Conversions between barycentric, RKFUN, and Newton representations of rational interpolants

Elsworth, Steven and Güttel, Stefan

2017

MIMS EPrint: **2017.40**

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

Conversions between barycentric, RKFUN, and Newton representations of rational interpolants

Steven Elsworth[a,b], Stefan Güttel[a,c]

[a] *School of Mathematics, The University of Manchester, M13 9PL Manchester, UK*
[b] *steven.elsworth@manchester.ac.uk*
[c] *stefan.guettel@manchester.ac.uk*

Abstract

We derive explicit formulas for converting between rational interpolants in barycentric, rational Krylov (RKFUN), and Newton form. We show applications of these conversions when working with rational approximants produced by the AAA algorithm [Y. Nakatsukasa, O. Sète, L. N. Trefethen, *SIAM J. Sci. Comput.* 40 (3), 2018] within the Rational Krylov Toolbox and for the solution of nonlinear eigenvalue problems.

Keywords: rational interpolation, rational Krylov, barycentric formula, nonlinear eigenvalue problem, AAA-NLEIGS
2010 MSC: 97N50, 30E10

1. Introduction

The Rational Krylov Toolbox (RKToolbox) is a collection of scientific computing tools based on rational Krylov techniques [1]. This MATLAB toolbox implements, among other algorithms, Ruhe's rational Arnoldi method [2] and the RKFIT method for nonlinear rational approximation [3]; see also the example collection on `http://rktoolbox.org`. At the heart of many RKToolbox algorithms are so-called RKFUNs (short for *rational Krylov functions*), which are matrix pencil-based representations of rational functions [4, 3]. The toolbox overloads more than 30 MATLAB commands for RKFUN objects using object-oriented programming, including root and pole finding (methods `roots` and `poles`), the efficient evaluation for scalar and matrix arguments (`feval`), addition, multiplication, and conversion to continued fraction form (`contfrac`). See [5, Chapter 7] for more details on the RKFUN calculus.

Since version 2.7 the RKToolbox supports the conversion of rational interpolants from their barycentric form into the RKFUN format. In this work we explain this conversion and also establish an explicit relation to rational Newton interpolation. We then apply these connections to the problem of sampling a nonlinear eigenvalue problem, making use of the recently developed AAA algorithm [6]. But let us first review the various rational function representations.

Assume that we are given pairs of complex numbers (z_j, f_j) with the z_j being pairwise distinct $(j = 0 : m)$. Then a rational interpolant of type $[m, m]$ can be written in *barycentric form* as

$$r(z) = \frac{\displaystyle\sum_{j=0}^{m} \frac{w_j f_j}{z - z_j}}{\displaystyle\sum_{j=0}^{m} \frac{w_j}{z - z_j}}, \tag{1}$$

where the barycentric weights w_j are nonzero but can otherwise be chosen freely (see, e.g., [7, 8, 9, 6]). It is easily verified that $r(z) \to f_j$ as $z \to z_j$, so r is indeed a rational interpolant for the data (z_j, f_j).

The *RKFUN representation* of a type $[m, m]$ rational function is

$$r \mathrel{\widehat{=}} (\underline{H_m}, \underline{K_m}, \mathbf{c}_m),$$

where $(\underline{H_m}, \underline{K_m})$ is an $(m+1) \times m$ unreduced upper-Hessenberg pencil (*unreduced* meaning that $\underline{H_m}$ and $\underline{K_m}$ have no common zero entries on their subdiagonals) and $\mathbf{c}_m = [c_0, c_1, \ldots, c_m]^T \in \mathbb{C}^{m+1}$ is a coefficient vector [3]. The matrix pencil gives rise to a sequence of rational basis functions $r_0, r_1, \ldots, r_m$ satisfying the *rational Arnoldi decomposition* [2, 4]

$$z\,[r_0(z), r_1(z), \ldots, r_m(z)]\underline{K_m} = [r_0(z), r_1(z), \ldots, r_m(z)]\underline{H_m}, \tag{2}$$

and r is defined as a linear combination of these basis functions:

$$r(z) = \sum_{j=0}^{m} c_j r_j(z). \tag{3}$$

Of course, at least one of the functions r_j needs to be fixed in order for the decomposition (2) to uniquely determine all other basis functions. In RKToolbox the convention is that $r_0 \equiv 1$, but other choices are possible.

Finally, a rational interpolant of type $[m, m]$ is in *Newton form*

$$r(z) = \sum_{j=0}^{m} d_j b_j(z)$$

if the basis functions b_j follow a recursion

$$b_0 \equiv 1, \quad b_j(z) = \frac{z - \sigma_{j-1}}{\beta_j(h_j - k_j z)} b_{j-1}(z), \quad j = 1 : m, \tag{4}$$

with complex scalars satisfying $\beta_j \neq 0$, $|h_j| + |k_j| \neq 0$, and $\sigma_{j-1} \neq h_j/k_j$ for all j. Note that the b_j form a rational Newton basis with each b_j having roots at the points σ_{i-1} and poles at $\xi_i := h_i/k_i$ for $i = 1 : j$.

2. From barycentric to RKFUN format

Our aim is to convert the rational function r from the barycentric form (1) into the RKFUN format (3). First we write

$$r(z) = \sum_{j=0}^{m} f_j r_j(z), \quad r_j(z) = \frac{\dfrac{w_j}{z - z_j}}{\displaystyle\sum_{i=0}^{m} \frac{w_i}{z - z_i}} \tag{5}$$

with the functions r_j satisfying the recursion

$$w_{j-1}(z - z_j)r_j(z) = w_j(z - z_{j-1})r_{j-1}(z), \quad j = 1 : m.$$

Note that in this representation the first basis function r_0 is not a constant when $m > 0$. Bringing terms containing z to the left,

$$z(w_{j-1}r_j(z) - w_j r_{j-1}(z)) = w_{j-1}z_j r_j(z) - w_j z_{j-1} r_{j-1}(z),$$

and collecting these equations in matrix form yields a rational Arnoldi decomposition

$$z\,[r_0(z), r_1(z), \ldots, r_m(z)]\underline{W_m} = [r_0(z), r_1(z), \ldots, r_m(z)]Z_m\underline{W_m} \tag{6}$$

with

$$Z_m = \begin{bmatrix} z_0 & & & & \\ & z_1 & & & \\ & & \ddots & & \\ & & & z_{m-1} & \\ & & & & z_m \end{bmatrix}, \quad \underline{W_m} = \begin{bmatrix} -w_1 & & & & \\ w_0 & -w_2 & & & \\ & \ddots & \ddots & & \\ & & & w_{m-2} & -w_m \\ & & & & w_{m-1} \end{bmatrix}.$$

Note that the $(m+1) \times m$ upper-Hessenberg pencil $(Z_m\underline{W_m}, \underline{W_m})$ is indeed unreduced as we assume nonzero barycentric weights w_j. We have therefore converted r into the RKFUN representation

$$r \mathrel{\widehat{=}} (Z_m\underline{W_m}, \underline{W_m}, \mathbf{f}_m), \quad \mathbf{f}_m = [f_0, f_1, \ldots, f_m]^T. \tag{7}$$

This form allows subsequent numerical operations on r as implemented in the RKToolbox. For example, the following result from [3, Section 5.2] summarizes how to compute the roots of r.

Theorem 1. *Given the RKFUN representation* (7) *with nonzero coefficient vector* $\mathbf{f}_m$*. Let* P *be an invertible* $(m+1) \times (m+1)$ *matrix such that* $P^{-1}\mathbf{f}_m = [\gamma, 0, \ldots, 0]^T$ *for some nonzero* γ*. Then the generalized eigenvalues of the lower* $m \times m$*-part of the pencil* $(P^{-1}Z_m\underline{W_m}, P^{-1}\underline{W_m})$ *correspond to the roots of* r *(with multiplicity).*

In the `roots` method in the RKToolbox, the matrix P is chosen as a Householder reflector $P = I_{m+1} - \sigma \mathbf{u}\mathbf{u}^H$. Note that Theorem 1 effectively enables us to find the m roots of a rational function in barycentric form (1) by solving an $m \times m$ generalized eigenvalue problem. The barycentric data z_j, f_j, w_j appear explicitly in the vector $\mathbf{f}_m$ and the matrices $Z_m, \underline{W_m}$. It is likely that the bidiagonal structure of $(Z_m\underline{W_m}, \underline{W_m})$ can be exploited in the solution of the generalized eigenproblem, but as m is typically moderate this is not our focus here.

The problem of root finding for polynomials and rational functions in (barycentric) Lagrange form has attracted quite some research, with the polynomial case slightly more explored. A popular approach for rational root finding, explained for example in [8], [9, Section 2.3.3], and also used in [6], involves the solution of an $(m+2) \times (m+2)$ eigenvalue problem, giving rise to two spurious eigenvalues at infinity. An exception is the very general class of CORK linearizations for nonlinear eigenvalue problems in [10], which contains our formulation and also leads to $m \times m$ eigenproblems (for scalar nonlinear eigenvalue problems, of which root finding is a special case). Theorem 1 spells out the conversion explicitly when r is given in the form (1), and it makes a direct link with the RKFUN root-finding approach in [3, Section 5.2].

For purposes other than root finding (e.g., pole finding) it may be desirable to modify (6) so that the first rational basis function r_0 is transformed into a constant. This can be achieved by utilizing the relation

$$\sum_{j=0}^{m} r_j(z) = 1.$$

Let Q be an invertible $(m+1) \times (m+1)$ matrix with first column $[1, 1, \ldots, 1]^T$. (For example, Q may be chosen as a multiple of a unitary matrix.) Then by inserting QQ^{-1} we transform (6) into the decomposition

$$z\,[\widehat{r}_0(z), \widehat{r}_1(z), \ldots, \widehat{r}_m(z)](Q^{-1}\underline{W_m}) = [\widehat{r}_0(z), \widehat{r}_1(z), \ldots, \widehat{r}_m(z)](Q^{-1}Z_m\underline{W_m})$$

where now $\widehat{r}_0 \equiv 1$. The equivalent to (7) is

$$r \mathrel{\widehat{=}} (Q^{-1}Z_m\underline{W_m}, Q^{-1}\underline{W_m}, Q^{-1}\mathbf{f}_m). \tag{8}$$

By computing a QZ decomposition of the lower $m \times m$-part of the pencil $(Q^{-1}Z_m\underline{W_m}, Q^{-1}\underline{W_m})$ we can restore the upper-Hessenberg structure and thereby obtain the RKFUN representation used in [1]. The following theorem is an immediate consequence of [1, Theorem 2.6].

Theorem 2. *Given the RKFUN representation* (8)*. Then the generalized eigenvalues of the lower $m \times m$-part of the pencil $(Q^{-1}Z_m\underline{W_m}, Q^{-1}\underline{W_m})$ correspond to the poles of r (with multiplicity).*

Again, the theorem requires an $m \times m$ eigenproblem to be solved and the involved matrices are products of structured matrices (for suitably chosen Q).

We have implemented the described method for converting a barycentric interpolant into RKFUN format in the utility function `util_bary2rkfun` provided with RKToolbox version 2.7. After the conversion, all methods overloaded for RKFUNs can be called on the barycentric interpolant. We demonstrate this with a simple example.

Example 1: In an example taken from [6, page 27] the AAA algorithm is used to compute a rational approximant $r(z)$ to the Riemann zeta function $\zeta(z)$ on the interval $[4-40i, 4+40i]$. After conversion into an RKFUN, we evaluate the matrix function $r(A)\mathbf{b} \approx \zeta(A)\mathbf{b}$ for a shifted skew-symmetric matrix $A \in \mathbb{R}^{20\times 20}$ having eigenvalues in $[4-40i, 4+40i]$ and a vector $\mathbf{b} = [1, 1, \ldots, 1]^T/\sqrt{20}$. This evaluation uses the efficient rerunning algorithm described in [3, Section 5.1] and requires no diagonalization of A. We then compute and display the error $\|\zeta(A)\mathbf{b} - r(A)\mathbf{b}\|_2$, which is found to be comparable to the default sampling accuracy of 10^{-13} used by AAA:

```
zeta = @(z) sum(bsxfun(@power, (1e5:-1:1)', -z));
[r, pol, res, zer, z, f, w] = aaa(zeta, linspace(4-40i,4+40i));
rat = util_bary2rkfun(z, f, w); % convert to RKFUN format
A = 10*gallery('tridiag', 10); S = 4*speye(10);
A = [ S , A ; -A , S ]; b = ones(20,1)/sqrt(20);
f = rat(A, b); % approximates zeta(A)*b
[V,D] = eig(full(A)); exact = V*(zeta(diag(D).').'.*(V\b));
norm(exact - f)

  ans = 1.6080e-13
```

3. From barycentric to Newton form

The recursion (4) for the Newton basis functions b_j can be linearized as

$$z\,[b_0(z), b_1(z), \ldots, b_m(z)]\underline{M_m} = [b_0(z), b_1(z), \ldots, b_m(z)]\underline{N_m} \tag{9}$$

with matrices

$$\underline{M_m} = \begin{bmatrix} 1 & & & & \\ \beta_1 k_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & \beta_{m-1}k_{m-1} & 1 & \\ & & & \beta_m k_m & \end{bmatrix}, \tag{10}$$

$$\underline{N_m} = \begin{bmatrix} \sigma_0 & & & & \\ \beta_1 h_1 & \sigma_1 & & & \\ & \ddots & \ddots & & \\ & & \beta_{m-1}h_{m-1} & \sigma_{m-1} & \\ & & & \beta_m h_m & \end{bmatrix}.$$

Comparing the pencil $(\underline{N_m}, \underline{M_m})$ in (9) with the pencil $(Z_m\underline{W_m}, \underline{W_m})$ in (6) we find that they are of the same structure. In particular, starting from (6) and defining for $j = 1 : m$ the quantities

$$\sigma_{j-1} := z_{j-1}, \quad \beta_j k_j := -\frac{w_{j-1}}{w_j}, \quad \beta_j h_j := -\frac{z_j w_{j-1}}{w_j},$$

we arrive at

$$z\,[r_0(z), r_1(z), \ldots, r_m(z)]\underline{M_m} = [r_0(z), r_1(z), \ldots, r_m(z)]\underline{N_m}$$

where $\underline{M_m}, \underline{N_m}$ are given in (10).

What we have just shown is that the basis functions r_j defined in (5) and associated with the barycentric interpolant can also be interpreted as Newton basis polynomials, except that the first function r_0 is not necessarily constant. This is a useful insight because numerical methods based on rational Newton interpolation may not need to be rewritten from scratch when switching to barycentric rational interpolation. The only change required is to convert the barycentric data z_j, f_j, w_j into the Newton data $\sigma_j, \xi_j = h_j/k_j, \beta_j$ using the explicit formulas provided above.

4. An application to nonlinear eigenproblems

Let $F : \Omega \to \mathbb{C}^{N\times N}$ be a matrix-valued function analytic on a domain $\Omega \subseteq \mathbb{C}$. We wish to compute points $\lambda \in \mathbb{C}$, the *eigenvalues of* F, at which $F(\lambda)$ is singular. This is the so-called *nonlinear eigenvalue problem* and many numerical methods have been developed for its solution; see, e.g., [11, 12] and the references therein. In this note we will focus on the NLEIGS method described in [13], which is based on an approximate expansion of F into a rational interpolant

$$R(z) = \sum_{j=0}^{m} b_j(z) D_j, \tag{11}$$

where the $D_j \in \mathbb{C}^{N\times N}$ are the matrix analogue to divided differences and the b_j are the rational Newton basis functions defined by (4). The rational interpolant $R(z)$ is then linearized into a (large, sparse and structured) matrix pencil $(\mathbf{A}_m, \mathbf{B}_m)$ of size $Nm \times Nm$. For completeness, we quote [13, Theorem 3.2].

Theorem 3. *Given a rational eigenvalue problem* (11) *with basis functions* b_j *defined by* (4)*. Then the linear pencil*

$$\mathbf{L}_m(z) = \mathbf{A}_m - z\mathbf{B}_m$$

with

$$\mathbf{A}_m = \begin{bmatrix} h_m D_0 & h_m D_1 & \cdots & h_m D_{m-2} & h_m D_{m-1} - h_m \sigma_{m-1} D_m/\beta_m \\ h_1\sigma_0 I & h_1\beta_1 I & & & \\ & \ddots & \ddots & & \\ & & h_{m-2}\sigma_{m-3} I & h_{m-2}\beta_{m-2} I & \\ & & & h_{m-1}\sigma_{m-2} I & h_{m-1}\beta_{m-1} I \end{bmatrix},$$

$$\mathbf{B}_m = \begin{bmatrix} k_m D_0 & k_m D_1 & \cdots & k_m D_{m-2} & k_m D_{m-1} - h_m D_m/\beta_m \\ h_1 I & k_1\beta_1 I & & & \\ & \ddots & \ddots & & \\ & & h_{m-2} I & k_{m-2}\beta_{m-2} I & \\ & & & h_{m-1} I & k_{m-1}\beta_{m-1} I \end{bmatrix},$$

is a strong linearization of $R(z)$*. If* $(\lambda, \mathbf{x})$*,* $\mathbf{x} \neq \mathbf{0}$*, is an eigenpair of* R*, that is* $R(\lambda)\mathbf{x} = \mathbf{0}$*, then* $(\lambda, b(\lambda) \otimes \mathbf{x})$ *with* $b(\lambda) := [b_0(\lambda), b_1(\lambda), \ldots, b_{m-1}(\lambda)]^T$ *is an eigenpair of* $\mathbf{L}_m(z)$*. Conversely, if* $(\lambda, \mathbf{y}_m)$ *is an eigenpair of* $\mathbf{L}_m(z)$ *then there exists a vector* $\mathbf{x}$ *such that* $\mathbf{y}_m = b(\lambda) \otimes \mathbf{x}$ *and* $(\lambda, \mathbf{x})$ *is an eigenpair of* R*.*

After linearization of $R(z)$ it remains to compute the eigenvalues of $(\mathbf{A}_m, \mathbf{B}_m)$ inside a *target set* Σ, a compact subset of Ω in which the desired eigenvalues of the nonlinear eigenproblem F are located. If $R \approx F$ uniformly on Σ, one can show (see, e.g., [12, Section 6]) that these eigenvalues are good approximants to the eigenvalues of F, and that there are *no spurious eigenvalues* inside Σ (i.e., eigenvalues of R which are not eigenvalues of F). Depending on the size Nm the linear eigenvalue problem can be solved either directly or iteratively (e.g., by a rational Krylov method [2]).

Note that the basis functions b_j in (4) depend on the sampling points σ_{i-1}, the poles $\xi_i = h_i/k_i$, and the scaling parameters β_i $(i, j = 1 : m)$, all of which have to be chosen so that $R \approx F$ uniformly on Σ. The NLEIGS sampling procedure described in [13, Section 5] serves this purpose, using greedy Leja–Bagby points as the parameters [14]. However, this approach requires the user to specify candidates for the pole parameters ξ_i, which may be a disadvantage in particular when the analytic form of F is not accessible.

In order to overcome this drawback, we consider instead of (11) the barycentric form

$$R(z) = \frac{\displaystyle\sum_{j=0}^{m} \frac{w_j}{z - z_j} F(z_j)}{\displaystyle\sum_{j=0}^{m} \frac{w_j}{z - z_j}} \tag{12}$$

and aim to choose the parameters z_j, w_j so that $R \approx F$ uniformly on Σ. The AAA algorithm [6] seems to provide a powerful tool for this, except that it is only applicable to scalar functions. However, if we use instead of the matrix-valued function F a *scalar surrogate function* f having the same region of analyticity and similar behaviour over Ω, we expect that the sampling parameters z_j and

the barycentric weights w_j computed for f are also good choices for interpolating the original function F via (12). Here we consider the scalar surrogate function $f(z) = \mathbf{u}^H F(z)\mathbf{v}$, where $\mathbf{u}, \mathbf{v} \in \mathbb{C}^N$ are random vectors of unit length.

Due to the direct link between the barycentric and Newton forms established in Section 3, we can modify the existing NLEIGS implementation in RKToolbox with minimal effort. Given the boundary of the target set Σ and the functionality to evaluate F in points on that boundary, the AAA-NLEIGS algorithm works as follows:

1. Discretize the boundary of the target set Σ with sufficiently many candidate points, collected in a set Z.
2. Run the AAA algorithm to compute a type $[m, m]$ rational interpolant r of the form (1) for the surrogate $f(z) = \mathbf{u}^H F(z)\mathbf{v}$ on the set Z. (The degree m is determined by the built-in stopping criterion of AAA.)
3. Convert the barycentric parameters z_j, f_j, w_j into the Newton parameters $\sigma_j, \xi_j = h_j/k_j, \beta_j$ using the formulas provided in Section 3.
4. Build the NLEIGS linearization in Theorem 3 using the Newton parameters. The matrices D_j are not affected by the conversion and we simply have $D_j = F(z_j) = F(\sigma_j)$.
5. Solve the linearized eigenproblem using, e.g., a rational Krylov iteration.

We illustrate this algorithm with the help of a simple numerical example.

Example 2: Consider the nonlinear eigenproblem $F(z) = A - z^{1/2}I$, where $A \in \mathbb{R}^{20\times 20}$ is the shifted skew-symmetric matrix from Example 1. The eigenvalues of F are the squares of the eigenvalues of A. As target set Σ we choose a disk of radius 50 centered at $10 + 50i$, inside of which are three eigenvalues of F. The boundary circle is discretized by 100 equispaced points, providing the set Z.

We apply the AAA algorithm to $f(z) = \mathbf{u}^H F(z)\mathbf{v}$, which returns with a rational interpolant r of degree $m = 15$ in barycentric form. The accuracy of r measured in the uniform norm over the set Z, as a function of the degree m, is shown on the left of Figure 1 (solid line). We also show the error $\max_{z\in Z} \|F(z) - R(z)\|_2$ for the corresponding matrix-valued interpolant R (dashed line) and confirm that both approximants converge at similar rates. For the degree $m = 15$ chosen by the AAA algorithm the accuracy of $r \approx f$ is $7.2 \cdot 10^{-14}$, while the accuracy of $R \approx F$ is $3.2 \cdot 10^{-13}$.

Following the algorithm outlined above, we convert the rational interpolant into Newton form and then supply the Newton parameters to the NLEIGS linearization code in RKToolbox. The resulting linear eigenvalue problem is of dimension $20m = 300$ and, for the purpose of this demonstration, small enough to be solved directly using MATLAB's `eig`.

On the right of Figure 1 we show the exact eigenvalues of F (red circles) and their approximants (magenta pluses), together with the sampling points z_j (black squares) and the poles ξ_j of the rational interpolant (green crosses). We observe that the three eigenvalues of F inside the target set Σ are well approximated by eigenvalues of the linearization, and that there are no spurious eigenvalues inside Σ.

Below we list the computed eigenvalue approximations in the target set Σ with their correct digits underlined:

```
-31.648445189114611 + 55.222282568754260i
  5.919823664567899 + 25.399434747010762i
 15.343672325361633 +  6.481124221680308i
```

This accuracy of about 13–15 digits is impressive given that only a degree of $m = 15$ was used for the linearization, and it is consistent with the accuracy of the matrix-valued interpolant $R \approx F$ over the target set Σ.

For a basic comparison with another popular solution approach, we have modified the `contour_eigensolver` code in [12, Figure 5.3] to solve the same problem. This method is closely related to the integral methods by Sakurai and Sugiura [15] and Beyn [16]. We found experimentally that at least `nc=230` quadrature nodes are required to locate the target eigenvalues with comparable accuracy. Computationally, this means that `nc=230` linear systems with coefficient matrices $F(z_j)$ have to be solved, one for each quadrature node z_j. The reason for this rather large number of nodes is the proximity of the circular contour (the boundary of Σ) to the branch cut of F, and the existence of some eigenvalues very close to this contour.

It is possible to construct specialised quadrature rules which may lead to a reduction of the number of required nodes, for example analytically when Σ is an interval [17, 18], or numerically via least squares optimization [19]. However, as [19] points out, it is still difficult for a non-experienced user to effectively choose the parameters of the involved optimization problems. In view of this we believe that the "black-box" AAA-NLEIGS approach might indeed be an alternative to contour-based methods, both in terms of convenience and efficiency.

5. AAA-NLEIGS code and possible extensions

To demonstrate the ease of use of the essentially parameter-free AAA-NLEIGS algorithm presented in Section 4, we include the complete MATLAB code for producing Figure 1 (right). It is assumed that RKToolbox version 2.7 and the matrix A from Example 1 are available.

```
F = @(z) A - z^.5*speye(20);         % NEP, A as in Example 1
exact = eig(full(A)).^2;             % compute and plot the
plot(exact,'ro'); hold on            % exact eigenvalues of F
Z = 10+50i+50*exp(2i*pi*(0:99)/100); % target set
plot(Z,'k:'); axis equal; axis([-100,60,-60,100])
[ratm,~,~,~,z] = util_aaa(F,Z);      % call AAA sampling
plot(z,'ks'), plot(poles(ratm),'gx') % plot samples and poles
Lm = linearize(ratm);                % linearize into pencil
[Am,Bm] = Lm.get_matrices();         % get pencil matrices
ee = eig(full(Am),full(Bm));         % compute and plot the
plot(ee,'m+')                        % eigenvalues of interpolant
legend('exact evs','target set','samples','poles','approx evs')
```

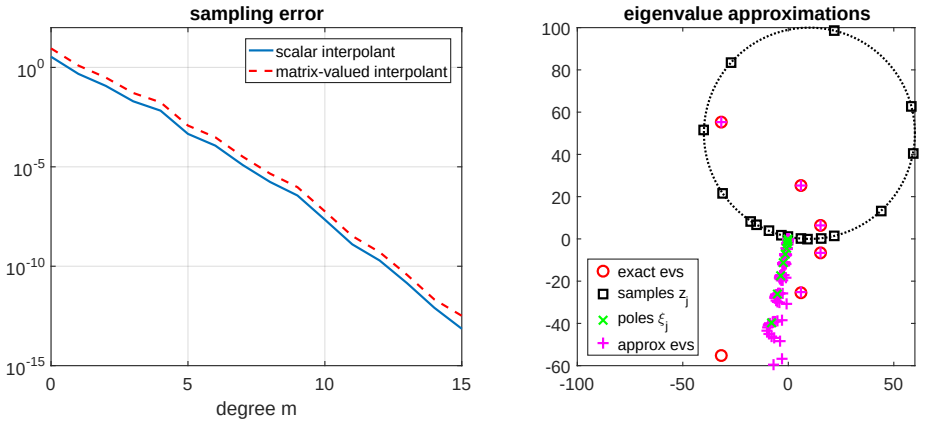

Figure 1: Solution of a nonlinear eigenvalue problem using the AAA-NLEIGS algorithm. Left: Accuracy of the AAA sampling procedure as the degree m of the rational interpolant increases. Right: Exact eigenvalues and their approximations, together with the sampling points z_j and the poles ξ_j of the rational interpolant.

Example 2 merely served as an illustration. For a more robust implementation of the AAA-NLEIGS algorithm one would need to monitor the actual accuracy of R as an approximation to F on the target set Σ, and in particular make sure that R has no poles there. In other tests we sometimes found it beneficial to place candidate points in Z not only on the boundary but also inside of Σ.

Further, the random choice of the projection vectors $\mathbf{u}, \mathbf{v}$ may not be the most appropriate when some more information on the nonlinear eigenvalue problem is accessible. For example, if the nonlinear variation in F is of low rank, i.e., $F(z) = A - zB + f(z)C$ with C being a low-rank matrix, then a better approach would be to choose $\mathbf{u}$ and $\mathbf{v}$ as (approximate) leading left and right singular vectors of C, respectively (or to sample f directly using the AAA algorithm). If $F(z)$ is only available as a black-box routine, then an effective projection direction $\mathbf{v}$ may still be inferable, e.g., by computing an SVD of $[F(z_1)\mathbf{w}, \dots, F(z_\ell)\mathbf{w}]$ with a randomly chosen vector $\mathbf{w}$ and a (hopefully moderate) number ℓ of sampling points, and similarly for $\mathbf{u}$.

Another AAA-based approach for sampling a nonlinear eigenvalue problem given in split form $F(z) = f_1(z)C_1 + \cdots + f_d(z)C_d$ with fixed matrices $C_j \in \mathbb{C}^{N\times N}$ has been proposed in [20]. Therein the scalar functions $\{f_1, \dots, f_d\}$ are simultaneously sampled with common sampling points and barycentric weights using a set-valued implementation of the AAA algorithm. The numerical comparisons in [20] indicate that this approach usually leads to linearizations of smaller degree than those achieved with Leja–Bagby interpolation.

6. Conclusions

We have derived explicit formulas for converting between rational interpolants in barycentric, rational Krylov (RKFUN), and Newton form, and have shown two examples where this conversion is useful within the RKToolbox. In particular we have shown a way to combine the AAA algorithm with the NLEIGS solver for nonlinear eigenvalue problems. In future work we plan to extend the sampling procedure to nonsquare problems, e.g., for sampling solutions of vector-valued initial and boundary value problems. Further, conversions to other structured representations of rational functions might be useful for performing certain operations (such as root-finding) more efficiently; see, e.g., the rank-structured representations in [21, Chapter 14].

Acknowledgements. The authors would like to thank the Engineering and Physical Sciences Research Council (EPSRC) and Sabisu for providing SE with a CASE PhD studentship. SG is grateful to Yuji Nakatsukasa, Olivier Sète, and Nick Trefethen for their hospitality during a sabbatical in 2016, and for many interesting discussions on the AAA algorithm and nonlinear eigenvalue problems. We also thank the two anonymous referees for their helpful comments.

References

[1] M. Berljafa, S. Elsworth, S. Güttel, A Rational Krylov Toolbox for MATLAB, MIMS EPrint 2014.56, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, available for download at `http://rktoolbox.org` (2017).

[2] A. Ruhe, Rational Krylov algorithms for nonsymmetric eigenvalue problems. II. Matrix pairs, Linear Algebra Appl. 198 (1994) 283–295.

[3] M. Berljafa, S. Güttel, The RKFIT algorithm for nonlinear rational approximation, SIAM J. Sci. Comput. 39 (5) (2017) A2049–A2071.

[4] M. Berljafa, S. Güttel, Generalized rational Krylov decompositions with an application to rational approximation, SIAM J. Matrix Anal. Appl. 36 (2) (2015) 894–916.

[5] M. Berljafa, Rational Krylov Decompositions: Theory and Applications, Ph.D. thesis, The University of Manchester, Manchester, UK, available as MIMS EPrint 2017.6 at `http://eprints.ma.man.ac.uk/2529/` (2017).

[6] Y. Nakatsukasa, O. Sète, L. N. Trefethen, The AAA algorithm for rational approximation, SIAM J. Sci. Comput. 40 (3) (2018) A1494–A1522.

[7] C. Schneider, W. Werner, Some new aspects of rational interpolation, Math. Comp. 47 (175) (1986) 285–299.

[8] P. W. Lawrence, R. M. Corless, Stability of rootfinding for barycentric Lagrange interpolants, Numer. Algorithms 65 (3) (2014) 447–464.

[9] G. Klein, Applications of Linear Barycentric Rational Interpolation, Ph.D. thesis, Université de Fribourg (2012).

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

[11] V. Mehrmann, H. Voss, Nonlinear eigenvalue problems: A challenge for modern eigenvalue methods, GAMM-Mitt. 27 (2004) 121–152.

[12] S. Güttel, F. Tisseur, The nonlinear eigenvalue problem, Acta Numer. 26 (2017) 1–94.

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

[14] T. Bagby, On interpolation by rational functions, Duke Math. J. 36 (1) (1969) 95–104.

[15] T. Sakurai, H. Sugiura, A projection method for generalized eigenvalue problems using numerical integration, J. Comput. Appl. Math. 159 (1) (2003) 119–128.

[16] W.-J. Beyn, An integral method for solving nonlinear eigenvalue problems, Linear Algebra Appl. 436 (10) (2012) 3839–3863.

[17] S. Güttel, E. Polizzi, P. T. P. Tang, G. Viaud, Zolotarev quadrature rules and load balancing for the FEAST eigensolver, SIAM J. Sci. Comput. 37 (4) (2015) A2100–A2122.

[18] A. P. Austin, L. N. Trefethen, Computing eigenvalues of real symmetric matrices with rational filters in real arithmetic, SIAM J. Sci. Comput. 37 (3) (2015) A1365–A1387.

[19] M. Van Barel, Designing rational filter functions for solving eigenvalue problems by contour integration, Linear Algebra Appl. 502 (2016) 346–365.

[20] P. Lietaert, J. Pérez, B. Vandereycken, K. Meerbergen, Automatic rational approximation and linearization of nonlinear eigenvalue problems, Tech. rep., arXiv preprint arXiv:1801.08622 (2018).

[21] R. Vandebril, M. Van Barel, N. Mastronardi, Matrix Computations and Semiseparable Matrices, Volume 2: Eigenvalue and Singular Value Methods, Johns Hopkins University Press, Baltimore, MD, USA, 2008.