SIAM J. MATRIX ANAL. APPL.
Vol. 42, No. 3, pp. 1172–1198

A RATIONAL EVEN-IRA ALGORITHM FOR THE SOLUTION OF T-EVEN POLYNOMIAL EIGENVALUE PROBLEMS*

PETER BENNER†, HEIKE FASSBENDER‡, AND PHILIP SALTENBERGER‡

Abstract. In this work we present a rational Krylov subspace method for solving real large-scale polynomial eigenvalue problems with T-even (that is, symmetric/skew-symmetric) structure. Our method is based on the EVEN-IRA algorithm [V. Mehrmann, C. Schröder, and V. Simoncini, *Linear Algebra Appl.*, 436 (2012), pp. 4070–4087]. To preserve the structure, a sparse T-even linearization from the class of block minimal bases pencils is applied (see [F. M. Dopico et al., *Numer. Math.*, 140 (2018), pp. 373–426). Due to this linearization, the Krylov basis vectors can be computed in a cheap way. Based on the ideas developed in [P. Benner and C. Effenberger, *Taiwanese J. Math.*, 14 (2010), pp. 805–823], a rational decomposition is derived so that our method explicitly allows for changes of the shift during the iteration. This leads to a method that is able to compute parts of the spectrum of a T-even matrix polynomial in a fast and reliable way.

Key words. polynomial eigenvalue problem, symmetric/skew-symmetric matrix polynomial, structure-preserving linearization, Krylov subspace method, rational Krylov decomposition

AMS subject classifications. 15A18, 15B57, 65F15, 65F30

DOI. 10.1137/20M1364485

1. Introduction.

Eigenvalue problems are ubiquitous in engineering, physics, mechanics, and many more scientific disciplines. Moreover, they lie at the heart of numerical linear algebra. As eigenproblems stemming from real-world applications are often subject to physical constraints and side conditions, they frequently and naturally inherit structure. For instance, mechanical vibration systems are usually described by symmetric mass, damping and stiffness matrices (see [21]). Optimal control problems often involve Hamiltonian/skew-Hamiltonian matrix pencils [22]. But, after all, which features and properties single out faithful numerical algorithms for structured problems from universal methods? In the first place, the occurrence of structure can be utilized to speed up algorithms and reduce memory requirements. This originates from the deeper focus on the true nature of the problem compared to standard methods. In addition to that, the adequate exploitation of structure is beneficial (and indispensable, sometimes) for the reliability of an algorithm. Indeed, a proper numerical treatment of structure will often produce more accurate and physically meaningful results. Consequently, it seems reasonable to design tailor-made algorithms instead of addressing structured problems without any care by standard means. We present an algorithm for real, T-even polynomial eigenvalue problems of large scale that takes into account all the aforementioned aspects. The method we propose is an implicitly restarted rational Krylov–Schur approach based on the EVEN-IRA algorithm introduced in [24] (see also [12]). In contrast to the EVEN-IRA algorithm and motivated by [3], our approach explicitly allows for changes of the shift parameter during the iteration. This leads to a flexible and adjustable rational Krylov algorithm.

*Received by the editors September 4, 2020; accepted for publication (in revised form) April 7, 2021; published electronically July 28, 2021.
https://doi.org/10.1137/20M1364485

†Max Planck Institute for Dynamics of Complex Technical Systems, 39106 Magdeburg, Germany (benner@mpi-magdeburg.mpg.de).

‡Institute for Numerical Analysis, TU Braunschweig, 38106 Braunschweig, Germany (h.fassbender@tu-braunschweig.de, philip.saltenberger@tu-braunschweig.de).

There exist various major applications, including the vibration of gyroscopic systems and control theory, that lead to T-even polynomial eigenproblems of large size (see, e.g., [24, 7] and the references therein). A *matrix polynomial* $P(\lambda)$ is an element from $\mathbb{R}^{m\times n}[\lambda]$, i.e.,

$$P(\lambda) = \sum_{k=0}^{\ell} P_k \lambda^k = P_\ell \lambda^\ell + P_{\ell-1}\lambda^{\ell-1} + \cdots + P_1\lambda + P_0 \tag{1.1}$$

with matrices $P_j \in \mathbb{R}^{m\times n}$. The degree $\deg(P)$ of $P(\lambda)$ is the largest index j with $P_j \neq 0$. Often, we write $P(\lambda)$ as a matrix with polynomial entries, i.e., as an element from $\mathbb{R}[\lambda]^{m\times n}$. Here, we are mostly interested in square matrix polynomials $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ with some particular structure in its matrix coefficients. We call $P(\lambda) \in \mathbb{R}^{n\times n}[\lambda]$ as in (1.1) T-even if $P_j = P_j^T$ holds whenever j is even and $P_j = -P_j^T$ holds otherwise (see [16, sect. 6]). Equivalently, $P(\lambda)^T = P(-\lambda)$. Eigenvalue/eigenvector pairs $(\mu, x) \in \mathbb{C}\times\mathbb{C}^n$ of $P(\lambda)$ are characterized by the relation $P(\mu)x = 0$. To find eigenvalues of $P(\lambda)$, it is a common approach to turn $P(\lambda)$ into a matrix polynomial $\mathcal{L}_P(\lambda) = \lambda X + Y$ of degree one[1] (e.g., the Frobenius companion form [20]) by linearization. Then, the eigenvalues of $P(\lambda)$ and $\mathcal{L}_P(\lambda)$ coincide and the generalized eigenproblem corresponding to the linearization $\mathcal{L}_P(\lambda)$ may be solved by, e.g., the standard QZ algorithm (cf. [25]). However, solving a structured (i.e., T-even) eigenvalue problem via the QZ algorithm and the Frobenius companion form is not conducive in the light of the problem's nature and structure.

In particular, the spectrum of T-even matrix polynomials has a *Hamiltonian structure*, that is, it is symmetric with respect to both the real and the imaginary axes. The algorithm we present takes care of this fact in two different ways. On the one hand, the linearization $\mathcal{L}_P(\lambda) = \lambda X + Y$ of $P(\lambda)$ we consider is a symmetric/skew-symmetric matrix pencil (i.e., $Y = Y^T, X = -X^T$). In particular, $\mathcal{L}_P(\lambda)$ itself is T-even and so it naturally preserves the Hamiltonian spectral structure of $P(\lambda)$. On the other hand, for any $\zeta \in \mathbb{C}$ outside the spectrum of $P(\lambda)$, we consider the special shift-and-invert transformation

$$\mathcal{L}_P(\zeta) = \zeta X + Y \;\mapsto\; K(\zeta) := \mathcal{L}_P(\zeta)^{-T} X \mathcal{L}_P(\zeta)^{-1} X$$

as proposed in [22, 24]. Each eigenvalue pair $(+\mu, -\mu)$ of $\mathcal{L}_P(\lambda)$ is transformed to only one eigenvalue $\theta = (\mu^2 - \zeta^2)^{-1}$ of $K(\zeta)$. Consequently, $K(\zeta)$ preserves eigenvalue pairings and the spectral symmetry inherent to the problem is respected.

The foundation of our method is the Even-IRA algorithm from [24]. This method is a sophisticated variant of the Krylov–Schur algorithm (see Stewart [29]) applied to $K(\zeta)$ for some appropriately chosen shift parameter ζ and a T-even linearization $\mathcal{L}_P(\lambda)$ for $P(\lambda)$. Rather than applying a (structure-preserving) symplectic Lanczos process as in [4, 5], our approach is related to the ideas established for the SHIRA algorithm in [22] (see also [3]). To define $K(\zeta)$, we take $\mathcal{L}_P(\lambda) = \lambda X + Y$ to be a special linearization from the class of block minimal bases pencils (see [9]). Due to the structure and sparsity of $\mathcal{L}_P(\lambda)$, the computation of matrix-vector-products $K(\zeta)x$ can be realized implicitly without ever forming $K(\zeta)$ at all. Moreover, linear systems with $\mathcal{L}_P(\zeta)$ and $\mathcal{L}_P(\zeta)^T$ (that arise in Arnoldi-like processes from matrix-vector-products involving $K(\zeta)$) can be solved implicitly through systems involving only $P(\zeta)$ and $P(-\zeta)$. Accordingly, the complexity of computing $K(\zeta)x$ is reduced

[1] Matrix polynomials of degree one are often called matrix pencils.

by a significant amount since the size of $P(\zeta)$ is substantially smaller than the size of $\mathcal{L}_P(\zeta)$. For the same reason, memory requirements (e.g., for storing matrix decompositions) can be decreased. These advantages of $\mathcal{L}_P(\lambda)$ over other linearizations (see, e.g., [16]) have already been successfully applied in [12] to the Even-IRA algorithm. However, as the Even-IRA algorithm does not allow for changes of the shift ζ during the iteration, this feature is incorporated in our method. Based on [3] and [26], our rational Even-IRA algorithm permits shift adjustments during the iteration without discarding the information that has been accumulated so far. Retaining the advantageous computational aspects, this endows our approach with more flexibility. As a consequence, the rational Even-IRA algorithm we present is a new reliable, flexible, and fast numerical method with reasonable costs.

This work is structured as follows:

1. The basic definitions regarding matrix polynomials and their eigenvalues are presented in section 2. We introduce the concept of linearization and show how a T-even linearization can be constructed.
2. In section 3, we briefly review the Even-IRA algorithm from [24]. It is the basis of our rational method to compute eigenvalues of T-even matrix polynomials in a structure-preserving way.
3. We show how the matrix-vector-multiplications involved in the Even-IRA algorithm can be carried out in a very efficient and implicit way in section 4. This is possible without forming the corresponding large-scale matrix at all.
4. Section 5 is dedicated to the rational Arnoldi decomposition. We show how a rational decomposition can be invoked for the Even-IRA algorithm and how it is applied in a useful fashion for our purpose.
5. We introduce the rational Even-IRA algorithm in section 6. We discuss the Krylov–Schur restart procedure in detail and also address the issue of infinite eigenvalues to guarantee a stable convergence of the algorithm.
6. Some numerical examples are given in section 7. We also illustrate how the shift strategy influences the algorithms success.

Some conclusions are given in section 8.

2. Definitions of matrix polynomials and notation. Recall that a matrix polynomial $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ as in (1.1) is said to be *T-even* if $P(\lambda)^T = P(-\lambda)$. The classes of regular, singular, and unimodular matrix polynomials are defined as follows:

1. A matrix polynomial $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ is called *regular* if $\det(P(\lambda)) \neq 0$ and *singular* otherwise (notice that $\det(P(\lambda)) \in \mathbb{R}[\lambda]$).
2. A matrix polynomial $Q(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ is called *unimodular* if $\det(Q(\lambda))$ is a nonzero constant.

Let $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ be regular. We call $\mu \in \mathbb{C}$ a (finite) *eigenvalue* of $P(\lambda)$ if

$$P(\mu) = \sum_{k=0}^{m} \mu^k P_k \in \mathbb{C}^{n\times n}$$

is a singular matrix. Thus, $\mu \in \mathbb{C}$ is an eigenvalue of $P(\lambda)$ if and only if $\det(P(\mu)) = 0$. Therefore, the set of all finite eigenvalues of $P(\lambda)$ coincides with the roots of $\det(P(\lambda)) \in \mathbb{R}[\lambda]$ [20, sect. 2]. The *algebraic multiplicity* of μ is defined as the multiplicity of μ as a root of $\det(P(\lambda))$. In addition, if $\mu \in \mathbb{C}$ is some eigenvalue of $P(\lambda)$, the corresponding nullspace $\operatorname{null}(P(\mu))$ is called the *eigenspace* for μ. Its dimension is referred to as the *geometric multiplicity* of μ.

We define for any $d \geq \deg(P)$

$$\operatorname{rev}_d P(\lambda) := \lambda^d P(\lambda^{-1}).$$

Then $\operatorname{rev}_d P(\lambda)$ is again a matrix polynomial, i.e., $\operatorname{rev}_d P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$. It is called the *d-reversal* corresponding to $P(\lambda)$ [20, Def. 2.2]. In case $d = \deg(P)$, we call $\operatorname{rev} P(\lambda) := \operatorname{rev}_d P(\lambda)$ the reversal of $P(\lambda)$. It is easily verified that the finite eigenvalues of $\operatorname{rev} P(\lambda)$ are the reciprocals of the eigenvalues of $P(\lambda)$. In accordance with this observation, we call ∞ an eigenvalue of $P(\lambda)$ if zero is an eigenvalue of $\operatorname{rev} P(\lambda)$. The algebraic and geometric multiplicities of the eigenvalue ∞ are defined in terms of $\operatorname{rev} P(\lambda)$ and its finite eigenvalue $\mu = 0$ [20, Def. 2.3]. The set of all eigenvalues of $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ is called the *spectrum* of $P(\lambda)$ and is denoted by $\sigma(P)$.

The following important property is intrinsic for the eigenvalues of T-even matrix polynomials $P(\lambda)$.

PROPOSITION 2.1. *The spectrum $\sigma(P)$ of real, T-even matrix polynomials $P(\lambda)$ has a Hamiltonian structure. That is, $\sigma(P)$ is symmetric with respect to both the real and the imaginary axis.*

Two matrix polynomials $S(\lambda), P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ are called unimodular *equivalent* if there exist unimodular matrix polynomials $U(\lambda), V(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ such that $S(\lambda) = U(\lambda)P(\lambda)V(\lambda)$ holds. Linearizations for matrix polynomials are defined by unimodular equivalence as follows.

DEFINITION 2.2 (Linearization [9, Def. 2.12]). *Let $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$.*

(i) *Any matrix polynomial $\mathcal{L}(\lambda) = \lambda X + Y$ that can be expressed as*

$$\mathcal{L}(\lambda) = U(\lambda) \begin{bmatrix} I_s & 0 \\ 0 & P(\lambda) \end{bmatrix} V(\lambda) \in \mathbb{R}[\lambda]^{(n+s)\times(n+s)} \tag{2.1}$$

for two unimodular matrix polynomials $U(\lambda), V(\lambda)$ of size $(n+s)\times(n+s)$ and some $s \in \mathbb{N}_0$ is called a linearization for $P(\lambda)$.

(ii) *Assume $\deg(P) = k$. A linearization $\mathcal{L}(\lambda)$ for $P(\lambda)$ as in (2.1) is called strong (linearization) whenever $\operatorname{rev}_1\mathcal{L}(\lambda)$ is a linearization for $\operatorname{rev}_k P(\lambda) = \operatorname{rev} P(\lambda)$, too.*

Notice that unimodular matrix polynomials do not have any finite eigenvalues. Therefore, any linearization $\mathcal{L}(\lambda)$ as in (2.1) of $P(\lambda)$ has the same finite eigenvalues (with the same algebraic and geometric multiplicities) as $P(\lambda)$ [9]. Furthermore, if $\mathcal{L}(\lambda)$ is strong, the same holds for the eigenvalue ∞ in the case $\infty \in \sigma(P)$.

The problem of finding linearizations for matrix polynomials $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ has been addressed in, e.g., [20, 28]. Particular research has been done on conditioning [13], structure-preservation [14], and nonstandard polynomial bases [11, 17, 18]. In [9], a new class of linearizations was introduced (so-called block minimal bases linearizations) that has recently attracted much attention. The linearization we present in Theorem 2.5 will belong to this class.

Due to Proposition 2.1, we are particularly interested in T-even linearizations (whenever $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ is T-even) to preserve the symmetries inherent to the spectrum of $P(\lambda)$. The structure of the T-even linearization $\mathcal{L}_P(\lambda)$ we define in (2.4) varies slightly depending on the parity of $\deg(P)$ (which can be even or odd). Thus we define $M_P(\lambda)$ in Definition 2.3 depending on the degree of $P(\lambda)$ to treat both cases in Theorem 2.5 in a common framework. Here and hereafter, we use the notation $\langle x, y\rangle$ to represent the scalar product $x^T y \in \mathbb{R}$ of two vectors x and y and $\oplus$ to denote the direct sum of matrices, i.e., $A \oplus B = \operatorname{diag}(A, B)$ for any $A, B \in \mathbb{R}^{n\times n}$.

DEFINITION 2.3. *Assume $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ is T-even and given as in* (1.1).
(a) *If* $\deg(P)$ *is odd, we define*

$$M_P(\lambda) := \bigoplus_{k=0}^{\ell-1} (-1)^k \big(\lambda P_{d-2k} + P_{d-2k-1}\big) \in \mathbb{R}[\lambda]^{\ell n\times \ell n} \tag{2.2}$$

with $d = \deg(P)$ *and* $\ell = (d+1)/2$.
(b) *If* $\deg(P)$ *is even, we define* $M_P(\lambda)$ *as in* (2.2) *above with* $d = \deg(P)+1$, $\ell = (d+1)/2$, *and* $P_d := 0_{n\times n}$.

Notice that $(\lambda P_{d-2k}+P_{d-2k-1})^T = -\lambda P_{d-2k}+P_{d-2k-1}$ holds for all summands in (2.2) regardless of the parity of $\deg(P)$. That means $M_P(\lambda)^T = M_P(-\lambda)$, so $M_P(\lambda)$ is always T-even. With the definition $\Lambda_k(\lambda) := [\,\lambda^k\ \lambda^{k-1}\ \cdots\ \lambda\ 1\,] \in \mathbb{R}[\lambda]^{1\times(k+1)}$ for any $k \geq 1$, we make the following important observation.

Remark 2.4. According to the construction of $M_P(\lambda)$ for $P(\lambda)$ as in (2.2) it can be verified by a direct calculation that

$$\big(\Lambda_{\ell-1}(-\lambda)\otimes I_n\big) M_P(\lambda) \big(\Lambda_{\ell-1}(\lambda)^T \otimes I_n\big) = P(\lambda)$$

holds. This property will be exploited in section 4.

Let $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ be T-even. With the use of $M_P(\lambda)$ and

$$L_k(\lambda) := \begin{bmatrix} 1 & -\lambda & & \\ & 1 & -\lambda & \\ & & \ddots & \ddots \\ & & & 1 & -\lambda \end{bmatrix} \in \mathbb{R}[\lambda]^{k\times(k+1)}, \quad k \geq 1, \tag{2.3}$$

we present a structure-preserving, i.e., T-even, linearization $\mathcal{L}_P(\lambda)$ for $P(\lambda)$ in the following Theorem 2.5. It is a block minimal bases pencil (as introduced in [9]) and was already used in [12]. In particular, Theorem 3.3 in [9] applies to the matrix pencil $\mathcal{L}_P(\lambda)$ defined in (2.4) below and confirms that it is in fact a linearization for $P(\lambda)$.

THEOREM 2.5. *Let $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ be T-even. Then the matrix pencil*

$$\mathcal{L}_P(\lambda) := \left[\begin{array}{c|c} M_P(\lambda) & L_{\ell-1}(-\lambda)^T \otimes I_n \\ \hline L_{\ell-1}(\lambda)\otimes I_n & 0 \end{array}\right] \in \mathbb{R}[\lambda]^{dn\times dn} \tag{2.4}$$

defined for $P(\lambda)$ with $M_P(\lambda)$, d, and ℓ given as in Definition 2.3 *and $L_{\ell-1}(\lambda)$ as introduced in* (2.3) *is T-even. Moreover, $\mathcal{L}_P(\lambda)$ is a strong linearization for $P(\lambda)$ if* $\deg(P)$ *is odd and a linearization for $P(\lambda)$ if* $\deg(P)$ *is even.*

Due to the linearization property, the matrix pencil $\mathcal{L}_P(\lambda) \in \mathbb{R}[\lambda]^{dn\times dn}$ defined in (2.4) has the same finite eigenvalues as $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ (from whose matrix coefficients it is defined). Moreover, since $\mathcal{L}_P(\lambda)$ is T-even whenever $P(\lambda)$ is T-even, we call $\mathcal{L}_P(\lambda)$ a structure-preserving linearization for $P(\lambda)$. To illustrate the form of $\mathcal{L}_P(\lambda)$ consider the following example.

Example 2.6. The linearization $\mathcal{L}_P(\lambda)$ defined for a T-even matrix polynomial $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ has a very sparse and clear structure. This is illustrated below for $P(\lambda) = \sum_{k=0}^{7} \lambda^k P_k$ of degree seven. Writing $\mathcal{L}_P(\lambda)$ in the form $\lambda X + Y$ for two

$7n \times 7n$ matrices X and Y we have

$$\mathcal{L}_P(\lambda) = \left[\begin{array}{ccccc|ccc} -P_7 & & & & & I_n & & \\ & P_5 & & & & & I_n & \\ & & -P_3 & & & & & I_n \\ & & & P_1 & & & & I_n \\ \hline & -I_n & & & & & & \\ & & -I_n & & & & & \\ & & & -I_n & & & & \end{array}\right] \lambda + \left[\begin{array}{ccccc|ccc} -P_6 & & & & & I_n & & \\ & P_4 & & & & & I_n & \\ & & -P_2 & & & & & I_n \\ & & & P_0 & & & & \\ \hline I_n & & & & & & & \\ & I_n & & & & & & \\ & & I_n & & & & & \end{array}\right]. \tag{2.5}$$

Since $P(\lambda)$ was assumed to be T-even, it is seen directly that Y is symmetric while X is skew-symmetric. In addition, notice that if $P(\lambda)$ was only of degree six, $\mathcal{L}_P(\lambda)$ as defined in (2.4) would be as in (2.5) with $P_7 = 0$.

In general, determining $\sigma(P)$ for a matrix polynomial $P(\lambda)$ is sometimes referred to as the *polynomial eigenvalue problem* (PEP). If $P(\lambda) = \lambda X + Y$ is a matrix pencil, the term generalized eigenvalue problem (GEP) is often used. A common way to solve a PEP corresponding to $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ is to compute $\sigma(P)$ (or just a part of it) through a linearization $\mathcal{L}(\lambda)$ for $P(\lambda)$ using a method for GEPs. Notice that the size of $\mathcal{L}(\lambda)$ is usually much larger than the size of $P(\lambda)$ (depending on the degree of $P(\lambda)$). Thus it is often appropriate (or even necessary) not to compute all eigenvalues of $\mathcal{L}(\lambda)$ but only some (e.g., in a predefined area of the complex plane). For such purposes Krylov subspace methods are among the most appropriate algorithms (cf. [1] for an overview of different Krylov subspace algorithms). The area where eigenvalues are to be found is controlled via a shift parameter $\zeta \in \mathbb{C}$. With some abuse of terminology, a Krylov subspace method can be called rational if it admits changes of this shift parameter during its iteration (see [26, 27]).

All subsequent investigations mainly aim for the construction of a rational Krylov subspace algorithm to determine eigenvalues of $\mathcal{L}_P(\lambda)$ defined as in (2.4) for some given T-even matrix polynomial $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$. The T-even structure of $\mathcal{L}_P(\lambda)$ is exploited to preserve the spectral symmetries.

3. The Even-IRA algorithm. According to Proposition 2.1, the spectrum of a (real) T-even matrix polynomial is symmetric with respect to the real and imaginary axis. Numerical algorithms respecting this spectral symmetry will in general be more accurate than standard methods [19]. In addition, numerical methods that ignore the special structure may often produce (physically) less meaningful results [21]. Therefore, our focus on the development of a reliable eigensolver for T-even polynomial eigenvalue problems is twofold: on the one hand on the application of a structure-preserving linearization (see Theorem 2.5) and on the other hand on a method that profitably exploits this structure. One method taking the T-even structure into account is the Even-IRA algorithm presented in [24]. It belongs to the class of Krylov subspace methods and is a sophisticated variant of the Krylov–Schur algorithm [29] customized for real T-even generalized eigenvalue problems. Other methods for solving T-even polynomial eigenvalue problems can be found in, e.g., [2, 22].

The Even-IRA algorithm is designed to determine a part of the spectrum of a regular T-even matrix pencil $\mathcal{G}(\lambda) = \lambda X + Y \in \mathbb{R}[\lambda]^{m\times m}$ close to a predefined target in the complex plane. To preserve the Hamiltonian eigenvalue structure, a special spectral transformation is applied to preserve the $\pm$ matching pairs of eigenvalues.

In particular, whenever $\mathcal{G}(\lambda) = \lambda X + Y$ is regular and T-even, i.e., $X = -X^T$ and $Y = Y^T$ holds, and some shift $\zeta \notin \sigma(\mathcal{G})$ is given in a region of the complex plane where eigenvalues are to be found, then in [24] the transformation

$$\mathcal{G}(\zeta) = \zeta X + Y \mapsto K(\zeta) = \mathcal{G}(\zeta)^{-T} X \mathcal{G}(\zeta)^{-1} X \in \mathbb{C}^{m\times m} \tag{3.1}$$

is considered. Notice that a similar spectral transformation already appeared in [4, 22, 31] in the context of skew-Hamiltonian/Hamiltonian eigenvalue problems and the symplectic Lanczos process. Whenever $\mathcal{G}(\mu)x = 0$ holds for some $\mu \in \mathbb{C}$ and $x \in \mathbb{C}^m$, it is easily confirmed that $K(\zeta)x = \theta x$ follows, where $\theta = (\mu^2 - \zeta^2)^{-1}$. Thus, any two finite eigenvalues μ and $-\mu$ of $\mathcal{G}(\lambda)$ are mapped to the same eigenvalue $\theta \in \sigma(K(\zeta))$. Due to this fact, $\pm$ matching pairs of eigenvalues are preserved. On the other hand, all eigenvalues of $K(\zeta)$ are necessarily of even multiplicity. Notice the following two important facts:

- Whenever some eigenvalue $\theta \in \sigma(K(\zeta))$ has been found, it gives rise to a $\pm$ matching pair of two eigenvalues of $\mathcal{G}(\lambda)$, namely,

$$\mu = \sqrt{(1/\theta) + \zeta^2} \quad \text{and} \quad \widehat{\mu} = -\sqrt{(1/\theta) + \zeta^2}. \tag{3.2}$$

- The matrix $K(\zeta)$ from (3.1) will in general be complex but remains real whenever $\zeta \in \mathbb{R}$ or $\zeta \in \imath\mathbb{R}$. In the case $\zeta = a + b\imath$ with nonzero real and imaginary parts, a slightly different spectral transformation can be considered (see [24, Rem. 2.1]) to stay in real arithmetic.

In [24] the authors suggest applying the implicitly restarted Krylov–Schur method [29] to the matrix $K(\zeta)$ in (3.1) to find some, say, $s \in \mathbb{N}$, eigenvalues of $\mathcal{G}(\lambda)$. That is, if $v_1, \ldots, v_s$ is an orthonormal basis of the Krylov space

$$\mathcal{K}_s(K(\zeta), x) = \operatorname{span}\{x, K(\zeta)x, K(\zeta)^2 x, \ldots, K(\zeta)^{s-1}x\} \tag{3.3}$$

for some $x \in \mathbb{R}^m$ (computed by the Arnoldi method; see [1]) and $V = [\, v_1 \ \cdots \ v_s \,] \in \mathbb{R}^{m\times s}$, in general some of the eigenvalues of $K(\zeta)$ of largest magnitude are well approximated by some of the s eigenvalues of $V^T K(\zeta) V$. This process can now be (implicitly) restarted using the Krylov–Schur restart strategy [29, sect. 3] until all s eigenvalues of $V^T K(\zeta) V$ serve as good approximations to eigenvalues of $K(\zeta)$. This approach is called the Even-IRA algorithm (details on the practical implementation of the algorithm can be found in [24, sect. 4]). Additional information on how eigenvectors may be captured can be found in [24, p. 4074ff].

The basis of our algorithm is the Even-IRA algorithm. As this method is designed for T-even matrix pencils, it cannot be used directly for T-even matrix polyomials $P(\lambda)$ of degree > 1. To solve the PEP for $P(\lambda)$, we apply the Even-IRA algorithm to the structure-preserving linearization $\mathcal{L}_P(\lambda)$ from (2.4). The sparse block structure of $\mathcal{L}_P(\lambda)$ turns out to be very beneficial for the computation of matrix-vector-products $K(\zeta)x$ (which are necessary to build the Krylov space; see (3.3)). In fact, we show in section 4 that $K(\zeta)x$ can be computed in a cheap and reliable way without ever forming $K(\zeta)$ and $\mathcal{L}_P(\zeta)$ explicitly. In section 5, we will modify the Even-IRA algorithm so that it is able to handle changes of the shift parameter ζ during the Arnoldi iteration and the restart process. This makes it possible to accelerate convergence or to control/change the regions in the complex plane where eigenvalues are to be found.

4. The efficient computation of matrix-vector-products $K(\zeta)x$. Assume that $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ is some T-even matrix polynomial and let $\mathcal{L}_P(\lambda) \in \mathbb{R}[\lambda]^{dn\times dn}$ be defined as in (2.4). Recall that $P(\lambda)$ and $\mathcal{L}_P(\lambda)$ share the same finite eigenvalues. As outlined in Theorem 2.5, $\mathcal{L}_P(\lambda)$ is T-even, so the Even-IRA algorithm can be applied to $\mathcal{L}_P(\lambda)$ to determine a part of the finite spectrum of $P(\lambda)$. In consideration of large-scale-problems the sparsity and structure of $\mathcal{L}_P(\lambda)$ can be exploited to significantly increase the computational speed in calculating $K(\zeta)v$. This effective computational approach is explained in detail in this section (see also [12] and [28, sect. 5.2]).

Remark 4.1. As we are only considering polynomial eigenvalue problems given by real T-even matrix polynomials, matrix-vector-multiplications $K(\zeta)v$ will only involve real vectors $v \in \mathbb{R}^{dn}$ in all subsequent sections (even if ζ and $K(\zeta)$ are complex). However, the technique to perform matrix-vector-multiplications is valid even if $v \in \mathbb{C}^{dn}$, so we give a general treatment here.

To begin, assume $\zeta \in \mathbb{C}$ is not contained in $\sigma(P)$ and let $v \in \mathbb{C}^{dn}$ be given. Moreover, let $\mathcal{L}_P(\lambda) = \lambda X + Y$ as in (2.4). Explicitly, the matrix-vector-product $K(\zeta)v$ can be written as

$$K(\zeta)v = \left(\mathcal{L}_P(\zeta)^{-T} X \mathcal{L}_P(\zeta)^{-1} X\right) v. \tag{4.1}$$

Actually, (4.1) can be evaluated using four consecutive matrix-vector-multiplications. The matrix-vector-products with X, where $X \in \mathbb{R}^{dn\times dn}$, can be evaluated directly and quickly by exploiting the sparsity of X. Moreover, as X has a clear and determined block-structure, a matrix-vector-multiplication Xv can entirely be carried out implicitly, that is, without forming X at all, on its nonzero $n \times n$ blocks. The matrix-vector-products with $\mathcal{L}_P(\zeta)^{-1}$ and $\mathcal{L}_P(\zeta)^{-T} = \mathcal{L}_P(-\zeta)^{-1}$ can be realized by solving linear systems with $\mathcal{L}_P(\zeta)$ and $\mathcal{L}_P(-\zeta)$, respectively. However, the size of both matrices is $dn \times dn$ and, therefore, can be rather large. Fortunately, a linear-systems-solve with $\mathcal{L}_P(\zeta)$ can be traced back to solely $n \times n$ computations involving the solution of a linear system with $P(\zeta) \in \mathbb{C}^{n\times n}$. This provides an economic approach for the determination of these products since, for instance, the computational cost of an LU decomposition for $\mathcal{L}_P(\zeta)$ is within $\mathcal{O}(d^3n^3)$ while it is only $\mathcal{O}(n^3)$ for $P(\zeta)$ if no sparsity patterns are taken into account. For sparse matrices the cost is about $\mathcal{O}(d \cdot \mathrm{nz})$ and $\mathcal{O}(\mathrm{nz})$, respectively, where nz denotes the number of nonzero entries. Moreover, the storage requirements for the LU factors for $P(\zeta)$ are way below those for the LU factors of $\mathcal{L}_P(\lambda)$.

Assume that $\mathcal{L}_P(\zeta)^{-1}v$ is to be computed, i.e., the linear system

$$\mathcal{L}_P(\zeta)y = \left[\begin{array}{c|c} M_P(\zeta) & L_{\ell-1}(-\zeta)^T \otimes I_n \\ \hline L_{\ell-1}(\zeta) \otimes I_n & 0 \end{array}\right] \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x \tag{4.2}$$

is to be solved for a given vector $x \in \mathbb{C}^{dn}$. Let $y^\star$ be the solution of (4.2) which is unique since $\mathcal{L}_P(\zeta)$ is nonsingular (due to the fact that $\zeta \notin \sigma(P)$). Let $x, y \in \mathbb{C}^{dn}$ be partitioned as $x_1, y_1 \in \mathbb{C}^{\ell n}$ and $x_2, y_2 \in \mathbb{C}^{(\ell-1)n}$ and assume that $y^\star = [\,(y_1^\star)^T\ (y_2^\star)^T\,]^T$ is partitioned accordingly. The structure of $\mathcal{L}_P(\zeta)$ reveals that (4.2) can be rewritten as a system of two equations for the unknown vectors y_1 and y_2 as

$$M_P(\zeta)y_1 + \left(L_{\ell-1}(-\zeta)^T \otimes I_n\right) y_2 = x_1 \quad \text{and} \tag{4.3}$$

$$\left(L_{\ell-1}(\zeta) \otimes I_n\right) y_1 = x_2. \tag{4.4}$$

Notice that (4.4) is an underdetermined system with $L_{\ell-1}(\zeta) \otimes I_n \in \mathbb{C}^{(\ell-1)n\times \ell n}$. Moreover, $\mathrm{rank}(L_{\ell-1}(\zeta)\otimes I_n)) = (\ell-1)n$ holds regardless of the choice of ζ. Therefore,

the nullspace of $L_{\ell-1}(\zeta) \otimes I_n$ is always n-dimensional and easily determined since

$$\big(L_\ell(\zeta) \otimes I_n\big)\big(\Lambda_\ell(\zeta)^T \otimes I_n\big) = 0_{\ell n \times n}.$$

Therefore we have $\operatorname{null}(L_\ell(\zeta) \otimes I_n) = \{(\Lambda_\ell(\zeta)^T \otimes I_n)r\,;\, r \in \mathbb{C}^n\}$. Consequently, any solution y_1 for (4.4) has the form $y_1 = \widehat{y}_1 + (\Lambda_\ell(\zeta)^T \otimes I_n)r$, where $\widehat{y}_1 \in \mathbb{C}^{\ell n}$ solves (4.4) and $r \in \mathbb{C}^n$ is arbitrary. In fact, once some particular solution $\widehat{y}_1$ has been found, there exists some unique $r^\star$ such that $y_1^\star = \widehat{y}_1 + (\Lambda_\ell(\zeta)^T \otimes I_n)r^\star \in \mathbb{C}^{\ell n}$. With this characterization of $y_1^\star$ at hand, it follows from (4.3) that

$$M_P(\zeta)\big(\widehat{y}_1 + (\Lambda_\ell(\zeta)^T \otimes I_n)r^\star\big) + \big(L_\ell(-\zeta)^T \otimes I_n\big)\, y_2^\star = x_1 \tag{4.5}$$

holds. Multiplying (4.5) by $\Lambda_\ell(-\zeta) \otimes I_n$ from the left eliminates the second term since $(\Lambda_\ell(-\zeta) \otimes I_n)(L_\ell(-\zeta)^T \otimes I_n) = 0$. After some reordering we obtain from (4.5)

$$\underbrace{\big(\Lambda_\ell(-\zeta) \otimes I_n\big)M_P(\zeta)\big(\Lambda_\ell(\zeta)^T \otimes I_n\big)}_{=P(\zeta)}\, r^\star = x_1 - \big(\Lambda_\ell(-\zeta) \otimes I_n\big)M_P(\zeta)\widehat{y}_1. \tag{4.6}$$

Notice that the left-hand side of (4.6) simplifies to $P(\zeta)r^\star$ in accordance with Remark 2.4. In addition, as $P(\zeta)$ is nonsingular, $r^\star$ is the unique solution of (4.6). Thus, in other words, for any fixed particular solution $\widehat{y}_1$ of (4.4), the unique solution $r^\star$ of the $n \times n$ linear system

$$P(\zeta)r = x_1 - \big(\Lambda_\ell(-\zeta) \otimes I_n\big)M_P(\zeta)\widehat{y}_1 \tag{4.7}$$

determines the first part $y_1^\star = \widehat{y}_1 + (\Lambda_\ell(\zeta)^T \otimes I_n)r^\star \in \mathbb{C}^{\ell n}$ of the solution vector $y^\star$. Once $y_1^\star$ has been found, $y_2^\star \in \mathbb{C}^{(\ell-1)n}$ will be the unique solution of the overdetermined system

$$(L_{\ell-1}(-\zeta) \otimes I_n)\, y_2 = x_1 - M_P(\zeta)y_1^\star \tag{4.8}$$

since (4.3) and (4.4) are satisfied if and only if $y_1 = y_1^\star$ and $y_2 = y_2^\star$. The computations of a particular solution $\widehat{y}_1$ of (4.4) and the solution $y_2^\star$ of (4.8) can be carried out by forward and backward substitution and both require $\mathcal{O}(\ell n)$ flops. In particular, the following hold:

1. A solution $\widehat{y}_1 \in \mathbb{C}^{\ell n}$ for (4.4), i.e.,
   (4.9)
   $$\begin{bmatrix} I_n & -\zeta I_n & & & \\ & I_n & -\zeta I_n & & \\ & & \ddots & \ddots & \\ & & & I_n & -\zeta I_n \end{bmatrix} \begin{bmatrix} y_{1,1} \\ y_{1,2} \\ \vdots \\ y_{1,\ell-1} \\ y_{1,\ell} \end{bmatrix} = \begin{bmatrix} v_{1,1} \\ v_{1,2} \\ \vdots \\ v_{1,\ell-1} \end{bmatrix}, \quad y_{1,k}, v_{1,k} \in \mathbb{C}^n,$$
   can be found by backward substitution. If $\widehat{y}_1$ and $v_1 \in \mathbb{C}^{(\ell-1)n}$ are partitioned as in (4.9) and $\widehat{y}_{1,\ell} = 0$ is chosen, then $\widehat{y}_{1,1}, \ldots, \widehat{y}_{1,\ell-1}$ are uniquely determined through the recurrence relation $\widehat{y}_{1,k} = v_{1,k} + \zeta\widehat{y}_{1,k+1}$ for $k = \ell-1, \ldots, 1$.
2. The unique solution $y_2^\star$ of (4.8), i.e.,
   (4.10)
   $$\begin{bmatrix} I_n & & & \\ \zeta I_n & I_n & & \\ & \zeta I_n & \ddots & \\ & & \ddots & I_n \\ & & & \zeta I_n \end{bmatrix} \begin{bmatrix} y_{2,1} \\ y_{2,2} \\ \vdots \\ y_{2,\ell-1} \end{bmatrix} = x_1 - M_P(\zeta)y_1^\star =: \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{\ell-1} \\ w_\ell \end{bmatrix}, \; y_{2,k}, w_k \in \mathbb{C}^n,$$

can be found by forward substitution. If $y_2^\star$ is partitioned as y_2 in (4.10), then $y_{2,1}^\star = w_1$ and $y_{2,2}^\star, \ldots, y_{2,\ell}^\star$ are uniquely determined by the recurrence $y_{2,k}^\star = w_k - \zeta y_{2,k-1}^\star$ for $k = 2, \ldots, \ell$.

Remark 4.2. Notice that although (4.10) is an overdetermined system for $y_2 \in \mathbb{C}^{(\ell-1)n}$ which usually need not have a solution, there is a unique solution $y_2^\star$ for (4.10) since we assumed $\mathcal{L}_P(\zeta)y = x$ to be uniquely solvable.

For the determination of matrix-vector-products $\mathcal{L}_P(\zeta)^{-T}v$, the T-even structure of $\mathcal{L}_P(\zeta)$ can be exploited. In particular, $\mathcal{L}_P(\zeta)^{-T} = (\mathcal{L}_P(\zeta)^T)^{-1} = \mathcal{L}_P(-\zeta)^{-1}$. In order to find $\mathcal{L}_P(-\zeta)^{-1}v$, the same approach as above can be used involving $-\zeta$ instead of ζ. In particular, in (4.7) the matrix $P(-\zeta)$ instead of $P(\zeta)$ will show up. If an LU decomposition $P(\zeta) = LU$ has been computed to solve the linear system with $P(\zeta)$ in (4.7), this factorization can be reused to solve the system with $P(-\zeta)$ since $P(-\zeta) = P(\zeta)^T = U^T L^T$.

In conclusion, the procedure described in this section presents an efficient way to calculate matrix-vector-products of the form (4.1). Whenever $d \ll n$, the complexity of the overall method is dominated by the cost of the LU decomposition of $P(\zeta)$, which is $\mathcal{O}(\mathrm{nz})$ or $\mathcal{O}(n^3)$ depending on whether $P(\zeta)$ is sparse or not.

5. The rational Arnoldi decomposition. Let $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ be some T-even matrix polynomial and let $\mathcal{L}_P(\lambda) = \lambda X + Y \in \mathbb{R}[\lambda]^{dn\times dn}$ and

$$K(\zeta) = \mathcal{L}_P(\zeta)^{-T} X \mathcal{L}_P(\zeta)^{-1} X = \mathcal{L}_P(-\zeta)^{-1} X \mathcal{L}_P(\zeta)^{-1} X, \quad \zeta \notin \sigma(P), \tag{5.1}$$

be defined for $P(\lambda)$ as in (2.4) and (3.1), respectively.

Recall from section 3 that $K(\zeta)$ as in (5.1) stays real whenever ζ is real or purely imaginary. We will assume for the moment that either of them holds to stay within real arithmetic. So, let $v_1 \in \mathbb{R}^{dn}$ be some normalized vector and suppose that (for instance, as part of the Even-IRA algorithm) $m \in \mathbb{N}$ steps of the Arnoldi process (cf. [1, Alg. 7.3]) have been performed for $K(\zeta)$. That is, we have an Arnoldi decomposition for $K(\zeta) \in \mathbb{R}^{dn\times dn}$ of the form

$$K(\zeta)\begin{bmatrix} V_m \end{bmatrix} = \begin{bmatrix} V_m \end{bmatrix}\begin{bmatrix} T_m \end{bmatrix} + t_{m+1,m} v_{m+1} e_m^T = \begin{bmatrix} V_{m+1} \end{bmatrix}\begin{bmatrix} \underline{T}_m \end{bmatrix}, \tag{5.2}$$

where $V_{m+1} = [\, v_1 \ \cdots \ v_{m+1} \,] = [\, V_m \ v_{m+1} \,] \in \mathbb{R}^{dn\times(m+1)}$. The following statements hold for the vectors and matrices involved in (5.2):

1. The columns $v_1, \ldots, v_{m+1} \in \mathbb{R}^{dn}$ of V_{m+1} form an orthonormal basis of the Krylov space $\mathcal{K}_{m+1}(K(\zeta), v_1)$, where v_1 is the starting vector of the Arnoldi iteration.
2. The matrices $T_m = [t_{i,j}]_{i,j} \in \mathbb{R}^{m\times m}$ and $\underline{T}_m := I_{m+1,m} T_m + t_{m+1,m} v_{m+1} e_m^T \in \mathbb{R}^{(m+1)\times m}$ have upper-Hessenberg structure, where e_m denotes the mth unit vector in $\mathbb{R}^m$.

The eigenvalues of $T_m \in \mathbb{R}^{m\times m}$ are called Ritz values of $K(\zeta)$ with respect to $\mathcal{K}_{m+1}(K(\zeta), v_1)$. According to the Rayleigh–Ritz principle (cf. [8, sect. 7]), these values are used as approximations to the eigenvalues of $K(\zeta)$ by the Even-IRA algorithm (cf. [24, pp. 4074ff]; see also section 3). Recall that the spectral transformation $\mu \mapsto (\mu^2 - \zeta^2)^{-1}$ corresponding to the transformation (3.1) causes eigenvalues μ of $\mathcal{L}_P(\lambda)$ close to ζ to be mapped to eigenvalues of large magnitude. In the first place, these will be well approximated by eigenvalues of T_m. Starting with the decomposition

(5.2), the Even-IRA algorithm performs several Krylov–Schur restarts (see [29]) until convergence of the desired number of eigenvalues was observed. As soon as $t_{m+1,m}$ in (5.2) becomes zero, $K(\zeta)V_m = V_m T_m$ holds and all eigenvalues of T_m are exact eigenvalues of $K(\zeta)$. Finally, the reverse transformation (3.2) reveals eigenvalues of $\mathcal{L}_P(\lambda)$ close to ζ.

Now notice that $K(\zeta)$ in (5.1) is nonsingular if and only if X is nonsingular. Therefore, assuming X to be nonsingular, we have

$$\begin{aligned} K(\zeta)^{-1} &= X^{-1}\mathcal{L}_P(\zeta)X^{-1}\mathcal{L}_P(-\zeta) = X^{-1}\big(\zeta X + Y\big)X^{-1}\big(-\zeta X + Y\big) \\ &= -\zeta^2 I_{dn} + X^{-1}YX^{-1}Y = \big(X^{-1}Y\big)^2 - \zeta^2 I_{dn} \end{aligned} \tag{5.3}$$

so that $K(\zeta) = ((X^{-1}Y)^2 - \zeta^2 I_{dn})^{-1}$. For all further considerations we let $G := (X^{-1}Y)$ whenever it exists so that $K(\zeta) = (G^2 - \zeta^2 I_{dn})^{-1}$.

Whenever $X \in \mathbb{R}^{dn\times dn}$ is nonsingular and G exists, (5.3) can be taken into account and (5.2) may be rewritten in terms of G^2 as

$$\left[\begin{array}{|c|}\hline G^2 \\ \hline\end{array}\right]\left[\begin{array}{|c|}\hline V_{m+1} \\ \hline\end{array}\right]\left[\begin{array}{|c|}\hline \underline{T}_m \\ \hline\end{array}\right] = \left[\begin{array}{|c|}\hline V_{m+1} \\ \hline\end{array}\right]\left[\begin{array}{|c|}\hline \underline{H}_m \\ \hline\end{array}\right], \tag{5.4}$$

where $\underline{H}_m := \zeta^2\underline{T}_m + I_{m+1,m} \in \mathbb{R}^{(m+1)\times m}$ is again of upper-Hessenberg form. A decomposition of the form (5.4) is called a generalized rational Arnoldi decomposition for G^2 in [6, eq. (1.1)], so we adapt this terminology here. When working with $K(\zeta) = (G^2 - \zeta^2 I_{dn})^{-1}$ we will mainly consider rational decompositions as in (5.4) instead of Arnoldi decompositions as in (5.2) from now on.

Regarding (5.4), the eigenvalues of the matrix pencil $\lambda T_m - H_m \in \mathbb{R}[\lambda]^{m\times m}$ (where T_m and H_m are given by the first m rows of $\underline{T}_m$ and $\underline{H}_m$, respectively) will in general be good approximations of the eigenvalues of G^2 (see [27]). Certainly it holds that $\sigma(G^2) = \sigma(G)^2 = \sigma(\mathcal{L}_P)^2$, so the square roots $+\sqrt{\theta}$ and $-\sqrt{\theta}$ of eigenvalues $\theta \in \mathbb{C}$ found from $\lambda T_m - H_m$ approximate eigenvalues of $\mathcal{L}_P(\lambda)$ and, in turn, $P(\lambda)$. It is a crucial observation regarding the rational Even-IRA algorithm presented in section 6 that this relationship holds even if G (and hence G^2) does not exist.

Now it is important to notice that, in (5.4), only $\underline{T}_m$ and $\underline{H}_m$ directly depend on ζ but G^2 does not. In comparison to the Arnoldi decomposition (5.2)—where $K(\zeta)$ appears on the left-hand side and explicitly depends on ζ—this enables us to extend the decomposition (5.4) while changing the shift ζ to some newly chosen value $\xi \in \mathbb{C}$. This is not possible for the standard Arnoldi decomposition (5.2) and cannot be realized in the Even-IRA algorithm. In particular, instead of calculating and orthogonalizing $K(\zeta)v_{m+1}$ to extend (5.4) (as in the Arnoldi iteration), we may use the vector $K(\xi)v_{m+1}$ for some new shift ξ.

Remark 5.1. We will assume throughout this section that $G \in \mathbb{R}^{dn\times dn}$ exists since this will be helpful to illustrate the forthcoming computations. This assumption will be dropped in the next section since the regularity of X is actually not required to perform the rational Even-IRA algorithm outlined in section 6.

In section 5.1 we show how the rational Arnoldi decomposition (5.4) can be extended to increase the dimension of the underlying Krylov space (spanned by the columns of V_{m+1}). To this end, we distinguish between the cases where either $\xi \in \mathbb{R}$ or $\xi \in \imath\mathbb{R}$ holds (only in these cases ξ^2 and $K(\xi)$ are real) and where $\xi = a + b\imath$ is complex with nonzero real and imaginary parts (which implies $K(\xi)$ is nonreal). In

the second case, we may still remain in real arithmetics if the real and imaginary parts of $K(\xi)v_{m+1}$ are considered separately.[2]

5.1. The extension of a rational Arnoldi decomposition. Assume we are given a decomposition as in (5.4) obtained from m steps of the Arnoldi iteration applied to $K(\zeta)$ for some shift $\zeta \notin \sigma(\mathcal{L}_P)$. Now let $\xi \notin \sigma(\mathcal{L}_P)$ be some new shift parameter. First assume that either $\xi \in \mathbb{R}$ or $\xi \in \imath\mathbb{R}$ holds, so $K(\xi)$ is a real matrix and $K(\xi)v_{m+1}$ is a real vector of size dn. The Gram–Schmidt-orthogonalization of $K(\xi)v_{m+1}$ against $v_1, \ldots, v_{m+1}$ yields

$$\widetilde{v}_{m+2} = K(\xi)v_{m+1} - \begin{bmatrix} v_1 & \cdots & v_{m+1} \end{bmatrix} \begin{bmatrix} t_{1,m+1} \\ \vdots \\ t_{m+1,m+1} \end{bmatrix}, \tag{5.5}$$

where $t_{i,m+1} = \langle K(\xi)v_{m+1}, v_i \rangle$, $i = 1, \ldots, m+1$, and $v_{m+2} = (t_{m+2,m+1})^{-1}\widetilde{v}_{m+2}$ with $t_{m+2,m+1} = \|\widetilde{v}_{m+2}\|_2$. Then (5.5) can be rearranged to $K(\xi)v_{m+1} = V_{m+2}t_{m+1}$, where $V_{m+2} := [\, V_{m+1} \; v_{m+2} \,] \in \mathbb{R}^{dn\times(m+2)}$ and where $t_{m+1} = [t_{k,m+1}]_{k=1}^{m+2} \in \mathbb{R}^{m+2}$. Putting the expression $K(\xi) = (G^2 - \xi^2 I_{dn})^{-1}$ from (5.3) in use we obtain

$$G^2 V_{m+2} t_{m+1} = v_{m+1} + \xi^2 V_{m+2} t_{m+1}. \tag{5.6}$$

The relation established in (5.6) can now be incorporated into the decomposition (5.4) easily by defining

$$\underline{T}_{m+1} := \left[\begin{array}{ccc|c} & & & t_{1,m+1} \\ & \underline{T}_m & & \vdots \\ & & & t_{m,m+1} \\ & & & t_{m+1,m+1} \\ \hline 0 & \cdots & 0 & t_{m+2,m+1} \end{array}\right] \in \mathbb{R}^{(m+2)\times(m+1)} \tag{5.7}$$

and

$$\underline{H}_{m+1} := \left[\begin{array}{ccc|c} & & & \xi^2 t_{1,m+1} \\ & \underline{H}_m & & \vdots \\ & & & \xi^2 t_{m,m+1} \\ & & & 1 + \xi^2 t_{m+1,m+1} \\ \hline 0 & \cdots & 0 & \xi^2 t_{m+2,m+1} \end{array}\right] \in \mathbb{R}^{(m+2)\times(m+1)}, \tag{5.8}$$

which gives a new decomposition $G^2 V_{m+2}\underline{T}_{m+1} = V_{m+2}\underline{H}_{m+1}$ that has the same structure as in (5.4).

Inspired by [3], we will work with a slightly modified form of the generalized rational Arnoldi decomposition from (5.4) in all further discussions. This decomposition will turn out to be adequate for the realization of the Krylov–Schur restart discussed in section 6. To illustrate the idea, assume that $v_1 \in \mathbb{R}^{dn}$ with $\|v_1\|_2 = 1$ is given and $m = 1$. The computations in (5.5), (5.7), and (5.8) yield

$$V_2 = \begin{bmatrix} v_1 & v_2 \end{bmatrix} \in \mathbb{R}^{dn\times 2}, \quad \underline{T}_1 = \begin{bmatrix} t_{1,1} \\ t_{2,1} \end{bmatrix} \in \mathbb{R}^{2\times 1}, \quad \underline{H}_1 = \begin{bmatrix} h_{1,1} \\ h_{2,1} \end{bmatrix} \in \mathbb{R}^{2\times 1}, \tag{5.9}$$

[2]Notice that the authors from [24] deal with complex shifts in another way by changing the definiton of $K(\zeta)$ (see [24, Rem. 2.1]).

so that $G^2 V_2 \underline{T}_1 = V_2 \underline{H}_1$ holds. Now there exists a Givens rotation $F \in \mathbb{R}^{2\times 2}$ such that the second entry in $F\underline{T}_1$ is zero. Redefining $\underline{T}_1$ as $F\underline{T}_1$, V_2 as $V_2 F^T = [\, v_1 \; v_2 \,]$, and $\underline{H}_1$ as $F\underline{H}_1$, we have computed a new equivalent decomposition $G^2 V_2 \underline{T}_1 = V_2 \underline{H}_1$. Now, notice that the left-hand side can also be expressed as $G^2 V_1 T_1$, where $V_1 = [\, v_1 \,]$ consists only of the first column of V_2 and $T_1 = [\, t_{1,1} \,]$, where $t_{1,1}$ is the first entry of $\underline{T}_1$. In particular, T_1 is now (trivially) an upper-triangular matrix.

For the further extension of the decomposition $G^2 V_1 T_1 = V_2 \underline{H}_1$, it is appropriate and feasible to keep $T_k \in \mathbb{R}^{k\times k}$, $k \geq 2$, in upper-triangular form (instead of upper-Hessenberg form) throughout while the upper-Hessenberg structure of $\underline{H}_k$ is preserved. This can be achieved by applying a special bulge-chasing after every extension step. We describe this procedure in general for a given decomposition of the above form of size $m \geq 1$, i.e.,

$$\left[\begin{array}{|c|}\hline G^2 \\ \hline \end{array}\right] \left[\begin{array}{|c|}\hline V_m \\ \hline \end{array}\right] \left[\begin{array}{|c|}\hline T_m \\ \hline \end{array}\right] = \left[\begin{array}{|c|}\hline V_{m+1} \\ \hline \end{array}\right] \left[\begin{array}{|c|}\hline \underline{H}_m \\ \hline \end{array}\right], \tag{5.10}$$

where $V_{m+1} \in \mathbb{R}^{dn\times(m+1)}$ has orthonormal columns $v_1, \ldots, v_{m+1}$, $T_m \in \mathbb{R}^{m\times m}$ is upper-triangular, and $\underline{H}_m \in \mathbb{R}^{(m+1)\times m}$ has upper-Hessenberg structure.

Let the scalars $t_{k,m+1} \in \mathbb{R}$, $k = 1, \ldots, m+2$, and the vector $v_{m+2} \in \mathbb{R}^{dn}$ be computed as in (5.5) and let $\widehat{V}_{m+2} := [\, V_{m+1} \; v_{m+2} \,]$. The matrices in (5.7) and (5.8) now have the special form

$$\underline{\widehat{T}}_{m+1} := \left[\begin{array}{ccc|c} & & & t_{1,m+1} \\ & T_m & & \vdots \\ & & & t_{m,m+1} \\ \hline 0 & \cdots & 0 & t_{m+1,m+1} \\ 0 & \cdots & 0 & t_{m+2,m+1} \end{array}\right] \in \mathbb{R}^{(m+2)\times(m+1)} \tag{5.11}$$

and

$$\underline{\widehat{H}}_{m+1} := \left[\begin{array}{ccc|c} & & & \xi^2 t_{1,m+1} \\ & \underline{H}_m & & \vdots \\ & & & \xi^2 t_{m,m+1} \\ & & & 1 + \xi^2 t_{m+1,m+1} \\ \hline 0 & \cdots & 0 & \xi^2 t_{m+2,m+1} \end{array}\right] \in \mathbb{R}^{(m+2)\times(m+1)}. \tag{5.12}$$

From $\widehat{V}_{m+2}$ and the upper-Hessenberg matrices in (5.11) and (5.12) we may now recover the structures from (5.10), i.e., $V_{m+2} \in \mathbb{R}^{dn\times(m+2)}$ with orthonormal columns, $T_{m+1} \in \mathbb{R}^{(m+1)\times(m+1)}$ with upper-triangular form, and $\underline{H}_{m+1} \in \mathbb{R}^{(m+2)\times(m+1)}$ with upper-Hessenberg structure, so that $G^2 V_{m+1} T_{m+1} = V_{m+2} \underline{H}_{m+1}$ holds. In fact, two orthogonal matrices $Q \in \mathbb{R}^{(m+2)\times(m+2)}$ and $Z \in \mathbb{R}^{(m+1)\times(m+1)}$ may be found such that $\underline{H}_{m+1} := Q \underline{\widehat{H}}_{m+1} Z \in \mathbb{R}^{(m+2)\times(m+1)}$ and

$$Q \underline{\widehat{T}}_{m+1} Z =: \left[\begin{array}{|c|}\hline T_{m+1} \\ \hline 0 \cdots 0 \\ \hline \end{array}\right] \quad \text{with } T_{m+1} \in \mathbb{R}^{(m+1)\times(m+1)} \text{ upper-triangular.} \tag{5.13}$$

Finally, defining $V_{m+2} := \widehat{V}_{m+2} Q^T$, we obtain the new decomposition

$$G^2 V_{m+1} T_{m+1} = V_{m+2} \underline{H}_{m+1}.$$

This equation is of the same form as (5.10) except that V_{m+1} has been extended by one column—which corresponds to the extension of the underlying Krylov space by one dimension—and that T_{m+1} and $\underline{H}_{m+1}$ have increased in their sizes by one. The matrices Q and Z can be set up as a product of Givens rotations by the bulge-chasing procedure described in Algorithm 5.1 for a real or purely imaginary shift.

Algorithm 5.1 Bulge-chasing-procedure (real/imaginary shift).

1: At first, a Givens rotation is applied to $\widehat{\underline{T}}_{m+1}$ (from the left) on rows $m+2$ and $m+1$ to eliminate $t_{m+2,m}$. Applying this transformation to $\widehat{\underline{H}}_{m+1}$ introduces a bulge in the position $(m+2, m)$. This bulge can be eliminated by applying a Givens rotation (from the right) to $\widehat{\underline{H}}_{m+1}$ acting on columns m and $m+1$. A bulge will now show up in the position $(m+1, m)$ in $\widehat{\underline{T}}_{m+1}$.

2: The bulge in the position $(m+1, m)$ in $\widehat{\underline{T}}_{m+1}$ created in (a) can be eliminated by a Givens rotation applied (from the left) on rows m and $m+1$ of $\widehat{\underline{T}}_{m+1}$. This introduces a new bulge in $\widehat{\underline{H}}_{m+1}$ at the position $(m+1, m-1)$. The elimination of this bulge can be achieved by applying a Givens rotation (from the right) to $\widehat{\underline{H}}_{m+1}$ acting on the columns $m-1$ and m. In consequence, a new bulge will appear in $\widehat{\underline{T}}_{m+1}$ in the position $(m, m-1)$.

3: The elimination process described in steps 1 and 2 continues in the same manner until the bulge in $\widehat{\underline{T}}_{m+1}$ is chased off the top-left corner.

Next, we discuss the case where $\xi \in \mathbb{C}$ has nonzero real and imaginary parts. Let us begin directly with a real rational Arnoldi decomposition as in (5.10). As $K(\xi) \in \mathbb{C}^{dn \times dn}$ is now a complex matrix, the resulting vector $K(\xi)v_{m+1}$ will also be complex (although v_{m+1} is still real). To remain in real arithmetics, we decompose $K(\xi)v_{m+1}$ as $\mathrm{Re}(K(\xi)v_{m+1}) + \mathrm{Im}(K(\xi)v_{m+1})\imath$ into its real and imaginary part. Now we apply the Gram–Schmidt process to both vectors one after the other. That is, for $\mathrm{Re}(K(\xi)v_{m+1})$ we obtain, analogously to (5.5),

$$\widetilde{v}_{m+2} = \mathrm{Re}(K(\xi)v_{m+1}) - \begin{bmatrix} v_1 & \cdots & v_{m+1} \end{bmatrix} \begin{bmatrix} t_{1,m+1} \\ \vdots \\ t_{m+1,m+1} \end{bmatrix} \tag{5.14}$$

with $t_{i,m+1} = \langle \mathrm{Re}(K(\xi)v_{m+1}), v_i \rangle$ and set $v_{m+2} := (t_{m+2,m+1})^{-1}\widetilde{v}_{m+2}$ with $t_{m+2,m+1} = \|\widetilde{v}_{m+2}\|_2$. Having computed v_{m+2}, we may now orthogonalize $\mathrm{Im}(K(\xi)v_{m+1})$ against $v_1, \ldots, v_{m+2}$ to obtain

$$\widetilde{v}_{m+3} = \mathrm{Im}(K(\xi)v_{m+1}) - \begin{bmatrix} v_1 & \cdots & v_{m+2} \end{bmatrix} \begin{bmatrix} t_{1,m+2} \\ \vdots \\ t_{m+2,m+2} \end{bmatrix} \tag{5.15}$$

with $t_{i,m+2} = \langle \mathrm{Im}(K(\xi)v_{m+1}), v_i \rangle$. Again we define $v_{m+3} := (t_{m+3,m+2})^{-1}\widetilde{v}_{m+3}$, where $t_{m+3,m+2} = \|\widetilde{v}_{m+3}\|_2$. Now we set $\widehat{V}_{m+3} := [\, V_{m+1}\ v_{m+2}\ v_{m+3}\,]$,

$$t_{m+1} := \begin{bmatrix} t_{1,m+1} \\ \vdots \\ t_{m+2,m+1} \\ 0 \end{bmatrix} \in \mathbb{R}^{m+3} \quad \text{and} \quad t_{m+2} := \begin{bmatrix} t_{1,m+2} \\ \vdots \\ t_{m+2,m+2} \\ t_{m+3,m+2} \end{bmatrix} \in \mathbb{R}^{m+3}.$$

From (5.14) and (5.15) we obtain $\mathrm{Re}(K(\xi)v_{m+1}) = \widehat{V}_{m+3}t_{m+1}$ and $\mathrm{Im}(K(\xi)v_{m+1}) = \widehat{V}_{m+3}t_{m+2}$, so that $K(\xi)v_{m+1} = \widehat{V}_{m+3}(t_{m+1} + it_{m+2})$ follows. Putting again $K(\xi) = (G^2 - \xi^2 I_{dn})^{-1}$ from (5.3) in use we get

$$G^2\widehat{V}_{m+3}\big(t_{m+1} + \imath t_{m+2}\big) = \widehat{V}_{m+3}\big(e_{m+1} + \xi^2 t_{m+1} + \imath\xi^2 t_{m+2}\big), \tag{5.16}$$

where e_{m+1} denotes the $(m+1)$st unit vector from $\mathbb{R}^{m+3}$. Furthermore, from (5.16) the splitting of ξ^2 as $\xi^2 = \rho + \eta\imath$ with $\rho := \mathrm{Re}(\xi^2)$ and $\eta := \mathrm{Im}(\xi^2)$ yields

$$G^2\widehat{V}_{m+3}\big(t_{m+1} + \imath t_{m+2}\big) = \widehat{V}_{m+3}\big[e_{m+1} + \rho t_{m+1} - \eta t_{m+2} + \imath\big(\eta t_{m+1} + \rho t_{m+2}\big)\big]$$

and, decomposing this once more into its real and imaginary parts, we arrive at

$$G^2\widehat{V}_{m+3}t_{m+1} = \widehat{V}_{m+3}\big(e_{m+1} + \rho t_{m+1} - \eta t_{m+2}\big) \qquad \text{and} \tag{5.17}$$

$$G^2\widehat{V}_{m+3}t_{m+2} = \widehat{V}_{m+3}\big(\eta t_{m+1} + \rho t_{m+2}\big). \tag{5.18}$$

The two relations (5.17) and (5.18) can now be incorporated into the decomposition (5.10). To this end, we define

$$\underline{\widehat{T}}_{m+2} = \left[\begin{array}{ccc|cc} & & & t_{1,m+1} & t_{1,m+2} \\ & T_m & & \vdots & \vdots \\ & & & t_{m,m+1} & t_{m,m+2} \\ \hline 0 & \cdots & 0 & t_{m+1,m+1} & t_{m+1,m+2} \\ 0 & \cdots & 0 & t_{m+2,m+1} & t_{m+2,m+2} \\ 0 & \cdots & 0 & 0 & t_{m+3,m+2} \end{array}\right] \tag{5.19}$$

and

$$\underline{\widehat{H}}_{m+2} \left[\begin{array}{ccc|cc} & & & \rho t_{1,m+1} - \eta t_{1,m+2} & \eta t_{1,m+1} + \rho t_{1,m+2} \\ & \underline{H}_m & & \vdots & \vdots \\ & & & \rho t_{m,m+1} - \eta t_{m,m+2} & \eta t_{m,m+1} + \rho t_{m,m+2} \\ & & & 1 + \rho t_{m+1,m+1} - \eta t_{m+1,m+2} & \eta t_{m+1,m+1} + \rho t_{m+1,m+2} \\ \hline 0 & \cdots & 0 & \rho t_{m+2,m+1} - \eta t_{m+2,m+2} & \eta t_{m+2,m+1} + \rho t_{m+2,m+2} \\ 0 & \cdots & 0 & -\eta t_{m+3,m+2} & \rho t_{m+3,m+2} \end{array}\right]. \tag{5.20}$$

From $\widehat{V}_{m+3}$ and the matrices in (5.19) and (5.20) we may again recover the structures from (5.10), that is, $G^2 V_{m+2}T_{m+2} = V_{m+3}\underline{H}_{m+2}$, where $T_{m+2} \in \mathbb{R}^{(m+2)\times(m+2)}$ is upper-triangular and $\underline{H}_{m+2} \in \mathbb{R}^{(m+3)\times(m+2)}$ has upper-Hessenberg form. As before, a special bulge-chasing procedure is appropriate to determine two orthogonal matrices $Q \in \mathbb{R}^{(m+3)\times(m+3)}$ and $Z \in \mathbb{R}^{(m+2)\times(m+2)}$ such that $\underline{H}_{m+2} := Q\underline{\widehat{H}}_{m+2}Z \in \mathbb{R}^{(m+3)\times(m+2)}$ and

$$Q\underline{\widehat{T}}_{m+2}Z = \left[\begin{array}{c} \\ T_{m+2} \\ \\ \hline 0 \cdots 0 \end{array}\right] \quad \text{where } T_{m+2} \in \mathbb{R}^{(m+2)\times(m+2)} \text{ is upper-triangular.} \tag{5.21}$$

The matrices Q and Z can be set up as a product of Givens rotations by Algorithm 5.2. With $V_{m+3} := \widehat{V}_{m+3}Q^T$ we obtain the desired decomposition.

Algorithm 5.2 Bulge-chasing procedure (complex shift).

1: At first, a Givens rotation is applied to $\underline{\widehat{T}}_{m+2}$ (from the left) on rows $m+1$ and $m+2$ to eliminate $t_{m+2,m+1}$. Subsequently, another Givens rotation is applied to the resulting matrix on rows $m+2$ and $m+3$ to eliminate $t_{m+3,m+2}$. Applying both transformations to $\underline{\widehat{H}}_{m+2}$ introduces a bulge in the positions $(m+2,m)$ and $(m+3,m)$. We now apply two Givens rotations (from the right) to $\underline{\widehat{H}}_{m+2}$ acting on columns m and $m+1$ to eliminate the element in position $(m+3,m)$ and, subsequently, acting on columns $m+1$ and $m+2$ to eliminate the element in position $(m+3,m+1)$. Two new bulges will show up in the positions $(m+1,m)$ and $(m+2,m+1)$ in $\underline{\widehat{T}}_{m+2}$. The additional element in the $(m+2,m)$ position of $\underline{\widehat{H}}_{m+2}$ remains in its position and is eliminated in step 2.

2: The bulges in the positions $(m+1,m)$ and $(m+2,m+1)$ in $\underline{\widehat{T}}_{m+2}$ created in step 1 can be eliminated by two Givens rotations applied (from the left) on rows m and $m+1$ (to eliminate the bulge in position $(m+1,m)$) and on rows $m+1$ and $m+2$ (to eliminate the bulge in position $(m+2,m+1)$) of $\underline{\widehat{T}}_{m+2}$. This introduces new additional nonzero elements in $\underline{\widehat{H}}_{m+2}$ at the position $(m+1,m-1)$ and $(m+2,m-1)$. We apply two subsequent Givens rotations (from the right) to $\underline{\widehat{H}}_{m+2}$ acting on the columns $m-1$ and m (to eliminate the element in $(m+2,m-1)$) and on columns m and $m+1$ (to eliminate the element in $(m+2,m)$). Notice that the $(m+2,m)$-element we eliminate now was the one that remained in step 1. Now new bulges will appear in $\underline{\widehat{T}}_{m+2}$ in the positions $(m,m-1)$ and $(m+1,m)$. The additional element in the $(m+1,m-1)$ position of $\underline{\widehat{H}}_{m+2}$ remains in its position and is eliminated in the next step.

3: The elimination process described in steps 1 and 2 continues in the same manner until the bulge in $\underline{\widehat{T}}_{m+2}$ is chased off the top-left corner.

Starting with some $v_1 \in \mathbb{R}^{dn}, \|v\|_1 = 2$, the previously described procedures are appropriate to construct and extend a rational Arnoldi decomposition of the form (5.10). In each run, a new shift parameter $\xi \notin \sigma(\mathcal{L}_P)$ can be chosen. Per iteration, the decomposition grows in size by one if ξ is real or purely imaginary and by two otherwise.

Now recall that, whenever X is singular, G and, consequently, a decomposition of the form (5.10), does not exist. Nevertheless, the vectors $K(\xi)v_{m+1}$ and v_{m+2} can still be calculated as in (5.5) and the matrices T_m, $\underline{H}_m$ can be extended as in (5.11) and (5.12) if ξ is real or purely imaginary. The bulge-chasing procedure from Algorithm 5.1 applies and recovers the matrix structures from (5.10). If ξ is not real or purely imaginary, $K(\xi)v_{m+1}$ can be split into its real and imaginary parts and the calculations in (5.14) and (5.15) can be carried out as described above. The extension of T_m and $\underline{H}_m$ works as explained in (5.19) and (5.20) and the bulge-chasing procedure from Algorithm 5.2 recovers the upper-triangular and upper-Hessenberg structures.

In conclusion, for any $m \geq 1$, the matrix pencil $\lambda T_m - H_m \in \mathbb{R}^{m\times m}$ can be formed even if G cannot. Moreover, its eigenvalues can be used to approximate the eigenvalues in $\sigma(\mathcal{L}_P)^2$ as before. We will permanently drop the assumption that X needs to be nonsingular and that G^2 needs to exist from now on. In other words, we explicitly allow $\mathcal{L}_P(\lambda)$ to have eigenvalues at infinity. Therefore, the following derivations will mostly be dealing only with the matrices $V_k, \underline{T}_k$, and $\underline{H}_k$ as in (5.10) instead of the

decomposition $G^2V_mT_m = V_{m+1}\underline{H}_m$. These matrices and their modifications in the upcoming section should always be understood in the context of a rational Arnoldi decomposition as in (5.2) whenever such a decomposition exists. We have summarized the method to generate (or extend) a rational Arnoldi decomposition in Algorithm 5.3.

Algorithm 5.3 Rational Arnoldi expansion.

INPUT: The linearization $\mathcal{L}_P(\lambda) = \lambda X + Y \in \mathbb{R}[\lambda]^{dn\times dn}$ for a T-even matrix polynomial $P(\lambda) \in \mathbb{R}[\lambda]^{n\times n}$ defined in (2.4). A matrix $V_{k+1} = [\, v_1 \ \cdots \ v_{k+1}\,] \in \mathbb{R}^{dn\times(k+1)}$ with orthonormal columns, $T_k \in \mathbb{R}^{k\times k}$ upper-triangular, and $\underline{H}_k \in \mathbb{R}^{(k+1)\times k}$ in upper Hessenberg form satisfying $G^2V_kT_k = V_{k+1}\underline{H}_k$ if G^2 exists. In case $k = 0$, we set $T_0 = [\,]$ and $\underline{H}_0 := [\,]$. A number $m \in \mathbb{N}$, $m > k$.
OUTPUT: Matrices $V_{m+1} = [\, v_1 \ \cdots \ v_{m+1}\,] \in \mathbb{R}^{dn\times(m+1)}$ with orthonormal columns, $T_m \in \mathbb{R}^{m\times m}$ upper-triangular, and $\underline{H}_m \in \mathbb{R}^{(m+1)\times m}$ in upper Hessenberg form that satisfy (5.4) in case G^2 exists.
for $j = \ell+1, \ldots, m$ **do**
  pick a shift $\zeta_j \in \mathbb{C}$
  compute $w := K(\zeta_j)v_j = (\mathcal{L}_P(\zeta_j)^{-T}X\mathcal{L}_P(\zeta_j)^{-1}X)v_j$ using section 4
  if $\zeta_j \in \mathbb{R}$ or $\zeta_j \in \imath\mathbb{R}$ **then**
    orthogonalize w against $v_1, \ldots, v_j$ and obtain $t_{1,j}, \ldots, t_{j,j} \in \mathbb{R}$ as in (5.5)
    set v_{j+1} to obtain $t_{j+1,j} \in \mathbb{R}$
    form $\widehat{\underline{T}}_j \in \mathbb{R}^{(j+1)\times j}$ and $\widehat{\underline{H}}_j \in \mathbb{R}^{(j+1)\times j}$ as in (5.11) and (5.12)
    set $\widehat{V}_{j+1} = [\, v_1 \ \cdots \ v_{j+1}\,]$
    apply the bulge-chasing procedure described in Algorithm 5.1 to determine
    orthogonal matrices $Q \in \mathbb{R}^{(j+1)\times(j+1)}$ and $Z \in \mathbb{R}^{j\times j}$ such that
      ▷ $Q\widehat{\underline{T}}_jZ$ is upper-triangular with zeros in its last row and
      ▷ $Q\widehat{\underline{H}}_jZ =: \underline{H}_j$ has upper-Hessenberg structure
    define T_j to be the first j rows of $Q\widehat{\underline{T}}_jZ$
    define $V_{j+1} := \widehat{V}_{j+1}Q^T$
  else
    orthogonalize $\mathrm{Re}(w)$ against $v_1, \ldots, v_j$ and get $t_{1,j}, \ldots, t_{j,j} \in \mathbb{R}$ as in (5.14)
    set v_{j+1} to obtain $t_{j+1,j} \in \mathbb{R}$
    orthogonalize $\mathrm{Im}(w)$ against $v_1, \ldots, v_j, v_{j+1}$ and obtain $t_{1,j+1}, \ldots, t_{j+1,j+1} \in \mathbb{R}$ as in (5.15)
    set v_{j+2} to obtain $t_{j+2,j+1} \in \mathbb{R}$
    form $\widehat{\underline{T}}_{j+1} \in \mathbb{R}^{(j+2)\times(j+1)}$ and $\widehat{\underline{H}}_j \in \mathbb{R}^{(j+2)\times(j+1)}$ as in (5.19) and (5.20)
    set $\widehat{V}_{j+2} = [\, v_1 \ \cdots \ v_{j+1}\ v_{j+2}\,]$
    apply the bulge-chasing procedure described in Algorithm 5.2 to determine
    orthogonal matrices $Q \in \mathbb{R}^{(j+2)\times(j+2)}$ and $Z \in \mathbb{R}^{(j+1)\times(j+1)}$ such that
      ▷ $Q\widehat{\underline{T}}_{j+1}Z$ is upper-triangular with zeros in its last row
      ▷ $Q\widehat{\underline{H}}_{j+1}Z =: \underline{H}_{j+1}$ has upper-Hessenberg structure
    define T_{j+1} to be the first $j+1$ rows of $Q\widehat{\underline{T}}_{j+1}Z$
    define $V_{j+2} := \widehat{V}_{j+2}Q^T$
  end if
end for

6. The rational Even-IRA algorithm. The rational Even-IRA algorithm presented in this section is a method that unifies the Krylov–Schur restart strategy [29] with the spectral-preserving transformation $K(\zeta)$ (see section 3 and [24, 22, 31]) and the shift flexibility offered by the rational Arnoldi process [27, 6, 26]. The sparse and structured form of the linearization $\mathcal{L}_P(\lambda)$ (see Theorem 2.5) is exploited for evaluating matrix-vector-products with $K(\zeta)$ implicitly and efficiently without ever forming $K(\zeta)$ at all (see section 4). Hence, the memory requirement of the method is essentially that of storing the given matrix polynomial and the vectors from the current search space. In a nutshell, this approach yields a powerful Krylov-subspace algorithm for the computation of some eigenvalues for T-even polynomial eigenvalue problems. The rational Even-IRA algorithm presented next consists of several phases. In the initialization phase (section 6.1) a rational Arnoldi decomposition is constructed which is the start and end point of each Krylov–Schur cycle. In the expansion phase (section 6.2) the size of this decomposition is increased. After the expansion, a generalized real Schur decomposition is applied (section 6.3) to identify eigenvalues that have converged during the current run and which are to be locked (section 6.4). To initialize the algorithm's next cycle, the decomposition is truncated (section 6.5) and the upper-triangular and upper-Hessenberg forms of the matrices are recovered (section 6.6). The next iteration then begins with the expansion phase. We now describe the different phases in detail.

6.1. The initialization phase. Let $M \in \mathbb{N}$ be the number of desired eigenvalues for $P(\lambda)$ ($\mathcal{L}_P(\lambda)$, respectively). Matrices

$$\left[\boxed{V_{M+1}}\right] \in \mathbb{R}^{dn\times(M+1)}, \quad \left[\begin{array}{c|c} & \\ \hline & T_M \end{array}\right] \in \mathbb{R}^{M\times M}, \quad \underline{H}_M = \left[\begin{array}{c} H_M \\ \hline B_M \end{array}\right] \in \mathbb{R}^{(M+1)\times M}, \tag{6.1}$$

where $B = h_{M+1,M}e_M^T$ for some scalar $h_{M+1,M} \in \mathbb{R}$, are computed by Algorithm 5.3. Note that the columns of V_{m+1} are orthonormal, T_M is upper-triangular, and H_M has upper-Hessenberg form. In the case X is nonsingular, $(X^{-1}Y)^2V_MT_M = V_{M+1}\underline{H}_M$ holds. If $P(\lambda)$ has no eigenvalues at infinity, Algorithm 5.3 may be initialized with $V_1 = [\,v_1\,]$ (arbitrary and normalized), $T_0 = [\,]$, and $\underline{H}_0 = [\,]$. In the case of the presence of infinite eigenvalues, a different initialization should be chosen (see section 6.7).

A cycle of the rational Krylov–Schur algorithm begins and ends with matrices of the form (6.1). Now suppose, at some stage of the algorithm, $s \in \mathbb{N}_0$ eigenvalues have already converged. Assume these had been locked so that they are located in the top-left $s \times s$ corner of T_M and H_M (of course, beginning with the algorithms first run, $s = 0$).

6.2. The expansion phase. The first step of the algorithm is the expansion phase where the above matrices are extended up to a size $m > M$. This is achieved by performing $m - M$ additional steps of Algorithm 5.3 with the input matrices from (6.1). We call $m - M$ the extension size for the algorithm. Now we obtain matrices

$$\left[\boxed{V_{m+1}}\right] \in \mathbb{R}^{dn\times(m+1)}, \quad \left[\begin{array}{c|c} & \\ \hline & T_m \end{array}\right] \in \mathbb{R}^{m\times m}, \quad \underline{H}_m = \left[\begin{array}{c} H_m \\ \hline B_m \end{array}\right] \in \mathbb{R}^{(m+1)\times m}, \tag{6.2}$$

where $B_m = h_{m+1,m}e_m^T$, T_m is upper-triangular and H_m has upper-Hessenberg structure. We partition the matrices in (6.2) in accordance with the number s of locked

eigenvalues as

(6.3)
$$V_m = \begin{bmatrix} V_s & V \end{bmatrix}, \quad T_m = \begin{bmatrix} T_s & T' \\ 0 & T \end{bmatrix}, \quad H_m = \begin{bmatrix} H_s & H' \\ 0 & H \end{bmatrix} \quad \text{and} \quad B_m = \begin{bmatrix} B_s & B \end{bmatrix},$$

where $T_s, H_s \in \mathbb{R}^{s\times s}$, $B_s^T = [\,0\ \cdots\ 0\,]^T \in \mathbb{R}^s$, and $B^T = [\,0\ \cdots\ 0\ h_{m+1,m}\,]^T \in \mathbb{R}^k$, where we have set $k := m - s$ implying $T, H \in \mathbb{R}^{k\times k}$. Moreover, recall that the eigenvalues from the matrix pair $\lambda T_s - H_s$ are the ones we assumed to be locked.

6.3. The decomposition and reordering phase. We may now enter the decomposition and reordering phase of the algorithm. To this end, we first compute a generalized real Schur decomposition of the matrix pair (T, H). For this purpose, orthogonal matrices $Q_1, Z_1 \in \mathbb{R}^{k\times k}$ can be determined so that $Q_1^T T Z_1 = T^\star \in \mathbb{R}^{k\times k}$ remains upper-triangular while $Q_1^T H Z_1 = H^\star \in \mathbb{R}^{k\times k}$ becomes quasi upper-triangular (with solely 1×1 and 2×2 blocks along its diagonal). At this point, a reordering procedure (see, e.g., [15]) can be applied to $\lambda T^\star - H^\star$ to move unwanted eigenvalues of $\lambda T^\star - H^\star$ into the trailing part of its generalized real Schur decomposition. That is, two additional orthogonal transformations $Q_2, Z_2 \in \mathbb{R}^{k\times k}$ can be found, so that unwanted eigenvalues of $\lambda T^\star - H^\star$ move to the south-east corner of the matrices $T^\diamond := Q_2^T T^\star Z_2 \in \mathbb{R}^{k\times k}$ and $H^\diamond := Q_2^T H^\star Z_2 \in \mathbb{R}^{k\times k}$. Thereby, the matrices $T^\diamond \in \mathbb{R}^{k\times k}$ and $H^\diamond \in \mathbb{R}^{k\times k}$ stay upper-triangular and quasi upper-triangular, respectively. Finally, defining $Q^T := Q_2^T Q_1^T$ and $Z := Z_1 Z_2$, we update (6.3) as follows:

(6.4)
$$\widehat{V}_m = \begin{bmatrix} V_s & VQ \end{bmatrix}, \quad \widehat{T}_m = \begin{bmatrix} T_s & T'Z \\ 0 & T^\diamond \end{bmatrix}, \quad \widehat{H}_m = \begin{bmatrix} H_s & H'Z \\ 0 & H^\diamond \end{bmatrix}, \quad \text{and} \quad \widehat{B}_m = \begin{bmatrix} B_s & B^\diamond \end{bmatrix}$$

with $B^\diamond := B^T Z$. Notice that $B^\diamond$ will now be, in general, a full vector.

6.4. The inspection-of-convergence phase. With (6.4) the inspection-of-convergence phase of the algorithm begins. That is, the leading components of $B^\diamond$ are inspected for convergence and eigenvalues are locked whenever convergence has taken place. Let $H^\diamond = [h_{i,j}]_{i,j}, T^\diamond = [t_{i,j}]_{i,j}$ with $1 \le i, j \le k$ and let $B^\diamond = [\,b_{s+1}\ \cdots\ b_m\,]$. Starting with $r \equiv 1$ we now consider the following cases:

(a) Whenever $h_{r+1,r} = 0$ and $|b_{s+r}|$ is below a given tolerance `tol`, we consider the corresponding eigenvalue $h_{r,r}/t_{r,r}$ as converged. The element b_{s+r} is set to zero and the number r of converged eigenvalues in the current run is increased by one.

(b) Whenever $h_{r+1,r} \neq 0$ but $\|[\,b_{s+r}\ b_{s+r+1}\,]\|_2$ is below the given tolerance `tol`, we consider the pair of complex conjugate eigenvalues corresponding to the 2×2 matrix pencil
$$\lambda \begin{bmatrix} t_{r,r} & t_{r,r+1} \\ 0 & t_{r+1,r+1} \end{bmatrix} + \begin{bmatrix} h_{r,r} & h_{r,r+1} \\ h_{r+1,r} & h_{r+1,r+1} \end{bmatrix}$$
as converged. The elements b_{s+r} and b_{s+r+1} are both set to zero. Finally, the number r of converged eigenvalues in the current run is increased by two.

We repeat the locking of eigenvalues as long as (a) or (b) reveals convergence. Once no further convergence is observed notice that $\widehat{B}_m := [\,0\ \cdots\ 0\ b_{s+r+1}\ \cdots\ b_m\,]$ (where r is now the total number of locked eigenvalues during the current run). The new number of converged eigenvalues in total is now $s^\star = s + r$. If $s^\star \ge M$ (the number of desired eigenvalues), we are done. Otherwise the matrices are truncated to prepare a restart of the algorithm.

Remark 6.1. Notice that a practical implementation of the rational Even-IRA algorithm and its different phases may also include a repetition of the phases described in sections 6.3 and 6.4. In particular, instead of preparing a restart once the above criteria reveal no more converged eigenvalues, we may apply the reordering procedure from [15] another time to move the first unconverged eigenvalue to the southeast corner of the matrix. In a second check according to (a) and (b) above, we might then detect other eigenvalues that meet the convergence conditions and can be locked.

6.5. The truncation phase. If $s^\star < M$, the size of the matrices in (6.4) is now decreased to size $M \times M$ in the truncation phase to initialize a restart of the process. In particular, let $\widehat{V}_M \in \mathbb{R}^{dn\times M}$ be the first M columns of $\widehat{V}_m$ and $\widehat{V}_{M+1} = [\,\widehat{V}_M\ v_{m+1}\,]$, where v_{m+1} denotes the last column from V_{m+1} in (6.2) (note that v_{m+1} has not been touched in all steps up to this point). Moreover, denote the top-left $M \times M$ submatrices of $\widehat{T}_m$ and $\widehat{H}_m$ by $\widehat{T}_M$ and $\widehat{H}_M$, respectively, and the vector obtained from the first M components of $\widehat{B}_m$ by $\widehat{B}_M$, i.e., $\widehat{B}_M = [\,0\ \cdots\ 0\ b_{s^\star+1}\ \cdots\ b_M\,]$. In the form (6.2) we have

$$\left[\boxed{\widehat{V}_{M+1}}\right] \in \mathbb{R}^{dn\times(M+1)},\quad \left[\begin{array}{c|c} & \\ \hline & \widehat{T}_M \end{array}\right] \in \mathbb{R}^{M\times M},\quad \underline{\widehat{H}}_M := \left[\begin{array}{c} \widehat{H}_M \\ \hline \widehat{B}_M \end{array}\right] \in \mathbb{R}^{(M+1)\times M}.$$

Notice that $\underline{\widehat{H}}_M$ will not have Hessenberg structure at this stage of the algorithm since $\widehat{B}_M$ will have more nonzero elements than just b_M.

Remark 6.2. It is unfortunate to truncate the matrices as above whenever the $(M+1, M)$-element in $\widehat{H}_m$ is nonzero. In this case a 2×2 block is split which should be avoided by decreasing or increasing M by one.

Analogously to (6.3) and (6.4) we may now partition $\widehat{V}_M, \widehat{T}_M$, and $\underline{\widehat{H}}_M$ according to the new number $s^\star$ of locked and converged Ritz values. This highlights the active part of the decomposition and separates it from the locked part (which does not need to be touched again). In particular, we partition

(6.5)
$$\widehat{V}_M = \begin{bmatrix} V_{s^\star} & V^\circ \end{bmatrix},\quad \widehat{T}_M = \begin{bmatrix} T_{s^\star} & T'' \\ 0 & T^\circ \end{bmatrix},\quad \widehat{H}_M = \begin{bmatrix} H_{s^\star} & H'' \\ 0 & H^\circ \end{bmatrix},\quad \widehat{B}_M = \begin{bmatrix} B_{s^\star} & B^\circ \end{bmatrix},$$

where $V_{s^\star} \in \mathbb{R}^{dn\times s^\star}, T_{s^\star}, H_{s^\star} \in \mathbb{R}^{s^\star\times s^\star}$, and $B_{s^\star}^T = [0\ \cdots\ 0]^T \in \mathbb{R}^{s^\star}$. Recall that B° is in general a full vector with all nonzero entries. Set $k^\star = M - s^\star$ so that $T^\circ, H^\circ \in \mathbb{R}^{k^\star\times k^\star}$.

6.6. The recovery phase. Our next goal is to tranform the matrices in (6.5) back to a decomposition of the form (6.1) in the recovery phase. That is, we determine orthogonal matrices $Q, Z \in \mathbb{R}^{k^\star\times k^\star}$ such that $Q^T T^\circ Z =: T \in \mathbb{R}^{k^\star\times k^\star}$ is still upper-triangular, $Q^T H^\circ Z =: H \in \mathbb{R}^{k^\star\times k^\star}$ remains in upper-Hessenberg form and $B = B^\circ Z =: h_{M+1,M} e_{k^\star}^T$ for some scalar $h_{M+1,M} \in \mathbb{R}$. Then we update (6.5) to obtain

(6.6)
$$V_M = \begin{bmatrix} V_{s^\star} & V^\circ Q \end{bmatrix},\quad T_M = \begin{bmatrix} T_{s^\star} & T''Z \\ 0 & T \end{bmatrix},\quad H_M = \begin{bmatrix} H_{s^\star} & H''Z \\ 0 & H \end{bmatrix},\ B_M = \begin{bmatrix} B_{s^\star} & B \end{bmatrix}$$

and with $V_{M+1} = [\,V_M\ v_{m+1}\,]$, we have matrices

(6.7)
$$\left[\boxed{V_{M+1}}\right] \in \mathbb{R}^{dn\times(M+1)},\quad \left[\begin{array}{c|c} & \\ \hline & T_M \end{array}\right] \in \mathbb{R}^{M\times M},\quad \underline{H}_M = \left[\begin{array}{c} H_M \\ \hline B_M \end{array}\right] \in \mathbb{R}^{(M+1)\times M}$$

as in (6.1), where $B_M = h_{M+1,M} e_M^T$. The next cycle of the algorithm then begins with the expansion phase as described in section 6.2. The recovery phase can be carried out by the bulge-chasing process described in section 6.9.

The overall goal of this algorithm is to achieve $B_M = [\,0 \ \cdots \ 0\,]$ in (6.7) after some cycles of the restarting procedure described above. As soon as this situation takes place, the M eigenvalues of $\lambda T_M - H_M$ are exact eigenvalues of $\mathcal{L}_P(\lambda)^2$ and, in turn, their plus/minus square roots exact eigenvalues of $P(\lambda)$.

6.7. The eigenvalue infinity. A matrix polynomial $P(\lambda)$ might have eigenvalues at infinity (see section 2). The rational Even-IRA algorithm will eventually detect infinite eigenvalues, i.e., in computations in real arithmetic, eigenvalues of very large magnitude might be found. This is detrimental for the algorithm's performance since (i) the detection of very large eigenvalues is, in this case, a wrong result, and (ii) the convergence results after the detection of such an eigenvalue are of unsatisfying accuracy. Therefore, it seems reasonable to a priori eliminate any possibility of convergence to infinity. This will guarantee a good performance throughout and reliable results.

Assuming $P(\lambda) = \sum_{k=1}^{d} P_k \lambda^k \in \mathbb{R}[\lambda]^{n\times n}$ of degree $d \geq 1$ is regular, the eigenvectors for the eigenvalue $\mu = \infty$ are the nullvectors of P_d. These can be found by solving the $n \times n$ linear system $P_d x = 0$ with an appropriate method. These vectors can now be used to initialize our algorithm so that convergence for the eigenvalue infinity has already taken place. For this purpose, let $\dim(\mathrm{null}(P_d)) = t$ and $\{v_1, v_2, \ldots, v_t\} \subset \mathbb{R}^n$ for some orthonormal basis of $\mathrm{null}(P_d)$. We define

$$V_{t+1} = \left[\begin{array}{ccc|c} v_1 & \cdots & v_t & \\ 0 & & 0 & \\ \vdots & & \vdots & v_{t+1} \\ 0 & \cdots & 0 & \end{array}\right] \in \mathbb{R}^{dn\times(t+1)}, \quad \underline{H}_t = \left[\begin{array}{c} I_t \\ \hline 0 \ \cdots \ 0 \end{array}\right],$$

and $T_t = 0_{t\times t} \in \mathbb{R}^{t\times t}$. The vector $v_{t+1} \in \mathbb{R}^{dn}$ can be chosen arbitrarily so that the columns of V_{t+1} are orthonormal. From here on, we start the initialization phase of the rational Even-IRA algorithm described in section 6 with V_{t+1}, T_t, and $\underline{H}_t$ in Algorithm 5.3. Moreover, we define, right from this point on, the number s of converged eigenvalues to be t. In other words, with this initialization, convergence to infinity and locking has already occurred before the algorithm actually starts. The algorithm will not reveal further eigenvalues at infinity. The number of desired eigenvalues has to be increased from M to $t + M$.

6.8. The shift strategy. The appropriate choice of shifts is a delicate problem that often depends on user-specified priorities. According to section 4, a matrix-vector-multiplication with $K(\zeta) \in \mathbb{C}^{dn\times dn}$ essentially reduces to a system solve with $P(\zeta)$ (and $P(\zeta)^T$). If an LU decomposition of $P(\zeta)$ is computed, it can be reused as long as the shift does not change. One the other hand, every change of shift requires the computation of a (costly) new decomposition. Hence, there is a trade-off between the acceleration of convergence obtained by changing to a new (good) shift and the computational cost involved with the shift change. Two general shift strategies are given below.

(a) Assume the current run of the rational Even-IRA algorithm revealed convergence and (in total) $s^\star$ eigenvalues are locked—this corresponds to the situation (6.5). Let $T^\circ = [t^\circ_{i,j}]_{i,j}, H^\circ = [h^\circ_{i,j}]_{i,j}$, $B^\circ = [\,b_{s^\star+1} \ \cdots \ b_M\,]$ and consider the case $h^\circ_{2,1} = 0$. In particular, assuming G^2 exists and regarding

$\widehat{V}_{s^\star+1}$, $\widehat{T}_{s^\star+1}$, and $\widehat{H}_{s^\star+1}$ we have in view of (6.5)

$$G^2\widehat{V}_{s^\star+1}\widehat{T}_{s^\star+1} = \widehat{V}_{s^\star+1}\widehat{H}_{s^\star+1} + b_{s^\star+1}v_{m+1}e_{s^\star+1}^T.$$

In other words,

$$\|G^2\widehat{V}_{s^\star+1}\widehat{T}_{s^\star+1} - \widehat{V}_{s^\star+1}\widehat{H}_{s^\star+1}\|_2 = |b_{s^\star+1}| \tag{6.8}$$

because $\|v_{m+1}\|_2 = 1$. Therefore, the absolute value of the first entry $b_{s^\star+1}$ of B° displays the first residual which was not below the given tolerance `tol` since, otherwise, the corresponding eigenvalue $\xi := h^\circ_{1,1}/t^\circ_{1,1}$ of $\lambda T^\circ - H^\circ$ located in the top-left 1×1 block would have been identified as converged. Nevertheless, ξ may serve as a good approximation to the next eigenvalue that is about to converge and ξ might now be chosen as the next shift parameter. Analogously, whenever $h^\circ_{2,1} \neq 0$, an eigenvalue of the 2×2 top-left corner of $\lambda T^\circ + H^\circ$ can be chosen as a new shift.

(b) The shift strategy from (a) can be modified so that the next shift parameter is chosen as $\xi = h^\circ_{1,1}/t^\circ_{1,1}$ only if the corresponding residual $|b_{s^\star+1}|$ is above a given tolerance. In particular, if $|b_{s^\star+1}|$ is already very small, a change of shift is probably not necessary since the algorithm seems to be "on the right way" to reveal the next convergence soon (e.g., within the next cycle). However, $|b_{s^\star+1}|$ being above a given tolerance might indicate that the current shift is not heading off to reveal further convergence in the near future. Thus, changing the shift could be an appropriate means to speed up convergence in such a situation.

Certainly, other shift strategies beside (a) and (b) above and mixtures of both are conceivable. In particular, if one is interested in eigenvalues in a particular region of the complex plane, the shift should, of course, be chosen appropriately.

6.9. The recovery phase. We now consider the recovery phase of the rational EVEN-IRA algorithm in detail. Therefore, reconsider the matrices obtained in (6.5). We now show how to construct two orthogonal matrices $Q, Z \in \mathbb{R}^{k^\star \times k^\star}$ such that $Q^T T^\circ Q = T \in \mathbb{R}^{k^\star \times k^\star}$ remains upper-triangular, $Q^T H^\circ Z = H \in \mathbb{R}^{k^\star \times k^\star}$ has upper-Hessenberg form, and $B^\circ Z = [\,0 \ \cdots \ 0 \ h_{M+1,M}\,]$ is a vector of zeros except for some scalar $h_{M+1,M} \in \mathbb{R}$ in the last position. The matrices Q and Z are the products of a sequence of Givens rotations that constitute our bulge-chasing procedure. A Givens rotation $\tilde{Q}^T \in \mathbb{R}^{2\times 2}$ from the left acts on two rows i and j (with $1 \le i, j \le k^\star$) of T° and H°. Each transformation $\tilde{Q}^T$ needs to be applied via $\tilde{Q}$ to the columns i and j of V°, too. This is implicitly understood in all the following derivations. A Givens rotation $\tilde{Z} \in \mathbb{R}^{2\times 2}$ from the right acts on two columns i and j, $1 \le i, j \le k^\star$, of T°, H°, and B°. These transformations do not influence the matrix V°.

Now let $B^\circ = [\,b_1 \ \cdots \ b_{k^\star}\,]$. The bulge-chasing process proceeds as follows:

(a) We apply a Givens rotation Z_1 from the right on the first two columns to eliminate b_1 using b_2. This introduces a bulge in the position $(3,1)$ in H° and in the position $(2,1)$ in T°. A rotation Q_1^T acting on rows one and two from the left can be used to eliminate the bulge in T°. The bulge on the second subdiagonal in H° remains in its position. Now the first element in B° is zero and the analogous process can be used to eliminate the second element in B°. As before, the new bulge in the position $(4,2)$ in H° (i.e., on the second subdiagonal in H°) remains it its position.

(b) The third element in B° can be eliminated as in (a) above and two new elements in the positions $(4,3)$ in T° and $(5,3)$ in H° show up. As in (a), this

bulge in H° is accepted for the moment. However, with the elimination of the bulge in T° with a Givens rotations from the left on rows three and four an additional bulge in H° will appear in the position $(4,1)$ (i.e., on the third subdiagonal in H°). This bulge can be eliminated by a Givens rotation from the right on the first and second column of H° introducing again a bulge in the $(2,1)$ position in T°. A rotation applied to the first two rows from the left is used to eliminate the bulge in T°.

(c) The process from (b) now continues for all $t > 3$. That is, the elimination of the tth entry in B° is achieved by a Givens rotation from the right on columns t and $t+1$. Consequently, bulges appear in $(t+2,t)$ in H° and $(t+1,t)$ in T°. The elimination of the bulge in T° by a Givens rotation from the left introduces an additional bulge $(t+1,t-2)$ in H°. This bulge is chased off the top-left corner of T° and H° by applying a Givens rotation alternatingly from left and right.

If the bulge-chasing process described in (a) to (c) is completely carried out, in the end, T° is still of upper-triangular form, $B^\circ = [\,0 \ \cdots \ 0 \ h_{M+1,M}\,]$, and H° is a matrix that now has two full subdiagonals (i.e., all entries below the second subdiagonal of H° are zero). Now the transformation process can be continued and the second subdiagonal in H° can be eliminated from the lower right corner to the top-left corner. A standard bulge-chasing (Givens rotations alternatingly from left and right) is adequate to achieve this. It is important to note that no Givens rotation is required that touches the last column of T°, H°, and B°. Therefore, B° remains as it is and we obtain the desired form.

7. Numerical experiments. In this section, we briefly describe the results of numerical experiments for two T-even eigenproblems to give a proof of concept for the algorithm described in the previous section. To this end, we set up a basic implementation of the rational Even-IRA algorithm in MATLAB R2020a and compared our results to those found with the MATLAB function `polyeig`. As the degree of $P(\lambda)$ is even in both examples, $\mathcal{L}_P(\lambda)$ was constructed as in (2.4) with $M_P(\lambda)$ from Definition 2.3(b). We initialize the algorithm as explained in section 6.7. In contrast to the computation of eigenvalues with `polyeig`, the rational Even-IRA algorithm is designed to find only a few eigenvalues of a matrix polynomial. Therefore, a comparison of the computational times for both algorithms seems inappropriate here.

Our first example is taken from [23]; see also `butterfly` in [7]. Here, the matrix polynomial $P(\lambda) = \sum_{j=0}^{4} P_j \lambda^j$ under consideration is of degree four. The matrix coefficients are built from several Kronecker products as follows: we set $m = 10$ and $n = m^2 = 100$. Let N denote the $m \times m$ nilpotent Jordan matrix with ones on the first subdiagonal and define $\tilde{P}_0 = (1/6)(4I_m + N + N^T), \tilde{P}_1 = N - N^T$, $\tilde{P}_2 = -(2I_m - N - N^T)$, $\tilde{P}_3 = \tilde{P}_1$, and $\tilde{P}_4 = -\tilde{P}_2$. Moreover, we set

$$P_i = c_{i1} I_m \otimes \tilde{P}_i + c_{i2} \tilde{P}_i \otimes I_m$$

with positive constants c_{ij} chosen as $c_{01} = 0.6, c_{02} = 1.3, c_{11} = 1.3, c_{12} = 0.1, c_{21} = 0.1, c_{22} = 1.2, c_{31} = c_{32} = c_{41} = c_{42} = 1.0$ (as in [23]). Now the matrix polynomial $P(\lambda) = \sum_{j=0}^{4} P_j \lambda^j$ has size 100×100. We intend to find the 12 eigenvalues of largest magnitude. We ran several experiments with different initial shifts where either the shift was changed according to the procedure described in section 6.8 (where a change was initialized if the first nonzero residual is not less than 10^{-5}; see (6.8)) or the shift was fixed at the start and not allowed to change during the iterations. An eigenvalue is

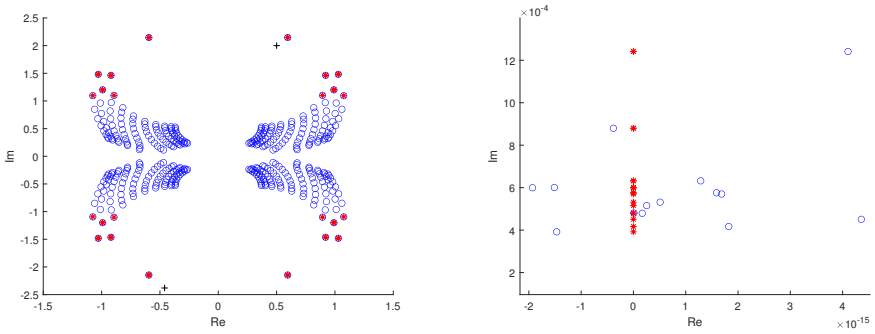

FIG. 1. *Left:* 24 *eigenvalues found by the rational* EVEN-IRA *algorithm (red stars) for the example* `butterfly` *in* [7]. *Blue circles indicate the eigenvalues computed via* `polyeig`. *Due to the preservation of* $\pm$ *matching eigenvalue pairs, only twelve eigenvalues were required to be computed by the rational* EVEN-IRA *algorithm. Crosses indicate the two shifts that have been used. Right:* 14 *eigenvalues (in the upper half plane) found by the rational* EVEN-IRA *algorithm (red stars) for* $\operatorname{rev} P(\lambda)$ *with* $P(\lambda) = \lambda^2 M + \lambda G + K$ *for a gyroscopic system (cf.* [4, sect. 4.2]*). Blue circles correspond to eigenvalues of* $\operatorname{rev} P(\lambda)$ *computed via* `polyeig`. *In contrast to* `polyeig`, *the rational* EVEN-IRA *algorithm recognizes the fact that the eigenvalues are all located on the imaginary axis.*

considered as converged if its corresponding residual becomes less than 10^{-9}. Within our experiments, we observed the following typical behavior of the algorithm.

- With the initial shift $\zeta \in \mathbb{C}$ chosen as $0.5 + 2\imath$, a change of shift takes place once and the rational EVEN-IRA algorithm finds the eigenvalues displayed in Figure 1 (left plot) in 18 iterations. If no shift change is incorporated, the algorithm takes just three more iterations.
- The initial shift $0.5 - 1\imath$ will reveal convergence of all desired eigenvalues in 18 iterations, thereby changing the shift once. Now, if a change of shift is not incorporated, more than 30 iterations are required to achieve full convergence of all those eigenvalues. Thus, in general, the option for changing the shift takes a "bad" initial shift to a region that more easily admits convergence.
- Finally, taking $1 + 3\imath$ as the initial shift, the rational EVEN-IRA algorithm converges in 19 iterations without initializing any change of shift at all.

In all these experiments, we observe an accordance in both the real and imaginary parts of the computed values to those found by the MATLAB function `polyeig` of at least the first ten decimal places.

Remark 7.1. Recall that the spectral transformation of $\mathcal{L}_P(\zeta) = \zeta X + Y$ to

$$K(\zeta) = \mathcal{L}_P(\zeta)^{-T} X \mathcal{L}_P(\zeta)^{-1} X$$

preserves $\pm$ matching pairs of eigenvalues $(+\mu, -\mu)$ as both are mapped to the same eigenvalue $\theta = (\mu^2 - \zeta^2)^{-1}$ (see section 3). Therefore, each eigenvalue of $K(\zeta)$ has even multiplicity. In exact arithmetic, multiple eigenvalues will not be captured (see [24, sect. 3]) by the Arnoldi iteration. However, as round-off may eventually create them, the authors of the EVEN-IRA algorithm suggest an additional X-orthogonalization of the Krylov basis (see [24, Lem. 2.3]). Requiring that the basis of the underlying Krylov space is X-orthogonal (that is, $\langle v_i, X v_j \rangle = 0$ for all $i \neq j$) will hinder the algorithm to find multiple copies of the same eigenvalue. The X-orthogonalization procedure suggested in [24] cannot be directly applied here because the rational EVEN-IRA algorithm as outlined in section 6 handles complex shifts differently.

For our second example we chose the model of a rolling tire (see [10] or [4, sect. 4.2.2]. Here $P(\lambda) = \lambda^2 M + \lambda G + K$, where M, G, K are of size 2697×2697. The matrices M and K are symmetric whereas G is skew-symmetric. Moreover, M and K are positive definite, which implies that $P(\lambda)$ has eigenvalues exclusively on the imaginary axis (see [21, sect. 1]). Those vary in magnitude from about 10^3 to $5 \cdot 10^5$. Here we intend to find the eigenvalues of smallest magnitude. To this end, we consider $\operatorname{rev} P(\lambda) = \lambda^2 K + \lambda G + M$ since the eigenvalues of $\operatorname{rev} P(\lambda)$ of largest magnitude correspond via their reciprocals to the eigenvalues of $P(\lambda)$ of smallest magnitude. We have applied the rational Even-IRA to $\operatorname{rev} P(\lambda)$ with the same parameters as in the previous example, an initial shift of $10^{-2}\imath$ and the shift strategy from section 6.8 to find 14 eigenvalues of $\operatorname{rev} P(\lambda)$ of largest magnitude. The eigenvalues of $\operatorname{rev} P(\lambda)$ computed with the rational Even-IRA algorithm and `polyeig` are displayed in Figure 1 (right plot). Eight restarts have been performed. The imaginary parts of the eigenvalues found by the MATLAB function `polyeig` and those values on the imaginary axis found by the rational Even-IRA algorithm coincide to at least ten significant digits. Clearly, in contrast to `polyeig`, the rational Even-IRA algorithm anticipates the fact that all eigenvalues are located on the imaginary axis.

Remark 7.2. The shift strategy explained in section 6.8 does not perform optimally for finding the eigenvalues of $P(\lambda)$ with smallest magnitude directly, i.e., when $P(\lambda)$ instead of $\operatorname{rev} P(\lambda)$ is used. The shift often increases during the algorithm's run and tends to find eigenvalues of larger magnitudes. Thus, according to our experiments, the basic shift strategy from section 6.8 is not appropriate in this situation and a more sophisticated strategy has to be used.

In conclusion, the overall success of our algorithm depends in large amounts on the chosen shift strategy. The method described in section 6.8 works well if one is interested in accelerating the convergence. However, if certain areas of the complex plane are to be "scanned" for eigenvalues, a more subtle shift-technique is needed. This is not further discussed here.

8. Conclusions. In this work, we have presented a method to compute parts of the spectrum of a T-even matrix polynomial. We developed our algorithm on the basis of the Even-IRA algorithm from [24] which is a method for computing a few eigenvalues of a T-even (i.e., symmetric/skew-symmetric) matrix pencil and the ideas developed in [3] on the rational SHIRA algorithm. Given a T-even matrix polynomial, we introduced a special linearization $\mathcal{L}_P(\lambda)$ for $P(\lambda)$ to preserve its T-even structure. We showed that the specific block-structure and sparsity of $\mathcal{L}_P(\lambda) = \lambda X + Y$ enables us to solve systems $\mathcal{L}_P(\zeta)x = y$ in an efficient way. We applied this technique to accelerate the computation of matrix-vector-products for the matrix $K(\zeta) = \mathcal{L}_P(\zeta)^{-T} X \mathcal{L}_P(\zeta)^{-1} X$ to build the underlying Krylov space. An eigenvalue θ of $K(\zeta)$ gives rise to a $\pm$ matching pair of eigenvalues $+\sqrt{(1/\theta) + \zeta^2}$ and $-\sqrt{(1/\theta) + \zeta^2}$ of $\mathcal{L}_P(\lambda)$. As suggested in [24], we used this spectral transformation (i.e., the matrix $K(\zeta)$) and the implicitly restarted Krylov–Schur algorithm to find eigenvalues of $K(\zeta)$. Moreover, we modified the Even-IRA algorithm and turned it into a rational method that is able to handle changes of the shift parameter during the iteration.

A question for future work that is naturally related to our algorithm is the existence of a compact representation of the Krylov basis similar to the one developed in [30] for other types of linearizations (which are not of the same form as $\mathcal{L}_P(\lambda)$). This would be an appropriate means to decrease the cost for storing the Krylov basis vectors.

Acknowledgments. Work on this manuscript started when all three authors visited the Courant Institute of New York University. We would like to give a special thanks to our host Michael Overton, who made this research stay possible! Moreover, we thank two anonymous referees for their valuable comments and suggestions on this work.

REFERENCES

[1] Z. Bai, D. Demmel, J. Dongarra, A. Ruhe, and H. Van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems*, SIAM, Philadelphia, 2000.
[2] M. Bassour, *Hamiltonian polynomial eigenvalue problems*, J. Appl. Math. Phys., 8 (2020), pp. 609–619.
[3] P. Benner and C. Effenberger, *A rational SHIRA method for the Hamiltonian eigenvalue problem*, Taiwanese J. Math., 14 (2010), pp. 805–823.
[4] P. Benner, H. Fassbender, and M. Stoll, *Solving large-scale quadratic eigenvalue problems with Hamiltonian eigenstructure using a structure-preserving Krylov subspace method*, Electron. Trans. Numer. Anal., 29 (2008), pp. 212–229.
[5] P. Benner, H. Fassbender, and M. Stoll, *A Hamiltonian Krylov–Schur-type method based on the symplectic Lanczos process*, Linear Algebra Appl., 435 (2011), pp. 578–600.
[6] M. Berljafa and S. Güttel, *Generalized rational Krylov decompositions with an application to rational approximation*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 894–916.
[7] T. Betcke, N. J. Higham, V. Mehrmann, and F. Tisseur, *NLEVP: A collection of nonlinear eigenvalue problems*, ACM Trans. Math. Software, 39 (2011).
[8] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
[9] F. M. Dopico, P. W. Lawrence, J. Pérez, and P. Van Dooren, *Block Kronecker linearizations of matrix polynomials and their backward errors*, Numer. Math., 140 (2018), pp. 373–426.
[10] K. Elssel and H. Voss, *Reducing huge gyroscopic eigenproblems by automated multi-level substructuring*, Arch. Appl. Mech., 76 (2006), pp. 171–179.
[11] H. Fassbender and P. Saltenberger, *On vector spaces of linearizations for matrix polynomials in orthogonal bases*, Linear Algebra Appl., 525 (2017), pp. 59–83.
[12] H. Fassbender and P. Saltenberger, *On a modification of the EVEN-IRA algorithm for the solution of T-even polynomial eigenvalue problems*, in Proceedings in Applied Mathematics and Mechanics (PAMM), 2018.
[13] N. J. Higham, D. S. Mackey, and F. Tisseur, *The conditioning of linearizations of matrix polynomials*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 1005–1028.
[14] N. J. Higham, D. S. Mackey, N. Mackey, and F. Tisseur, *Symmetric linearizations for matrix polynomials*, SIAM J. Matrix Anal. Appl., 29 (2006), pp. 143–159.
[15] B. Kagstrom, *A direct method for reordering eigenvalues in the generalized real Schur form of a regular matrix pair* (A, B), in Linear Algebra for Large Scale and Real-Time Applications, M. S. Moonen, G. H. Golub, and B. L. R. De Moor, eds., Kluwer Academic Publishers, Amsterdam, 1993, pp. 195–218.
[16] D. S. Mackey, *Structured Linearizations for Matrix Polynomials*, MIMS EPrint 2006.68, Mancheter Institute for Mathematical Sciences, Manchester, 2006.
[17] D. S. Mackey and V. Perovic, *Linearizations of matrix polynomials in Bernstein bases*, Linear Algebra Appl., 501 (2016), pp. 162–197.
[18] D. S. Mackey and V. Perovic, *Linearizations of matrix polynomials in Newton bases*, Linear Algebra Appl., 556 (2018), pp. 1–45.
[19] D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann, *Structured polynomial eigenvalue problems: good vibrations from good linearizations*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 1029–1051.
[20] D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann, *Vector spaces of linearizations for matrix polynomials*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 971–1004.
[21] K. Meerbergen and F. Tisseur, *The quadratic eigenvalue problem*, SIAM Rev., 43 (2001), pp. 235–286.
[22] V. Mehrmann and D. Watkins, *Structure-preserving methods for computing eigenpairs of large sparse skew-Hamiltoninan/Hamiltonian pencils*, SIAM J. Sci. Comput., 22 (2001), pp. 1905–1925.
[23] V. Mehrmann and D. Watkins, *Polynomial eigenvalue problems with Hamiltonian structure*, Electron. Trans. Numer. Anal., 13 (2002), pp. 106–118.

[24] V. Mehrmann, C. Schröder, and V. Simoncini, *An implicitly-restarted Krylov subspace method for real symmetric/skew-symmetric eigenproblems*, Linear Algebra Appl., 436 (2012), pp. 4070–4087.
[25] C. B. Moler and G. W. Stewart, *An algorithm for generalized matrix eigenvalue problems*, SIAM J. Numer. Anal., 10 (1973), pp. 241–256.
[26] A. Ruhe, *Rational Krylov sequence methods for eigenvalue computation*, Linear Algebra Appl., 58 (1984), pp. 391–405.
[27] A. Ruhe, *Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils*, SIAM J. Sci. Comput., 19 (1998), pp. 1535–1551.
[28] P. Saltenberger, *On Different Concepts for the Linearization of Matrix Polynomials and Canonical Decompositions of Structured Matrices with Respect to Indefinite Sesquilinear Forms*, Logos Verlag, Berlin, 2019.
[29] G. W. Stewart, *A Krylov–Schur algorithm for large eigenproblems*, SIAM J. Matrix Anal. Appl., 23 (2002), pp. 601–614.
[30] R. Van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for nonlinear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 820–838.
[31] D. Watkins, *On Hamiltonian and symplectic Lanczos processes*, Linear Algebra Appl., 385 (2004), pp. 23–45.