

Key words. rational approximation, linear rational interpolation, nonlinear eigenvalue problem

AMS subject classifications. 65F15, 65F35, 15A18, 41A20, 47J10

1. Introduction.

Consider a matrix-valued function $F\colon \Omega \to \mathbb{C}^{n\times n}$ defined on a nonempty open subset Ω of the complex plane and its rational approximant

$$R^{(m)}(z) = b_0(z)R_0 + b_1(z)R_1 + \cdots + b_m(z)R_m \tag{1.1}$$

on the compact *target set* $\Sigma_T \subset \Omega$. The $R_j \in \mathbb{C}^{n\times n}$ in (1.1) are constant-coefficient matrices and the b_j are polynomials of degree at most m or rational functions of type (m,m), that is, quotients of polynomials of degree at most m. Such approximants play an important role in model order reduction [1, 2] and the solution of the *nonlinear eigenvalue problem* (NEP) for F on Σ_T [13, sect. 6]. The latter consists of finding all the pairs $(\lambda, v) \in \Sigma_T \times \mathbb{C}^n \setminus \{0\}$ such that

$$F(\lambda)v = 0. \tag{1.2}$$

A scalar λ satisfying (1.2) is an eigenvalue of F and $v \neq 0$ is the corresponding eigenvector. If $R^{(m)}$ approximates F well on Σ_T then, rather than solving the NEP (1.2), we can solve the *rational eigenvalue problem*

$$R^{(m)}(\lambda)v = 0, \tag{1.3}$$

which is still nonlinear in λ but simpler to solve numerically as long as $R^{(m)}$ is expressed in an appropriate basis. Indeed, in this case linearization techniques exist that allow the $n \times n$ rational eigenproblem (1.3) to be rewritten as a linear eigenproblem

$$L(\lambda)x = 0$$

*Version of November 14, 2020.

†Department of Mathematics, The University of Manchester, Manchester, M13 9PL, UK (stefan.guettel@manchester.ac.uk, gianmaria.negriporzio@manchester.ac.uk, francoise.tisseur@manchester.ac.uk).

of larger dimension (typically $nm \times nm$) but which can be solved by a variety of algorithms [13]. All that is left to do is to recover approximate eigenpairs (λ, v) of $F(\lambda)$ from approximate eigenpairs (λ, x) of $L(\lambda)$. The quality of such NEP eigensolvers depends on a good understanding and careful implementation of all these steps. In particular, it is important that $R^{(m)}$ is a good uniform approximation to F on the target set Σ_T so that the eigenpairs of the two eigenproblems are related.

In this paper we focus on the numerical construction of rational approximants that are robust in the sense that they are reliable for a given tolerance and are scaling-independent. To be more specific, for the numerical construction of such approximants we use a discrete compact set $\Sigma \subseteq \Sigma_T$ in place of the target set Σ_T (Σ is usually a fine mesh of the compact superset Σ_T or its boundary contour). For a given tolerance ε, we describe four algorithms that, when they succeed, return rational approximants $R^{(m)}$ satisfying

$$\|F - R^{(m)}\|_\Sigma \leq \varepsilon \|F\|_\Sigma, \tag{1.4}$$

where

$$\|F\|_\Sigma = \max_{z \in \Sigma} \|F(z)\|_2. \tag{1.5}$$

When F and $R^{(m)}$ are continuous on Σ_T, we argue in section 2 that (1.4) implies that

$$\|F - R^{(m)}\|_{\Sigma_T} \leq \varepsilon c_\Sigma \|F\|_{\Sigma_T} \tag{1.6}$$

holds for some constant $c_\Sigma > 1$. As a consequence of (1.6), we show that any eigenpair (λ, v) of $R^{(m)}$ with $\lambda \in \Sigma_T$ is an exact eigenpair of the perturbed matrix-valued function $F + \Delta F$ with $\|\Delta F\|_{\Sigma_T} \leq c_\Sigma \varepsilon \|F\|_{\Sigma_T}$. We consider both the case where F is provided in the split form[1]

$$F(z) = \sum_{j=1}^{s} f_j(z) A_j, \tag{1.7}$$

with $A_j \in \mathbb{C}^{n \times n}$ and the $f_j(z)$ being functions defined on Ω, and the case where F is provided as a black box that only returns evaluations $F(z_0)$ for $z_0 \in \Omega$.

There is a growing body of work on the approximation or interpolation of matrix-valued functions. Taylor approximation has been applied successfully, in particular to NEPs arising in delay differential equations [16]. Chebyshev interpolants are more appropriate for NEPs with eigenvalues lying on a smooth curve in the complex plane [7]. For the nonlinear eigensolver NLEIGS, Güttel et al. used a rational Leja–Bagby sampling approach to construct a rational approximant $R^{(m)}$, where the b_j in (1.1) are rational Newton basis functions [12]. Hochman [15] and Lietaert et al. [17] showed how to extend the adaptive Antoulas–Anderson (AAA) algorithm [18] from scalar functions to a set of multiple scalar functions, with the rational approximants for each function expressed in barycentric form. This led to the fastAAA algorithm for rational fitting [15] and the set-valued AAA algorithm for NEPs [17], which require F to be provided in the split form (1.7). Saad et al. [19] used the Cauchy integral formula to approximate the scalar functions f_j in (1.7) by a rational function and applied this approach successfully to solve acoustic nonlinear problems [8]. Güttel and Elsworth [9] showed how to construct a matrix-valued rational approximant with the

[1] Observe that we can always express $F(z)$ in that form with at most $s = n^2$ matrix coefficients.

AAA algorithm when F is provided as a black box function, using a scalar surrogate function f for F, and named this approach surrogate AAA. This surrogate approach in combination with the NLEIGS method is implemented in the NEP module of the SLEPc library [4]. A drawback of the surrogate AAA approach is that there might be a significant gap between the accuracy of the scalar surrogate approximant for f and the accuracy of the matrix-valued approximant for the original function F. In the SLEPc implementation this is addressed by polynomially expanding the surrogate rational approximant with poles placed at infinity.

In all the above approaches the computed families of rational approximants share the same barycentric scalar support points and weights, and therefore all functions share the same set of poles. A recent preprint by Gosea and Güttel describes a block-AAA algorithm which is based on a barycentric formula with matrix-valued weights [11]. This approach can deliver accurate rational approximants of lower order compared to algorithms using scalar weights, but the involved linear algebra computations currently make it only suitable for problems of very small dimensions. None of the above contributions provide an error analysis of the expected approximation accuracy returned by their algorithms.

The main contributions of our work are as follows.

- In section 2 we provide an error analysis for the eigenpairs of F computed from a rational approximant $R^{(m)} \approx F$. This analysis gives insights into the sampling accuracy required to solve NEPs with a guaranteed backward error, an essential requirement for a robust eigensolver.
- In section 3 we apply the new error analysis for the development of a stopping criterion for rational approximation procedures that exploit the split form (1.7). Our criterion includes weights that are chosen relative to the "importance" of each function f_j. This leads to a new variant of the set-valued AAA algorithm, called weighted AAA, that guarantees to return a rational approximant with a user chosen accuracy and possibly lower degree than set-valued AAA.
- In section 4 we consider the problem of approximating F given only as a black box. Here we introduce several two-phase methods that combine the ability of surrogate AAA to identify good pole parameters with the robustness of the Leja–Bagby approach in NLEIGS. Here our contributions over previous works are three-fold.
  - Instead of refining the surrogate AAA approximant by poles at infinity as in [4], we argue that poles should instead be ordered in the Leja–Bagby manner and then be repeated cyclically.
  - Instead of discarding previous evaluations of the surrogate $f(z)$ at sampling points z_j, which involves expensive evaluations of $F(z_j)$, we show how to reuse these computations for the second phase of our algorithm. While the resulting rational approximant $R^{(m)}$ combines data in barycentric and Newton form, the rational eigenproblem (1.3) is easily linearized as shown in Appendix A. Hence this rational approximant in mixed bases can be used as part of an NEP eigensolver.
  - We propose a stopping criterion that is less strict than the divided difference based criterion in [12], typically leading to approximants of slightly lower degree without sacrificing accuracy.
- Finally, section 5 contains a comprehensive comparison of the discussed algorithms on a large range of problems from the NLEVP collection (version 4.1) representing a variety of matrix-valued functions with different sizes and

properties. We believe this is the first comparison of so much detail available in the literature. MATLAB codes needed to reproduce these experiments are available from https://github.com/Gmnp/nep2rat. Our set of benchmark problems may be useful for future developments of eigensolvers for nonlinear eigenvalue problems.

2. Error analysis of approximate nonlinear eigenpairs. Let $\Sigma \subseteq \Sigma_T \subset \Omega \subseteq \mathbb{C}$ be the sets introduced in section 1 and let $F\colon \Omega \to \mathbb{C}^{n\times n}$. One of the objectives of this paper is to numerically construct an approximant $R^{(m)}$ of F on the discrete set Σ such that (1.4) holds. This does not in general guarantee that the relative error of $R^{(m)}$ on the target set Σ_T will be bounded above by the given tolerance ε. But assuming that F and $R^{(m)}$ are (uniformly) continuous on Σ_T, we can argue using [5, Lem. 2, p. 86] that (1.4) implies

$$\|F - R^{(m)}\|_{\Sigma_T} \le \omega_F(\delta) + \omega_{R^{(m)}}(\delta) + \varepsilon\|F\|_{\Sigma_T},$$

where

$$\omega_F(\delta) = \sup_{|x-y|\le\delta} \|F(x) - F(y)\|_2$$

is the modulus of continuity of F (and likewise for $R^{(m)}$) and

$$\delta = \max_{z\in\Sigma_T} \min_{\sigma\in\Sigma} |\sigma - z|$$

is the 'density' of Σ in the target set Σ_T. If for a given ε we choose δ such that

$$\omega_F(\delta) + \omega_{R^{(m)}}(\delta) \le \varepsilon\|F\|_{\Sigma_T}$$

then (1.6) holds for $c_\Sigma = 2$. Hence the relative error on the target set Σ_T can be controlled by the relative error on the finite set Σ provided that δ is sufficiently small, i.e., Σ is sufficiently dense in Σ_T.

The above argument also applies if both F and $R^{(m)}$ are holomorphic in Ω and Σ is only a sufficiently fine discretization of the boundary $\partial\Sigma_T$. In this case (1.4) implies (1.6) with Σ_T replaced by $\partial\Sigma_T$, i.e.,

$$\|F - R^{(m)}\|_{\partial\Sigma_T} \le \varepsilon\, c_{\partial\Sigma_T}\|F\|_{\partial\Sigma_T}$$

with some constant $c_{\partial\Sigma_T}$. By the maximum norm principle (see, e.g., [6, Thm. 2]), $\|F\|_{\partial\Sigma_T} = \|F\|_{\Sigma_T}$ and therefore

$$\|F - R^{(m)}\|_{\Sigma_T} \le \varepsilon c_{\partial\Sigma_T}\|F\|_{\Sigma_T}.$$

Now suppose we have an approximant $R^{(m)}$ to F satisfying (1.6). Can we use any computed eigenpair of $R^{(m)}$ as an approximate eigenpair of F? Let us look at backward errors. A natural definition for the backward error $\eta_F(\widehat{\lambda},\widehat{v})$ of an approximate eigenpair $(\widehat{\lambda},\widehat{v})$ of F with $\widehat{\lambda}$ in the target set Σ_T is given by

$$\eta_F(\widehat{\lambda},\widehat{v}) := \min\{\varepsilon :\ (F(\widehat{\lambda}) + \Delta F(\widehat{\lambda}))\widehat{v} = 0,\ \|\Delta F\|_{\Sigma_T} \le \varepsilon\|F\|_{\Sigma_T}\}. \tag{2.1}$$

Starting from $(F(\widehat{\lambda}) + \Delta F(\widehat{\lambda}))\widehat{v} = 0$, we find that

$$\|F(\widehat{\lambda})\widehat{v}\|_2 = \|\Delta F(\widehat{\lambda})\widehat{v}\|_2 \le \|\Delta F(\widehat{\lambda})\|_2\|\widehat{v}\|_2 \le \|\Delta F\|_{\Sigma_T}\|\widehat{v}\|_2 \le \varepsilon\|F\|_{\Sigma_T}\|\widehat{v}\|_2,$$

so that

$$\eta_F(\widehat{\lambda},\widehat{v}) \geq \frac{\|F(\widehat{\lambda})\widehat{v}\|_2}{\|F\|_{\Sigma_T}\|\widehat{v}\|_2}.$$

This lower bound on $\eta_F(\widehat{\lambda},\widehat{v})$ is attained by the perturbation

$$\Delta F(z) = -F(\widehat{\lambda})\frac{\widehat{v}\widehat{v}^*}{\widehat{v}^*\widehat{v}}.$$

Indeed, for this perturbation we have that $(F(\widehat{\lambda})+\Delta F(\widehat{\lambda}))\widehat{v} = 0$, so the first constraint in (2.1) is satisfied. Moreover,

$$\|\Delta F\|_{\Sigma_T} = \sup_{z\in\Sigma_T}\|\Delta F(z)\|_2 = \frac{\|F(\widehat{\lambda})\widehat{v}\|_2}{\|\widehat{v}\|_2} = \frac{\|F(\widehat{\lambda})\widehat{v}\|_2}{\|F\|_{\Sigma_T}\|\widehat{v}\|_2}\|F\|_{\Sigma_T}.$$

Hence, for the backward error $\eta_F(\widehat{\lambda},\widehat{v})$ in (2.1) we have the explicit expression

$$\eta_F(\widehat{\lambda},\widehat{v}) = \frac{\|F(\widehat{\lambda})\widehat{v}\|_2}{\|F\|_{\Sigma_T}\|\widehat{v}\|_2}. \tag{2.2}$$

Now if we can construct an approximant $R^{(m)}$ of F such that (1.6) holds and if $(\widehat{\lambda},\widehat{v})$ is a computed eigenpair of $R^{(m)}$ with backward error $\eta_{R^{(m)}}(\widehat{\lambda},\widehat{v})$, i.e., $\big(R^{(m)}(\widehat{\lambda}) + \Delta R^{(m)}(\widehat{\lambda})\big)\widehat{v} = 0$ with $\|\Delta R^{(m)}\|_{\Sigma_T} = \eta_{R^{(m)}}(\widehat{\lambda},\widehat{v})\|R^{(m)}\|_{\Sigma_T}$, then

$$\begin{aligned}\eta_F(\widehat{\lambda},\widehat{v}) = \frac{\|F(\widehat{\lambda})\widehat{v}\|_2}{\|F\|_{\Sigma_T}\|\widehat{v}\|_2} &= \frac{\|F(\widehat{\lambda})\widehat{v} - R^{(m)}(\widehat{\lambda})\widehat{v} - \Delta R^{(m)}(\widehat{\lambda})\widehat{v}\|_2}{\|F\|_{\Sigma_T}\|\widehat{v}\|_2}\\ &\leq \frac{\|F-R^{(m)}\|_{\Sigma_T}}{\|F\|_{\Sigma_T}} + \frac{\|\Delta R^{(m)}\|_{\Sigma_T}}{\|F\|_{\Sigma_T}}\\ &\leq c_\Sigma\varepsilon + \frac{\|R^{(m)}\|_{\Sigma_T}}{\|F\|_{\Sigma_T}}\,\eta_{R^{(m)}}(\widehat{\lambda},\widehat{v}).\end{aligned} \tag{2.3}$$

This implies that if $(\widehat{\lambda},\widehat{v})$ with $\widehat{\lambda}\in\Sigma_T$ is a computed eigenpair of $R^{(m)}$ with backward error $\eta_{R^{(m)}}(\widehat{\lambda},\widehat{v}) \leq \varepsilon$ then as long as $\|R^{(m)}\|_{\Sigma_T}/\|F\|_{\Sigma_T} \approx 1$ and c_Σ is not too large, we can expect $(\widehat{\lambda},\widehat{v})$ to be an approximate eigenpair of F with a backward error $\eta_F(\widehat{\lambda},\widehat{v}) \lesssim \varepsilon$. We illustrate this in Example 2.1 below.

The explicit formula in (2.2) is not practical due to the presence of the Σ_T-norm in the denominator. Hence, when computing backward errors, we use instead the upper bound

$$\widehat{\eta}_F(\widehat{\lambda},\widehat{v}) = \frac{\|F(\widehat{\lambda})\widehat{v}\|_2}{\|F\|_{\Sigma}\|v\|_2}.$$

This affects the bounds in (2.3) by a factor $\|F\|_{\Sigma_T}/\|F\|_{\Sigma}$.

EXAMPLE 2.1. *Let us consider the* 2×2 *matrix-valued function*

$$F(z) = \begin{bmatrix} \mathrm{e}^{\mathrm{i}z^2} & 1\\ 1 & 1\end{bmatrix}, \tag{2.4}$$

which has eigenvalues

$$\lambda_{1,2} = 0, \quad \lambda_3 = \sqrt{2\pi}, \quad \lambda_4 = \mathrm{i}\sqrt{2\pi}, \quad \lambda_5 = -\mathrm{i}\sqrt{2\pi}, \quad \lambda_6 = -\sqrt{2\pi}$$

Table 2.1: Backward errors for approximate eigenpairs of F in (2.4) computed as eigenpairs of $R^{(m)}$.

m	$\frac{\|F-R^{(m)}\|_\Sigma}{\|F\|_\Sigma}$	$\widehat{\eta}_F(\widehat{\lambda}_1,\widehat{v}_1)$	$\widehat{\eta}_F(\widehat{\lambda}_2,\widehat{v}_2)$	$\widehat{\eta}_F(\widehat{\lambda}_3,\widehat{v}_3)$	$\widehat{\eta}_F(\widehat{\lambda}_4,\widehat{v}_4)$	$\widehat{\eta}_F(\widehat{\lambda}_5,\widehat{v}_5)$	$\widehat{\eta}_F(\widehat{\lambda}_6,\widehat{v}_6)$
14	3.3e-5	2.5e-8	2.6e-8	7.0e-7	1.6e-6	1.7e-6	1.2e-6
18	1.8e-7	1.4e-10	1.4e-10	2.0e-9	2.9e-9	4.8e-9	2.0e-9
22	3.6e-10	9.1e-14	9.1e-14	2.7e-12	7.8e-13	3.7e-12	3.2e-12
28	1.5e-14	1.4e-15	1.4e-15	3.4e-15	7.4e-16	1.5e-15	4.9e-15

in the target set Σ_T consisting of the disc of center 0 and radius 3. For the finite set $\Sigma \subset \Sigma_T$, we generate 1000 random points inside Σ_T and another 200 points uniformly distributed on the boundary of Σ_T. We then construct four rational approximants of F on Σ having different relative errors and degrees m—see columns 1 and 2 in Table 2.1. The computed eigenpairs $(\widehat{\lambda}_j,\widehat{v}_j)$ of $R^{(m)}$ with eigenvalues $\widehat{\lambda}_j \in \Sigma_T$ are returned as approximate eigenpairs of F. The corresponding backward errors are displayed in Table 2.1. We checked that for all these eigenpairs, $\widehat{\eta}_{R^{(m)}}(\widehat{\lambda}_j,\widehat{v}_j) \le 7\times 10^{-15}$. As predicted by (2.3), the backward errors $\widehat{\eta}_F(\widehat{\lambda}_j,\widehat{v}_j)$ are smaller than $\varepsilon = \|F-R^{(m)}\|_\Sigma/\|F\|_\Sigma$. We also see that the smaller $\|F-R^{(m)}\|_\Sigma/\|F\|_\Sigma$ is, the smaller the backward errors $\widehat{\eta}_F(\widehat{\lambda}_j,\widehat{v}_j)$ are.

Since we know the exact eigenvalues, we can compute the absolute errors for the double and defective eigenvalue at 0 and the relative errors for the other nonzero eigenvalues. These are provided in Table 2.2. Assuming that they satisfy the rule of thumb that

$$\textit{forward error} \lesssim \textit{condition number} \times \textit{backward error},$$

we anticipate that the nonzero simple eigenvalues $\lambda_3,\dots,\lambda_6$ have a condition number of order 10^3. A normwise condition number for a nonzero simple eigenvalue $\lambda \in \Sigma_T$ of F with eigenvector v that is consistent with the backward error definition in (2.1) is given by

$$\kappa_F(\lambda) = \limsup_{\varepsilon\to 0}\left\{\frac{|\Delta\lambda|}{\varepsilon|\lambda|} : (F(\lambda+\Delta\lambda)+\Delta F(\lambda+\Delta\lambda))(v+\Delta v)=0, \|\Delta F\|_{\Sigma_T} \le \varepsilon\|F\|_{\Sigma_T}\right\}.$$

Following the proof of [13, Thm. 2.20] we find that

$$\kappa_F(\lambda) = \frac{\|F\|_{\Sigma_T}\|w\|_2\|v\|_2}{|\lambda||w^*F'(\lambda)v|},$$

where w is a left eigenvector of F with corresponding eigenvalue λ. Now for F in (2.4), we find that $\|F\|_{\Sigma_T} \approx \mathrm{e}^9 \approx 8\times 10^3$. Also, all the left and right eigenvectors of F are nonzero multiples of $\left[\begin{smallmatrix}1\\-1\end{smallmatrix}\right]$. An easy calculation shows that

$$\|w\|_2\|v\|_2/(|\lambda_j||w^*F'(\lambda_j)v|) = 1/(2\pi)$$

so that $\kappa_F(\lambda_j) \approx 1.3\times 10^3$, $j=3,\dots,6$ as anticipated from the numerical experiments.

3. Matrix-valued functions in split form. A natural approach to approximate a matrix-valued function F provided in split form (1.7) with a rational approximant $R^{(m)}$ consists of approximating the scalar functions f_j with scalar rational

Table 2.2: Absolute and relative errors for approximate eigenvalues of F in (2.4) computed as eigenvalues of $R^{(m)}$.

m	$\frac{\|F-R^{(m)}\|_\Sigma}{\|F\|_\Sigma}$	$\lvert\lambda_1-\widehat{\lambda}_1\rvert$	$\lvert\lambda_2-\widehat{\lambda}_2\rvert$	$\frac{\lvert\lambda_3-\widehat{\lambda}_3\rvert}{\lvert\lambda_3\rvert}$	$\frac{\lvert\lambda_4-\widehat{\lambda}_4\rvert}{\lvert\lambda_4\rvert}$	$\frac{\lvert\lambda_5-\widehat{\lambda}_5\rvert}{\lvert\lambda_5\rvert}$	$\frac{\lvert\lambda_6-\widehat{\lambda}_6\rvert}{\lvert\lambda_6\rvert}$
14	3.3e-5	1.7e-2	1.7e-2	6.4e-4	1.5e-3	1.5e-3	1.1e-3
18	1.8e-7	1.2e-3	1.3e-3	1.8e-6	2.7e-6	4.4e-6	1.8e-6
22	3.6e-10	3.2e-5	3.2e-5	2.5e-9	7.1e-10	3.4e-9	2.9e-9
28	1.5e-14	4.0e-6	4.0e-6	3.1e-12	6.8e-13	1.3e-12	4.5e-12

functions $r_j^{(m)}$, $j = 1, \ldots, s$ and let

$$R^{(m)}(z) := \sum_{j=1}^{s} r_j^{(m)}(z) A_j =: \sum_{i=0}^{m} b_i(z) R_i, \tag{3.1}$$

where the $b_i(z)$ are as in (1.1). The rational approximants $r_j^{(m)}$ can be constructed in several ways. We concentrate on the following.

Cauchy approximation. When the f_j in (1.7) are holomorphic on $\widehat{\Sigma}_T \supset \Sigma_T$, Saad et al. [19] use Cauchy's integral formula to rewrite the f_j as

$$f_j(z) = \frac{1}{2\pi \mathrm{i}} \int_{\partial\widehat{\Sigma}_T} \frac{f_j(u)}{u - z} du, \quad z \in \widehat{\Sigma}_T \setminus \partial\widehat{\Sigma}_T, \quad j = 1, \ldots, s.$$

If the boundary $\partial\widehat{\Sigma}_T$ of $\widehat{\Sigma}_T$ is piecewise regular with parametrization $\gamma : [0, 2\pi] \to \partial\widehat{\Sigma}_T$ then the substitution $u = \gamma(t)$ leads to

$$f_j(z) = \frac{1}{2\pi \mathrm{i}} \int_0^{2\pi} \frac{f_j(\gamma(t))\gamma'(t)}{\gamma(t) - z} dt.$$

A quadrature rule with $m + 1$ nodes σ_i and weights ω_i is then employed to approximate the f_j with the rational functions

$$r_j^{(m)}(z) = \sum_{i=i}^{m} \frac{w_i f_j(\sigma_i)}{\sigma_i - z}, \quad j = 1, \ldots, s. \tag{3.2}$$

AAA approximation. The adaptive Antoulas–Anderson (AAA) algorithm aims at interpolating a scalar function $f \colon \Omega \to \mathbb{C}$ by a rational function $r^{(m)}$ expressed in barycentric form [18]

$$r^{(m)}(z) = \sum_{i=0}^{m} \frac{f(\sigma_i) w_i}{z - \sigma_i} \Big/ \sum_{i=0}^{m} \frac{w_i}{z - \sigma_i}. \tag{3.3}$$

The core of the procedure is a greedy selection of support points σ_i, one at a time, from a given discrete set Σ of M points. To be more specific, at step m, the next support point σ_m is chosen such that

$$\max_{\sigma \in \Sigma^{(m-1)}} |f(\sigma) - r^{(m-1)}(\sigma)| = |f(\sigma_m) - r^{(m-1)}(\sigma_m)|,$$

where $\Sigma^{(m-1)} = \Sigma \backslash \{\sigma_0, \ldots, \sigma_{m-1}\}$. The vector of weights $w = [w_0, w_2, \ldots, w_m]^T$ is obtained by solving a least squares problem $\min_{\|w\|_2=1} \|A^{(m)} w\|_2$ with an

$(M-m-1)\times(m+1)$ Loewner matrix $A^{(m)}$ (see [18] for detail). The construction stops when

$$\|f-r^{(m)}\|_\Sigma = \|f-r^{(m)}\|_{\Sigma^{(m)}} \le \varepsilon\|f\|_\Sigma, \tag{3.4}$$

where the norm on Σ defined in (1.5) reduces to $\|g\|_\Sigma = \max_{z\in\Sigma}|g(z)|$ for a scalar function g.
For matrix-valued functions in split form (1.7), Hochman [15] and Lietaert et al. [17] propose to approximate all the scalar functions f_j with the AAA procedure using the same support points and weights for all the functions f_j. This leads to the rational approximant

$$r_j^{(m)}(z) = \sum_{i=0}^{m} \frac{f_j(\sigma_i)w_i}{z-\sigma_i} \Big/ \sum_{i=0}^{m} \frac{w_i}{z-\sigma_i}, \quad j=1,\dots,s. \tag{3.5}$$

How well the rational $r_j^{(m)}$ in (3.2) or (3.5) approximates f_j on Σ has a direct consequence on how well $R^{(m)}$ in (3.1) approximates F on Σ. Indeed,

$$\|F-R^{(m)}\|_\Sigma = \max_{z\in\Sigma}\Big\|\sum_{j=1}^{s}(f_j(z)-r_j^{(m)}(z))A_j\Big\|_2 \le \sum_{j=1}^{s}\|f_j-r_j^{(m)}\|_\Sigma\|A_j\|_2. \tag{3.6}$$

So we propose to use the stopping criterion

$$\sum_{j=1}^{s}\|f_j-r_j^{(m)}\|_\Sigma\|A_j\|_F \le \varepsilon\beta, \tag{3.7}$$

where β is a lower bound on $\|F\|_\Sigma$ that we assume can be computed cheaply. Then under the assumption that (3.7) holds, we have

$$\|F-R^{(m)}\|_\Sigma \le \sum_{j=1}^{s}\|f_j-r_j^{(m)}\|_\Sigma\|A_j\|_2 \le \sum_{j=1}^{s}\|f_j-r_j^{(m)}\|_\Sigma\|A_j\|_F \le \varepsilon\beta \le \varepsilon\|F\|_\Sigma.$$

Hence if $r_j^{(m)}$ in (3.2) or (3.5) satisfies (3.7) then the resulting rational approximation $R^{(m)}$ in (3.1) is guaranteed to satisfy (1.4). A lower bound β on $\|F\|_\Sigma$ can be computed as follows: let $u\in\mathbb{C}^n$ be some normally distributed vector of unit length and let $u_j = A_j u$, $j=1,\dots,s$. Then we let

$$\beta := \max_{z\in\Sigma}\|\sum_{j=1}^{s} f_j(z)u_j\|_2 \le \|F\|_\Sigma. \tag{3.8}$$

The lower bound can be attained, but it can be several orders of magnitude smaller than $\|F\|_\Sigma$ with a bad choice for u. A poor lower bound will result in a few unnecessary extra steps in the construction of the approximant and hence a larger degree m than is needed. Note that the stopping criterion (3.7) is scaling independent. It returns the same approximation when applied to the split form $F(z)$ in (1.7) and to $F(z)=\sum_{j=1}^{s}g_j(z)B_j$ with $g_j(z)=\alpha_j f_j(z)$ and $A_j=\alpha_j B_j$, $\alpha_j\neq 0$. Also, the criterion (3.7) returns $\alpha R^{(m)}$ when applied to αF, where $R^{(m)}$ is the approximant to F with this criterion. We call the AAA approximant $R^{(m)}$ obtained with the stopping criterion (3.7), the *weighted AAA rational approximant*.

For their *set-valued AAA algorithm*, Lietaert et al. [17] use

$$\max_{1\le j\le s} \|f_j - r_j^{(m)}\|_\Sigma / \|f_j\|_\Sigma \le \varepsilon \tag{3.9}$$

as stopping criterion. The latter satisfies

$$\|F - R^{(m)}\|_\Sigma \le \sum_{j=1}^{s} \|f_j - r_j^{(m)}\|_\Sigma \|A_j\|_2 \le \Big(s \max_{1\le j\le s} \|f_j\|_\Sigma \|A_j\|_2\Big)\varepsilon. \tag{3.10}$$

Since $(s \max_{1\le j\le s} \|f_j\|_\Sigma \|A_j\|_2)/\|F\|_\Sigma \ge 1$ we cannot conclude that (1.4) holds. Our stopping criterion (3.7) takes into account the magnitude of the coefficient matrices while (3.9) does not. In particular, (3.7) assigns more importance to scalar functions f_j associated with coefficient matrices A_j of large norm. As a consequence, if the lower bound β on $\|F\|_\Sigma$ is sharp, we can expect the construction of $R^{(m)}$ to stop earlier with (3.7) than with (3.9). We will illustrate this behaviour in section 5.

The Cauchy approximant (3.2) for holomorphic functions is in principle easy to construct. However, it requires choosing a set $\widehat{\Sigma}_T \supset \Sigma_T$ such that the functions f_j are still holomorphic on $\widehat{\Sigma}_T$ and with interpolation points on the boundary $\partial\widehat{\Sigma}_T$ that are not too close to the boundary of Σ_T. This makes an automatic implementation of the Cauchy approximant difficult. The numerical experiments we conducted with the trapezium rule on a variety of problems usually returned Cauchy approximants of degree higher than that of the corresponding set-valued or weighted AAA approximants. On the other hand, for a holomorphic F on Σ_T, the set-valued or the weighted AAA procedure can return a rational $R^{(m)}$ with poles in Σ_T (although we did not observe this happening in the experiments we conducted).

EXAMPLE 3.1. *Let F be the 2×2 matrix-valued function in (2.4) with target set Σ_T the disc of centre 0 and radius $\rho = 3$. Rewrite F in split form as*

$$F(z) = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} + \mathrm{e}^{\mathrm{i}z^2} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.11}$$

and construct a Cauchy approximant $R^{(m)}$ using the trapezium rule and contours $\partial\widehat{\Sigma}_T$ that are circles of centre 0 and radius $\alpha\rho$, $\alpha \in \{1.05, 1.1, 1.25, 1.5\}$. For each value of α, we plot in Figure 3.1 the relative error $\|F - R^{(m)}\|_\Sigma / \|F\|_\Sigma$ as the degree, or equivalently, the number of quadrature nodes m increases. For the finite set Σ, we generate 400 random points inside Σ_T and add 100 uniformly distributed points on the boundary of Σ_T. While the convergence is slow for α close to 1, it improves for larger values of α but the limiting accuracy of the approximation increases as well. For comparison, we also plotted the convergence of the AAA approximant of F in (3.11).

The AAA approximation can stagnate when the tolerance ε is too small. We then see numerical *Froissart doublets* appear. These are poles with very small residues or pole-support point pairs that are so close together that they nearly cancel [10]. We use the "clean-up" procedure described in [18, Sec. 5] to remove them.

4. Two phase approximation of black box matrix-valued functions. Elsworth and Güttel [9] use the support points σ_i and weights w_i, $i = 1, \ldots, m$ from the AAA approximant $r^{(m)}$ to a surrogate scalar function

$$f(z) = u^* F(z) v, \tag{4.1}$$

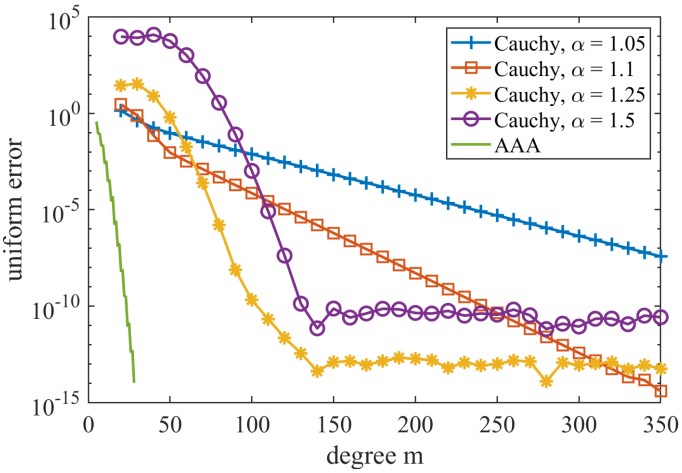

Fig. 3.1: Demonstration of four different choices for the contour $\partial\widehat{\Sigma}_T$ (a circle of centre 0 and radius $\alpha\rho$) for the Cauchy approximation of F in (2.4) on Σ_T, the disc of centre 0 and radius $\rho = 3$. Refer to Example 3.1 for a detailed discussion.

where u and v are normally distributed vectors of unit length. The rational matrix-valued approximant is then defined as

$$R^{(m)}(z) = \sum_{i=0}^{m} \frac{F(\sigma_i)w_i}{z-\sigma_i} \Bigg/ \sum_{i=0}^{m} \frac{w_i}{z-\sigma_i}. \tag{4.2}$$

They refer to this approach as *surrogate AAA*, the latter being applicable to matrix-valued functions provided in black box or split form. They argue that f has a region of analyticity similar to F and so the support points and the weights computed for the AAA approximant of f should be good choices for the function F as well. The procedure stops when (3.4) holds. Unfortunately, there is no guarantee that $R^{(m)} \approx F$ with good accuracy; see also section 5. However, when combined with a refinement phase, the surrogate approach can be a viable option. In particular, the AAA approximant to the surrogate function (4.1) also provides information about the location of poles that can then be fed to the NLEIGS Leja–Bagby sampling procedure [12, Section 5]. Indeed, the m poles of the AAA approximation $r^{(m)}$ of f in (4.1) or equivalently, the m poles of $R^{(m)}$ in (4.2) can be computed by following the procedure in [18], i.e., by first constructing the $(m+2)\times(m+2)$ pencil

$$\begin{bmatrix} 0 & w_0 & w_1 & \dots & w_m \\ 1 & \sigma_0 & & & \\ 1 & & \sigma_1 & & \\ \vdots & & & \ddots & \\ 1 & & & & \sigma_m \end{bmatrix} - z \begin{bmatrix} 0 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix} \tag{4.3}$$

then computing its eigenvalues, discarding the two eigenvalues at infinity, and returning the remaining m eigenvalues as poles $\{\lambda_1, \dots, \lambda_m\} =: \Xi$ of $R^{(m)}$.

Also, the first step of the AAA procedure applied to the surrogate function f in (4.1) allows the computation of a lower bound for $\|F\|_\Sigma$ at almost no extra cost.

Indeed, for surrogate AAA, $f(z) = u^* g(z)$ with $g(z) = F(z)v$ is evaluated for all $z \in \Sigma$ so that

$$\widetilde{\beta} = \max_{z\in\Sigma} \|g(z)\|_2 \le \|F\|_\Sigma \tag{4.4}$$

is easy to compute. Having access to such a lower bound is useful when constructing a stopping criterion that ensures that the rational approximant $R^{(m)}$ to F has a relative error bounded by ε for some given tolerance ε.

4.1. NLEIGS with poles from surrogate AAA. The new NEP module inside the SLEPc library for the solution of nonlinear eigenvalue problems [4] starts by constructing a surrogate AAA approximant to the scalar function f in (4.1) and then feeds its poles to the NLEIGS Leja–Bagby sampling procedure [12, Section 5]. This approach combines the convenience of the AAA algorithm, which only requires as inputs samples of the function to be approximated, with the robustness and full parameter control of the Leja–Bagby approach. By the latter we mean in particular that poles returned by AAA can be preprocessed before being used for the Leja–Bagby procedure. This might be necessary in case where AAA returns unwanted poles in the target set Σ_T. The *NLEIGS with poles from surrogate AAA algorithm* works as follows.

step 1. Run the AAA algorithm on the surrogate function (4.1) and discretized target set Σ to compute the $d+1$ support points σ_i and weights w_i, $i = 0, \dots, d$ defining the AAA rational approximant $r^{(d)}(z) = \sum_{i=0}^{d} \frac{f(\sigma_i)w_i}{z-\sigma_i} \Big/ \sum_{i=0}^{d} \frac{w_i}{z-\sigma_i}$.

step 2. Compute the d poles of $r^{(d)}$ (i.e., the eigenvalues of the pencil (4.3) with $m = d$ minus the two extra eigenvalues at infinity).

step 3. Apply *NLEIGS with Leja–Bagby sampling*, i.e., start with a random point $\sigma_0 \in \Sigma$ and let

$$R_0 = F(\sigma_0), \qquad b_0(z) = 1$$

be the first term in (1.1). The Leja–Bagby pairs (σ_k, ξ_k) are formed one at a time using

$$\sigma_k = \arg\max_{z\in\Sigma} |s_{k-1}(z)|, \quad \xi_k = \arg\min_{z\in\Xi} |s_{k-1}(z)|, \qquad k = 1, 2, \dots, \tag{4.5}$$

where

$$s_k(z) = \prod_{j=0}^{k} (z - \sigma_j) \Big/ \prod_{\substack{j=1 \\ \xi_j \neq \infty}}^{k} (z - \xi_j).$$

Then the coefficient matrix R_k and rational function b_k in (1.1) are computed as [12]

$$R_k = \frac{F(\sigma_k) - \sum_{i=0}^{k-1} b_i(\sigma_k) R_i}{b_k(\sigma_k)}, \qquad b_k(z) = \frac{z - \sigma_{k-1}}{\beta_k (1 - z/\xi_k)} b_{k-1}(z) \tag{4.6}$$

with scaling parameters β_k such that $\|b_k\|_{\partial\Sigma} = \max_{z\in\partial\Sigma} |b_k(z)| = 1$.

4.1.1. Stopping criterion for the Leja–Bagby procedure. Güttel et al. [12] stop the Leja–Bagby sampling procedure at step $k = m$ when

$$\|R_m\|_F \le \varepsilon \|R_0\|_F. \tag{4.7}$$

But if the Leja–Bagby procedure converges then

$$\|F - R^{(m)}\|_\Sigma = \max_{z\in\Sigma} \Big\| \sum_{k=0}^{\infty} b_k(z)R_k - \sum_{k=0}^{m} b_k(z)R_k \Big\|_2 \le \sum_{k=m+1}^{\infty} \max_{z\in\Sigma} \|b_k(z)\|_2 \|R_k\|_F.$$

If none of the poles ξ_k are inside Σ then the rational functions b_k in (4.6) are holomorphic on Σ and by construction, $\max_{z\in\Sigma} \|b_k(z)\|_2 = \|b_k\|_\Sigma = \|b_k\|_{\partial\Sigma} = 1$ so that

$$\|F - R^{(m)}\|_\Sigma \le \sum_{j=m+1}^{\infty} \|R_j\|_2.$$

Also, $\|F\|_\Sigma \ge n^{-1/2} \max_{0\le k\le m} \|F(\sigma_k)\|_F$ and since we assume the Leja–Bagby procedure converges then for m large enough we have $\|R_j\|_F < \|R_m\|_F$ for $j > m$ and $\sum_{j=m+1}^{\infty} \|R_j\|_2 \le \kappa \|R_m\|_F$ for some constant $\kappa > 1$ (we use $\kappa = 3$ in our implementation). Hence, instead of (4.7) we suggest to stop the Leja–Bagby procedure when

$$\|R_m\|_F \le \frac{\varepsilon}{\kappa} \max_{0\le k\le m} \|F(\sigma_k)\|_F, \tag{4.8}$$

which, once convergence has taken place, guarantees $\|F - R^{(m)}\|_\Sigma \le \varepsilon \|F\|_\Sigma$. Note that (4.8) is usually less strict than (4.7) since $R_0 = F(\sigma_0)$ so with (4.8) as stopping criterion, NLEIGS with Leja–Bagby sampling returns a rational approximant with the required accuracy but with a smaller degree m than when (4.7) is used.

4.1.2. On the choice of poles to feed to the Leja–Bagby sampling procedure. When the number of poles $\xi_1, \ldots, \xi_d$ in the approximant $R^{(d)}$ returned by the surrogate AAA procedure is smaller than the degree m needed to satisfy the NLEIGS stopping criterion, i.e., $d < m$, Campos and Roman [4] add extra poles at infinity. Using the denominator polynomial

$$q_d(z) = (z - \xi_1) \cdots (z - \xi_d)$$

and the nodal polynomial

$$s_d(z) = (z - \sigma_0)(z - \sigma_1) \cdots (z - \sigma_d)$$

of the interpolation nodes σ_j, we show that this approach is equivalent to computing a polynomial interpolant P of degree $m - d - 1$ to the error function

$$E(z) := q_d(z)(F(z) - R^{(d)}(z))/s_d(z) \tag{4.9}$$

and then setting

$$R^{(m)}(z) := R^{(d)}(z) + s_d(z)P(z)/q_d(z). \tag{4.10}$$

To this end we need to recall a basic fact from linearized rational interpolation: a rational function $r(z) = p(z)/q_d(z)$ of type (m, m) with fixed prescribed denominator q_d is uniquely determined by $m + 1$ interpolation conditions $r(\sigma_j) = f_j$ at distinct points σ_j. The only requirement is that $q_d(\sigma_j) \ne 0$ for all $j = 0, 1, \ldots, m$. The same result holds for a rational matrix-valued interpolant of the form $R(z) = P(z)/q_d(z)$, where now P is a matrix polynomial of degree m because every matrix entry is a scalar rational function with the interpolation property.

The interpolant obtained after d iterations of the surrogate approach is of the form $R^{(d)}(z) = Q(z)/q_d(z)$, with a matrix polynomial Q of degree d, and it interpolates F at the nodes $\sigma_0, \sigma_1, \ldots, \sigma_d$ by construction. After $m-d$ more steps of the Leja–Bagby procedure with poles $\xi_{d+1} = \cdots = \xi_m = \infty$ and interpolation nodes $\sigma_{d+1}, \ldots, \sigma_m$, we obtain a degree m rational interpolant $R^{(m)}(z) = \widetilde{Q}(z)/q_d(z)$ satisfying $m+1$ interpolation conditions at $\sigma_0, \sigma_1, \ldots, \sigma_m$. By uniqueness of the interpolant, this function must coincide with $R^{(m)}$ defined in (4.10), which has the same denominator q_d and satisfies the same $m+1$ interpolation conditions by construction.

The above discussion shows that choosing poles at infinity in the refinement phase, amounts to falling back to polynomial interpolation of the error function defined in (4.9). The asymptotic convergence of this process is governed by the region of analyticity of E in a neighbourhood of the target set Σ_T. Note that the roots σ_j of s_d do *not* introduce singularities in $E(z)$, as $F(\sigma_j) - R^{(d)}(\sigma_j) = 0$ due to the interpolation conditions ($j = 0, 1, \ldots, d$). Also, $q_d(z)R^{(d)}$ is a matrix polynomial. Hence, the asymptotic convergence of this polynomial interpolation process is entirely governed by the region of analyticity of the original function F.

In situations where rational interpolation is of advantage, e.g. if F has singularities nearby the target set Σ_T, falling back to polynomial interpolation might lead to significant inefficiencies in the sampling procedure. On the other hand, in the refinement phase we only have d pole parameters at our disposal. Our recommendation is therefore to repeat these d poles cyclically in Leja–Bagby order, i.e., once the number k of Leja–Bagby points generated by the sampling procedure exceeds d, the expression for the poles ξ_k in (4.5) is replaced with

$$\xi_k = \xi_{1+(k-1 \mod d)} \tag{4.11}$$

until we reach the final degree $k = m$ required to satisfy a stopping criterion. This is no longer a true Leja–Bagby procedure as d pole parameters are repeated cyclically, so we refer to it as *d-cyclic Leja–Bagby procedure*. The Leja–Bagby ordering of the poles ensures that the scalar basis functions $b_k(z)$ defined in (4.6) vary only mildly over the target set Σ, thus avoiding problems with numerical under- or overflow.

EXAMPLE 4.1. *In order to illustrate the above discussion, it suffices to consider a scalar 1×1 NEP $F(z) = f(z) = 0.2\sqrt{z} - 0.6\sin(2z)$, which can also be found in the introduction of [12]. As target set we use the interval $\Sigma_T = [10^{-2}, 4]$, discretized by 10^3 logarithmically spaced points (the discrete set Σ). Figure 4.1 shows four convergence curves each corresponding to one of the approaches discussed above.*

- AAA (solid blue): *This curve shows the uniform error $\max_{z\in\Sigma_T} |f(z) - r^{(m)}(z)|$ of the degree m interpolant of f obtained by AAA. The algorithm resolves f efficiently, with the degree 19 approximant achieving an error below 10^{-14}, but we note the spikes in the curve for some degrees and overall stagnation behaviour with noisy fluctuations around $\approx 10^{-13}$.*
- Leja–Bagby (dashed red with square markers): *This curve corresponds to the original NLEIGS approach in [12], computing Leja–Bagby points on Σ_T and the singularity set of f, which is $\Xi = (-\infty, 0]$. The observed geometric convergence behaviour is robust, at an asymptotic rate that is given in terms of the logarithmic capacity of the condenser $1/\mathrm{cap}(\Sigma_T, \Xi) \approx 0.569$, i.e.,*

$$\limsup_{m\to\infty} \|f(z) - r^{(m)}(z)\|_{\Sigma_T} \leq \exp(-1/\mathrm{cap}(\Sigma_T, \Xi));$$

see e.g. [12, 20] for more details. The downside of the Leja–Bagby approach

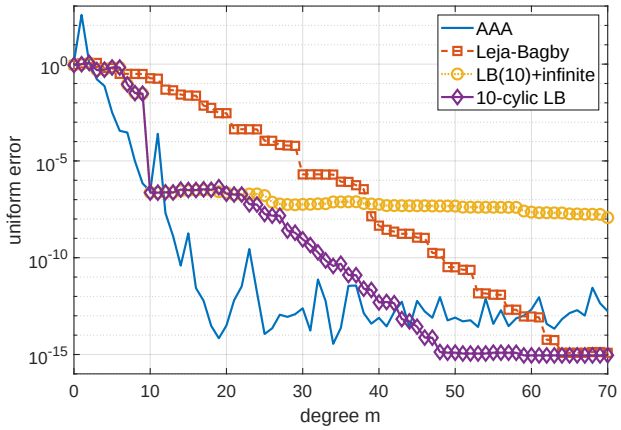

Fig. 4.1: Demonstration of four different choices for the rational approximation of a scalar function, including our proposed d-cyclic Leja–Bagby procedure ($d = 10$). Refer to Example 4.1 for a detailed discussion.

is that the singularity set Ξ needs to be specified by the user, which might be difficult in particular for NEPs given as a black box.

- LB(10)+infinite (dotted yellow with circles): *This is similar to what is used in [4], namely an initial phase of AAA (in this example $d = 10$ iterations), followed by a Leja–Bagby procedure using the d poles obtained from the AAA approximant, and then using poles at infinite for the remaining process. We observe a sudden drop in error for degree* 10, *approximately to the level of the AAA approximant of the same degree, but then a very slow convergence in the refinement phase for degrees $m > 10$. As explained above, this approach amounts to using polynomial interpolation in the refinement phase. For this particular problem an asymptotic geometric convergence rate of only $(\sqrt{k} - 1)/(\sqrt{\kappa} + 1) \approx 0.905$ is expected, where $\kappa = \max(\Sigma_T)/\min(\Sigma_T)$. We refer to the discussion in [12].*
- 10-cyclic LB (solid purple with diamonds): *This is our recommendation of running an initial phase of AAA to get $d = 10$ poles, and then repeat them cyclically in Leja–Bagby order. This approach combines the convenience of not having to specify the singularity set with the robust convergence of the Leja–Bagby approach.*

4.2. Surrogate AAA with exact/relaxed search. The Leja–Bagby construction of the support points in (4.5) does not use information from F unlike the surrogate AAA construction through the surrogate function $f(z) = u^*F(z)v$. So instead of constructing $R^{(m)}$ entirely with NLEIGS as explained above we propose to refine the rational approximation

$$R^{(d)}(z) = \sum_{i=0}^{d} \frac{F(\sigma_i)w_i}{z - \sigma_i} \Bigg/ \sum_{i=0}^{d} \frac{w_i}{z - \sigma_i} \tag{4.12}$$

obtained after d steps of surrogate AAA. To this end we modify the surrogate AAA steps as follows. The support points are determined from

$$\sigma_k = \underset{z\in\Sigma^{(k-1)}}{\arg\max} \|F(z) - R^{(k-1)}(z)\|_F, \quad k > d, \tag{4.13}$$

where $\Sigma^{(k-1)} = \Sigma \setminus \{\sigma_0, \ldots, \sigma_{k-1}\}$. The weights w_i are computed as for the surrogate AAA approximation. The procedure terminates at step $k = m$ when

$$\max_{z\in\Sigma^{(m)}} \|F(z) - R^{(m)}(z)\|_F \le \varepsilon\widetilde{\beta} \tag{4.14}$$

holds, where $\widetilde{\beta}$ is the lower bound for $\|F\|_\Sigma$ in (4.4). The rational approximant $R^{(m)}$ then satisfies (1.4). We named this approach *surrogate AAA with exact search on* Σ.

Computing σ_k in (4.13) is expensive so we suggest two ways to reduce the complexity.

1. When $F(z)$ is holomorphic in Ω, we can remove all the sampling points in the interior part of Σ and only work on the boundary $\partial\Sigma = \Sigma \cap \partial\Sigma_T$, which we call *surrogate AAA with exact search on* $\partial\Sigma$.
2. In addition, if the points of $\Sigma^{(k)}$ are randomly shuffled, instead of choosing σ_k as in (4.13), we take the first support point $\sigma_k \in \Sigma^{(k)}$ such that

$$\|F(\sigma_k) - R^{(k-1)}(\sigma_k)\|_F > \varepsilon\widetilde{\beta}.$$

We refer to this approach as *surrogate AAA with relaxed search.*

Our numerical experiments in section 5 show that this approach works well when the tolerance ε in (1.4) is not too small. For small tolerance, the fact that the weights are computing using the surrogate function (4.1) prevents the procedure to reach the required accuracy.

4.3. Surrogate AAA with cyclic Leja–Bagby refinement. Instead of refining the surrogate AAA approximation $R^{(d)}$ in (4.12) using surrogate AAA with exact or relaxed search, we propose refining $R^{(d)}$ with a d-cyclic Leja–Bagby procedure. To be more specific, the *surrogate AAA with cyclic Leja–Bagby refinement algorithm* works as follows:

step 1. Run the AAA algorithm on the surrogate function (4.1) and discretized target set Σ to compute the $d+1$ support points σ_i and weights w_i, $i = 0, \ldots, d$ defining the type $[d, d]$ rational approximant $R^{(d)}$ in (4.12) The degree d is determined by the built-in stopping criterion of the AAA implementation `aaa` in [18] and provided in (3.4).

step 2. Compute the d poles as in **step 1** of the NLEIGS with poles from surrogate AAA procedure and reorder them using the Leja–Bagby ordering in (4.5) to give $\xi_1, \ldots, \xi_d$.

step 3. Apply the d-cyclic Leja–Bagby procedure, i.e., for each new pair of node and pole $(\sigma_{d+i}, \xi_{d+i})$, where σ_{d+i} is computed as in (4.5) and ξ_{d+i} as in (4.11) construct

$$R_{d+i} = \frac{F(\sigma_{d+i}) - R^{(d+i-1)}(\sigma_{d+i})}{b_{d+i}(\sigma_{d+i})}, \quad b_{d+i}(z) = \prod_{j=1}^{d+i} \frac{z - \sigma_{j-1}}{\beta_j(1 - z/\xi_j)}, \tag{4.15}$$

with β_j chosen such that $\|b_{d+i}\|_{\partial\Sigma} = \max_{z\in\partial\Sigma}|b_{d+i}(z)| = 1$. Terminate the d-cyclic Leja–Bagby procedure at step $k = m$ with m such that (4.8) holds.

This algorithm yields a rational approximation of the form

$$R^{(m)}(z) = \sum_{i=0}^{d} \frac{F(\sigma_i)w_i}{z-\sigma_i} \Bigg/ \sum_{i=0}^{d} \frac{w_i}{z-\sigma_i} + b_{d+1}(z)R_{d+1} + \cdots + b_m(z)R_m \tag{4.16}$$

that satisfies (1.4). We show in Appendix A how to rewrite the rational eigenvalue problem $R^{(m)}(\lambda)v = 0$ as a linear eigenproblem when $R^{(m)}$ is provided in the form (4.16).

5. Numerical experiments. We test the robustness of the algorithms described in sections 3–4, i.e.,

(i) for matrix-valued functions provided in split form:
- the *set-valued AAA* algorithm as in Lietaert et al. [17][2]
- the *weighted AAA* algorithm, which is the set-valued AAA algorithm with stopping criterion (3.7) with β as in (3.8);

(ii) for matrix-valued functions provided as black box:
- the *surrogate AAA* algorithm proposed by Elsworth and Güttel [9];
- the *surrogate AAA algorithm with exact search on* $\Sigma \cap \partial\Sigma_T$ as described in section 4.2;
- *NLEIGS with poles from surrogate AAA* as described in section 4.1 with the new stopping criterion (3.7) (the d-cyclic Leja–Bagby procedure is employed for the poles as explained in section 4.1.2);
- the *surrogate AAA with cyclic Leja–Bagby refinement* algorithm as described in section 4.3.

We omit the Cauchy approximation (3.1)–(3.2) since we do not know of an automatic way to choose an "optimal" contour for the Cauchy integral formula.

To benchmark these algorithms, we use the test problems from the NLEVP collection[3, 14] listed in Table 5.1 (24 problems). These problems are selected to represent a variety of matrix-valued functions with different sizes and properties. All our computations are done in MATLAB R2020a.

Experiment 1. For our first set of experiments, we discretize the target sets Σ_T (either discs or half a discs) as follows. We generate 300 random points inside Σ_T plus another set of 100 uniformly distributed points on the contour of Σ_T. This gives a total of 400 points for the finite set Σ. We set the maximum number of steps to 60.

For a given tolerance ε and each problem listed in Table 5.1, we test if an algorithm fails to construct an approximant $R^{(m)}$ with accuracy $\|F - R^{(m)}\|_\Sigma / \|F\|_\Sigma$ below a given tolerance ε or if it does not converge within the maximum number of steps. We also compare the degrees of the approximants. The results are reported in Table 5.2 for $\varepsilon = 10^{-7}$, Tables 5.3 and 5.4 for $\varepsilon = 10^{-10}$, and Table 5.5 for $\varepsilon = 10^{-13}$. Results for the rational problems `loaded_string`, `railtrack2_rep`, and `railtrack_rep` are only reported in Table 5.2 since for these problems and any tolerance $\varepsilon \le 10^{-5}$, all the algorithms return (rightly) a degree 2 rational approximant with a relative error of about 10^{-15}. The tables show that for our benchmark of test problems and chosen tolerances $\varepsilon \in \{10^{-7}, 10^{-10}, 10^{-13}\}$:

(a) The set-valued and weighted AAA algorithms always return an approximant $R^{(m)}$ with relative error below the required accuracy. For problems that are holomorphic on the target sets, surrogate AAA with cyclic Leja–Bagby

[2] We use an implementation provided by the authors of [9] with a few adjustments for the solution of the least squares problems and cleanup of the Froissart doublets.

Table 5.1: List of benchmark examples from the NLEVP collection [3], their type and size, the target set Σ_T (disc or half disc), and the number of eigenvalues in Σ_T. For the `canyon_particle` problem, γ =-9e-2+1e-6i. The `fiber` and `sandwich_beam` problems are holomorphic on their respective target set if we remove the negative real numbers. Similarly, `schrodinger_abc` is holomorphic on $\Sigma_T \setminus [-15, -10)$.

Name	type	size	center	radius	half disc	#evs	holomorphic
`bent_beam`	nonlinear	6	60	30	yes	2	yes
`buckling_plate`	nonlinear	3	11	9	no	12	no
`canyon_particle`	square root	55	γ	0.1	yes	15	yes
`clamped_beam_1d`	exponential	100	0	10	no	101	yes
`distributed_delay1`	nonlinear	3	0	2	no	2	yes
`fiber`	nonlinear	2400	0	0.002	yes	1	no
`gun`	square root	9956	62500	50000	yes	21	yes
`hadeler`	exponential	200	-30	11.5	no	14	yes
`loaded_string`	rational	100	362	358	no	9	yes
`nep1`	nonlinear	2	0	3	no	6	yes
`nep2`	nonlinear	3	0	2	no	4	yes
`nep3`	nonlinear	10	5i	2	no	14	yes
`neuron_dde`	exponential	2	0	15	no	11	yes
`pdde_symmetric`	exponential	81	0	2	no	59	yes
`photonic_crystal`	nonlinear	288	11	9	no	28	yes
`pillbox_small`	square root	20	0.08	0.05	yes	1	yes
`railtrack2_rep`	rational	1410	3	2	no	53	yes
`railtrack_rep`	rational	1005	-3	2	no	2	yes
`sandwich_beam`	nonlinear	168	0	2	no	≈ 168	no
`schrodinger_abc`	nonlinear	10	-10	5	no	6	no
`square_root`	square root	20	10+50i	50	no	3	yes
`time_delay`	exponential	3	0	15	no	8	yes
`time_delay2`	exponential	2	0	15	no	11	yes
`time_delay3`	exponential	10	2	3	no	38	yes

refinement and NLEIGS with poles from surrogate AAA also return an approximant $R^{(m)}$ with relative error below the required accuracy.

(b) The set-valued and weighted AAA algorithms typically return the approximants $R^{(m)}$ of lowest degrees. The degrees of the set-valued AAA and weighted AAA approximants are more or less the same: they are usually either equal or they differ by one. There are exceptions though such as with the `sandwich_beam` and `time_delay3` problems, for which weighted AAA returns a lower degree approximant. These two problems have the particularity that, when viewed in split form, the norms of their coefficient matrices have large variations. The latter is exploited by the weighted AAA algorithm but ignored by the set-valued AAA algorithm.

(c) The surrogate AAA approach often fails to return a rational approximant with relative accuracy below ε. There is no surprise here since there is no guarantee of any accuracy with the stopping criterion used by this algorithm.

(d) As expected by our analysis, surrogate AAA with exact search either returns a rational approximant with relative error below the tolerance or fails to converge. There is an exception though for the `fiber` problem and tolerance $\varepsilon = 10^{-7}$ (see Table 5.2), where the constructed rational approximant $R^{(m)}$ is such that $\|F - R^{(m)}\|_\Sigma / \|F\|_\Sigma = 1.6 \times 10^{-7} > \varepsilon$, and hence marked as a failed in the table. The reason why the relative error is slightly above the

tolerance is that the exact search is done on $\partial\Sigma = \Sigma \cap \partial\Sigma_T$ and since F is not holomorphic on Σ_T, $\|\cdot\|_\Sigma \geq \|\cdot\|_{\partial\Sigma}$ so the stopping criterion (4.14) does not guarantee that (1.4) holds. But the surrogate AAA with exact search on Σ in place of $\partial\Sigma$ returns a rational approximant $R^{(m)}$ for the `fiber` problem and tolerance $\varepsilon = 10^{-7}$ with relative error 4e-8, i.e., below the tolerance.
Also, Tables 5.2, 5.4 and 5.5 show that for a few problems, the algorithm reaches the maximum number of steps, i.e., 60 (indicated by red stars) while returning approximants of degree less than 60. This is due to the removal of the Froissard doublets (see end of Section 3).

(e) NLEIGS with poles from surrogate AAA repeated in a cyclic way has a behaviour similar to that of surrogate AAA with cyclic Leja–Bagby refinement: they typically return rational approximants of the same degree. The `sandwich_beam` problem is an exception, though. First, it is not holomorphic on the target set so the theoretical results do not hold and unexpected behaviour can occur. For an accuracy tolerance of 1e-7, Table 5.2 shows that the approximant returned by surrogate AAA is a constant which is constructed from a single sample point σ_0. This constant matrix approximant is refined with cyclic Leja–Bagby using σ_0 and a single pole at infinity. This leads to a cubic approximant once the refinement step ends. In contrast, NLEIGS works with an initial random point σ_0' and a single pole at infinity and the procedure stops exactly after it reaches a degree equal to $60 \gg 3$. Table 5.3 for ε=1e-10 shows the opposite behaviour: NLEIGS returns an approximant of lower degree than surrogate AAA with cyclic Leja–Bagby refinement. The reason lies in the regime of (non-)convergence of the algorithms, which is really slow and "noisy": therefore slightly different initial conditions cause the algorithms to return approximants whose degrees vary a lot. Note that for ε=1e-13, both NLEIGS and surrogate AAA with cyclic Leja–Bagby refinement fail to return an approximant with the required accuracy and within the maximum number of steps for two problems that are not holomorphic on the target set.

Experiment 2. We now visualize where the algorithms place the interpolation nodes σ_i and the poles ξ_i in and around the target sets Σ_T for a small subset of the problems in Table 5.1. To avoid clutter, Figure 5.1 only displays the nodes and poles for the weighted AAA and surrogate AAA with cyclic Leja–Bagby refinement. For the latter, we distinguish the nodes chosen by the surrogate AAA phase from the nodes chosen by the refinement phase. We leave out the poles that are too far from Σ_T. The discretization of Σ_T consists of 300 points randomly generated inside Σ_T plus another 50 points uniformly distributed on the contour $\partial\Sigma_T$. The maximum number of steps is set to 60 and the tolerance $\varepsilon = 10^{-10}$.

The results are as anticipated by the theory. For the holomorphic problems, the nodes lie on the contour $\partial\Sigma_T$, while the poles form a pattern outside Σ_T. For instance, they are aligned towards the branch points for the `gun` problem, which contain two square roots. For the `nep1` problem, which is a scalar function camouflaged as a matrix-valued one, both algorithms use the same interpolation nodes and poles. Interestingly, one of the nodes lies inside Σ_T despite the problem being holomorphic. Finally, the poles and the nodes of `buckling_plate` do not follow the same pattern, because this problem is not holomorphic in the chosen region.

6. Conclusion. We have developed an error analysis for the eigenpairs of a matrix-valued function F computed from a rational approximant $R^{(m)} \approx F$ on a

Table 5.2: Degree of $R^{(m)}$ for ε = 1e-7 and 24 problems. The lowest degrees are highlighted in bold/blue including any within one of the lowest and excluding those corresponding to failed required accuracy that are provided within square brackets. A '★' indicates that the algorithm reached the maximum number of steps.

Problem	set-valued AAA	weighted AAA	surrogate AAA	surrogate+ exact search	NLEIGS AAA poles	surrogate+ LB refine
bent_beam	**7**	**7**	[4]	**8**	11	10
buckling_plate	**23**	**23**	[21]	26	42	42
canyon_particle	**13**	**12**	[7]	17	20	20
clamped_beam_1d	**11**	**11**	[10]	16	23	24
distributed_delay1	**6**	**6**	[6]	9	11	10
fiber	13	**10**	[6]	[16]	22	21
gun	**9**	**9**	[5]	[60]★	15	15
hadeler	**2**	4	**2**	6	15	5
loaded_string	**2**	**2**	**2**	**2**	**2**	**2**
nep1	**21**	**20**	**20**	**20**	**20**	**20**
nep2	**13**	**13**	[9]	[13]★	20	19
nep3	**8**	**8**	[7]	10	16	16
neuron_dde	**13**	**13**	[12]	**13**	**14**	**14**
pdde_symmetric	**8**	**8**	[7]	11	16	15
photonic_crystal	**5**	**5**	**4**	**5**	9	9
pillbox_small	**7**	**6**	[4]	9	11	11
railtrack2_rep	**2**	**2**	**2**	**2**	**2**	**2**
railtrack_rep	**2**	**2**	**2**	**2**	**2**	**2**
sandwich_beam	35	2	**0**	2	60	3
schrodinger_abc	**11**	**11**	[10]	16	20	20
square_root	**9**	**10**	**9**	**10**	21	14
time_delay	**12**	**13**	**12**	**12**	**13**	**12**
time_delay2	**12**	**13**	**12**	**12**	**13**	**12**
time_delay3	16	**13**	**12**	**12**	**13**	**12**
# of fails	0	0	13	3	0	0
# of lowest degree	21	22	11	11	8	8

discretization of the target set Σ_T. We showed in particular that if (λ, v) with $\lambda \in \Sigma_T$ is a computed eigenpair of $R^{(m)}$ with backward error η, then in order to guarantee a backward error of η for the eigenpair (λ, v) when considered as an approximate eigenpair of F, we need the approximant $R^{(m)}$ to have a relative accuracy $\varepsilon \leq \eta$.

We have shown that the weighted AAA algorithm, a variant of the set-valued AAA algorithm, is a robust procedure to approximate matrix-valued functions that are provided in split form: it is scaling independent and returns a rational approximant with a user chosen accuracy on the discretized target set Σ, as long as Σ contains enough points and the chosen accuracy is not too low. This is achieved through the use of a stopping criterion that includes weights relative to the importance of each scalar function in the split form.

For black box matrix-valued functions that are holomorphic on the target set, we have developed a two-phase algorithm that we called surrogate AAA algorithm with cyclic Leja–Bagby refinement. This algorithm, while more computationally expensive that the weighted AAA algorithm, only requires the ability to evaluate the matrix-valued function at the point in the target set. It is scaling independent and returns a rational approximant with a user chosen accuracy on the discretized target set Σ, as long as Σ contains enough points. Our algorithm combines the strength of surrogate

Table 5.3: Accuracy $\|F - R^{(m)}\|_\Sigma / \|F\|_\Sigma$ for $\varepsilon =$ 1e-10 and 21 problems. Any relative error above ε is highlighted in red and considered as a fail. A '★' indicates that the algorithm reached the maximum number of steps, i.e., 60.

Problem	set-valued AAA	weighted AAA	surrogate AAA	surrogate+ exact search	NLEIGS AAA poles	surrogate+ LB refine
bent_beam	4e-11	7e-12	2e-06	8e-07★	1e-12	2e-11
buckling_plate	3e-11	2e-11	1e-07	2e-06★	3e-11	5e-11
canyon_particle	4e-12	1e-11	2e-06	7e-08★	2e-11	1e-11
clamped_beam_1d	1e-11	1e-11	1e-06	4e-09★	3e-13	5e-13
distributed_delay1	1e-12	5e-13	3e-06	1e-08★	2e-12	2e-12
fiber	4e-13	3e-11	2e-06	1e-06★	1e-11	2e-11
gun	4e-13	3e-13	6e-07	2e-07★	1e-12	1e-12
hadeler	2e-11	5e-12	4e-10	2e-11	2e-12	4e-12
nep1	1e-11	9e-12	9e-12	9e-12	3e-11	9e-12
nep2	3e-12	3e-12	2e-05	5e-06★	1e-11	4e-12
nep3	5e-11	4e-12	2e-07	2e-09★	9e-12	1e-11
neuron_dde	6e-11	3e-11	8e-09	7e-11	8e-12	2e-11
pdde_symmetric	4e-11	3e-13	2e-07	2e-09★	2e-12	4e-12
photonic_crystal	6e-16	7e-16	1e-15	1e-15	2e-15	1e-15
pillbox_small	3e-13	8e-12	3e-07	1e-09★	1e-11	7e-12
sandwich_beam	2e-13	3e-12	5e-11	5e-11	9e-11	1e-11
schrodinger_abc	2e-11	5e-12	1e-07	2e-08★	5e-12	7e-12
square_root	6e-12	4e-12	2e-12	2e-12	3e-12	2e-12
time_delay	3e-11	7e-12	7e-11	1e-11	8e-12	1e-10
time_delay2	3e-11	2e-12	2e-09	4e-11	8e-12	9e-11
time_delay3	6e-12	4e-12	4e-10	4e-11	7e-12	2e-11
# of fails	0	0	16	12	0	0

AAA to identify good pole parameters in the first phase with the robustness of the Leja–Bagby approach in the second phase.

We have conducted an extensive numerical comparison of algorithms for matrix-valued functions on a large set of test problems from the NLEVP collections. We hope this test suite will be useful for other developers of NEP eigensolvers.

Our numerical experiments were performed with the same tolerance for the two phases of the two-phase algorithms. However, a stricter tolerance for phase 1 (i.e., surrogate AAA) could provide phase 2 with a better set of poles and a better approximant at the start of the Leja–Bagby refinement steps. Also, we limited our experiments to uniformly distributed (random) sampling points for the discrete set $\Sigma \subset \Sigma_T$. However, a uniform grid or a choice that reflects the properties of the specific matrix-valued function could lead to lower degree rational approximants but this is outside the scope of this work.

Appendix A. Linearization of $R^{(m)}$ in (4.16). We show how to rewrite $R^{(m)}(\lambda)v = 0$ as a linear eigenproblem $L(\lambda)x = 0$, when $R^{(m)}$ is expressed in the mixed form (4.16). We rewrite $R^{(m)}$ in the form

$$R^{(m)}(z) = \sum_{i=0}^{d} w_i F(z_i) b_i(z) + b_{d+1}(z) R_{d+1} + \cdots + b_m(z) R_m,$$

with $b_{d+i}(z)$, $i = 1, \ldots m-d$, as in (4.6) and $b_i(z) = \frac{1}{z-\sigma_i} \big/ \sum_{i=0}^{d} \frac{w_i}{z-\sigma_i}$, $i = 1, \ldots, d$.

Table 5.4: Degree of $R^{(m)}$ for ε = 1e-10 and 21 problems. The lowest degrees are highlighted in bold/blue including any within one of the lowest and excluding those corresponding to failed required accuracy that are provided within square brackets). A '★' indicates that the algorithm reached the maximum number of steps, i.e., 60.

Problem	set-valued AAA	weighted AAA	surrogate AAA	surrogate+ exact search	NLEIGS AAA poles	surrogate+ LB refine
bent_beam	**9**	**9**	[6]	[23]★	13	12
buckling_plate	**26**	**27**	[24]	[35]★	47	48
canyon_particle	**18**	**17**	[11]	[17]★	29	30
clamped_beam_1d	**13**	**13**	[12]	[16]★	29	29
distributed_delay1	**8**	**8**	[7]	[9]★	15	15
fiber	18	**15**	[11]	[19]★	30	30
gun	**12**	**12**	[8]	[60]★	21	21
hadeler	**7**	**8**	[6]	23	21	19
nep1	**25**	**24**	**24**	**24**	**24**	**24**
nep2	**16**	**16**	[11]	[14]★	22	22
nep3	**10**	**10**	[9]	[11]★	17	17
neuron_dde	**16**	**15**	[14]	18	31	31
pdde_symmetric	**9**	**10**	[9]	[11]★	18	18
photonic_crystal	**7**	**6**	**6**	**6**	**6**	**6**
pillbox_small	**10**	**9**	[7]	[11]★	16	16
sandwich_beam	45	25	**19**	23	26	54
schrodinger_abc	**13**	**13**	[12]	[60]★	21	21
square_root	**13**	**13**	**13**	**13**	**13**	**13**
time_delay	**15**	**15**	**14**	**15**	31	20
time_delay2	**15**	**16**	[14]	17	31	24
time_delay3	20	**16**	[14]	**16**	31	31
# of lowest degree	18	19	5	5	3	3

Then,

$$b_d(z) = \frac{1}{z-\sigma_d} \Bigg/ \sum_{i=0}^{d} \frac{w_i}{z-\sigma_i} = \prod_{j=0}^{d-1}(z-\sigma_j) \Bigg/ \sum_{i=0}^{d} w_i \prod_{j\neq i}(z-\sigma_j) \ .$$

The denominator of the right-hand side is, up to a scalar multiple, the same scalar denominator as that of $R^{(d)}$. Further, the numerator is the same as that of the Newton basis function obtained after d iterations of Newton interpolation. Consequently, $b_d(z)$ (the last barycentric basis function) is a scalar multiple of the basis function we would have obtained had we used Newton's interpolation from the start. We can therefore write down the recursions for all $b_0(z), \ldots, b_m(z)$ throughout, with the first d recursions corresponding to the barycentric basis functions, and then switching to $m-d$ steps of Newton:

$$b_0(z) = \frac{1}{z-\sigma_0} \Bigg/ \sum_{i=0}^{d} \frac{w_i}{z-\sigma_i} \ ,$$

$$(z-\sigma_{i+1})b_{i+1}(z) = (z-\sigma_i)b_i(z), \quad i = 0, \ldots, d-1,$$

$$\beta_{d+i+1}(1-z/\xi_{d+i+1})b_{d+i+1}(z) = (z-\sigma_{d+i})b_{d+i}(z), \quad i = 0, \ldots, m-d-1.$$

This recursion allows us to write the following linearization.

Table 5.5: Degree of $R^{(m)}$ for ε = 1e-13 and 21 problems. The lowest degrees are highlighted in bold/blue including any within one of the lowest and excluding those corresponding to failed required accuracy that are provided within square brackets). A '★' indicates that the algorithm reached the maximum number of steps, i.e., 60.

Problem	set-valued AAA	weighted AAA	surrogate AAA	surrogate+ exact search	NLEIGS AAA poles	surrogate+ LB refine
bent_beam	**12**	**11**	[7]	[23]★	16	16
buckling_plate	**30**	**30**	[26]	[35]★	[60]★	[60]★
canyon_particle	**23**	**22**	[16]	[18]★	38	39
clamped_beam_1d	**15**	**15**	[14]	[16]★	32	32
distributed_delay1	**9**	**9**	[8]	[9]★	16	16
fiber	22	**20**	[16]	[19]★	42	38
gun	**15**	**15**	[11]	60★	27	27
hadeler	**10**	**11**	[9]	[59]★	23	23
nep1	**29**	**28**	**28**	**28**	**28**	**28**
nep2	**18**	**19**	[12]	[14]★	24	24
nep3	**12**	**12**	[10]	[11]★	22	22
neuron_dde	**19**	**18**	[17]	[60]★	30	32
pdde_symmetric	**11**	**11**	[11]	[11]★	21	20
photonic_crystal	**7**	**6**	**6**	**6**	**6**	**6**
pillbox_small	**13**	**12**	[9]	[11]★	20	22
sandwich_beam	55	**38**	[31]	[60]★	[60]★	[60]★
schrodinger_abc	**15**	**15**	[13]	[60]★	26	26
square_root	**16**	**16**	**15**	**16**	33	**15**
time_delay	**18**	**18**	[17]	23	24	22
time_delay2	**17**	**18**	[17]	[60]★	28	29
time_delay3	23	**19**	[17]	[60]★	27	24
# of fails	0	0	18	17	2	2
# of lowest degree	17	20	3	3	2	3

THEOREM A.1. *Given the rational matrix-valued function $R^{(m)}$ in (4.16), the rational eigenvalue problem $R^{(m)}(\lambda)v = 0$ is equivalent to $Ax = \lambda Bx$, where*

$$A = \begin{bmatrix} w_0F(\sigma_0) & w_1F(\sigma_1) & \cdots & w_dF(\sigma_d) & R_{d+1} & \cdots & R_{m-2} & R_{m-1} - \frac{\sigma_{m-1}}{\beta_m}R_m \\ \sigma_0 I & -\sigma_1 I & & & & & & \\ & \ddots & \ddots & & & & & \\ & & \sigma_{d-1}I & -\sigma_d I & & & & \\ & & & \frac{\sigma_d}{\beta_{d+1}}I & I & & & \\ & & & & \ddots & \ddots & & \\ & & & & & \ddots & \ddots & \\ & & & & & & \frac{\sigma_{m-2}}{\beta_{m-1}}I & I \end{bmatrix}$$

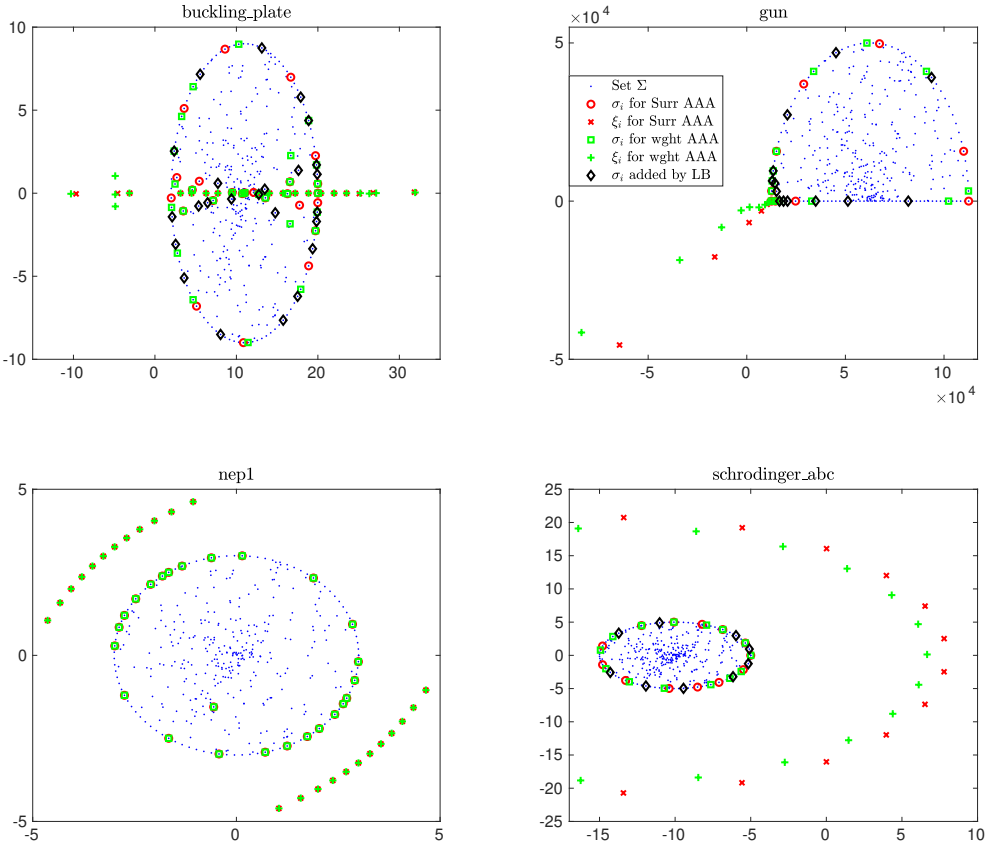

Fig. 5.1: The set Σ, the interpolation nodes σ_i and the poles ξ_i nearest to Σ for a subset of the problems in Table 5.1.

and

$$B = \begin{bmatrix} \frac{w_0 F(\sigma_0)}{\xi_m} & \frac{w_1 F(\sigma_1)}{\xi_m} & \cdots & \frac{w_d F(\sigma_d)}{\xi_m} & \frac{R_{d+1}}{\xi_m} & \cdots & \frac{R_{m-2}}{\xi_m} & \frac{R_{m-1}}{\xi_m} - \frac{R_m}{\beta_m} \\ I & -I & & & & & & \\ & \ddots & \ddots & & & & & \\ & & I & -I & & & & \\ & & & \frac{I}{\beta_{d+1}} & \frac{I}{\xi_{d+1}} & & & \\ & & & & \ddots & \ddots & & \\ & & & & & \ddots & \ddots & \\ & & & & & & \frac{I}{\beta_{m-1}} & \frac{I}{\xi_{m-1}} \end{bmatrix},$$

while $x = b(\lambda) \otimes v$ *with* $b(\lambda) = [b_0(\lambda)\, b_1(\lambda)\, \ldots, b_{m-1}(\lambda)]^T$.

REFERENCES

[1] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2005.

[2] A. C. Antoulas, C. A. Beattie, and S. Gugercin, *Interpolatory Methods for Model Reduction*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2020.
[3] T. Betcke, N. J. Higham, V. Mehrmann, C. Schröder, and F. Tisseur, *NLEVP: A collection of nonlinear eigenvalue problems*, ACM Trans. Math. Software, 39 (2013), pp. 7:1–7:28.
[4] C. Campos and J. E. Roman, *NEP: a module for the parallel solution of nonlinear eigenvalue problems in SLEPc*, Technical Report arXiv:1910.11712, 2019.
[5] E. W. Cheney, *Introduction to Approximation Theory*, Chelsea, New York, second ed., 1982.
[6] A. A. Condori, *Maximum principles for matrix-valued analytic functions*, The American Mathematical Monthly, 127 (2020), pp. 331–343.
[7] C. Effenberger and D. Kressner, *Chebyshev interpolation for nonlinear eigenvalue problems*, BIT, 52 (2012), pp. 933–951.
[8] M. El-Guide, A. Międlar, and Y. Saad, *A rational approximation method for solving acoustic nonlinear eigenvalue problems*, Eng. Anal. Bound. Elem., 111 (2020), pp. 44–54.
[9] S. Elsworth and S. Güttel, *Conversions between barycentric, RKFUN, and Newton representations of rational interpolants*, Linear Algebra Appl., 576 (2019), pp. 246–257.
[10] M. Froissart, *Approximation de Padé: application à la physique des particules élémentaires*, in RCP, Programme No. 25, vol. 9, CNRS, Strasbourg, 1969, pp. 1–13.
[11] I. V. Gosea and S. Güttel, *Algorithms for the rational approximation of matrix-valued functions*, Technical Report arXiv:2003.06410v1, 2020.
[12] S. Güttel, R. V. Beeumen, K. Meerbergen, and W. Michiels, *NLEIGS: A class of robust fully rational Krylov methods for nonlinear eigenvalue problems*, SIAM J. Sci. Comput., 36 (2014), pp. A2842–A2864.
[13] S. Güttel and F. Tisseur, *The nonlinear eigenvalue problem*, in Acta Numerica, vol. 26, Cambridge University Press, 2017, pp. 1–94.
[14] N. J. Higham, G. M. N. Porzio, and F. Tisseur, *An updated set of nonlinear eigenvalue problems*, Tech. Report 2019.5, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, Mar. 2019.
[15] A. Hochman, *FastAAA: A fast rational-function fitter*, in 2017 IEEE 26th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS), Oct 2017, pp. 1–3.
[16] E. Jarlebring, W. Michiels, and K. Meerbergen, *A linear eigenvalue algorithm for the nonlinear eigenvalue problem*, Numer. Math., 122 (2012), pp. 169–195.
[17] P. Lietaert, J. Pérez, B. Vandereycken, and K. Meerbergen, *Automatic rational approximation and linearization of nonlinear eigenvalue problems*, Technical Report arXiv:1801.08622v2, 2018.
[18] Y. Nakatsukasa, O. Sète, and L. N. Trefethen, *The AAA algorithm for rational approximation*, SIAM J. Sci. Comput., 40 (2018), pp. A1494–A1522.
[19] Y. Saad, M. El-Guide, and A. Międlar, *A rational approximation method for the nonlinear eigenvalue problem*, Technical Report arXiv:1901.01188v2, June 2020.
[20] E. B. Saff and V. Totik, *Logarithmic Potentials with External Fields*, vol. 316, Springer Science & Business Media, 2013.