NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2016; **23**:607–628
Published online 3 March 2016 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nla.2043

A rank-exploiting infinite Arnoldi algorithm for nonlinear eigenvalue problems

Roel Van Beeumen[1,*,†], Elias Jarlebring[2] and Wim Michiels[1]

[1]*Department of Computer Science, KU Leuven, University of Leuven, 3001 Heverlee, Belgium*
[2]*Department of Mathematics, NA group, KTH Royal Institute of Technology, 100 44 Stockholm, Sweden*

SUMMARY

We consider the nonlinear eigenvalue problem $M(\lambda)x = 0$, where $M(\lambda)$ is a large parameter-dependent matrix. In several applications, $M(\lambda)$ has a structure where the higher-order terms of its Taylor expansion have a particular low-rank structure. We propose a new Arnoldi-based algorithm that can exploit this structure. More precisely, the proposed algorithm is equivalent to Arnoldi's method applied to an operator whose reciprocal eigenvalues are solutions to the nonlinear eigenvalue problem. The iterates in the algorithm are functions represented in a particular structured vector-valued polynomial basis similar to the construction in the infinite Arnoldi method [Jarlebring, Michiels, and Meerbergen, Numer. Math., 122 (2012), pp. 169–195]. In this paper, the low-rank structure is exploited by applying an additional operator and by using a more compact representation of the functions. This reduces the computational cost associated with orthogonalization, as well as the required memory resources. The structure exploitation also provides a natural way in carrying out implicit restarting and locking without the need to impose structure in every restart. The efficiency and properties of the algorithm are illustrated with two large-scale problems. Copyright © 2016 John Wiley & Sons, Ltd.

Received 22 July 2014; Revised 29 September 2015; Accepted 3 February 2016

KEY WORDS: nonlinear eigenvalue problem; Arnoldi method; low-rank

1. INTRODUCTION

Suppose $\Omega \subset \mathbb{C}$ is an open subset of the complex plane containing the origin and let $M : \Omega \to \mathbb{C}^{n\times n}$ be a matrix with elements that are analytic in Ω. We will consider the problem of finding $(\lambda, x) \in \Omega \times \mathbb{C}^n \backslash \{0\}$ such that

$$M(\lambda)x = 0. \tag{1.1}$$

This nonlinear eigenvalue problem occurs in many situations. For instance, they arise in the study of stability of higher-order differential equations where they give rise to quadratic and polynomial eigenvalue problems [1, 2]; in the study of delay-differential equations [3]; and in the study of fluid-solid interaction where $M(\lambda)$ contains rational functions [4]. There are also problems involving boundary integral operators [5]. For summary works and benchmark collections on nonlinear eigenvalue problems, we refer to [6–8].

There are many algorithms in various generality settings for solving nonlinear eigenvalue problems, for example, based on Arnoldi's method [4], Jacobi–Davidson methods [9, 10], methods which can be seen as flavors and extensions of Newton's method [11–13], and contour integral formulations [14, 15] as well as methods exploiting min-max properties of Hermitian nonlinear eigenvalue

*Correspondence to: Roel Van Beeumen, Department of Computer Science, KU Leuven, University of Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium.
†E-mail: Roel.VanBeeumen@cs.kuleuven.be

problems [16]. There are also several approaches based on first approximating $M(\lambda)$ and subsequently linearizing the approximation. This gives rise to companion-type linearizations from which the structure can be exploited [17–19]. However, our approach here is similar to [20] and based on directly applying the Arnoldi method on an operator reformulation of the nonlinear eigenvalue problem (1.1).

In the recent literature, there are several nonlinear eigenvalue methods that exploit low-rank structures in different settings, for example, the approach for low-rank modifications of symmetric eigenvalue problems in [21], the linearization approach for rational eigenvalue problems in [22], and for nonlinear eigenvalue problems in [18, 19]. In [23], a quadratic eigenvalue problem with low-rank damping term is turned into a problem as addressed in the paper (nonlinear eigenvalue problem with low-rank terms corresponding to higher order terms) and subsequently solved using a Paé approximation and linearization, exploiting the low-rank property. In this paper, we will address such a type of low-rank structure in the framework of the infinite Arnoldi method [20]. This method is equivalent to Arnoldi's method applied to a linear operator, and the restarting procedure is based on the structure of the invariant subspace presented in [24].

In this work, we will construct an algorithm that exploits a particular commonly occuring structure, that is, the high-order coefficients in the Taylor expansion of M have low rank r (see Assumption 2 for a precise statement). The proposed algorithm will be particularly suitable for large n and situations where $r \ll n$. This low-rank property often appears in the discretizations of PDE eigenvalue problems that have been constructed with non-reflecting boundary conditions. In numerical examples, we will also give an example of how localized delayed feedback control can give rise to this type of low-rank structure.

The paper is organized as follows. In Section 2, we state the main assumptions and show that the solutions to (1.1) are reciprocal eigenvalues of an operator $\mathcal{F}\mathcal{B}$. This is similar to [20] where an equivalence was shown for $\mathcal{B}$. The additional operator $\mathcal{F}$ stems from the low-rank structure and allows for considerable perfomance improvement. In Section 3, we consider the Arnoldi method for the operator $\mathcal{F}\mathcal{B}$, where we represent the iterates in a polynomial basis. We also show that, if we start the iteration in a particular way and use a particular vector-valued polynomial basis, we can carry out the Arnoldi method for $\mathcal{F}\mathcal{B}$ very efficiently. In comparison to [20], the basis matrix in this algorithm grows slower yielding a reduction of the computation time required for the orthogonalization and the memory resources required to store the basis matrix. The slower growth also allows for a natural way to carry out implicit restarting and locking. This is derived in Section 4. It is well-known that the Arnoldi's method usually converges quickly to extreme isolated eigenvalues (e.g., [25, Section 6.7]). As a consequence of the fact that we carry out the Arnoldi method on an operator with inverted eigenvalue set, the construction is likely to find solutions to (1.1) close to the origin quickly, similar to shift-and-invert Arnoldi method. This as well as other efficiency properties are illustrated in the numerical experiments in Section 5.

2. EQUIVALENT LINEAR OPERATOR EIGENVALUE PROBLEM

Similarly as in the infinite Arnoldi method [20], the basis of the algorithm will be a characterization of the solutions to (1.1) as reciprocal eigenvalues of a linear operator. Here, we will use an operator that also takes the low-rank structure into account.

If we let

$$B(\lambda) := \frac{1}{\lambda} M(0)^{-1}(M(0) - M(\lambda)),$$

we have that

$$\lambda B(\lambda)x = x, \qquad \lambda \in \mathbb{C}, \quad x \in \mathbb{C}^n \backslash \{0\}, \tag{2.1}$$

unless $\lambda = 0$, and define $B(0)$ with analytic continuation.

Throughout the paper, we make the following assumptions.

DOI: 10.1002/nla

Assumption 1
The Taylor series expansion of B at $\lambda = 0$ converges for all $\lambda \in \Omega$.

We note that Assumption 1 is satisfied with $\Omega = \mathbb{C}$ if M is an entire function, and similarly, if M is analytic in a neighborhood of the closed disk of radius r centered at the origin, Assumption 1 is satisfied with Ω selected as the corresponding open disk.

We also assume that $M(\lambda)$ and $B(\lambda)$ satisfy a low-rank property, in the following sense.

Assumption 2 (Low rank property)
There exist strictly positive integers p, r satisfying $r < n$, a matrix $Q_i \in \mathbb{C}^{n\times r}$ with orthonormal columns and matrices $U_i \in \mathbb{C}^{n\times r}$, $i \geqslant p$, such that the function B can be decomposed as

$$B(\lambda) = B_{\text{pol}}(\lambda) + B_{\text{rem}}(\lambda),$$

where

- B_{pol} is a polynomial matrix of a given degree $p-1$ corresponding to the first p terms in the Taylor series of B, that is, it can be expanded as

$$B_{\text{pol}}(\lambda) = B_0 + B_1\frac{\lambda}{1!} + B_2\frac{\lambda^2}{2!} + \cdots + B_{p-1}\frac{\lambda^{p-1}}{(p-1)!}.$$

- B_{rem} is the remainder of the Taylor expansion of B and can be expanded as

$$B_{\text{rem}}(\lambda) = B_p\frac{\lambda^p}{p!} + B_{p+1}\frac{\lambda^{p+1}}{(p+1)!} + \cdots, \quad \lambda \in \Omega,$$

  where B_i satisfies

$$B_i = U_i Q^*, \qquad i \geqslant p. \tag{2.2}$$

In terms of the original formulation (1.1), Assumption 2 implies that in the Taylor expansion of M

$$M(\lambda) = \sum_{i=0}^{\infty} M_i\lambda^i, \tag{2.3}$$

all matrices M_i with $i > p$ have rank r and span a common column space. Our interest in the context of eigenvalue computations consists of exploiting the situation where $r \ll n$.

The construction of the proposed algorithm requires two operators:

- Operator $\mathcal{B} : \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n) \to \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n)$, which served as a basis of the derivation in [20], is defined by

$$\begin{aligned} \mathcal{D}(\mathcal{B}) &= \left\{\phi \in \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n) : \left(B\left(\frac{d}{d\theta}\right)\phi\right)(0) < \infty\right\}, \\ (\mathcal{B}\phi)(\theta) &= \int_0^\theta \phi(s)ds + \left(B\left(\frac{d}{d\theta}\right)\phi\right)(0), \quad \phi \in \mathcal{D}(\mathcal{B}), \end{aligned} \tag{2.4}$$

  where

$$\left(B\left(\frac{d}{d\theta}\right)\phi\right)(0) = \sum_{i=0}^{\infty}\frac{1}{i!}B_i\phi^{(i)}(0).$$

- Operator $\mathcal{F} : \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n) \to \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n)$ is defined by

$$(\mathcal{F}\phi)(\theta) = \sum_{i=0}^{p-1}\phi^{(i)}(0)\frac{\theta^i}{i!} + \sum_{i=p}^{\infty} QQ^*\phi^{(i)}(0)\frac{\theta^i}{i!}. \tag{2.5}$$

DOI: 10.1002/nla

Note that $\phi \in \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n)$ implies $\mathcal{F}\phi \in \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n)$ because QQ^* can be factored out in the second term of (2.5).

We now relate (2.1) with the operator eigenvalue problem

$$\lambda(\mathcal{F}\mathcal{B})\phi = \phi, \qquad \phi \in \mathcal{C}^\infty(\mathbb{R}, \mathbb{C}^n), \quad \lambda \in \mathbb{C}, \quad \phi \not\equiv 0. \tag{2.6}$$

The eigenvalue problems (2.1) and (2.6) are equivalent in the following sense.

Theorem 2.1 (Operator equivalence)
Suppose that B satisfies Assumptions 1–2. Then, the following implications are satisfied.

(1) Let the pair (λ, x), with $\lambda \in \Omega \setminus \{0\}$, be a solution of (2.1). Then (λ, ϕ) is a solution of (2.6), where

$$\phi(\theta) = \sum_{i=0}^{p-1} \frac{(\lambda\theta)^i}{i!} x + \sum_{i=p}^{\infty} \frac{(\lambda\theta)^i}{i!} QQ^* x. \tag{2.7}$$

(2) Let the pair (λ, ϕ), with $\lambda \in \Omega \setminus \{0\}$, be a solution of (2.6). Then (λ, x), with $x = \phi(0)$, is a solution of (2.1).

Proof
The proof consists of two parts.

Statement 1. Let ϕ be given by (2.7). Because of Assumption 1, we have that $B(\frac{d}{dt})\phi(0) < \infty$ and thus $\phi \in \mathcal{D}(\mathcal{B})$. Moreover, we can express

$$\left(B_{\text{pol}}\left(\frac{d}{d\theta}\right)\phi \right)(0) = B_{\text{pol}}(\lambda)x, \tag{2.8}$$

$$\left(B_{\text{rem}}\left(\frac{d}{d\theta}\right)\phi \right)(0) = B_{\text{rem}}(\lambda)QQ^*x. \tag{2.9}$$

From the definition of $\mathcal{B}$, it follows that

$$(\mathcal{B}\phi)(\theta) = \sum_{i=0}^{p-1} \frac{\lambda^i\theta^{i+1}}{(i+1)!} x + \sum_{i=p}^{\infty} \frac{\lambda^i\theta^{i+1}}{(i+1)!} QQ^*x + B_{\text{pol}}(\lambda)x + B_{\text{rem}}(\lambda)QQ^*x.$$

Consequently

$$((\mathcal{F}\mathcal{B})\phi)(\theta) = \sum_{i=1}^{p-1} \frac{\lambda^{i-1}\theta^i}{i!} x + \sum_{i=p}^{\infty} \frac{\lambda^{i-1}\theta^i}{i!} QQ^*x + B_{\text{pol}}(\lambda)x + B_{\text{rem}}(\lambda)QQ^*x.$$

From (2.2) and the fact that (λ, x) is an eigenpair, we obtain

$$B_{\text{pol}}(\lambda)x + B_{\text{rem}}(\lambda)QQ^*x = B_{\text{pol}}(\lambda)x + B_{\text{rem}}(\lambda)x = B(\lambda)x = \frac{1}{\lambda}x,$$

and it follows that

$$\lambda(\mathcal{F}\mathcal{B})\phi = \phi.$$

Statement 2. Let (λ, ϕ) be an eigenpair of $\mathcal{F}\mathcal{B}$. Hence, $\phi \in \mathcal{D}(\mathcal{B})$. Denote the power series expansion of ϕ by

$$\phi = \sum_{i=0}^{\infty} \theta^i x_i.$$

DOI: 10.1002/nla

Hence, we obtain

$$\mathcal{B}\phi = \left(B\left(\frac{d}{d\theta}\right)\phi\right)(0) + \sum_{i=0}^{\infty} \frac{\theta^{i+1}}{i+1} x_i,$$
$$(\mathcal{F}\mathcal{B})\,\phi = \left(B\left(\frac{d}{d\theta}\right)\phi\right)(0) + \sum_{i=0}^{p-1} \frac{\theta^{i+1}}{i+1} x_i + \sum_{i=p}^{\infty} \frac{\theta^{i+1}}{i+1} Q Q^* x_i.$$

Equating the coefficients corresponding to powers of θ in $(\mathcal{F}\mathcal{B})\phi = \frac{1}{\lambda}\phi$ yields

$$\begin{aligned} B\left(\frac{d}{d\theta}\right)\phi(0) &= \frac{1}{\lambda} x_0 \\ \frac{x_0}{1} &= \frac{1}{\lambda} x_1 \\ &\vdots \\ \frac{x_p}{p-1} &= \frac{1}{\lambda} x_{p-1} \\ Q Q^* \frac{x_{p-1}}{p} &= \frac{1}{\lambda} x_p \\ Q Q^* \frac{x_p}{p+1} &= \frac{1}{\lambda} x_{p+1} \\ &\vdots \end{aligned} \tag{2.10}$$

We conclude directly that

$$x_i = \begin{cases} \frac{\lambda^i}{i!} x_0, & i = 1, \ldots, p-1, \\ \frac{\lambda^i}{i!} Q Q^* x_0, & i \geqslant p. \end{cases}$$

Because by assumption $\phi \not\equiv 0$, we see that $x_0 \neq 0$ must hold. Writing out the first equation of (2.10) yields

$$x_0 = \lambda \sum_{i=0}^{\infty} B_i x_i = \lambda B_{\text{pol}}(\lambda) x_0 + \lambda B_{\text{rem}}(\lambda) Q Q^* x_0 = \lambda B(\lambda) x_0.$$

We conclude that (λ, x_0) is an eigenpair of (2.1). The proof is completed by noticing that $x_0 = \phi(0)$. □

3. ARNOLDI'S METHOD ON $\mathcal{F}\mathcal{B}$

The infinite Arnoldi method [20] is equivalent to Arnoldi's method for the operator $\mathcal{B}$ whose reciprocal eigenvalues are solutions to (2.1). We know from Theorem 2.1 that the reciprocal eigenvalues of the operator $\mathcal{F}\mathcal{B}$ are also solutions to (2.1) for problems with low-rank structure. Analogous to the infinite Arnoldi method, we will now construct an algorithm by considering Arnoldi's method for $\mathcal{F}\mathcal{B}$. By carrying out k steps of Arnoldi's method for $\mathcal{F}\mathcal{B}$ with a starting function ϕ, we generate a sequence of functions $\phi_1, \ldots, \phi_k$ forming a basis of the Krylov subspace

$$\begin{aligned} \mathcal{K}_k\left(\mathcal{F}\mathcal{B},\, \phi\right) &:= \operatorname{span}\left\{\phi,\ \mathcal{F}\mathcal{B}\phi,\ \ldots,\ (\mathcal{F}\mathcal{B})^{k-1}\phi\right\}, \\ &:= \operatorname{span}\left\{\phi_1,\ \phi_2,\ \ldots,\ \phi_k\right\}. \end{aligned} \tag{3.1}$$

If we start the Arnoldi method with a constant function, we can show from the fact that $\mathcal{F}\mathcal{B}$ corresponds to integration, that $\phi_1, \ldots, \phi_k$ are vector-valued polynomials. However, unlike [20], we will see here that because of the application of $\mathcal{F}$, the functions can be represented with less informa-

DOI: 10.1002/nla

tion yielding a significant performance improvement. This will also allow the possibility to carry out implicit restarting in a natural way, which we shall explain in Section 4. The possibility to represent the functions in a compressed way stems from the fact that some polynomial coefficients can be represented with vectors of smaller size. This can be seen from the following lemma, where we see that the polynomial coefficients of degree higher than $p-1$ can be represented with vectors of length r, that is, the rank of the low-rank terms.

Lemma 3.1
Suppose $\phi_1(\theta) := z_0$ is constant. Then, there are constants $x_0, \ldots, x_{p-1} \in \mathbb{C}^n$ and $\hat{x}_p, \ldots, \hat{x}_k \in \mathbb{C}^r$ such that

$$\left((\mathcal{F}\mathcal{B})^k \phi_1\right)(\theta) = x_0 + x_1\theta + x_2\theta^2 + \cdots + x_{p-1}\theta^{p-1} + Q\left(\hat{x}_p\theta^p + \cdots + \hat{x}_p\theta^k\right). \quad (3.2)$$

Proof
The result follows from the definitions of $\mathcal{B}$ and $\mathcal{F}$, by induction on the degree. □

It is also easy to show that Arnoldi's method applied to $\mathcal{F}\mathcal{B}$, that is, [20, Algorithm 1] with $\mathcal{F}\mathcal{B}$ instead of $\mathcal{B}$, generates iterates $\phi_1, \ldots, \phi_k$ corresponding to functions of the structure (3.2). In the implementation, we represent these iterates in a polynomial basis using p vectors of length n and $k-p$ vectors of length r.

For the operator $\mathcal{F}\mathcal{B}$, there is in general no obvious scalar product to be used in the construction. We will, similar to [20], couple the scalar product with the polynomial representation, in the sense that we will use the Euclidean inner product corresponding to the coefficient vectors in the basis. Consider any polynomial basis $g_0, g_1, \ldots$ and any two functions φ and ψ that can be expressed as

$$\varphi(\theta) = \sum_{i=0}^{\infty} g_i(\theta)x_i, \qquad \psi(\theta) = \sum_{i=0}^{\infty} g_i(\theta)y_i.$$

Then, we define the scalar product as follows:

$$\langle \varphi, \psi \rangle := \sum_{i=0}^{\infty} y_i^* x_i.$$

We will work out the algorithm for both the monomial basis and a (scaled) Chebyshev basis, although other choices of polynomial bases are also possible. The scalar product corresponding to the Chebyshev basis is expected to lead to a fast convergence of our algorithms for the delay eigenvalue problem. In [20, Section 5.2–5.3]), for the standard infinite Arnoldi algorithm, this is explained by a connection with a spectral discretization of a corresponding differential operator.

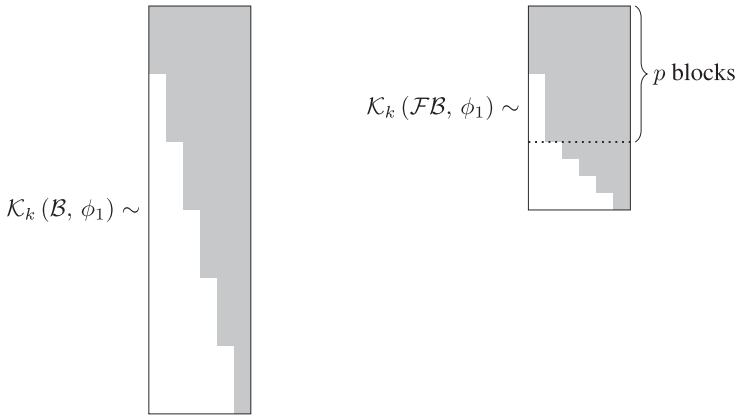
By using the coupling of basis and scalar product, we can carry out the scalar product of two functions directly in the compressed representation, as can be seen from the following lemma.

Lemma 3.2
Consider a polynomial basis $g_0, g_1, \ldots$ and suppose the vector-valued polynomials φ and ψ are given by $\varphi(\theta) = x_0 g_0(\theta) + \cdots + x_k g_k(\theta)$ and $\psi(\theta) = y_0 g_0(\theta) + \cdots + y_m g_m(\theta)$. Moreover, suppose the functions φ and ψ have the structure (3.2), that is, $x_i = Q\hat{x}_i$, $y_i = Q\hat{y}_i$ when $i \geqslant p$. Then

$$\langle \varphi, \psi \rangle := \sum_{i=0}^{p-1} y_i^* x_i + \sum_{i=p}^{\min(k,m)} \hat{y}_i^* \hat{x}_i.$$

Figure 1 illustrates graphically the non-zero structure of the basis representations constructed by the standard infinite Arnoldi method [20] as well as by its low-rank version. Figure 1(a) shows that when we apply Arnoldi's method to $\mathcal{B}$ with constant starting function, the non-zero part of the basis grows by a block row consisting of n rows. In contrast to this, Figure 1(b) shows that when we apply Arnoldi's method to $\mathcal{F}\mathcal{B}$ with constant starting function, the basis matrix is only expanded by

DOI: 10.1002/nla



(a) Standard Infinite Arnoldi method (b) Low-rank Infinite Arnoldi method

Figure 1. Structure of the basis representations of the standard infinite Arnoldi method [20] and its low-rank version presented in this paper. (a) Standard infinite Arnoldi method and (b) low-rank infinite Arnoldi method.

a block row consisting of r rows, because the vector coefficients for polynomials of degree higher than p can be represented with vectors of length r.

3.1. Taylor coefficient map

We wish to carry out Arnoldi's method for $\mathcal{FB}$ where the functions are represented in a polynomial basis. As a first step in this construction, we need the action of $\mathcal{FB}$ in the monomial basis. The following theorem specifies $\mathcal{FB}$ for functions of the structure (3.2).

Theorem 3.3 (General coefficient map for $\mathcal{FB}$ in the monomial basis)
Suppose that φ is given by

$$\varphi(\theta) := \sum_{i=0}^{p-1} \theta^i x_i + \sum_{i=p}^{N-1} \theta^i Q\hat{x}_i,$$

where $x_0, \ldots, x_{p-1}, Q\hat{x}_p, \ldots, Q\hat{x}_{N-1} \in \mathbb{C}[n]$ denote the vector coefficients in the monomial basis. Then, the coefficients of $\psi := \mathcal{FB}\varphi$, that is

$$\psi(\theta) = (\mathcal{FB}\varphi)(\theta) = \sum_{i=0}^{p-1} \theta^i y_i + \sum_{i=p}^{N} \theta^i Q\hat{y}_i,$$

are given by

$$y_0 = \sum_{i=0}^{p-1} B_i x_i + \sum_{i=p}^{N-1} U_i \hat{x}_i, \tag{3.3}$$

and

$$\begin{aligned} \begin{bmatrix} y_1 & \cdots & y_{p-1} \end{bmatrix} &= \begin{bmatrix} x_0/1 & \cdots & x_{p-2}/(p-1) \end{bmatrix}, \\ \hat{y}_p &= Q^* x_{p-1}/p, \\ \begin{bmatrix} \hat{y}_{p+1} & \cdots & \hat{y}_N \end{bmatrix} &= \begin{bmatrix} \hat{x}_p/(p+1) & \cdots & \hat{x}_{N-1}/N \end{bmatrix}, \end{aligned} \tag{3.4}$$

Proof
This follows directly from the definitions of $\mathcal{F}$ and $\mathcal{B}$. □

DOI: 10.1002/nla

In order to carry out the action in practice, some analysis is required for the specific problem in order to find an explicit and efficient expression for (3.3). Fortunately, we can simplify somewhat, if the problem is expressed in terms of the coefficient matrices M_i of the original nonlinear eigenvalue problem (1.1), because simple manipulations yield

$$\begin{aligned} y_0 &= -M_0^{-1}\left(\sum_{i=1}^{p-1} M_i y_i + \frac{1}{p} M_p x_{p-1} + \sum_{i=p+1}^{N} V_i \hat{y}_i\right), \\ &= -M_0^{-1}\left(\sum_{i=0}^{p-1} \frac{1}{i+1} M_{i+1} x_i + \sum_{i=p}^{N-1} \frac{1}{i+1} V_{i+1} \hat{x}_i\right). \end{aligned} \tag{3.5}$$

3.2. Chebyshev coefficient map

It was illustrated and explained in [20] that for certain problems, it is natural and much more efficient to work with the inner product corresponding to the Chebyshev basis, in particular, in terms of asymptotic convergence rate. We will now derive the coefficient map for $\mathcal{FB}$ in Chebyshev basis. In Section 5, we illustrate that we have a considerable improvement in performance for certain problems.

Let $T_0, T_1, \ldots$ be the Chebyshev polynomials of the first kind scaled to an interval $[a, b]$, that is

$$T_i(\theta) = \cos(i \arccos(k\theta + c)) \tag{3.6}$$

where $c = \frac{a+b}{a-b}$ and $k = \frac{2}{b-a}$. We will need an explicit representation of the integration of a polynomial expressed in the Chebyshev basis. Let L_N correspond to this map, that is, for any $N \in \mathbb{N}$ we have

$$\begin{bmatrix} T_0(\theta) \\ \vdots \\ T_{N-1}(\theta) \end{bmatrix} = L_N \begin{bmatrix} T_1'(\theta) \\ \vdots \\ T_N'(\theta) \end{bmatrix}.$$

Then, the matrix L_N is triangular, and an explicit expression is given in [20, Equation 21]. We will partition L_N into blocks as follows:

$$L_N =: \begin{bmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{bmatrix}, \tag{3.7}$$

where $L_{11} \in \mathbb{R}^{(p-1)(p-1)}$, $L_{22} \in \mathbb{R}$, and $L_{33} \in \mathbb{R}^{(N-p)(N-p)}$.

In the formulations of the coefficient map, we will need the coefficients transforming a Chebyshev polynomial into its monomial coefficients. Let this matrix be given by $U \in \mathbb{R}^{N\times N}$, that is

$$\begin{bmatrix} T_0(\theta) \\ T_1(\theta) \\ T_2(\theta) \\ \vdots \end{bmatrix} = \begin{bmatrix} u_{0,0} & 0 & 0 & 0 & \cdots \\ u_{1,0} & u_{1,1} & 0 & 0 & \cdots \\ u_{2,0} & u_{2,1} & u_{2,2} & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots & \ddots \end{bmatrix} \begin{bmatrix} 1 \\ \theta \\ \theta^2 \\ \vdots \end{bmatrix} =: U \begin{bmatrix} 1 \\ \theta \\ \theta^2 \\ \vdots \end{bmatrix}. \tag{3.8}$$

Moreover, let $v_p \in \mathbb{R}[p]$ be defined as follows:

$$v_p^T = \begin{bmatrix} u_{p,0} & u_{p,1} & \cdots & u_{p,p-1} \end{bmatrix} \begin{bmatrix} u_{0,0} & 0 & \cdots & 0 \\ u_{1,0} & u_{1,1} & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ u_{p-1,0} & u_{p-1,1} & \cdots & u_{p-1,p-1} \end{bmatrix}^{-1}. \tag{3.9}$$

In contrast to the monomial case, the application of $\mathcal{F}$ modifies all first $p+1$ coefficients when it is represented in the Chebyshev basis. More precisely, the coefficients are modified as follows.

DOI: 10.1002/nla

Lemma 3.4 (The representation of the operator $\mathcal{F}$ in the Chebyshev basis)
Suppose φ is given by

$$\varphi(\theta) = \sum_{i=0}^{p} T_i(\theta) x_i + \sum_{i=p+1}^{N} T_i(\theta) Q \hat{x}_i, \tag{3.10}$$

with $x_0, \ldots, x_p \in \mathbb{C}^n$ and $\hat{x}_{p+1}, \ldots, \hat{x}_N \in \mathbb{C}^r$ and $T_0, T_1, \ldots$ are defined by (3.6). Then

$$(\mathcal{F}\varphi)(\theta) = \sum_{i=0}^{p-1} T_i(\theta) y_i + \sum_{i=p}^{N} T_i(\theta) Q \hat{y}_i,$$

where

$$\begin{aligned} \begin{bmatrix} y_0 & \cdots & y_{p-1} \end{bmatrix} &= \begin{bmatrix} x_0 & \cdots & x_{p-1} \end{bmatrix} + (I - QQ^*) x_p v_p^T, \\ \hat{y}_p &= Q^* x_p, \\ \begin{bmatrix} \hat{y}_{p+1} & \cdots & \hat{y}_N \end{bmatrix} &= \begin{bmatrix} \hat{x}_{p+1} & \cdots & \hat{x}_N \end{bmatrix}, \end{aligned}$$

with v_p is defined by (3.9).

Proof
Because the operator $\mathcal{F}$ is defined in the monomial basis, we transform φ (3.10) to the monomial basis, carry out the operation $\mathcal{F}$, and then transform it back to the Chebyshev basis. Let $X_1 = \begin{bmatrix} x_0 & \cdots & x_{p-1} \end{bmatrix}$ and $X_3 = \begin{bmatrix} \hat{x}_{p+1} & \cdots & \hat{x}_N \end{bmatrix}$. We have

$$\begin{aligned} \varphi(\theta) &= \begin{bmatrix} X_1 & x_p & QX_3 \end{bmatrix} \begin{bmatrix} T_0(\theta) & \cdots & T_N(\theta) \end{bmatrix}^*, \\ &= \begin{bmatrix} X_1 & x_p & QX_3 \end{bmatrix} U \begin{bmatrix} 1 & \theta & \cdots & \theta^N \end{bmatrix}^*, \end{aligned} \tag{3.11}$$

where U is defined in (3.8). We now partition U according to

$$U := \begin{bmatrix} U_{11} & 0 & 0 \\ U_{21} & U_{22} & 0 \\ U_{31} & U_{32} & U_{33} \end{bmatrix},$$

where $U_{11} \in \mathbb{R}^{p \times p}$, $U_{22} \in \mathbb{R}$, and $U_{33} \in \mathbb{R}^{(N-p)\times(N-p)}$. By carrying out the multiplication in (3.11), we have

$$\varphi(\theta) = \begin{bmatrix} X_1 U_{11} + x_p U_{21} + QX_3 U_{31} & x_p U_{22} + QX_3 U_{32} & QX_3 U_{33} \end{bmatrix} \begin{bmatrix} 1 & \theta & \cdots & \theta^N \end{bmatrix}^*.$$

Recall that $\mathcal{F}$ corresponds to multiplying all monomial coefficients of degree equal or higher than p by QQ^*. By using that Q has orthonormal columns, we have that

$$\begin{aligned} (\mathcal{F}\varphi)(\theta) &= \begin{bmatrix} X_1 U_{11} + x_p U_{21} + QX_3 U_{31} & QQ^* x_p U_{22} + QX_3 U_{32} & QX_3 U_{33} \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ \theta^N \end{bmatrix}, \\ &= \begin{bmatrix} X_1 + (I - QQ^*) x_p v_p^T & QQ^* x_p & QX_3 \end{bmatrix} \begin{bmatrix} U_{11} & 0 & 0 \\ U_{21} & U_{22} & 0 \\ U_{31} & U_{32} & U_{33} \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ \theta^N \end{bmatrix}, \end{aligned}$$

where $v_p^T = U_{21} U_{11}^{-1}$. Finally, noting that

$$U \begin{bmatrix} 1 & \theta & \cdots & \theta^N \end{bmatrix}^* = \begin{bmatrix} T_0(\theta) & \cdots & T_N(\theta) \end{bmatrix}^*.$$

completes the proof. □

DOI: 10.1002/nla

By combining the lemma characterizing the coefficient map corresponding to $\mathcal{F}$, we can now derive the coefficient map corresponding to $\mathcal{FB}$.

Theorem 3.5 (General coefficient map for $\mathcal{FB}$ in the Chebyshev basis)
Suppose φ is given by

$$\varphi(\theta) := \sum_{i=0}^{p-1} T_i(\theta) x_i + \sum_{i=p}^{N-1} T_i(\theta) Q \hat{x}_i,$$

where $T_0, T_1, \ldots$ are defined by (3.6) and the columns of $X \in \mathbb{C}[n][N]$

$$X := \begin{bmatrix} X_1 & x_{p-1} & QX_3 \end{bmatrix},$$

denote the vector coefficients of φ, that is, $X_1 := \begin{bmatrix} x_0 & \cdots & x_{p-2} \end{bmatrix}$ and $X_3 := \begin{bmatrix} \hat{x}_p & \cdots & \hat{x}_{N-1} \end{bmatrix}$. Then, the expansion of $\psi := \mathcal{FB}\varphi$, that is

$$\psi(\theta) = (\mathcal{FB}\varphi)(\theta) = \sum_{i=0}^{p-1} T_i(\theta) y_i + \sum_{i=p}^{N} T_i(\theta) Q \hat{y}_i,$$

is given by

$$\begin{aligned} y_0 = &\left(\sum_{i=0}^{p-1} \left(B\left(\frac{d}{d\theta}\right) T_i(\theta) x_i \right) + \sum_{i=p}^{N-1} \left(B\left(\frac{d}{d\theta}\right) T_i(\theta) Q \hat{x}_i \right) \right)_{\theta=0} \\ &- \begin{bmatrix} X_1 & x_{p-1} & QX_3 \end{bmatrix} L_N \begin{bmatrix} T_1(0) \\ \vdots \\ T_N(0) \end{bmatrix} + (I - QQ^*) x_{p-1} v_{p,1}, \end{aligned} \tag{3.12}$$

and

$$\begin{aligned} \begin{bmatrix} y_1 & \cdots & y_{p-1} \end{bmatrix} &= X_1 L_{11} + x_{p-1} L_{21} + QX_3 L_{31} + (I - QQ^*) x_{p-1} L_{22} \tilde{v}_p^T, \\ y_p &= Q^* x_{p-1} L_{22} + X_3 L_{32}, \\ \begin{bmatrix} \hat{y}_{p+1} & \cdots & \hat{y}_N \end{bmatrix} &= X_3 L_{33}, \end{aligned} \tag{3.13}$$

where L_{ij} are defined by (3.7) and we denote $v_p^T =: \left(v_{p,1}, \tilde{v}_p^T\right)$ with $v_{p,1} \in \mathbb{R}$, $\tilde{v}_p \in \mathbb{R}[p-1]$.

Proof
The proof consists of two parts. Firstly, we will use the general coefficient map defined in [20, Theorem 4] in order to obtain the coefficients of $\mathcal{B}\varphi$. Next, we apply Lemma 3.4 resulting in the coefficients of $\psi := \mathcal{FB}\varphi$.

Let $Z := \begin{bmatrix} z_0 & Z_1 & z_p & QZ_3 \end{bmatrix}$ with $Z_1 := \begin{bmatrix} z_1 & \cdots & z_{p-1} \end{bmatrix}$ and $Z_3 := \begin{bmatrix} z_{p+1} & \cdots & z_N \end{bmatrix}$ denote the coefficients of the function $\mathcal{B}\varphi$, that is

$$(\mathcal{B}\varphi)(\theta) = \sum_{i=0}^{p} T_i(\theta) z_i + \sum_{i=p+1}^{N} T_i(\theta) Q z_i.$$

Then, from the general coefficient map defined in [20, Theorem 4], we have

$$\begin{bmatrix} Z_1 & z_p & QZ_3 \end{bmatrix} = \begin{bmatrix} X_1 & x_{p-1} & QX_3 \end{bmatrix} \begin{bmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{bmatrix},$$

which yields

DOI: 10.1002/nla

$$Z_1 = X_1 L_{11} + x_{p-1} L_{21} + Q X_3 L_{31}, \tag{3.14a}$$
$$z_p = x_{p-1} L_{22} + Q X_3 L_{32}, \tag{3.14b}$$
$$Z_3 = X_3 L_{33}. \tag{3.14c}$$

An explicit expression for the vector $z_0 \in \mathbb{C}^n$ can be found by noting that [20, Equation 22] can be rephrased using [20, Equation 12] and specialized for φ. This leads to

$$z_0 = \sum_{i=0}^{p-1} \left(B\left(\frac{d}{d\theta}\right) T_i(\theta) x_i \right)(0) + \sum_{i=p}^{N-1} \left(B\left(\frac{d}{d\theta}\right) T_i(\theta) Q \hat{x}_i \right)(0) \tag{3.15}$$
$$- \begin{bmatrix} X_1 & x_{p-1} & Q X_3 \end{bmatrix} L_N \begin{bmatrix} T_1(0) \\ \vdots \\ T_N(0) \end{bmatrix}$$

To complete the proof, we apply Lemma 3.4, which results in $y_0 = z_0 + (I - QQ^*) x_{p-1} v_{p,1}$ and (3.12)–(3.13). □

Similar to the monomial case, we need to find an efficient, accurate, and explicit expressions for y_0 in (3.12). The difficulty in deriving such an expression should not be underestimated. Unfortunately, unlike the monomial case, in particular (3.5), expressing y_0 in terms of the coefficients of the original nonlinear eigenvalue problem $M_0, M_1, \ldots$ does not considerably simplify the problem, although a general approach based on manipulations similar to [20, Appendix A] is feasible but somewhat tedious in general.

In this work, we will derive explicit expressions for an important special case. We consider delay eigenvalue problems and specialize the result as follows. Suppose

$$M(\lambda) = -\lambda I + A_0 + A_1 e^{-\tau\lambda}. \tag{3.16}$$

where $A_1 = VQ^*$, such that we can set $p = 1$. Such problems occurs in the stability analysis of PDE with pointwise delayed feedback, as we shall further illustrate in Section 5.2. The coefficient map (Theorem 3.5) simplifies as follows for (3.16).

Corollary 3.1 (Delay eigenvalue problem with single delay and $p = 1$)
Consider the nonlinear eigenvalue problem (1.1) where $M(\lambda)$ is given by (3.16) with $A_1 = VQ^*$ and $Q \in \mathbb{C}^{n\times r}$ has orthonormal columns. Let $T_0, T_1, \ldots$ be the Chebyshev polynomials of the first kind (3.6) scaled to the interval $(a, b) = (-\tau, 0)$. Moreover, suppose φ is given by

$$\varphi(\theta) = T_0(\theta) x_0 + \sum_{i=1}^{N-1} T_i(\theta) Q \hat{x}_i, \tag{3.17}$$

where $x_0 \in \mathbb{C}^n$ and $x_1, x_2, \ldots \in \mathbb{C}^r$. Then, the expansion of $\psi := \mathcal{F}\mathcal{B}\varphi$, that is

$$\psi(\theta) = (\mathcal{F}\mathcal{B}\varphi)(\theta) = T_0(\theta) y_0 + \sum_{i=1}^{N} T_i(\theta) Q y_i,$$

is given by

$$y_0 = (A_0 + A_1)^{-1} \left(x_0 + Q \sum_{i=1}^{N-1} \hat{x}_i - A_0 \left[z_1 + Q \sum_{i=2}^{N} \hat{y}_i \right] \right.$$
$$\left. - A_1 \left[z_1 T_1(-\tau) + Q \sum_{i=2}^{N} T_i(-\tau) \hat{y}_i \right] \right) + \frac{\tau}{2} (I - QQ^*) x_0,$$

DOI: 10.1002/nla

and

$$\begin{bmatrix} y_1 & \cdots & y_N \end{bmatrix} = \begin{bmatrix} Q^* x_0 & x_1 & \cdots & x_{N-1} \end{bmatrix} L_N, \tag{3.18}$$

with

$$z_1 = \begin{cases} \frac{\tau}{2} x_0 - \frac{\tau}{4} Q x_2 & N \geqslant 3 \\ \frac{\tau}{2} x_0 & \text{otherwise.} \end{cases} \tag{3.19}$$

Proof
Note that when $p = 1$, several matrices in Theorem 3.5 should be interpreted as empty matrices, in particular L_{11}, L_{21}, L_{31}, Z_1 and X_1, implying that $\begin{bmatrix} y_1 & \cdots & y_N \end{bmatrix}$ can be directly computed from (3.18).

The formula for y_0 is simplified if we introduce the variable z_1 in (3.14b). Because of the partitioning of L_N in (3.7) and the explicit formula for L_N in [20, Equation 21] with $(a, b) = (-\tau, 0)$, we have in our case that $L_{22} = \tau/2$ and $L_{32} = \begin{pmatrix} 0 & -\frac{\tau}{4} & 0 & \cdots & 0 \end{pmatrix}^T$. Hence, the definition of z_1 in (3.14b) simplifies to (3.19).

We derive the formula of y_0 from the fact that $y_0 = z_0 + (I - QQ^*)z_1$, similar to the last step in the proof of Theorem 3.5. In our setting, $v_p = 1$ and $z_1 = x_0 L_{22}$ with $L_{22} = 2(b - a)/4 = \tau/2$ from (3.7). The expression for z_0 is found by inserting the definition of $M(\lambda)$ and $B(\lambda)$ into (3.15). □

3.3. Low-rank infinite Arnoldi method

From Section 3.1 and Section 3.2, we now know how to compute the action of $\mathcal{FB}$ in monomial basis or Chebyshev basis for functions with the structure (3.2). Lemma 3.2 suggests a natural way to carry out the scalar product for such functions. The actions of the operator and the scalar product are the only ingredients needed in order to carry out Arnoldi's method in an operator setting, that is, [20, Algorithm 1].

Algorithm 1: Low-rank infinite Arnoldi method

Input : $V_1 = v \in \mathbb{C}^n$, k is number of steps
Output: Reciprocal Ritz values $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_k$

Set $\underline{H}_0 = 0$,
for $j = 1, 2, \ldots, k$ **do**
  $[V_{j+1}, \underline{H}_j]$ = Algorithm 2 with input $[V_j, \underline{H}_{j-1}]$
end
Return approximate solutions of (2.1) $\tilde{\lambda}_j = 1/\mu_j$ where $\mu_j \in \sigma(H_k)$, $j = 1, \ldots, k$.

As usual for the Arnoldi method, we will denote the upper block of the rectangular Hessenberg matrix $\underline{H}_k \in \mathbb{C}^{(k+1)\times k}$ by $H_k \in \mathbb{C}^{k\times k}$, and the (i, j) element of $\underline{H}_k$ is denoted $h_{i,j}$. We summarize the algorithm in Algorithms 1 and 2, where we have separated the algorithm into two parts in order to simplify the presentation of the restarting in the following sections. We will for reasons of efficiency stack the coeffients into vectors and matrices such that the orthogonalization can be carried out with simple operations on larger matrices and vectors as illustrated in Figure 1.

Remark 3.1 (Extraction of eigenvectors)
The result of Algorithm 1 is the matrix H_k and V_k. The approximate eigenvalues $\tilde{\lambda}_j$ are the reciprocal eigenvalues of H_k. We also need to form approximations of the corresponding eigenvectors v_j. With V_k, we have a representation of an approximate eigenfunctions of $\mathcal{FB}$. We propose here to compute approximate eigenvectors by function φ at $\theta = 0$, because exact eigenfunctions satisfies $\varphi(0) = v_j$ (according to Theorem 2.1). In the Taylor version (Section 3.1), this corresponds to using the first n rows of V_k, whereas for the Chebyshev version (Section 3.2), we make the corresponding evaluation by computing $T_i(0)$.

DOI: 10.1002/nla

Algorithm 2: Low-rank infinite Arnoldi step

Input : V_j, $\underline{H}_{j-1}$
Output: V_{j+1}, $\underline{H}_j$

1 Let $\begin{bmatrix} x_0^* & \cdots & x_{p-1}^* & \hat{x}_p^* & \cdots & \hat{x}_j^* \end{bmatrix}^* := v_j^*$.
2 Compute $y_0, \ldots, y_{p-1} \in \mathbb{C}[n]$ and $\hat{y}_p, \hat{y}_{p+1}, \ldots, \hat{y}_{j+1} \in \mathbb{C}[r]$ by either

(a) the Taylor coefficient map according to (3.4)–(3.5); or
(b) the Chebyshev coefficient map according to (3.12)–(3.13).

3 Let $w_j := \begin{bmatrix} y_0^* & y_1^* & \cdots & y_{p-1}^* & \hat{y}_p^* & \hat{y}_{p+1}^* & \cdots & \hat{y}_{j+1}^* \end{bmatrix}^*$.
4 Expand V_j with one block row with n or r rows.
5 Orthogonalize $\hat{w}_j := w_j - V_j h_j$, where $h_j = V_j^* w_j$.
6 Compute $\beta_j = \|\hat{w}_j\|_2$ and let $v_{j+1} = \hat{w}_j / \beta_j$.
7 Let $\underline{H}_j = \begin{bmatrix} \underline{H}_{j-1} & h_j \\ 0 & \beta_j \end{bmatrix} \in \mathbb{C}[(j+1)][j]$.
8 Expand V_j into $V_{j+1} = \begin{bmatrix} V_j & v_{j+1} \end{bmatrix}$.

4. IMPLICIT RESTARTING AND LOCKING

Due to the fact that Algorithm 1 is equivalent to the Arnoldi method applied to the operator $\mathcal{FB}$, the result will also satisfy an Arnoldi relation. More precisely, the function-setting Arnoldi relation generated by Algorithm 1 can be formulated as follows. Let $\Phi_j(\theta) \in \mathbb{C}^{n \times j}$ corresponds to the matrix consisting of columns $\varphi_1(\theta), \ldots, \varphi_j(\theta)$, that is

$$\Phi_j(\theta) := \begin{bmatrix} \varphi_1(\theta) & \cdots & \varphi_j(\theta) \end{bmatrix}.$$

Then, the output of Algorithm 1 satisfies

$$(\mathcal{FB}\,\Phi_j)(\theta) = \Phi_{j+1}(\theta)\,\underline{H}_j, \tag{4.1}$$

where we can express Φ_j explicitly as

$$\Phi_j(\theta) = \begin{bmatrix} (q_0(\theta), \ldots, q_{p-1}(\theta)) \otimes I_n & (q_p(\theta), \ldots, q_{j-1}(\theta)) \otimes Q \end{bmatrix} V_j, \tag{4.2}$$

and $q_i(\theta) = \theta^i$ or $q_i(\theta) = \hat{T}_i(\theta)$, $i = 0, \ldots, j$ depending on which basis is used.

We will now see that as a consequence of the fact that we have an Arnoldi relation (4.1), we will be able to carry out implicit restarting very similar to the implicit restarting procedures for the standard Arnoldi method. The restarting can be seen as a procedure to (essentially) compress the Arnoldi relation resulting in a basis matrix with a smaller number of columns. Due to the fact that the infinite Arnoldi method (Algorithm 1) has a growth also in the height of the basis matrix, we will have a growth with each restart. However, if r is small, this growth is moderate, and the growth in the height of the basis matrix is not restrictive. This is illustrated in Figure 2.

4.1. Krylov-Schur style implicit restarting and locking

In order to carry out implicit restarting and locking easily, we will work with a Krylov–Schur recurrence relation [26] in every iteration of the algorithm. This can be achieved by computing a Schur decomposition of H_j

$$H_j = Z_j S_j Z_j^*, \tag{4.3}$$

where S_j is an upper quasitriangular matrix and $Z_j^* Z_j = I_j$. Using the Schur decomposition (4.3), we can transform (4.1) into the following Krylov–Schur recurrence relation

DOI: 10.1002/nla

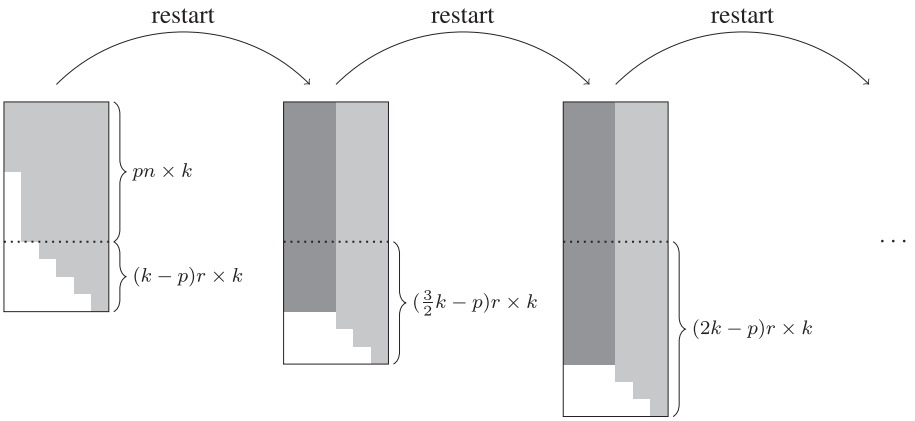


Figure 2. Graphical illustration of implicit restarting on the Krylov subspace $\mathcal{K}_k(\mathcal{FB}, \varphi_1)$ for the case $p = 2$. The light grey shade areas represent the non-zero structure of the growing coefficient vectors, whereas the dark grey shade represent the non-zero structure of the coefficient vectors after an implicit restart where the subspace is reduced from dimension k to $k/2$ in every cycle.

$$(\mathcal{FB}\,\Psi_j)(\theta) = \Psi_{j+1}(\theta)\,\underline{S}_j, \tag{4.4}$$

where

$$\Psi_{j+1}(\theta) := \begin{bmatrix} \Phi_j(\theta)Z_j & \varphi_{j+1} \end{bmatrix}, \tag{4.5}$$

and

$$\underline{S}_j := \begin{bmatrix} S_j \\ h_{j+1}^* Z_j \end{bmatrix}.$$

Let S_j be an ordered Schur decomposition of H_k

$$S_j = Z_j^* H_j Z_j = \begin{bmatrix} Z_1 & Z_2 & Z_3 \end{bmatrix}^* H_j \begin{bmatrix} Z_1 & Z_2 & Z_3 \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{13} \\ & S_{22} & S_{23} \\ & & S_{33} \end{bmatrix}, \tag{4.6}$$

where $S_{11} \in \mathbb{C}[\ell][\ell]$, $S_{22} \in \mathbb{C}[(m-\ell)][(m-\ell)]$, and $S_{33} \in \mathbb{C}[(k-m)][(k-m)]$ are upper quasitriangular matrices. The ordering is as follows: the eigenvalues of S_{11}, S_{22}, and S_{33} are, respectively, the very accurate Ritz values, the wanted but not yet converged Ritz values, and the unwanted Ritz values. Hence

$$\underline{S}_j = \begin{bmatrix} S_{11} & S_{12} & S_{13} \\ & S_{22} & S_{23} \\ & & S_{33} \\ b_1^* & b_2^* & b_3^* \end{bmatrix}, \tag{4.7}$$

where $b_i^* = h_{j+1}^* Z_i$, for $i = 1, 2, 3$.

Note that by using the ordered Schur decomposition of H_j (4.6)

$$(\mathcal{FB}) \begin{bmatrix} \Psi_1(\theta) & \Psi_2(\theta) \end{bmatrix} = \begin{bmatrix} \Psi_1(\theta) & \Psi_2(\theta) & \psi_{j+1}(\theta) \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \\ b_1^* & b_2^* \end{bmatrix},$$

where

$$\Psi_1(\theta) := \begin{bmatrix} \psi_1(\theta) & \cdots & \psi_\ell(\theta) \end{bmatrix},$$
$$\Psi_2(\theta) := \begin{bmatrix} \psi_{\ell+1}(\theta) & \cdots & \psi_m(\theta) \end{bmatrix},$$

DOI: 10.1002/nla

is also a Krylov–Schur decomposition. Thus, the purging problem can be solved by moving the unwanted Ritz values into the southeast corner of the matrix S and truncating the decomposition. Next, the Arnoldi process is restarted.

The use of the ordered Schur decomposition has also the advantage that deflation and locking of the converged Ritz pairs corresponds to setting $b_1^* = 0$ in (4.7), yielding the following recurrence relation

$$(\mathcal{FB})\begin{bmatrix} \Psi_1(\theta) & \Psi_2(\theta) \end{bmatrix} = \begin{bmatrix} \Psi_1(\theta) & \Psi_2(\theta) & \psi_{j+1}(\theta) \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \\ 0 & b_2^* \end{bmatrix} + O(\|b_1\|).$$

Note that b_1 is a measure for the (unstructured) backward error of the corresponding eigenvalues of S_{11}. Hence, $\|b_1\|$ will be zero if the eigenvalues of S_{11} are exact. For more information, we refer to [27].

4.2. Implicit restarting and locking of Algorithm 1

The operations outlined in the previous section can now be combined with Algorithms 1 and 2. In (4.5), we multiply $\Phi_j(\theta)$ by Z_j from the left. Note that, because of the relation (4.2), this corresponds to multiplying the basis matrix V_j from the left by Z_j. We provide the algorithm details in Algorithm 3.

Algorithm 3: Implicit Restarted Infinite Arnoldi (IRIA) method

Input : $x_0 \in \mathbb{R}[n], k, m \in \mathbb{N}$
Output: Reciprocal Ritz values $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_k$

Let $V_1 = x_0/\|x_0\|_2$ and $\underline{H}_0 = [\,]$.
for $j = 1, 2, \ldots$ **do**
  Compute V_{j+1} and $\underline{H}_j$ by Algorithm 2 based on V_j and $\underline{H}_{j-1}$.
  Compute the ordered Schur factorization of H_j according to (4.6).
  Let $\begin{bmatrix} b_1^* & b_2^* & b_3^* \end{bmatrix} := h_{j+1}^* \begin{bmatrix} Z_1 & Z_2 & Z_3 \end{bmatrix}$.
  Let $\begin{bmatrix} U_1 & U_2 & U_3 \end{bmatrix} := \begin{bmatrix} V_1 & V_2 & V_3 \end{bmatrix} Z_j$.
  if $j \geqslant k$ ***and*** $\operatorname{mod}(j-k, k-m) = 0$ **then**
    Let $V_{j+1} = \begin{bmatrix} U_1 & U_2 & v_{j+1} \end{bmatrix}$.
    Let $\underline{H}_j = \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \\ 0 & b_2^* \end{bmatrix}$.
  else
    Let $V_{j+1} = \begin{bmatrix} U_1 & U_2 & U_3 & v_{j+1} \end{bmatrix}$.
    Let $\underline{H}_j = \begin{bmatrix} S_{11} & S_{12} & S_{13} \\ 0 & S_{22} & S_{23} \\ 0 & 0 & S_{33} \\ 0 & b_2^* & b_3^* \end{bmatrix}$.
  end
end

Remark 4.1
Algorithms 1 and3 start from a given rank revealing decomposition (use of matrices Q and U_i). If the determination of these factors involves a low-rank approximation, then the algorithms compute eigenvalues of a *perturbed problem* obtained by replacing the original matrices by the corresponding low-rank approximations. The effect of such an approximation can be assessed *a posteriori* by checking the residual of the computed eigenvalue approximation or by computing the

DOI: 10.1002/nla

eigenvalue condition number $1/(z^*M'(\lambda)x)$, with z and x the normalized left and eigenvectors corresponding to eigenvalue λ. See, e.g., [28, 29] for nonlinear eigenvalue perturbation problems and pseudospectra.

5. NUMERICAL EXPERIMENTS

Before presenting the results of the numerical experiments, we first introduce the following notation in order to simplify referencing to the different variants of the algorithms.

- T$\mathcal{B}$: Taylor variant for operator $\mathcal{B}$, that, Taylor variant of [20, Algorithm 2],
- T$\mathcal{FB}$: Taylor variant for operator $\mathcal{FB}$, that is, Taylor variant of Algorithm 2,
- C$\mathcal{B}$: Chebyshev variant for operator $\mathcal{B}$, that is, Chebyshev variant of [20, Algorithm 2],
- C$\mathcal{FB}$: Chebyshev variant for operator $\mathcal{FB}$, that is, Chebyshev variant of Algorithm 2.

We will also add an R to denote the implicitly restarted variant, for example, T$\mathcal{FB}$R and C$\mathcal{FB}$R denote respectively the implicitly restarted Taylor and Chebyshev variants with low-rank exploitation of Algorithm 3.

Note that the dynamic variant of NLEIGS [19] corresponds to the variant $T\mathcal{B}$ in the special case where the shifts in the Rational Krylov method are all chosen zero (Hermite interpolation) and the poles of the approximation of M at infinity (polynomial approximation). See [20] for the connection between the Taylor variant of the infinite Arnoldi algorithm and dynamic polynomial approximation.

5.1. A random example

We illustrate the generality and efficiency of the algorithm by applying it to a problem with randomly generated matrices. Suppose

$$M(\lambda) = A_0 + \lambda A_1 + A_2\lambda^4 + A_3\sin(\lambda), \tag{5.1}$$

where $A_0, A_2 \in \mathbb{R}^{n\times n}$ are random sparse matrices with normal-distributed elements, $A_1 = -I$, and $A_3 = UQ^T$ with $U, Q \in \mathbb{R}^{n\times 2}$ randomly generated matrices and $Q^TQ = I$. For illustrative reasons, we set $n = 1000$. In order to make the results reproducible, we have made the matrices available online[‡].

For this example, it is natural to select $p = 4$, and the expansion (2.3) is explicitly given by

$$\begin{aligned} M_0 &= A_0 & M_3 &= -UQ^T \\ M_1 &= A_1 + UQ^T & M_4 &= 4!\cdot A_2 \\ M_2 &= 0 & M_i &= V_iQ^T, \qquad i \geqslant 5 \end{aligned}$$

where

$$\begin{aligned} V_{2k+1} &= (-1)^kU, \qquad k \geqslant 2, \\ V_{2k} &= 0, \qquad\quad\ k \geqslant 3. \end{aligned}$$

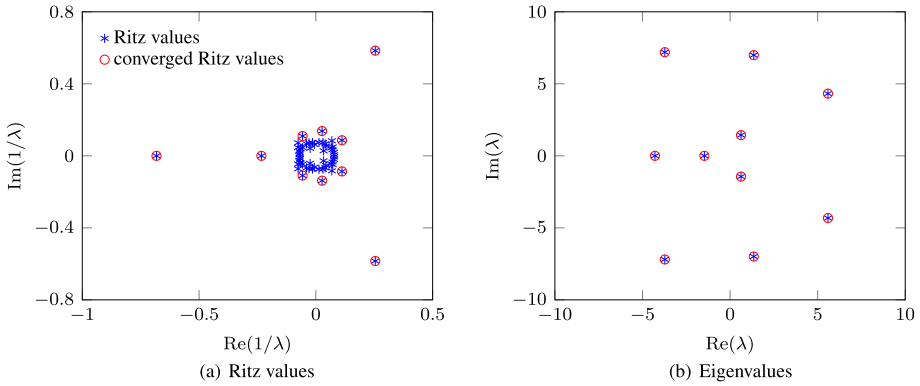
The goal in this experiment is to compute the 10 eigenvalues closest to the origin. For measuring the convergence of an approximate eigenpair (λ, x), we used the following relative residual norm:

$$E(\lambda, x) = \frac{\|M(\lambda)x\|_2/\|x\|_2}{\|A_0\|_1 + |\lambda| + \|A_2\|_1|\lambda|^4 + \|A_3\|_1|\sin(\lambda)|}.$$

In the implementation, we precompute the LU-factorization of M_0 in order to use in the formula for y_0, given by (3.5), and the terms involving M_1 and M_3 are computed as follows: $M_1y_1 = A_1y_1 + V(Q^Ty_1)$ and $M_3y_3 = -V(Q^Ty_1)$, respectively.

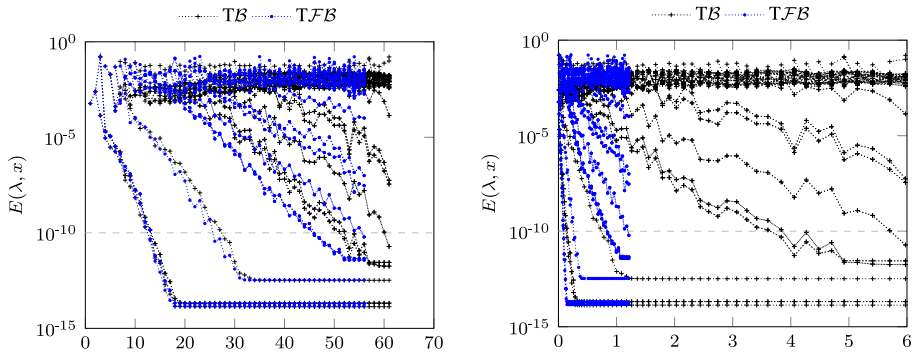
[‡]http://www.math.kth.se/~eliasj/src/lowranknep/example1_matrices_final.mat.

DOI: 10.1002/nla



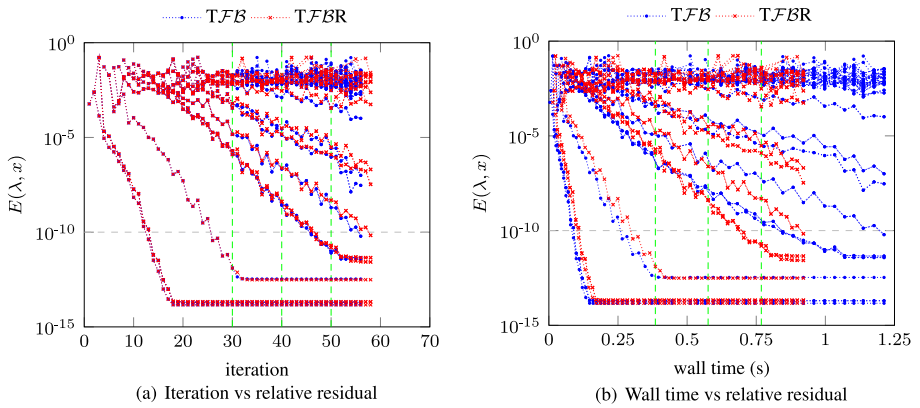
(a) Ritz values (b) Eigenvalues

Figure 3. (a) Ritz values and (b) approximate eigenvalues, that is, reciprocal Ritz values, of the random example computed with variant T$\mathcal{FB}$. An eigenvalue is classified as converged if $E(\lambda, x) \leqslant 10^{-10}$.



(a) Iteration vs relative residual (b) Wall time vs relative residual

Figure 4. Comparison of T$\mathcal{B}$ and T$\mathcal{FB}$ for (5.1). The variants generate very similar results per iteration, but the computation time grows much faster for T$\mathcal{B}$ than for T$\mathcal{FB}$. (a) Iteration versus relative residual and (b) wall time versus relative residual.



(a) Iteration vs relative residual (b) Wall time vs relative residual

Figure 5. Comparison of T$\mathcal{FB}$ and T$\mathcal{FB}R$ for (5.1). The convergence is only slightly slowed down by the restart (in terms of result per iteration), whereas the computation is further reduced. The vertical dashed lines indicate the restarts. (a) Iteration versus relative residual and (b) wall time versus relative residual.

We first solved the nonlinear eigenvalue problem (5.1) by the variants T$\mathcal{B}$ and T$\mathcal{FB}$. The eigenvalues and results of this experiment are shown in Figures 3 and 4, respectively. We observe in Figure 4(a) that the application of the operator $\mathcal{F}$ has very little impact on the approximations

DOI: 10.1002/nla

generated by each iteration of the two variants. On the other hand, using a compressed representation for T$\mathcal{FB}$ as illustrated in Figure 1(b) gives a significant performance improvement in the sense that each iteration can be carried out in less computation time. As shown in Figure 4(b), this results in a much lower total computation time for T$\mathcal{FB}$ compared with T$\mathcal{B}$.

Next, we solved (5.1) by the implicitly restarted variant T$\mathcal{FB}R$. The results are illustrated in Figure 5. We observe in Figure 5(a) that the convergence speed as a function of iteration is slightly worsened by the restarting. But as expected from restarting, we notice in Figure 5(b) that the convergence speed as a function of computation time is further improved, such that we obtain for computing 10 eigenvalues a speed up factor of 6 from T$\mathcal{B}$ to T$\mathcal{FB}R$. In Figure 6, we illustrate the growth of the computation time of the restarted variants as a function of iteration. As a comparison, the computation time grows slower when exploiting the structure. Note that without low-rank exploitation, we need $O(j^2k)$ scalar products between vectors of size n up to iteration $j > k$, with k the maximum dimension of the subspace and j the total number of iterations. On the other hand, by exploiting the low-rank structure, we only need $O(jpk)$ scalar products between vectors of size n and $O(j^2k)$ between vectors of size r. We observe in Figure 6 that the computation time grows quadratic with the iteration for T$\mathcal{B}R$. However, for T$\mathcal{FB}R$, the computation cost grows essentially linearly after the first restart, because the quadratic term in j is negligible due to the fact that $jr \ll pn$.

In what follows, we present a comparison of the variants T$\mathcal{B}$, T$\mathcal{FB}$, and T$\mathcal{FB}R$ to the static variant of the fully rational Krylov method (NLEIGS) [19]. We have selected the parameters of the NLEIGS software package in order to improve fairness of comparison. In particular, we chose all shifts in the rational Krylov process equal to zero such that we also have a polynomial (not rational)

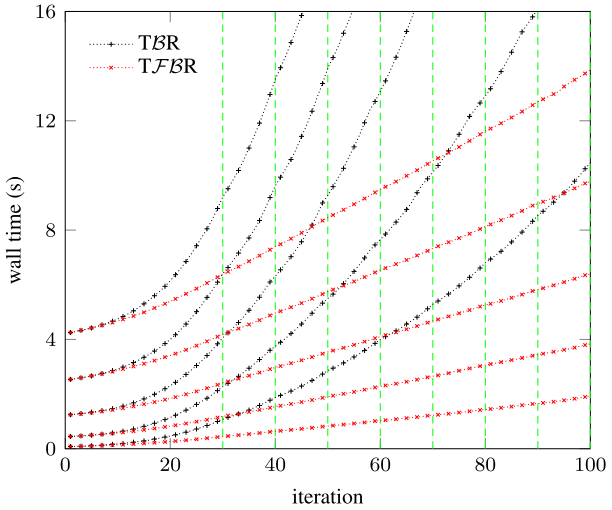


Figure 6. The computation time of T$\mathcal{B}R$ and T$\mathcal{FB}R$ for (5.1) as a function of iteration for $n = 1000, 2000, 3000, 4000, 5000$. The total computation time for T$\mathcal{FB}R$ is considerably lower than for T$\mathcal{B}R$ for larger n. The vertical dashed lines indicate the restarts.

Table I. Comparison of T$\mathcal{FB}$, T$\mathcal{FB}R$, and the static variant of NLEIGS for (5.1) with $n = 1000$.

Method	# Conv. λ	# Iter.	Wall time
T$\mathcal{B}$	10	61	5.97 s
T$\mathcal{FB}$	10	56	1.21 s
T$\mathcal{FB}R$	10	58	0.92 s
NLEIGS - disk (center = 0; radius = 1)	0	20	0.76 s
NLEIGS - disk (center = 0; radius = 5)	4	34	1.25 s
NLEIGS - disk (center = 0; radius = 10)	14	91	4.51 s

DOI: 10.1002/nla

Arnoldi process. The interpolation nodes for (5.1) are automatically placed on the boundary in a Leja–Bagby fashion.

In contrast to the variants of Algorithm 1, which require a target point, the NLEIGS software package requires a target set as input in which the algorithm computes the eigenvalues. However, how to choose this target set is less clear but crucial as illustrated in Table I. If the target set is too small, we find no eigenvalues. On the other hand, if the target set is chosen too large, we will do an unnecessary extra amount of computation work because NLEIGS tries to compute all eigenvalues inside the target accurately.

5.2. A delay eigenvalue problem

We model a one-dimensional clamped beam and delayed feedback control localized at the endpoint with a partial delay differential equation. See [30, 31] for PDEs with delays. More precisely, we consider the one-dimensional DDE

$$u_t(x,t) = u_{xx}(x,t) + \delta(x-0.5)u(0.5,t-\tau),$$

with boundary conditions $u(0,t) = u_x(1,t) = 0$ and $\delta(x)$ a Dirac impulse such that the problem corresponds to delayed pointwise feedback at $x = 0.5$. The finite difference discretization with n intervals results in the following delay eigenvalue problem:

$$M(\lambda) = \lambda I + A_0 + A_1 e^{-\tau\lambda}, \tag{5.2}$$

where $A_0 \in \mathbb{C}^{n\times n}$ is a tridiagonal matrix and A_1 a rank 1 matrix. The goal in this experiment is to compute the rightmost eigenvalues of (5.2). For measuring the convergence of an approximate eigenpair (λ, x), we used the following relative residual norm:

$$E(\lambda, x) = \frac{\|M(\lambda)x\|_2/\|x\|_2}{|\lambda| + \|A_0\|_1 + \|A_1\|_1|e^{-\tau\lambda}|}.$$

In every iteration of Algorithm 2, y_0 is computed with the formulas of Corollary 3.1.

The delay eigenvalue problem (5.2) with $n = 10.001$ and $\tau = 1$ is solved by variants C$\mathcal{B}$ and C$\mathcal{FB}$, where the Chebyshev polynomials are scaled and shifted from the interval $[-1,\ 1]$ to the interval $[-\tau,\ 0]$. Figure 7 shows the Ritz values and the 15 converged eigenvalues. In this figure, we see that the iteration of Algorithm 1 converges first to the (wanted) extreme eigenvalues of the inverted spectrum (which are well isolated). Note that even though there is no guarantee that all rightmost eigenvalues are found, an indication is that the converged part of the spectrum starts to capture the asymptotic eigenvalue chains typical for delay problems [32], containing eigenvalues with a very high imaginary part.

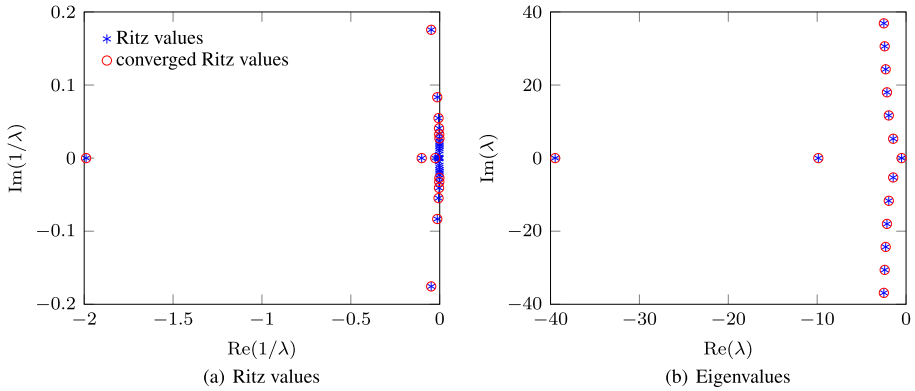
The advantage of using the operator $\mathcal{F}$ together with the compact representation is reported in Figure 8. If we only consider the relative error as a function of the iteration, we observe in Figure 8(a) that the eigenvalues converge faster for C$\mathcal{FB}$ than for C$\mathcal{B}$. But the major advantage of the compact representation can be seen in Figure 8(b), where we compare the relative error generated by C$\mathcal{B}$ and C$\mathcal{FB}$ as a function of wall time. In this figure, we see that the total computation time for C$\mathcal{FB}$ is several orders of magnitude less than for C$\mathcal{B}$. As illustrated in Figure 1(a), the subspace vectors grow in every iteration of variant C$\mathcal{B}$ with a block of size $n = 10.001$. On the other hand, in variant C$\mathcal{FB}$, they grow after the first iteration only with blocks of size $r = 1$. Therefore, C$\mathcal{B}$ handles in this experiment with vectors of size $O(10^6)$, whether C$\mathcal{FB}$ only deals with vectors of size $O(10^4)$.

We finally present a comparison of the variants C$\mathcal{B}$ and C$\mathcal{FB}$ to the static variant of NLEIGS, where we again chose all shifts in the rational Krylov process equal to zero and use polynomial interpolation. In Table II, we notice that the choice of the target set has again a large impact on the number of eigenvalues as well as on the total computation time. Furthermore, the choice of the target set in this example is more crucial than in Section 5.1 because we aim in this example to compute the rightmost eigenvalues. Also, notice that in the case of an optimal choice of the target set (last row in Table II), the static variant of NLEIGS is significantly slower than C$\mathcal{FB}$ because the Chebyshev scalar product used in C$\mathcal{FB}$ corresponds to a rational approximation of (5.2) in λ with

DOI: 10.1002/nla

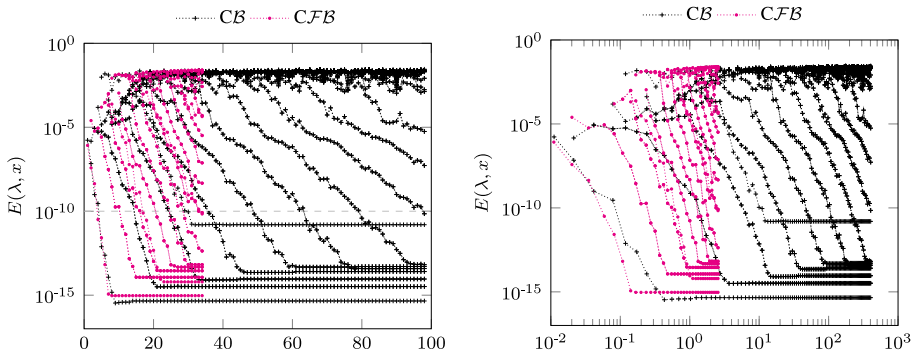
implicitly optimal pole selection, whereas NLEIGS only uses a polynomial approximation because (5.2) has no singularities.

It should be noted that a balanced comparison to NLEIGS is difficult, given that both methods are fundamentally different. The static variant of NLEIGS belongs to the class of 'discretize-first' methods, where a rational approximation of $M(\lambda)$ and corresponding linearization are constructed first and whose eigenvalues are subsequently computed with a method of choice. This is different from the infinite Arnoldi method, which is equivalent to the application of a method for the linear eigenvalue problem in an operator setting. As a consequence, their preferred use is different.



(a) Ritz values (b) Eigenvalues

Figure 7. (a) Ritz values and (b) approximate eigenvalues, that is, reciprocal Ritz values, of the delay problem computed with variant C$\mathcal{FB}$. An eigenvalue is classified as converged if $E(\lambda, x) \leqslant 10^{-10}$.



(a) Iteration vs relative residual (b) Wall time vs relative residual

Figure 8. Comparison of C$\mathcal{B}$ and C$\mathcal{FB}$ for (5.2) with $n = 10.001$. The variant C$\mathcal{FB}$ converges in less iterations than variant C$\mathcal{B}$. Furthermore, the computation time differs several orders of magnitude.

Table II. Comparison of C$\mathcal{FB}$ and the static variant of NLEIGS for (5.2) with $n = 10.001$.

Method	# Conv. λ	# Iter.	Wall time
C$\mathcal{B}$	15	98	403.64 s
C$\mathcal{FB}$	15	34	2.56 s
NLEIGS - disk (center = 0; radius = 1)	1	20	1.19 s
NLEIGS - disk (center = 0; radius = 5)	1	20	1.99 s
NLEIGS - disk (center = 0; radius=10)	4	20	3.75 s
NLEIGS - rectangle (center = 10; width = 20; height = 200)	0	20	1.63 s
NLEIGS - rectangle (center = 0; width = 2; height = 20)	1	21	1.44 s
NLEIGS - rectangle (center = 0; width = 4; height = 40)	5	46	3.56 s
NLEIGS - rectangle (center = 0; width = 8; height = 80)	13	97	11.58 s

DOI: 10.1002/nla

First, as we have seen in the previous examples, the infinite Arnoldi method is particularly suitable for the fast computation of eigenvalues close to a target. On the other hand, the static variant of NLEIGS is a very robust method for computing all eigenvalues in a predefined compact target set. The latter requires, however, that the target set is given a priori or that there is an obvious way how to choose it. Second, NLEIGS allows *explicit* rational approximation, where the user can specify the poles. This is a very powerful property, provided there is an obvious way how to select the poles (and explains why we have chosen them at infinity in the aforementioned examples). This is, for instance, the case when computing eigenvalues close to branch points or branch cuts, where NLEIGS is the method of preference [19]. The infinite Arndoldi method is not based on direct approximation of $M(\lambda)$, and consequently, no poles or interpolation nodes but an inner product needs to be chosen. The Chebyshev inner product is particularly suitable for the delay problem, which can be related to a rational approximation of exponentials. The corresponding 'good poles' are not prescribed by the user but induced by the spectral discretization of the operator on a Chebyshev grid [33].

6. CONCLUSIONS AND OUTLOOK

We have presented a new procedure to compute solutions to a type of nonlinear eigenvalue problem with a particular low-rank structure. We have constructed the algorithm such that it is equivalent to the Arnoldi method on a (infinite dimensional) linear operator, and the behavior in the numerical examples is very similar to the Arnoldi method, including the restarting features. Although the construction is general, some specific adaptations, such as efficient formulas for y_0 in (3.5) and (3.12), are necessary in order to implement the algorithm for a specific problem. The numerical examples have illustrated that for large-scale problems, the low-rank exploitation can result in significant lower computation times because of the much lower orthogonalization and memory costs.

As demonstrated by the examples, an important choice of the algorithm is the scalar product, as it affects the quality of the projections. Besides the delay problem, where significant improvements of the Chebyshev version with respect to the Taylor version can be attributed to the connection with a spectral discretization, there is currently no obvious choice, and further research is present. It should, however, be noted that even for the standard eigenvalue problem, the choice of scalar product in Krylov methods, which includes an optimal scaling of the matrix, is not yet fully understood.

Several possible continuations of this result appear feasible. There are several variants of the Arnoldi method that might be extendible, for example, a block Krylov–Schur [34] or advanced filtering techniques [35]. The understanding of the algorithm can also certainly be improved by further adapting results known for the standard Arnoldi method (for matrixes) (e.g., [36, 37]).

ACKNOWLEDGEMENTS

This research was supported by the Dahlquist research fellowship, the Programme of Interuniversity Attraction Poles of the Belgian Federal Science Policy Office (IAP P6-DYSCO), by OPTEC, the Optimization in Engineering Center of the KU Leuven, by projects STRT1-09/33 and OT/10/038 of the KU Leuven Research Council, and by project G.0712.11N of the Research Foundation-Flanders (FWO).

REFERENCES

1. Bai Z, Demmel J, Dongarra J, Ruhe A, van der Vorst HA. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM: Philadelphia, 2000.
2. Tisseur F, Meerbergen K. The quadratic eigenvalue problem. *SIAM Review* 2001; **43**(2):235–286.
3. Jarlebring E, Meerbergen K, Michiels W. A Krylov method for the delay eigenvalue problem. *SIAM Journal of Scientific Computing* 2010; **32**(6):3278–3300.
4. Voss H. An Arnoldi method for nonlinear eigenvalue problems. *BIT* 2004; **44**:387–401.
5. Unger G. Analysis of boundary element methods for Laplacian eigenvalue problems. In *Monographic Series TU Graz, Computation in Engineering and Science,* vol. 6. TU Graz/Büroservice, Austria, 2009.
6. Betcke T, Higham N J, Mehrmann V, Schröder C, Tisseur F. NLEVP: a collection of nonlinear eigenvalue problems. *ACM Transactions on Mathematical Software* 2013; **39**(2):1–28.
7. Voss H. *Handbook of Linear Algebra, Second Edition*, , chap. Nonlinear Eigenvalue Problems. No. 164 in Discrete Mathematics and Its Applications. Chapman and Hall/CRC: Boca Raton, FL, USA, 2013.

DOI: 10.1002/nla

8. Mehrmann V, Voss H. Nonlinear eigenvalue problems: a challenge for modern eigenvalue methods. *GAMM-Mitteilungen* 2004; **27**:121–152.
9. Voss H. A Jacobi–Davidson method for nonlinear and nonsymmetric eigenproblems. *Computers and structures* 2007; **85**:1284–1292.
10. Sleijpen GL, Booten AG, Fokkema DR, van der Vorst HA. Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems. *BIT* 1996; **36**(3):595–633.
11. Neumaier A. Residual inverse iteration for the nonlinear eigenvalue problem. *SIAM Journal on Numerical Analysis* 1985; **22**:914–923.
12. Kressner D. A block Newton method for nonlinear eigenvalue problems. *Numerical Mathematics* 2009; **114**(2): 355–372.
13. Effenberger C. Robust solution methods for nonlinear eigenvalue problems. *PhD Thesis*, EPFL, Lausanne, 2013.
14. Asakura J, Sakurai T, Tadano H, Ikegami T, Kimura K. A numerical method for nonlinear eigenvalue problems using contour integrals. *JSIAM Letters* 2009; **1**:52–55.
15. Beyn W J. An integral method for solving nonlinear eigenvalue problems. *Linear Algebra Applications* 2012; **436**(10):3839–3863.
16. Szyld D, Xue F. Preconditioned eigensolvers for large-scale nonlinear Hermitian eigenproblems with variational characterizations. I. Conjugate gradient methods. *Technical Report*, Dep. of Math: Temple Univ, 2014.
17. Effenberger C, Kressner D. Chebyshev interpolation for nonlinear eigenvalue problems. *BIT* 2012; **52**(4):933–951.
18. Van Beeumen R, Meerbergen K, Michiels W. A rational Krylov method based on Hermite interpolation for nonlinear eigenvalue problems. *SIAM Journal of Scientific Computing* 2013; **35**(1):A327–A350.
19. Güttel S, Van Beeumen R, Meerbergen K, Michiels W. A class of fully rational Krylov methods for nonlinear eigenvalue problems. *NLEIGS: SIAM Journal of Scientific Computing* 2014; **36**(6):A2842–A2864.
20. Jarlebring E, Michiels W, Meerbergen K. A linear eigenvalue algorithm for the nonlinear eigenvalue problem. *Numerical Mathematical* 2012; **122**(1):169–195.
21. Voss H, Yildiztekin K, Huang X. Nonlinear low rank modification of a symmetric eigenvalue problem. *SIAM Journal on Matrix Analysis and Applications* 2011; **32**(2):515–535.
22. Su Y, Bai Z. Solving rational eigenvalue problems via linearization. *SIAM Journal on Matrix Analysis and Applications* 2011; **32**(1):201–216.
23. Lu D, Huang X, Bai Z, Su Y. A Padé. *International Journal for Numerical and Methods in Enginnering* 2015; **103**(11):840–858.
24. Jarlebring E, Meerbergen K, Michiels W. Computing a partial Schur factorization of nonlinear eigenvalue problems using the infinite Arnoldi method. *SIAM Journal on Matrix Analysis and Applications* 2014; **35**(2):411–436.
25. Saad Y. *Numerical Methods for Large Eigenvalue Problems*, 2011.
26. Stewart GW. A Krylov–Schur algorithm for large eigenproblems. *SIAM Journal on Matrix Analysis and Applications* 2001; **23**(3):601–614.
27. Lehoucq R, Sorensen D. Deflation techniques for an implicitly restarted Arnoldi iteration. *SIAM Journal on Matrix Analysis and Applications* 1996; **17**(4):789–821.
28. Bindel D, Hood A. Localization theorems for nonlinear eigenvalue problems. *SIAM Journal on Matrix Analysis and Applications* 2013; **34**(4):1728–1749.
29. Michiels W, Green K, Wagenknecht T, Niculescu SI. Pseudospectra and stability radii for analytic matrix functions with application to time-delay systems. *Linear Algebra and its Applications* 2006; **418**(1):315–335.
30. Wu J. *Theory and Applications of Partial Functional Differential Equations. Applied Mathematical Sciences*, Vol. 119. Springer: New York, NY, 1996.
31. Gugat M. Boundary feedback stabilization by time delay for one-dimensional wave equations. *IMA Journal of Mathematical Control and Information* 2010; **27**(2):189–203.
32. Michiels W, Niculescu SI. *Stability, Control, and Computation for Time-Delay Systems* (2edn.) PA: Philadelphia, 2014.
33. Gumussoy S, Michiels W. A predictor–corrector type algorithm for the pseudospectral abscissa computation of time-delay systems. *Automatica Journal IFAC* 2010; **46**(4):657–664.
34. Zhou Y, Saad Y. Block Krylov-Schur method for large symmetric eigenvalue problems. *Numerical Algorithms* 2008; **47**(4):341–359.
35. Bujanović Z, Drmač Z. A new framework for implicit restarting of the Krylov-Schur algorithm. *Numerical Linear Algebra with Applications* 2015; **22**:220–232.
36. Bellalij M, Saad Y, Sadok H. Further analysis of the Arnoldi process for eigenvalue problems. *SIAM Journal on Numerical Analysis* 2010; **48**(2):393–407.
37. Jia Z. The convergence of generalized Lanczos methods for large unsymmetric eigenproblems. *SIAM Journal on Matrix Analysis and Applications* 1995; **16**(3):843–862.

DOI: 10.1002/nla