Efficient Ehrlich–Aberth iteration for finding intersections of interpolating polynomials and rational functions

Leonardo Robol[a,1], Raf Vandebril[b]

[a] *ISTI, Area della ricerca CNR, Via G. Moruzzi 1, 56124 Pisa, Italy.*
[b] *Department of Computer Science, KU Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium.*

Abstract

We analyze the problem of carrying out an efficient iteration to approximate the eigenvalues of some rank structured pencils obtained as linearization of sums of polynomials and rational functions expressed in (possibly different) interpolation bases. The class of linearizations that we consider has been introduced by Robol, Vandebril and Van Dooren in [17]. We show that a traditional QZ iteration on the pencil is both asymptotically slow (since it is a cubic algorithm in the size of the matrices) and sometimes not accurate (since in some cases the deflation of artificially introduced infinite eigenvalues is numerically difficult). To solve these issues we propose to use a specifically designed Ehrlich–Aberth iteration that can approximate the eigenvalues in $O(kn^2)$ flops, where k is the average number of iterations per eigenvalue, and n the degree of the linearized polynomial. We suggest possible strategies for the choice of the initial starting points that make k asymptotically smaller than $O(n)$, thus making this method less expensive than the QZ iteration. Moreover, we show in the numerical experiments that this approach does not suffer of numerical issues, and accurate results are obtained.

Keywords: Matrix polynomials, Rational functions, Ehrlich–Aberth, Rank structure, Eigenvalues, Polynomial roots
2010 MSC: 00-01, 99-00

1. Introduction

Polynomials and rational functions are used extensively in mathematics and engineering, for modeling and as approximations of smooth functions [2, 3, 21].

Email addresses: leonardo.robol@isti.cnr.it (Leonardo Robol), raf.vandebril@cs.kuleuven.be (Raf Vandebril)

[1] This research has been partially supported by the Region of Tuscany (Project "MOSCARDO - ICT technologies for structural monitoring of age-old constructions based on wireless sensor networks and drones", 2016- 2018, FAR FAS), and by the GNCS/INdAM project "Metodi numerici avanzati per equazioni e funzioni di matrici con struttura".

A particularly relevant application is the analysis of closed loop linear systems, which involves also matrices of rational functions when MIMO systems are considered [16]. Often one is interested in finding the roots of sums of polynomials or rational functions that are expressed in different bases, such as interpolation bases with distinct nodes. Robol, Vandebril and Van Dooren introduced a framework [17] that provides the possibility to linearize[2] rational functions of the form:

$$f(\lambda) = \frac{p_1(\lambda)}{q_1(\lambda)} + \frac{p_2(\lambda)}{q_2(\lambda)},$$

where $p_i(\lambda)$ and $q_i(x)$ can be expressed in different polynomial bases. More general forms with more than 2 summands are possible (see [17] for further details). The linearizations obtained in this setting, as we will see in Section 2, have particular rank structures, which suggests that a fast method for finding their eigenvalues might be formulated. This is precisely the aim of this work.

We will concentrate on the case where $p_i(\lambda)$ and $q_i(\lambda)$ are expressed in interpolation bases, namely the *Newton* and *Lagrange* ones. The framework can be extended to cover the case of rational and polynomial eigenproblems, that is to the problem of finding the values of λ that make the matrix

$$F(\lambda) := P_1(\lambda)^{-1} Q_1(\lambda) + P_2(\lambda) Q_2^{-1}(\lambda) \tag{1}$$

singular, even when the bases in which the $P_i(\lambda)$ and the $Q_i(\lambda)$ are represented do not match. This problem arises, for example, when one wants to verify that a transfer function associated to a linear time invariant system has all the eigenvalues in the left plane, thus ensuring that the associated system is stable [16]. When the factors of the transfer functions have been computed using different interpolation nodes the problem fits precisely in the framework that we are describing.

Linearizations are widely used to find roots of polynomials and matrix polynomials. Given a polynomial $p(\lambda)$ one usually constructs a pencil $\mathcal{L}(\lambda) := A - \lambda B$ such that $\det \mathcal{L}(\lambda) = p(\lambda)$, and then computes its eigenvalues using an approximation method. This strategy has the advantage of relying on well-tested and efficient numerical software for the approximation of eigenvalues, usually the QZ iteration (or the QR when the pencil is monic).

However, there are some drawbacks to this approach. Since we rephrase the rootfinding problem as an eigenvalue one, applying an unstructured method leads to a cubic computational cost in the degree and possibly to a higher condition number. In fact, once the coefficients of the polynomial are embedded in a companion matrix the set of possible perturbations becomes larger, and the condition number of the eigenvalue problem can grow due to this fact [10]. Motivated by the introduction of a new class of linearizations for sums of polynomials and rational functions in [17], we develop a class of structured iterations for the approximation of the eigenvalues of such pencils.

[2] Here by linearize we mean constructing a linear pencil whose eigenvalues are the solution of the given nonlinear equation.

Our approach is based on the Ehrlich–Aberth method, which is a functional iteration for the approximation of roots of polynomials [1, 11]. We will shorten it as EAI in the following. One advantage is that, even if the linearizations can have spurious infinite eigenvalues, the EAI can implicitly deflate them at no additional cost and without introducing numerical errors. In contrast, the QZ iteration would need an explicit deflation step (either a priori or a posteriori). Moreover, the EAI relies on the original input data at each step of the iteration, unlike the QR algorithms, making it much easier to exploit the structure of the problem.

The advantage of this approach compared to just running the EAI on the scalar polynomial is that it provides a backward stable evaluation method. This can be transparently applied to any polynomial basis with a two-term recurrence relation (like monomials, Newton and Lagrange, which are described here — and with small adaptations could also be extended to three terms recurrence relations). Moreover, the matrix polynomial and rational case of (1) can be handled with minimal modifications.

In Section 2 we briefly review the structure and the construction of the pencils $A - \lambda B$ introduced in [17]. In Section 3, we recap the definition of the Ehrlich–Aberth iteration and we provide efficient strategies for the selection of the starting points. In Section 3 we show that computing the Newton correction is the main ingredient in order to apply the EAI. In Section 4, we show how such structure can be exploited to compute it in a fast and accurate way. Finally, numerical experiments are reported in Section 5.

2. Linearizing interpolation polynomials

It is shown in [17] that linearizations for sums of polynomials and rational functions can be realized easily if one knows the so-called *dual bases* related to the polynomial bases of interest. We will briefly recall these concepts and then show how the construction can be performed in the Newton and Lagrange cases. These definitions, which are here adapted for our needs, go back to the work of Forney [13].

Definition 2.1. Let $\phi_0(\lambda), \dots, \phi_k(\lambda)$ be a basis for the vector space of scalar polynomials of degree at most k. We say that a $k \times (k+1)$ linear pencil $A - \lambda B$ is *dual to the polynomial basis* $\phi_j(\lambda)$, $j = 0, \dots, k$, if

$$(A - \lambda B) \begin{bmatrix} \phi_k(\lambda) \\ \vdots \\ \phi_0(\lambda) \end{bmatrix} = 0. \tag{2}$$

In the following we will use $\pi_\phi(\lambda)$ to denote the column vector containing $\phi_k(\lambda) \dots \phi_0(\lambda)$.

The concept of *duality* introduced in [13] is much more general than what is described here, since it handles bases with different sizes and degrees. Another

important concept defined by Forney is the one of minimality. For its definition we rely on the row-degree, which can be defined as the maximum of the degrees of the entries in a row. As an example, the row-degrees of

$$\begin{bmatrix} \lambda & \lambda^2 - 1 & 1 \\ 1 & \lambda & 0 \end{bmatrix}$$

are 2 and 1.

Definition 2.2. We say that a matrix polynomial $P(\lambda) = \sum_{i=0}^{d} P_i \lambda^i \in \mathbb{C}[\lambda]^{k_1 \times k_2}$ is *minimal* if its rows form a basis of a subspace of $\mathbb{C}(\lambda)^{k_2}$, the vector space of k_2-tuples of rational functions, and the sum of its row-degrees is minimal among all the possible polynomial bases for that subspace.

The above definition is often difficult to check in practice, so that the following characterization will be useful.

Lemma 2.3. *A matrix polynomial* $P(\lambda) = \sum_{i=0}^{d} P_i \lambda^i$ *is* minimal *if and only if*

- *its row rank is maximal for every* $\lambda \in \mathbb{C}$*;*
- *the matrix whose rows are the highest degree coefficients of the polynomial rows of* $P(\lambda)$ *has full row rank;*

Remark 2.4. It is immediate to verify that the row vector $\pi_\phi^T(\lambda)$ containing the elements of a polynomial basis $\phi_j(\lambda)$ is always minimal according to the above definition. In fact, its highest degree coefficient is e_1^T, and so different from zero, and thus has rank 1. Moreover, if w is the column vector with the coordinates of 1 in the given basis then $\pi_\phi^T(\lambda) w = 1$ independently of λ, thus proving the rank 1 property for every λ.

The same can not be said of the pencils dual to $\pi_\phi^T(\lambda)$. However, when the minimality property holds, we say that the pencil is *minimal and dual to* $\phi_j(\lambda)$. Here we state a general result, adapted from the framework of [17], which eases the construction of linearizations for rational functions.

Theorem 2.5. *Let* $p_i(\lambda)$, $q_i(\lambda)$ *for* $i = 1, 2$ *be polynomials of degree* d *with no common factors. Denote by* p_i, q_i *the vectors of their coefficients in two bases which are dual to* $L_\phi(\lambda)$ *and* $L_\psi(\lambda)$*, respectively. Then the matrix pencil*

$$\mathcal{L}(\lambda) := \begin{bmatrix} p_1 q_2^T - p_2 q_1^T & L_\phi^T(\lambda) \\ L_\psi(\lambda) & 0 \end{bmatrix} \in \mathbb{C}^{(2d+1)\times(2d+1)}[\lambda]$$

is a linearization for the polynomial $p_1(\lambda)q_2(\lambda) - p_2(\lambda)q_1(\lambda)$ *so, in particular, has as finite eigenvalues the solutions of the rational equation*

$$\frac{p_1(\lambda)}{q_1(\lambda)} = \frac{p_2(\lambda)}{q_2(\lambda)}.$$

Remark 2.6. When $p_i(\lambda)$ and $q_i(\lambda)$ share a common factor the above construction is still a linearization for $p_1(\lambda)q_2(\lambda) - p_2(\lambda)q_1(\lambda)$. In this case, however, the common factors might appear as additional eigenvalues which are not roots of the rational equation.

The above result can be used to linearize sums of rational functions defined as quotient of polynomials expressed in different bases. We show that, when a certain structure is present in the matrices $L_\phi(\lambda)$ and $L_\psi(\lambda)$, one can apply a fast and stable functional iteration to approximate all the solutions.

The results that follow do not strictly depend on the rank 2 in the top-left block, and they are generalizable to rank k blocks with some $k \leqslant d$. One can check then that the obtained pencils are linearizations of polynomials of the form

$$p(\lambda) = \sum_{i=1}^{k} p_i(\lambda) q_i(\lambda).$$

Moreover, it is possible to formulate a block version of the above result which yields linearizations of the form

$$\mathcal{L}(\lambda) := \begin{bmatrix} p_1 q_2^T - p_2 q_1^T & L_\phi^T(\lambda) \otimes I_k \\ L_\psi(\lambda) \otimes I_k & 0 \end{bmatrix}, \qquad p_i, q_i \in \mathbb{C}^{dk \times k}, \tag{3}$$

whose eigenvalues coincides with the ones of the nonlinear matrix function

$$F(\lambda) := P_1(\lambda)^{-1} Q_1(\lambda) + P_2(\lambda) Q_2^{-1}(\lambda).$$

2.1. Newton linearizations

Let $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ be a (ordered) set of interpolation nodes in the complex plane. Then the Newton basis related to Σ is defined as follows:

$$n_{\Sigma,j}(\lambda) = \prod_{i \leqslant j} (\lambda - \sigma_i), \qquad j = 1, \ldots, k.$$

Given a function $f(\lambda)$ or, more generally, a set of points f_j for $j = 1, \ldots, k$, we can construct the interpolating polynomial $p(\lambda)$ such that $p(\sigma_j) = f_j$ by computing the so-called *divided differences*. This is a classical topic in interpolation theory, for which we refer to [22].

The following result gives a concrete recipe to construct a dual basis for the Newton case. The proof can be found in [17].

Lemma 2.7 (Section 3.6 of [17])**.** *The linear pencil $L_{\Sigma,k}(\lambda)$ of size $k \times (k+1)$ for the nodes $\sigma_1, \ldots, \sigma_k$ defined as follows*

$$L_{\Sigma,k}(\lambda) := \begin{bmatrix} 1 & -(\lambda - \sigma_k) & & \\ & \ddots & \ddots & \\ & & 1 & -(\lambda - \sigma_1) \end{bmatrix}.$$

is dual to the Newton basis associated with $\sigma_1, \ldots, \sigma_k$.

2.2. Lagrange linearizations

A construction for the Lagrange case can be given in a similar way. This case is also treated in [17], but we prefer to introduce a slight variation that makes the dual basis equal to the one used in [20] to linearize Lagrange polynomials.

Given a set of nodes $\sigma_1, \dots, \sigma_k$ we consider the set of (scaled) Lagrange polynomials defined as:

$$\ell_j(\lambda) := \theta_j \prod_{\substack{i=1 \\ i \neq j}}^{k} \frac{\lambda - \sigma_i}{\sigma_j - \sigma_i}, \qquad j = 1, \dots, k \tag{4}$$

Lemma 2.8. *Given a set of nodes σ_j, $j = 1, \dots, k$, the following matrix pencil is dual to the scaled Lagrange basis defined in (4) for any choice of non-zero θ_j:*

$$L_{k,\phi}(\lambda) = \begin{bmatrix} (\lambda - \sigma_k) & -(\lambda - \sigma_{k-1})\frac{\theta_k}{\theta_{k-1}} & & \\ & \ddots & \ddots & \\ & & (\lambda - \sigma_1) & -(\lambda - \sigma_0)\frac{\theta_1}{\theta_0} \end{bmatrix}.$$

Proof. It is easy to check that $L_{k,\phi}(\lambda)\pi_\phi(\lambda) = 0$. Moreover, the pencil $L_{k,\phi}(\lambda)$ is a row and column scaling of the one introduced in [17], and so it has the same property of maximal rank for any λ. □

In order to keep the growth of the coefficients under control it is often convenient to choose the parameter θ_j as the the barycentric weights of the nodes σ_j. We refer to [20] for the details concerning this choice.

3. The Ehrlich–Aberth iteration

The Ehrlich–Aberth method [1, 11] is a functional iteration that simultaneously approximates all the roots of a scalar polynomial $p(\lambda)$. It works by updating a set of d approximations $\lambda_1, \dots, \lambda_d$, where d is the degree of $p(\lambda)$, by means of the following formula:

$$\lambda_i^{(k+1)} = \lambda_i^{(k)} - \frac{N(\lambda_i)}{1 - \sum_{j \neq i} \frac{1}{\lambda_i^{(k)} - \lambda_j^{(k)}} \cdot N(\lambda_i)}, \qquad N(\lambda) = \frac{p(\lambda)}{p'(\lambda)},$$

where $N(\lambda)$ is Newton's correction of the polynomial at the point λ. This iteration can be seen as Newton's correction computed on the rational functions

$$R_i(\lambda) = \frac{p(\lambda)}{\prod_{j \neq i}(\lambda_i - \lambda_j)}, \qquad i = 1, \dots d.$$

Whenever the approximations $\lambda_i^{(k)}$ are near the roots of the polynomial for $i \neq j$, then $R_j(\lambda)$ is almost linear in a neighborhood of $\lambda_j^{(k)}$, and so Newton's method converges fast. In fact, it is possible to prove that the Ehrlich–Aberth iteration is locally cubically convergent on simple roots, and linearly on multiple ones [1].

In this work we discuss the applicability of the Ehrlich–Aberth method to the computation of the eigenvalues of a square $n \times n$ pencil $A - \lambda B$. A similar idea has been previosuly considered by Bini, Gemignani, and Tisseur in [4] and by Bini and Noferini in [6]. We know that (if no infinite eigenvalues are present) the degree of $\det(A - \lambda B)$ is equal to n, and its eigenvalues are the roots of this polynomial. We recall that computing the coefficients of the scalar polynomial $p(\lambda) := \det(A - \lambda B)$ starting from the matrices A and B is an ill-conditioned operation in general [9]. For this reason, we rely on the following formula for the application of the EAI.

Theorem 3.1 (Jacobi's formula)**.** *Let $\mathcal{A}(\lambda)$ be a C^1 matrix function. Then*

$$\frac{d}{d\lambda} \det \mathcal{A}(\lambda) = \operatorname{tr}\left(\operatorname{adj} \mathcal{A}(\lambda) \cdot \frac{d}{d\lambda}\mathcal{A}(\lambda)\right),$$

where $\operatorname{adj}(\cdot)$ *is the adjugate operator.*

Theorem 3.1 can be exploited to compute Newton's correction of $p(\lambda) := \det \mathcal{L}(\lambda)$. We have, in fact,

$$N(\lambda) = \left(\operatorname{tr}\left(\mathcal{A}(\lambda)^{-1} \frac{d}{d\lambda}\mathcal{A}(\lambda)\right)\right)^{-1}.$$

Applying the above formula to the pencil $\mathcal{L}(\lambda) := A - \lambda B$ yields the relation

$$N(\lambda) = -\left(\operatorname{tr}\left((A - \lambda B)^{-1} B\right)\right)^{-1}. \tag{5}$$

In Section 4 we will see how to exploit the structure to compute Newton's correction in a fast way.

3.1. Choosing the starting points

A non-trivial task in the implementation of the Ehrlich–Aberth iteration is the choice of the starting points. As suggested by Aberth in [1], a strategy that works well in most cases is to put them on a circle whose radius is slightly larger than the maximum modulus of all the roots. In order to do this we need to estimate the spectral radius of the pencil $A - \lambda B$. However, we have emphasized at the beginning that our pencil might have infinite eigenvalues, which we want to ignore. From now on, whenever we will mention the *spectral radius* of $A - \lambda B$, we will mean the maximum modulus of the *finite* eigenvalues.

We present two different strategies to provide starting points. The first is based on an adaptation of the power method, while the other relies on contour integration.

3.1.1. Power method

Given a pencil $A - \lambda B$ one can estimate the spectral radius by running a certain number of iterations of an adapted power method. Recall that, in

the standard eigenvalue problem setting, the power method associated with a matrix M is obtained by performing the iteration

$$x^{(k+1)} = Mx^{(k)}.$$

Assuming there exist a unique and simple dominant eigenvalue λ_1 so that $|\lambda_j| < |\lambda_1|$ for any $j > 1$, the ratio between the entries of $x^{(k+1)}$ and $x^{(k)}$ converges to λ_1 as $k \to \infty$. Renormalization of $x^{(k)}$ might be needed after some steps in order to avoid overflow or underflow situations.

This method can be generalized easily to a pencil when B is invertible by running the iteration

$$x^{(k+1)} = B^{-1}Ax^{(k)}$$

which is equivalent to the above when setting $M = B^{-1}A$. Notice, however, the explicit computation of the matrix M is not needed and one can perform the iteration by solving a certain number of linear systems.

In our case, however, B is singular[3], so we make use of Brauer's theorem, which is a simple yet powerful tool that allows one to move a specified eigenvalue of a matrix [8] and, more generally, of matrix functions expressed as Laurent series [5]. In our case we are interested in shifting an entire Jordan chain from the infinite eigenvalue to a zero one, such that it will not interfere with the power iteration and estimation of the dominant finite eigenvalue.

In order to achieve this result we prove a version of Brauer's theorem for pencils. This is a generalization of the original one in [8], and a particular case of [5]. Our formulation allows to transparently deal with the shift of infinity eigenvalues to 0, which is not achievable directly with the formulations in [5, 8]. To achieve this, we identify the eigenvalues of the pencil with the projective points in $\mathbb{P}^1(\mathbb{C})$.

Theorem 3.2 (Brauer)**.** *Let $\mu A - \lambda B$ a pencil with eigenvalues (λ_i, μ_i), and assume that v is a right eigenvector associated to a simple eigenvalue (λ_*, μ_*), i.e.,*

$$(\mu_* A - \lambda_* B)v = 0.$$

Let w be the only vector such that $Av = \lambda_* w$ and $Bv = \mu_* w$. Then, for any vectors u_A and u_B, the matrix pencil

$$\mu\tilde{A} - \lambda\tilde{B}, \qquad \tilde{A} := A + wu_A^T,\ \tilde{B} := B + wu_B^T$$

has the same eigenstructure of the original pencil $\mu A - \lambda B$ with the only exception of the eigenvalue (λ_*, μ_*) which is moved to $(\lambda_* + u_A^T v, \mu_* + u_B^T v)$.

Proof. We notice that the vector w is always well defined, since λ_* and μ_* cannot be zero at the same time. We then consider the Kronecker canonical form of

[3] In fact, the linearization of Theorem 2.5 has size $2d + 1$, but linearizes a polynomial of degree $2d$. This implies that the linear term of the pencil is singular. We refer to [17] for a details analysis of the eigenstructure of the pencil.

the pencil given by the upper triangular pencil $\mu T_A - \lambda T_B$ defined as follows

$$(\mu A - \lambda B)V = W(\mu T_A - \lambda T_B), \tag{6}$$

with V and W invertible matrices. Let $v_1 := Ve_1$ and $w_1 := We_1$, and assume that we ordered the diagonal elements so that λ_* and μ_* are found in position $(1,1)$ of T_A and T_B. For any choice of u_A and u_B the pencil

$$\mu\tilde{T}_A - \lambda\tilde{T}_B, \qquad \tilde{T}_A := T_A + e_1 u_A^T V, \quad \tilde{T}_B := T_B + e_1 u_B^T V$$

has the same same eigenvalues of $\mu T_A - \lambda T_B$ with the only exception of (λ_*, μ_*) which is moved to $(\lambda_* + u_A^T v, \mu_* + u_B^T v)$. Right multiplying (6) by V^{-1} after having replaced T_A and T_B with $\tilde{T}_A$ and $\tilde{T}_B$, respectively, yields

$$\mu\tilde{A} - \lambda\tilde{B} := W(\mu\tilde{T}_A - \lambda\tilde{T}_B)V^{-1}$$

which has the required eigenvalues by construction and is such that

$$\tilde{A} = A + wu_A^T, \qquad \tilde{B} = B + wu_B^T,$$

as requested. This completes the proof. □

Specializing the above result to eigenvalues of the form $(\lambda, 1)$ gives us the original Brauer's theorem from [8]. In our case, if ∞ is an eigenvalue of a pencil $A - \lambda B$ then $(\lambda, \mu) = (1, 0)$ is an eigenvalue of $\mu A - \lambda B$. Thus, we can choose

$$u_A = -\frac{v}{\|v\|^2}, \qquad u_B = \frac{v}{\|v\|^2}$$

so that the modified pencil has $(0, 1)$ as an eigenvalue. A simple generalization of the above result can be used to move an entire Jordan chain by perturbing it in the Kronecker canonical form. The proof is just more technical but uses the same ideas, so we omit it. The same result can be obtained by relying on the theorem in [5] twice, first moving the Jordan chain at infinity to some finite point and then moving it to zero.

Theorem 3.3. *Let $\mu A - \lambda B$ a pencil with a left and right deflating subspace spanned by the columns of W and V, that is there exist invertible $k \times k$ matrices M_A and M_B such that*

$$AV = WM_A, \qquad BV = WM_B.$$

Then, for any U_A, U_B in $\mathbb{C}^{n\times k}$ the modified pencil $\mu\tilde{A} - \lambda\tilde{B}$ with

$$\tilde{A} := A + WU_A^T, \qquad \tilde{B} := B + WU_B^T$$

has the same eigenstructure of $\mu A - \lambda B$ except the block corresponding to the deflating subspaces V and W, which is replaced by the eigenstructure of the (small) pencil $\mu(M_A + U_A^T V) - \lambda(M_B + U_B^T V)$.

The simplest case of a deflating subspace is to consider an eigenvector and its image under the multiplication by A and B, and this gives back Theorem 3.2. However, one might consider also a subspace spanned by the vectors of a Jordan chain and in this case the above result allows to move it to a completely different eigenstructure.

In view of the previous results, we assume the pencil $\tilde{A} - \lambda\tilde{B}$ has the infinite eigenvalue (and the Jordan chain associated, if it exists) shifted to 0. We can perform some iterations of the form

$$x^{(k+1)} = \tilde{B}^{-1}\tilde{A}x^{(k)}$$

in order to approximate the dominant finite eigenvalue. We can then use that approximation to select the initial approximations to start the EAI, by putting them equally distributed on a circle of radius equal to the spectral radius of the pencil.

In [17] it is shown that the linearizations of sums of rational functions only have 1 simple infinite eigenvalue, while the ones for sums of polynomials have an entire Jordan chain linked to infinity. For this reason, Theorem 3.2 is sufficient for the former case, while Theorem 3.3 is required for the latter. In both cases the explicit characterization of the Kronecker structure of the infinite eigenvalue allows to avoid its explicit computation.

3.1.2. Counting the eigenvalues by means of contour integration

Here we study a more refined version of the starting point selection procedure, which is based on the so-called *argument principle*. We recall its formulation from [15], for which we refer for the definition of a Jordan curve.

Theorem 3.4 (Argument principle, Theorem 4.10a in [15])**.** *Let $f(\lambda)$ a holomorphic function defined on a simply connected region R. Then, for any positively oriented Jordan curve Γ that borders in R and does not pass through any zero of $f(\lambda)$ we have*

$$\frac{1}{2\pi i}\int_{\Gamma}\frac{f'(\lambda)}{f(\lambda)}d\lambda = N$$

where N is the number of zeros of $f(\lambda)$ inside Γ, counted with multiplicities.

The above result applied to the holomorphic function $f(\lambda) := \det(A - \lambda B)$ allows to count the eigenvalues of the pencil $A - \lambda B$ inside a contour Γ.

Remark 3.5. The integrand of Theorem 3.4 is also called the *logarithmic derivative of* $f(\lambda)$. We notice that it is nothing else than the inverse of Newton's correction $f(\lambda)/f'(\lambda)$ evaluated at the point λ, according to (5). In the following we will show how to evaluate this function in $O(n)$ flops.

We propose the following strategy to count the roots inside a circle of center x_0 and radius $r > 0$. Let $I_k(x_0, r)$ be the approximation of the integral of Theorem 3.4 obtained by applying the trapezoidal rule with k points, and $B(x_0, r)$ the ball of center x_0 and radius r. We have

$$I(x_0, r) := \lim_{k\to\infty} I_k(x_0, r) = \frac{1}{2\pi i}\int_{\partial B(x_0,r)}\frac{f'(\lambda)}{f(\lambda)}d\lambda.$$

Since we are integrating a holomorphic function along a circle the trapezoidal rule converges exponentially fast to the integral thanks to the periodicity of the function [19] restricted to $\partial B(x_0, r)$. We choose k by means of the following procedure:

1. We evaluate the integrand at k points on the circle of center x_0 and radius r. We then compute $I_k(x_0, r)$ by appropriately combining the results of this evaluation.

2. We estimate the error by assuming $|I_k(x_0, r) - I_{2k}(x_0, r)| \approx |E_k(x_0, r)|$, where $E_k(x_0, r) := I(x_0, r) - I_k(x_0, r)$. If the absolute error is smaller than $\frac{1}{2}$ then we round the result to the nearest integer and exit, otherwise we go back to the first point doubling k.

3. We continue until convergence.

Notice that doubling the value of k allows to reuse the previous evaluations, so the cost for the integration will be $O(kn)$ where k is the minimum power of 2 such that the integration error can be bounded by $\frac{1}{2}$.

We can then use the above scheme to obtain an algorithm for the choice of the starting approximations. We first approximate the spectral radius by evaluating the number of eigenvalues in $B(0, 2^j)$ for various values of j. We find the smallest j such that all eigenvalues are contained inside $B(0, 2^j)$. Let it be j_2, and let j_1 the largest j such that $B(0, 2^j)$ does not contain any eigenvalue.

We then count the number of eigenvalues in each circle of radius 2^j for $j_1 < j < j_2$, and select the starting approximations accordingly. In our implementation we have chosen to place the approximations in each annulus $\{z \mid 2^j \leqslant |z| \leqslant 2^{j+1}\}$ on a circle of radius $\sqrt{2} \cdot 2^j$.

This strategy allows to match the moduli of the approximations to the ones of the eigenvalues. In order to complete this task one has to evaluate $r := j_2 - j_1 + 1$ integrals, plus the ones needed to find the spectral radius (that could be also computed with the scheme of the previous subsection).

We assume that the number of evaluations needed for each integral is bounded by n, in which case this will give a procedure that costs $O(rn^2)$. In particular, the two strategies for the choice of the starting points have a comparable cost.

In Figure 1 an example of starting points obtained with this strategy and the one of the previous section, along with the correct eigenvalues of the pencil, are displayed. The strategy relying on Theorem 3.4 is capable of estimating all the eigenvalues, not only the largest ones, and we will see in Section 5 that this yields a lower number of iterations for the EAI.

3.2. A suitable stopping criterion

When dealing with iterative methods it is important to understand when to stop. In order to take this decision we rely on some results of Henrici [15], and Bini and Noferini [6].

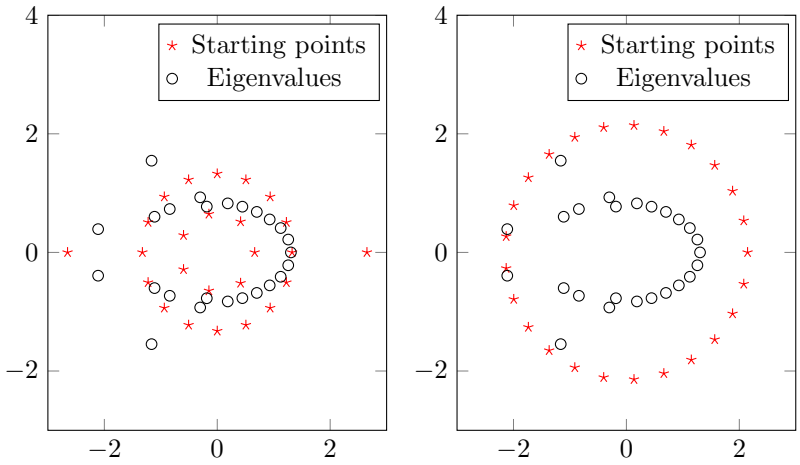

Figure 1: On the left: starting points generated with the algorithm relying on Theorem 3.4. The empty circles are the eigenvalues, while the stars represent the starting points computed with the above method. The radii have been chosen in the middle of the annuli containing a certain amount of eigenvalues. On the right: starting points generated relying on the power method.

3.2.1. Small Newton correction

The following result relates the modulus of Newton's correction with the accuracy of an approximation.

Theorem 3.6 (Corollary 6.4g of [15])**.** *Let $p(\lambda)$ be a polynomial of degree n. Then, for any λ such that $p'(\lambda) \neq 0$, the circle of radius $(n-1) \cdot |p(\lambda)/p'(\lambda)|$ and center λ contains at least one root of $p(\lambda)$.*

We can state the following immediate consequence of the above result, based on which we will formulate our stopping criterion.

Theorem 3.7. *Let $p(\lambda)$ be a polynomial of degree n and λ a point in the complex plane such that $|p(\lambda)/p'(\lambda)| \leqslant |\lambda|\epsilon$ for some $\epsilon > 0$. Then there exists a point ξ such that $p(\xi) = 0$ and $|\xi - \lambda| \leqslant (n-1)|\lambda|\epsilon$.*

The above states that whenever Newton's correction of $\det \mathcal{L}(\lambda)$ is of the order of the machine precision the point λ is nearby an eigenvalue of $\mathcal{L}(\lambda)$. Whenever this happens we can then stop our iteration, and this also automatically provides a bound on the forward error of the computed eigenvalue.

3.2.2. Checking the conditioning of the evaluated pencil

Another useful criterion to stop the iteration is checking the condition number of the matrix $A - \lambda B$ at a point λ. Since the pencil is singular whenever λ is an eigenvalue, we can expect the condition number $\kappa(A) := \|A\|\|A^{-1}\|$ to be high when λ is near an eigenvalue.

Intuitively, one could formulate a stopping condition by asking to stop the iterations when $\kappa(A - \lambda B) > t$ where t is some chosen threshold and $\kappa(\cdot)$ is the matrix condition number. Theorem 3.9 shows that in fact when we choose t to be approximately $\frac{1}{u}$, with u being the unit round-off, the above condition is equivalent to asking that λ is an eigenvalue of a slightly modified pencil.

Remark 3.8. We need to be careful with the definition of *slightly modified* in this context. In fact, what we would like to have is that a *structured* modification makes the pencil singular. Considering unstructured perturbations can cause the algorithm to stop too early since the unstructured condition number might be much higher than the structured one.

Here we state the following result, that gives a good stopping criterion for an unstructured pencil. Then we will rephrase it to make it applicable in our context so that structured perturbations can be considered instead. We note that this can be seen as a slight variation of Lemma 3 in [18], where $\kappa_2(\cdot)$ is used to denote the matrix condition number[4] with respect to the 2-norm.

Theorem 3.9. *Let $A - \lambda B$ a pencil. If $\kappa_2(A - \lambda B) \geqslant \frac{1}{\epsilon}$ then λ is an eigenvalue of a pencil whose coefficients have been perturbed relatively less than 2ϵ in norm.*

Proof. We need to prove that there exist two perturbations δA and δB, of norm relatively smaller than 2ϵ (compared to A and B, respectively), such that λ is an eigenvalue of $A + \delta A - \lambda(B + \delta B)$.

Recall that, in the 2-norm, $\kappa_2(A - \lambda B) = \frac{\sigma_1}{\sigma_n}$, where $\sigma_1 > \ldots > \sigma_n$ are the singular values of $A - \lambda B$. Let $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ be the associated left and right singular vectors. We then have that the matrix $A - \lambda B - \sigma_n u_n v_n^*$ is singular. Moreover, since $\|A - \lambda B\|_2 = \sigma_1$, either $\|A\|_2 \geqslant \frac{1}{2}\sigma_1$ or $\|B\|_2 \geqslant \frac{1}{2}\frac{\sigma_1}{|\lambda|}$. In the first case, we can define $\delta A := -\sigma_n u_n v_n^*$, and then we can verify that $A + \delta A - \lambda B$ is singular. In the second one, we can define $\delta B := \frac{\sigma_n}{\lambda} u_n v_n^*$, and then $A - \lambda(B + \delta B)$ is singular.

In both cases, the coefficients of the pencil $A - \lambda B$ can be perturbed with a perturbation relatively smaller than $\frac{2\sigma_n}{\sigma_1}$, so smaller than 2ϵ, so that λ is an eigenvalue. This concludes the proof. □

Notice that measuring the above condition number could be difficult in practice. However, as already mentioned in the previous remark, we are more interested in a structured condition number which is also easier to measure in our context.

Theorem 3.10. *Consider an invertible upper triangular matrix with the fol-*

[4] Here we refer to the standard condition number of the linear system associated to a certain matrix, that is, $\kappa_2(A) := \|A\|_2 \cdot \|A^{-1}\|_2$.

lowing structure:

$$M = \begin{array}{ccccccc} \times & \times & \star & \star & \star & \star & \\ & \times & \times & \star & \star & \star & \star \\ & & \times & \star & \star & \star & \star \\ & & & \gamma & \star & \star & \star \\ & & & & \times & \times & \\ & & & & & \times & \times \\ & & & & & & \times \end{array},$$

where the entries marked with $\star$ and γ form a block of rank k which can be written as UV^T for $U, V \in \mathbb{C}^{n\times k}$. Then there exists a perturbation of norm smaller than $\frac{|\gamma|}{\sqrt{\|e_n^T U\|^2 + \|e_1^T V\|^2}}$, at the first order, of U and V which makes the matrix singular.

Proof. Define the $n \times k$ matrices $\tilde{U}, \tilde{V}$ as follows:

$$\tilde{U} := U + \delta U, \qquad \tilde{V} := V + \delta V$$

Since the only element on the diagonal modified by changing δU and δV is γ, the only way to obtain a singular matrix is to choose them so that

$$e_n^T (U + \delta U)(V + \delta V)^T e_1 = 0. \tag{7}$$

The above shows that all the entries in δU and δV which are not on the last row and first row, respectively, have no effect on the singularity, thus we can set them to zero (since any other choice will increase the norm of the perturbations). Let $u, \delta u, v$ and δv be the last and first rows of these matrices. We can rephrase (7) as follows:

$$\langle u + \delta u, v + \delta v \rangle = 0.$$

Dropping second order terms one can verify that this is equivalent to finding the minimum norm solution to

$$\begin{bmatrix} u \\ v \end{bmatrix}^T \begin{bmatrix} \delta v \\ \delta u \end{bmatrix} = -\gamma$$

which is given by

$$\begin{bmatrix} \delta v \\ \delta u \end{bmatrix} = -\frac{\gamma}{\|u\|^2 + \|v\|^2} \begin{bmatrix} u \\ v \end{bmatrix} = \frac{-\gamma}{\sqrt{\|u\|^2 + \|v\|^2}} \left(\frac{1}{\sqrt{\|u\|^2 + \|v\|^2}} \begin{bmatrix} u \\ v \end{bmatrix} \right).$$

where we have normalized the vector $\left[\begin{smallmatrix} u \\ v \end{smallmatrix}\right]$ to make it of unit norm. The proof is then concluded since the size of the perturbation is exactly what we were aiming for. □

Using the above result we can say that if

$$\frac{|\gamma|}{\sqrt{\|u\|^2 + \|v\|^2}} \leqslant \epsilon \|UV^T\|$$

then a *structured* perturbation which is relatively smaller than ϵ can make the evaluated pencil singular. We will see in Section 4.3 that the matrix can be taken in this upper triangular form by means of Givens rotations. This is used to compute Newton's correction, so then we can easily check when we have reached convergence by testing whether $|\gamma| \leqslant Ku\|UV^T\|\sqrt{\|e_n^T U\|^2 + \|e_1^T V\|^2}$, where K is a small constant, depending also on the norm of U and V, and u the unit round-off. Since all these quantities are available during the computation of Newton's correction this condition can be checked almost for free, and provides an effective stopping criterion.

4. Efficient computation of Newton's correction

In this section we show how the previous results can be turned into a practical algorithm. The main issue is the efficient evaluation of Newton's correction at a point, which corresponds to computing the trace of the matrix $(A-\lambda B)^{-1}B$. In this section we present a strategy that works both for the Newton and Lagrange linearizations, with some specific results that only cover the Newton case.

4.1. Transformation into Hessenberg structure

As we have seen in Section 2.1 and 2.2, the linearizations that we are interested in have the following form:

$$\mathcal{L}(\lambda) = \begin{bmatrix} R & L_1^T(\lambda) \\ L_2(\lambda) & 0 \end{bmatrix} \tag{8}$$

with $L_j(\lambda), j = 1, 2$ being rectangular $k_j \times (k_j + 1)$ and upper bidiagonal and R being a rank 2 matrix. Without loss of generality, in the following we assume that $L_1(\lambda)$ and $L_2(\lambda)$ have the same size $k \times (k+1)$ and $R = UV^T$ with $U, V \in \mathbb{C}^{(k+1)\times 2}$.

Theorem 4.1. *Let $\mathcal{L}(\lambda)$ be a pencil as in (8). Then there exists a block column permutation that takes it to upper Hessenberg form. More precisely, we have that*

$$\mathcal{L}(\lambda)\Pi = \begin{bmatrix} L_1^T(\lambda) & R \\ 0 & L_2(\lambda) \end{bmatrix} =: \tilde{A} - \lambda\tilde{B}, \qquad \Pi = \begin{bmatrix} & I_{k+1} \\ I_k & \end{bmatrix}$$

is an upper Hessenberg pencil. Moreover, its leading coefficient is lower bidiagonal with a zero element on the diagonal in position $(k+1, k+1)$, and the constant coefficient is the sum of a bidiagonal matrix with an upper triangular rank 2 matrix.

Proof. Direct consequence of applying Π to the pencils defined in Sections 2.1 and 2.2. □

Something more can be said in the Newton case, where the leading coefficient is diagonal. Using an additional permutation, the pencil $\mathcal{L}(\lambda)$ can be endowed with an Hessenberg-Triangular structure. This is relevant if one wants to apply the QZ iteration, since the reduction to upper Hessenberg-Triangular form is

the usual preliminary step in this case. While this is not directly relevant for the EA approach, it is still a reduction that is interesting so we state the following result.

Lemma 4.2. *Let $\mathcal{L}(\lambda)$ the pencil obtained by linearizing the sum (or difference) of two polynomials expressed in the Newton basis. Then there exist two permutation matrices Π_1 and Π_2 such that*

$$\Pi_1 \mathcal{L}(\lambda) \Pi_2 = A - \lambda B,$$

with B diagonal and A upper Hessenberg.

Proof. We already know, thanks to Theorem 4.1, that we can choose $\Pi_{2,1}$ so that the pencil $\mathcal{L}(\lambda)\Pi_{2,1}$ is upper Hessenberg. Let J_k, $\Pi_{1,1}$ and $\Pi_{1,2}$ be defined as follows:

$$J_k = \begin{bmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{bmatrix}, \qquad \Pi_{1,1} = J_{k_1+1} \oplus I_{k_2}, \qquad \Pi_{1,2} = J_{k_1} \oplus I_{k_2+1}.$$

Multiplying $\mathcal{L}(\lambda)\Pi_{2,1}$ on the left by $\Pi_{1,1}$ acts on the first block row as the left multiplication by J_{k_1+1} and, analogously, the right multiplication by $\Pi_{1,2}$ acts on the right as J_{k_1}. These transformations preserve the rank of the top-right block and leave $L_2(\lambda)$ unchanged. Moreover, in the Newton case, $L_1(\lambda)^T$ is given by

$$L_1(\lambda)^T = H - \lambda \begin{bmatrix} 0_{k_1}^T \\ I_{k_1} \end{bmatrix}$$

where H is lower bidiagonal. It can be checked easily that $J_{k_1+1} H J_{k_1}$ is still lower bidiagonal and that

$$J_{k_1+1} L_1(\lambda)^T \; J_{k_1} = J_{k_1+1} H J_{k_1} - \lambda \begin{bmatrix} I_{k_1} \\ 0_{k_1}^T \end{bmatrix}$$

has the prescribed Hessenberg triangular structure when embedded in the larger pencil. Setting $\Pi_1 := \Pi_{1,1}$ and $\Pi_2 := \Pi_{2,1}\Pi_{1,2}$ completes the proof. □

4.2. A Sherman-Morrison based approach

In this section we focus on providing a method involving $O(n)$ flops for computing the trace of $(A - \lambda B)^{-1}B$, i.e., for the evaluation of the Newton correction of the polynomial $\det \mathcal{L}(\lambda)$. The method is based on the Sherman-Morrison formula [14].

Theorem 4.3 (Sherman-Morrison)**.** *Let M and $M + UV^T$ be two invertible matrices, where $M \in \mathbb{C}^{n\times n}$ and $U, V \in \mathbb{C}^{n\times k}$. Then*

$$(M + UV^T)^{-1} = M^{-1} - M^{-1}U(I + V^T M^{-1} U)^{-1} V^T M^{-1}.$$

The above formula provides a cheap method to evaluate the inverse of a low rank correction of a matrix whose inverse is known (or easily computable). This is exactly our case, since the pencil $\mathcal{L}(\lambda)$ can be written in the following form:

$$A - \lambda B = M(\lambda) + UV^T$$

where $M(\lambda)$ is a lower bidiagonal pencil and $U, V \in \mathbb{C}^{n\times 2}$. Unfortunately, the above decomposition does not satisfy the hypotheses of Theorem 4.3, since the bidiagonal matrix $M(\lambda)$ has a zero diagonal entry (see Theorem 4.1) and is not invertible.

However, we can rephrase the decomposition by modifying $M(\lambda)$ and putting a value $\alpha \neq 0$ in position $(k+1, k+1)$ and accordingly modify the rank 2 correction to a rank 3 one so that

$$A - \lambda B = \tilde{M}(\lambda) + UV^T - \alpha e_{k+1} e_{k+1}^T = \tilde{M}(\lambda) + \tilde{U}\tilde{V}^T.$$

In the above formulation the matrix $\tilde{M}(\lambda)$ is invertible and by the Sherman-Morrison formula we obtain:

$$(A - \lambda B)^{-1} = \tilde{M}(\lambda)^{-1} - \tilde{M}(\lambda)^{-1}\tilde{U}(I + \tilde{V}^T\tilde{M}(\lambda)^{-1}\tilde{U})^{-1}\tilde{V}^T\tilde{M}(\lambda)^{-1}, \quad (9)$$

which in turn leads to the following result.

Lemma 4.4. *Let $A - \lambda B$ be a pencil defined as in Theorem 4.1. Then, for any λ such that $A - \lambda B$ is invertible and for any $\alpha \neq 0$,*

$$\operatorname{tr}((A - \lambda B)^{-1}B) = \operatorname{tr}(\tilde{M}(\lambda)^{-1}B) - \operatorname{tr}(\hat{V}^T(\lambda)\hat{U}(\lambda))$$

where $\tilde{M}(\lambda), \tilde{U}(\lambda), \tilde{V}(\lambda)$ are defined as in (9) and

$$\hat{U}(\lambda) := \tilde{M}(\lambda)^{-1}\tilde{U}(I + \tilde{V}^T\tilde{M}(\lambda)^{-1}\tilde{U})^{-1}, \qquad \hat{V}(\lambda) = B^T\tilde{M}(\lambda)^{-T}\tilde{V}.$$

Proof. We can use the decomposition of (9) to get:

$$(A - \lambda B)^{-1}B = \tilde{M}(\lambda)^{-1}B - \tilde{M}(\lambda)^{-1}\tilde{U}(I + \tilde{V}^T\tilde{M}(\lambda)^{-1}\tilde{U})^{-1}\tilde{V}^T\tilde{M}(\lambda)^{-1}B.$$

Since the trace is a linear operator, we can split the trace of this sum as the sum of the traces, and using the fact that the trace of a matrix product is invariant under cyclic permutation of the factors we get the thesis. □

The trace of a matrix product can be characterized as follows.

Lemma 4.5. *Let M, N be two $n \times k$ matrices. Then*

$$\operatorname{tr}(MN^T) = \sum_{0\leqslant i,j\leqslant n} (M \circ N)_{ij}$$

where $\circ$ denotes the Hadamard product.

Remark 4.6. We emphasize that Lemma 4.4 provides an $O(n)$ algorithm for computing Newton's correction. In fact, to evaluate the first term of the sum we can use the relation given by Lemma 4.5:

$$\operatorname{tr}(\tilde{M}(\lambda)^{-1}B) = \sum_{i,j}(\tilde{M}(\lambda)^{-1} \circ B^T)_{ij}.$$

Since B^T has only nonzero elements on the diagonal and on the superdiagonal we have to compute the diagonal and superdiagonal of $\tilde{M}(\lambda)^{-1}$, which can be done in $O(n)$ flops given its bidiagonal structure.

Moreover, the second matrix of which we have to compute the trace is 3×3 and can be computed in $O(n)$ flops. These two facts together provide an $O(n)$ algorithm.

Whilst the above framework is theoretically satisfying, from a numerical perspective there are still some points that need to be handled carefully. A natural one is the choice of α. While any $\alpha \neq 0$ provides a mathematically correct formula, we are interested in choosing α in order to obtain the best possible numerical results. In practice we can choose α to be about the norm of the other diagonal elements, in order to avoid unbalancing in the matrix.

4.3. Using rotations

As we will see in Section 5 the algorithm of Section 4.2 can be unstable. For this reason, it is of interest to devise an alternative scheme based on unitary transformations that, as confirmed by numerical experiments in Section 5, is more robust in practice.

In view of Lemma 4.2 we know that, up to permutations, we can rewrite the pencil as $A - \lambda B$ where A and B have the following structure:

$$A = \begin{bmatrix} B_\phi^T & UV^T \\ 0 & B_\psi \end{bmatrix}, \qquad B = \begin{bmatrix} -B_{\phi,1}^T & \\ & -B_{\psi,1} \end{bmatrix},$$

where B_ϕ and B_ψ are (rectangular) bidiagonal matrices containing the interpolation nodes. The Newton case is particularly easy to deal with, since the matrix B is diagonal, with a zero entry in the middle. We have the following.

Lemma 4.7. *Let $A - \lambda B$ a linearization for a sum of two scalar polynomials expressed in two Newton bases as in (3). Then the trace of the matrix $(A - \lambda B)^{-1}B$ can be expressed as follows:*

$$\operatorname{tr}\left((A - \lambda B)^{-1}B\right) = \left(\sum_{\substack{i=1 \\ i \neq k+1}}^{n} [A - \lambda B]_{ii}^{-1} \right)^{-1}$$

where k is the degree of the polynomials whose sum is linearized.

Proof. It follows by recalling that $\operatorname{tr}(AB^T) = \sum_{i,j}(A \circ B)_{ij}$, where $\circ$ is the Hadamard product of the matrices A and B, see Lemma 4.5. □

An analogous result (which we do not state explicitly) also holds for the Lagrange case, where the linear combination of the diagonal and superdiagonal elements has to be done using the barycentric weights as coefficients.

In both cases, to ease the computation, we will split the inverse of $A-\lambda B$ in two parts. The linearity of the trace operator allows to compute these two parts separately and then sum the results. More precisely, we look for a decomposition $(A-\lambda B)^{-1} = M_1 + M_2$, so that we can compute $\mathrm{tr}((A-\lambda B^{-1})B) = \mathrm{tr}(M_1 B) + \mathrm{tr}(M_2 B)$. We rely on the following elementary result.

Lemma 4.8. *Let X be an upper bidiagonal matrix, defined as follows:*

$$X = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ & \ddots & \ddots & \\ & & \ddots & \beta_{n-1} \\ & & & \alpha_n \end{bmatrix}.$$

Then it admits a factorization as a sequence of $n-1$ Gauss transformations given by $X = X_{n-1} \dots X_1$ where

$$X_1 = \begin{bmatrix} \alpha_1 & \beta_1 & \\ & \alpha_2 & \\ & & I_{n-2} \end{bmatrix} \text{ and } X_i = \begin{bmatrix} I_{i-1} & & & \\ & 1 & \beta_i & \\ & & \alpha_{i+1} & \\ & & & I_{n-i-1} \end{bmatrix} \text{ for } i > 1.$$

Assume λ fixed and set $M := A - \lambda B$. We want to compute the elements of M^{-1}. Under the hypotheses above we have:

$$M = \begin{bmatrix} X_\phi^T & UV^T \\ 0 & X_\psi \end{bmatrix}, \qquad X_\phi \in \mathbb{C}^{k\times(k+1)},\ X_\psi \in \mathbb{C}^{k\times(k+1)},$$

where X_ϕ and X_ψ are the bidiagonal matrices relative to the nodes in the bases ϕ and ψ, respectively. As reported by the following lemma, the above structure allows for a structured upper triangular factorization of M.

Lemma 4.9. *Given a matrix M with the prescribed structure, it is possible to find two unitary matrices Q_U and Q_L such that*

$$R := Q_U M Q_L = \begin{bmatrix} \tilde{X}_\phi & \tilde{U}x_1 & \tilde{U}\tilde{V}^T \\ & \gamma & x_2^T \tilde{V}^T \\ & & \tilde{X}_\psi \end{bmatrix}, \quad \tilde{X}_\phi \in \mathbb{C}^{k\times k},\ \tilde{X}_\psi \in \mathbb{C}^{k\times k},\ x_1, x_2 \in \mathbb{C}^2.$$

and $\tilde{X}_\phi$ and $\tilde{X}_\psi$ are upper bidiagonal matrices. Moreover, $Q_U = Q_{U,S} \oplus I_k$ and $Q_L = I_k \oplus Q_{L,S}$ and $Q_{U,S}$ and $Q_{L,S}$ can be decomposed as the product of k Givens rotations. The matrices $\tilde{U}$ and $\tilde{V}$ are defined by $\tilde{U} = Q_{U,S}U$ and $\tilde{V} = Q_{L,S}V$.

Proof. The proof of the above result is constructive. We define $2k$ Givens rotations that reduce the top-left and bottom-right blocks to upper triangular form as reported in the following for the $k = 3$ case:

$$M = \underbrace{G_1 G_2 G_3}_{Q_U^\star} \begin{bmatrix} \times & \times & & \star & \star & \star & \star \\ & \times & \times & \star & \star & \star & \star \\ & & \times & \star & \star & \star & \star \\ & & & \star & \star & \star & \star \\ & & & & \times & \times & \\ & & & & & \times & \times \\ & & & & & & \times \end{bmatrix} \underbrace{G_4 G_5 G_6}_{Q_L^\star}$$

where the $\times$-es identify the entries of the bidiagonal blocks , the $\star$ are the entries of the low rank block, and G_i is a Givens rotation acting on the rows $(i, i+1)$. The rotations can be obtained computing a QR factorization of X_ϕ and an RQ factorization of X_ψ. □

The advantage of the above representation is that it eases the parametrization of the inverse of R in order compute its trace (even after performing the rotations). In fact we have the following.

Lemma 4.10. *The inverse of R is given by*

$$R^{-1} = \begin{bmatrix} \tilde{X}_\phi^{-1} & -\gamma^{-1}\tilde{X}_\phi^{-1}\tilde{U}x_1 & \times \\ & \gamma^{-1} & -\gamma^{-1}x_2^T\tilde{V}^T\tilde{X}_\psi^{-1} \\ & & \tilde{X}_\psi^{-1} \end{bmatrix}.$$

Moreover, the trace of $M^{-1} = Q_L R^{-1} Q_U$ does not depend on the entries that have been marked with the $\times$ symbol.

Proof. The structure of the inverse matrix can be obtained by performing a block-wise inversion of the upper triangular matrix R. The last claim can be obtained by decomposing R^{-1} as $R^{-1} = R_\times + R_{\times^\perp}$, where $R_\times$ contains the elements marked with $\times$ and $R_{\times^\perp}$ the others. The structure if Q_L and Q_U implies that $Q_L R_\times Q_U$ has a zero diagonal, thus giving a null contribution to the trace.

□

By exploiting the last statement of Lemma 4.10 and the linearity of the trace operator, we can rephrase the problem as follows for the Newton case.

Lemma 4.11. *The trace of $M^{-1}B$, where M and B have been built starting from a linearization in the Newton basis, can be written as*

$$\begin{aligned} \operatorname{tr}(M^{-1}B) = &- \operatorname{tr}(\tilde{X}_\phi^{-1} Q_{U,S}) - \operatorname{tr}(Q_{L,S}\tilde{X}_\psi^{-1}) \\ &+ \frac{1}{\gamma}\operatorname{tr}(e_{k+1}^T Q_{U,S} X_\phi^{-1}\tilde{U}x_1) + \frac{1}{\gamma}\operatorname{tr}(x_2^T\tilde{V}^T\tilde{X}_\psi^{-1}Q_{L,S}e_1) \end{aligned}$$

All these summands can be computed in $O(n)$ flops.

The above results allow to devise an $O(n)$ method to evaluate the Newton correction of $\det \mathcal{L}(\lambda)$ at any point in the complex plane.

Remark 4.12. The computation of the Q_L, Q_U and the inversion of the upper triangular matrix, can be all performed by means of backward stable operations. Moreover, given the structure of $A - \lambda B$, all the errors are offloaded either on the nodes or on the low rank part which contains the coefficients of the polynomials. This suggests that the procedure for the computation of Newton's correction is structurally backward stable, with respect to the bidiagonal plus low rank structure. In fact, the final result is the exact one obtained by slight perturbations on the nodes and on the coefficients. As we will see in the numerical experiments, this leads to a better accuracy with respect to non-structured backward stable methods, like the QZ algorithm.

5. Numerical experiments

In this section we report the numerical experiments that validate our approach. We have tested two different aspects of the algorithm: the accuracy and the asymptotic cost.

Regarding the former, we verified that in many common cases EAI delivers very accurate results. Moreover, we show that it easily overcomes the problems related to poor conditioning of the eigenvalues when considering the *unstructured condition number* of the eigenvalue problem.

5.1. Accuracy of the method

We consider the problem of finding the roots of a polynomial $r(\lambda)$ described as $r(\lambda) = p_1(\lambda) - p_2(\lambda)$, with $p_1(\lambda)$ and $p_2(\lambda)$ expressed in the Newton basis. As nodes for these two interpolation polynomials we have chosen the Chebyshev points, in order to have a set of points where the interpolation is reasonably conditioned. We have computed $2k$ nodes and we have used k of them to generate the basis for $p_1(\lambda)$ and k of them to build the basis for $p_2(\lambda)$, so they are expressed in a different basis. We have ordered the set of $2k$ nodes according to the canonical ordering on $\mathbb{R}$ and we have assigned the ones in the odd positions to the first interpolation basis, and the ones in the even position to the other, as depicted in Figure 2. The same kind of splitting has been used for the roots of unity, which have been employed for the numerical experiments in the Lagrange case reported in Table 3 (in this case they have been ordered by their angle).

In Table 1 we have reported the absolute forward errors[5] and the backward errors (on the matrix pencil) for the approximation of the roots using the Sherman-Morrison based strategy and the one based on Givens rotations. More precisely, we have computed the backward error $err_{(A,B)}(\lambda)$ for each eigenvalue

[5] Approximations for the roots with an arbitrary number of digits have been obtained using MPSolve [7], a multiprecision polynomial solver. The symbolic toolbox of MATLAB to compute the coefficients of the linearized polynomial.

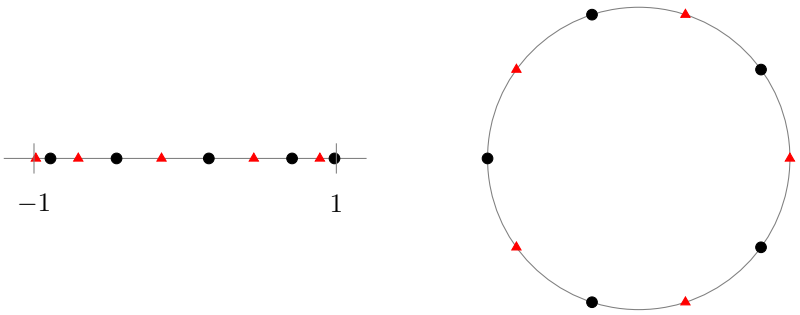

Figure 2: On the left, the splitting used to assign the $2k$ Chebyshev nodes to the first and second family of nodes, used for $p_1(\lambda)$ and $p_2(\lambda)$, respectively, is reported. On the right, the same splitting for the roots of unity is shown.

defined as $err_{(A,B)}(\lambda) := \sigma_n(A - \lambda B)$, where $\sigma_n(\cdot)$ is the smallest singular value. This can be proven to be the distance (in the Euclidean norm) to the closest pencil that has λ as an eigenvalue. We refer to the work of Tisseur [18] for a detailed error analysis.

It is clearly visible that the strategy based on rotations does not have stability issues, while the accuracy of the one based on Sherman-Morrison soon degrades as the degree increases. For this reason, in the following we will always consider the strategy based on rotations. The numbers reported are the norms of the vectors containing the errors for each approximation. For the examples that we have chosen there is not much difference between absolute and relative errors since most of the roots have modulus about 1.

Degree	Forward SM	Forward Rot	Backward SM	Backward Rot
2	$2.14 \cdot 10^{-16}$	$1.87 \cdot 10^{-16}$	$6.11 \cdot 10^{-17}$	$5.18 \cdot 10^{-17}$
5	$2.06 \cdot 10^{-15}$	$1.38 \cdot 10^{-16}$	$4.54 \cdot 10^{-16}$	$6.76 \cdot 10^{-17}$
10	$1.83 \cdot 10^{-13}$	$1.58 \cdot 10^{-16}$	$1.05 \cdot 10^{-14}$	$5.66 \cdot 10^{-17}$
15	$5.68 \cdot 10^{-11}$	$1.23 \cdot 10^{-16}$	$9.3 \cdot 10^{-12}$	$3.69 \cdot 10^{-17}$
20	$4.01 \cdot 10^{-6}$	$1.17 \cdot 10^{-16}$	$3.57 \cdot 10^{-8}$	$4.22 \cdot 10^{-17}$

Table 1: Comparison of the accuracies of the two strategies for the computation of Newton's correction. The columns marked with SM represents the data relative to the Sherman-Morrison based approach of Section 4.2, while the ones marked with Rot refer to the strategy based on Givens rotations of Section 4.3.

In Table 2 we have reported both absolute forward errors and backward errors (on the matrix pencil) for a wider range of degrees, and we have compared it with the QZ algorithm. However, the degradation in the quality of the approximations given by the QZ iteration is clearly visible. This is due to the fact that while giving backward stable results, they are backward stable in an unstructured sense, and they are not guaranteed to correspond to small perturbations in the polynomials. Since the EAI iteration relies on a structured

Degree	Forward EAI	Forward QZ	Backward EAI	Backward QZ
10	$5.1 \cdot 10^{-16}$	$3.64 \cdot 10^{-15}$	$1.02 \cdot 10^{-16}$	$1.43 \cdot 10^{-16}$
20	$5.2 \cdot 10^{-16}$	$5.65 \cdot 10^{-14}$	$1.55 \cdot 10^{-16}$	$1.94 \cdot 10^{-16}$
40	$7.96 \cdot 10^{-16}$	$3.59 \cdot 10^{-10}$	$2.33 \cdot 10^{-16}$	$2.66 \cdot 10^{-16}$
80	$5.93 \cdot 10^{-16}$	0.35	$3.38 \cdot 10^{-16}$	$4.35 \cdot 10^{-16}$
160	$1.41 \cdot 10^{-15}$	1.09	$4.62 \cdot 10^{-16}$	$6.71 \cdot 10^{-16}$

Table 2: Numerical accuracy of the EAI compared to the QZ iteration. We have generated 50 examples of sums of polynomials whose coefficients in the Newton basis are drawn by Gaussian distribution coefficients. The nodes of the Newton bases are Chebyshev points. The infinite eigenvalues in the QZ methods have been deflated a posteriori — and have always been exactly identified by the QZ method. In this cases a posteriori deflation is easy because of the special structure that the linearization has for degree-graded bases. This is not the case in general. The accuracies have been averaged over all the experiments. The backward error reported in the table is the one on the matrix pencil.

Degree	Forward EAI	Forward QZ	Backward EAI	Backward QZ
5	$7.25 \cdot 10^{-16}$	$2.15 \cdot 10^{-15}$	$1.35 \cdot 10^{-16}$	$2.21 \cdot 10^{-16}$
10	$5.85 \cdot 10^{-16}$	$1.68 \cdot 10^{-15}$	$1.01 \cdot 10^{-16}$	$2.33 \cdot 10^{-16}$
20	$1.52 \cdot 10^{-14}$	$2.69 \cdot 10^{-14}$	$8.03 \cdot 10^{-17}$	$2.02 \cdot 10^{-16}$
40	$1.58 \cdot 10^{-15}$	$1.22 \cdot 10^{-14}$	$4.82 \cdot 10^{-17}$	$8.37 \cdot 10^{-17}$
80	$6.8 \cdot 10^{-15}$	$6.78 \cdot 10^{-14}$	$2.99 \cdot 10^{-17}$	$4.56 \cdot 10^{-17}$

Table 3: Numerical accuracy of the EAI compared to the QZ iteration for sums of rational functions defined by ratios of Lagrange polynomials. The accuracies have been averaged over 10 runs, and the nodes have been chosen with interlacing properties as in the Newton example of Table 2 from the roots of unity of appropriate degree.

(and backward stable) solver to compute the Newton correction, evaluating a slightly perturbed *polynomial*, it leads to much better results in practice.

5.2. Asymptotic cost of the method

The speed of convergence of the EAI is strictly related to the quality of the starting approximations. In Section 3.1 we have discussed possible choices for the starting points, and here we study how these relate to the number of iterations before the stopping criterion presented in Section 3.2 is met on all the components.

In particular, we are interested in studying the *average number of iterations* per eigenvalue. Since an iteration costs $O(n)$ flops, keeping this number bounded by a constant makes the asymptotic cost $O(n^2)$.

More generally, assuming an instance of EAI has an average number of iterations equal to $t > 0$, we have a total cost for the algorithm of $O(tn^2)$. Our aim is to choose the starting points that make t as small as possible. The results in Figure 3 show that good starting points produce a very slow growth in the number of iteration, thus providing a practically quadratic method.

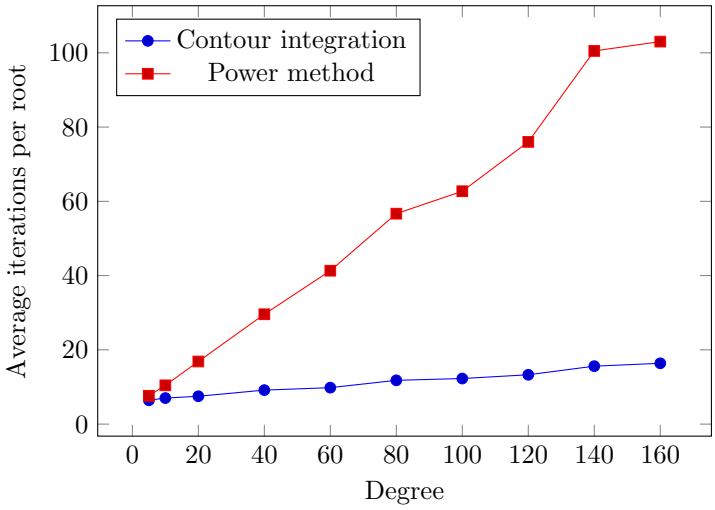

Figure 3: Average number of iterations for different choices of starting points. The tests refer to the computation of the roots of the sum of two polynomial expressed in the Newton basis with interlaced Chebyshev nodes as described in Figure 2.

Degree	Integration	Power method
5	6.42	7.62
10	7.02	10.45
20	7.5	16.86
40	9.16	29.59
60	9.82	41.32
80	11.79	56.67
100	12.28	62.72
120	13.29	76.01
140	15.6	100.51
160	16.38	103.02

Table 4: Average number of iterations with different criterion for the choice of the starting points.

To estimate the value of t we have run the following procedure:

1. We have randomly generated a sequence of rational functions, for various degrees from $n = 10$ up to $n = 160$ (here by degree we mean the degree of the numerator and the denominator). We have chosen the same kind of Newton basis for all of them and we have drawn random coefficients from a Gaussian distribution $N(0, 1)$.

2. We have run the EAI on these problems. 50 problems with the same degree have been tested and we have computed the average number of iterations for each degree.

The results of these tests are reported in Table 4 and in Figure 3. We have tested the two methods for the choice of the starting points that have been discussed in Section 3.1, that is the adapted power method and the integral approach to counting the number of eigenvalues inside a closed curve. Both methods manage to deliver the starting points in (at most) $O(n^2)$ flops, so they do not significantly contribute to the total complexity of the method. More precisely, we have fixed the number of integration points or iterations of the power method to be bounded by n, so that we have a guaranteed $O(n^2)$ complexity for the computation of the starting points.

Figure 3 shows how, as we have already stressed, even if the contour integration method still exhibits some growth in the average number of iteration as n grows, this effect is very mitigated compared to taking points on a circle of large enough radius.

The degraded performance of putting all the initial approximations on a circle with radius equal to the spectral radius of the pencil (ignoring infinite eigenvalues) can be informally explained by the fact that the approximation have to travel a long distance to reach the roots with smaller modulus.

5.3. Eigenvalues of matrix polynomials

To complete the section we show an application to the computation of eigenvalues of matrix polynomials and rational functions. More precisely, we consider the nonlinear eigenvalue problem

$$R(\lambda)v = 0, \qquad R(\lambda) := P_1(\lambda)^{-1}Q_1(\lambda) + P_2(\lambda)Q_2(\lambda)^{-1},$$

where as usual the matrix polynomials $P_1(\lambda)$ and $Q_1(\lambda)$ are expressed in a certain basis, and $P_2(\lambda)$ and $Q_2(\lambda)$ in another one. In this case we assume that they are both Newton bases, with different nodes.

The same approach of Section 4.3 can be used to evaluate the trace of the linearization of such a nonlinear eigenvalue problem at a certain point in the complex plane. Assuming the degree of all the matrix polynomials involved is d one can reduce the diagonal blocks to upper block bidiagonal form with $O(d)$ block Givens rotations, and then compute the inverse of the resulting block upper triangular matrix.

Degree	Forward EAI	Forward QZ	Backward EAI	Backward QZ
2	$1.63 \cdot 10^{-15}$	$1.04 \cdot 10^{-13}$	$1.02 \cdot 10^{-18}$	$2.4 \cdot 10^{-18}$
4	$6.94 \cdot 10^{-15}$	$1.37 \cdot 10^{-13}$	$1.11 \cdot 10^{-18}$	$1.93 \cdot 10^{-18}$
6	$2.21 \cdot 10^{-15}$	$2.05 \cdot 10^{-13}$	$7.77 \cdot 10^{-19}$	$1.61 \cdot 10^{-18}$
8	$1.62 \cdot 10^{-15}$	$3.26 \cdot 10^{-13}$	$5.77 \cdot 10^{-19}$	$1.2 \cdot 10^{-18}$
10	$9.39 \cdot 10^{-16}$	$2.57 \cdot 10^{-13}$	$7.1 \cdot 10^{-19}$	$1.05 \cdot 10^{-18}$
12	$5.2 \cdot 10^{-16}$	$3.43 \cdot 10^{-13}$	$4.98 \cdot 10^{-19}$	$7.79 \cdot 10^{-19}$
14	$1.06 \cdot 10^{-15}$	$3.29 \cdot 10^{-13}$	$5.16 \cdot 10^{-19}$	$8.17 \cdot 10^{-19}$
16	$3.79 \cdot 10^{-15}$	$5.07 \cdot 10^{-13}$	$5.45 \cdot 10^{-19}$	$9.03 \cdot 10^{-19}$
18	$8.42 \cdot 10^{-15}$	$7.12 \cdot 10^{-13}$	$7.09 \cdot 10^{-19}$	$2.19 \cdot 10^{-18}$
20	$1.77 \cdot 10^{-15}$	$8.62 \cdot 10^{-13}$	$4.51 \cdot 10^{-19}$	$8.61 \cdot 10^{-19}$
22	$8.02 \cdot 10^{-16}$	$1.88 \cdot 10^{-12}$	$4.34 \cdot 10^{-19}$	$6.06 \cdot 10^{-19}$
24	$3.28 \cdot 10^{-15}$	$7.23 \cdot 10^{-12}$	$4.96 \cdot 10^{-19}$	$6.76 \cdot 10^{-19}$

Table 5: Numerical accuracy of the EAI compared to the QZ algorithm in the computation of the eigenvalues of a nonlinear eigenvalue problem expressed as a sum of two rational functions in the Newton basis. The nodes of the Newton bases are Chebyshev points.

The cost of each evaluation of Newton's correction is cubic in the size of the coefficients, leading to a total computational cost of $O((dn) \cdot dn^3) = O(dn^4)$, so this approach is convenient only if the degree is large enough. We have compared the results obtained using the EA iteration to the QZ on the pencil, and also in this case one notices that the (forward) accuracy of the EA is much better than the one of the QZ. However, both algorithms deliver backward stable approximations, as reported in Table 5.

The coefficients of the matrix polynomials in this example are random 6×6 matrices with integer entries between -1000 and 1000. This setup has been chosen to allow the computation of the eigenvalues symbolically in order to check the computed results. The backward error computed (which is relative to the norms of the pencil) is always below the machine precision, and the results of the QZ algorithm show that the (unstructured) eigenvalue condition number of the pencil is still quite high compared to the structured one (that is, the one of the original problem).

6. Conclusions

We have shown the effectiveness of the Ehrlich–Aberth iteration as an approximation engine for the eigenvalues of some rank structured pencils which are encountered when linearizing sums of polynomials and rational functions expressed in Newton Lagrange bases. Our approach allows to treat a broad set of problems, such as (matrix) polynomials and rational functions expressed as sums in different bases.

This work has shown that the method is both fast, in the sense of having a lower asymptotic complexity than the QR and the QZ iterations, and more

accurate when looking at the forward errors. The gain is obtained by applying a *structured* solver that only allows perturbations on the original input data. Moreover, we have shown that the deflation of infinite eigenvalue is not an issue in this context, simplifying the analysis. Thus, even when some of the eigenvalues are ill-conditioned in the pencil no loss of accuracy is encountered with the EAI.

We have derived suitable strategies and methods for the estimation of the starting points which have shown to be effective in practice, and we have devised a practical criterion for the stopping conditions.

We think this proves both the flexibility of the EAI, which has been adapted to this case with the development of proper tools, and the importance of considering structured iterations for the approximation of eigenvalues of linearizations. This is particularly interesting for applications where the data is naturally expressed in different bases (or the same bases with different nodes), such as the transfer functions for closed loop linear systems [16], or the clipping problems in computer aided graphics [12].

References

[1] Aberth, O., 1973. Iteration methods for finding all zeros of a polynomial simultaneously. Mathematics of computation 27 (122), 339–344.

[2] Ball, J. A., Gohberg, I., et al., 2013. Interpolation of rational matrix functions. Vol. 45. Birkhäuser.

[3] Berrut, J.-P., Trefethen, L. N., 2004. Barycentric Lagrange interpolation. SIAM Review 46 (3), 501–517.

[4] Bini, D. A., Gemignani, L., Tisseur, F., 2005. The Ehrlich–Aberth Method for the Nonsymmetric Tridiagonal Eigenvalue Problem. SIAM Journal on Matrix Analysis and Applications 27 (1), 153–175.

[5] Bini, D. A., Meini, B., 2015. Generalization of the Brauer Theorem to Matrix Polynomials and Matrix Laurent Series. arXiv preprint arXiv:1512.07118.

[6] Bini, D. A., Noferini, V., 2013. Solving polynomial eigenvalue problems by means of the Ehrlich–Aberth method. Linear Algebra and its Applications 439 (4), 1130–1149.

[7] Bini, D. A., Robol, L., 2014. Solving secular and polynomial equations: A multiprecision algorithm. Journal of Computational and Applied Mathematics 272, 276292.

[8] Brauer, A., 1946. Limits for the characteristic roots of a matrix. Duke Mathematical Journal 13 (3), 387–395.

[9] Datta, B. N., 2010. Numerical Linear Algebra and Applications. SIAM.

[10] Edelman, A., Murakami, H., 1995. Polynomial roots from companion matrix eigenvalues. Mathematics of Computation 64 (210), 763–776.

[11] Ehrlich, L. W., 1967. A modified Newton method for polynomials. Communications of the ACM 10 (2), 107–108.

[12] Farin, G., 2014. Curves and surfaces for computer-aided geometric design: a practical guide. Elsevier.

[13] Forney Jr, G. D., 1975. Minimal bases of rational vector spaces, with applications to multivariable linear systems. SIAM Journal on Control 13 (3), 493–520.

[14] Gustafson, W. H., 1984. A note on matrix inversion. Linear algebra and its applications 57, 71–73.

[15] Henrici, P., 1974. Computational complex analysis. In: Proceedings Symposium Applied Mathematics. Vol. 20. pp. 79–86.

[16] Kailath, T., 1980. Linear systems. Vol. 156. Prentice-Hall Englewood Cliffs, NJ.

[17] Robol, L., Vandebril, R., Dooren, P. V., 2017. A framework for structured linearizations of matrix polynomials in various bases. SIAM Journal on Matrix Analysis and Applications 38 (1), 188–216.

[18] Tisseur, F., 2000. Backward error and condition of polynomial eigenvalue problems. Linear Algebra and its Applications 309 (1), 339–361.

[19] Trefethen, L. N., Weideman, J., 2014. The exponentially convergent trapezoidal rule. SIAM Review 56 (3), 385–458.

[20] Van Beeumen, R., Michiels, W., Meerbergen, K., 2015. Linearization of Lagrange and Hermite interpolating matrix polynomials. IMA Journal of Numerical Analysis 35 (2), 909–930.

[21] Walsh, J. L., 1935. Interpolation and approximation by rational functions in the complex domain. Vol. 20. American Mathematical Soc.

[22] Whittaker, S. E., Robinson, G., 1924. The Calculus of Observations. Blackie And Son Limited.