BIT Numer Math (2016) 56:241–262
DOI 10.1007/s10543-015-0566-9

Numerical methods for nonlinear two-parameter eigenvalue problems

Bor Plestenjak[1]

Received: 19 September 2014 / Accepted: 27 May 2015 / Published online: 21 July 2015

Abstract We introduce nonlinear two-parameter eigenvalue problems and generalize several numerical methods for nonlinear eigenvalue problems to nonlinear two-parameter eigenvalue problems. As a motivation we consider the computation of critical delays of delay-differential equations with multiple delays.

Keywords Nonlinear two-parameter eigenvalue problem · Nonlinear eigenvalue problem · Inverse iteration · Residual inverse iteration · Successive linear approximation · Jacobi–Davidson method · Delay-differential equations

Mathematics Subject Classification 65F15 · 15A18 · 15A69 · 47J10

1 Introduction

We consider the *nonlinear two-parameter eigenvalue problem* (*N2EP*)

$$\begin{aligned} T_1(\lambda,\mu)x_1 &= 0,\\ T_2(\lambda,\mu)x_2 &= 0, \end{aligned} \tag{1.1}$$

where $T_i(\lambda,\mu)$ is an $n_i \times n_i$ complex matrix, whose elements are analytic functions of $\lambda, \mu \in \mathbb{C}$, and $x_i \in \mathbb{C}^{n_i}$ for $i = 1, 2$. Matrices could be of different sizes, but to keep things simple, from now on we assume that $n_1 = n_2 = n$.

Communicated by Daniel Kressner.

✉ Bor Plestenjak
bor.plestenjak@fmf.uni-lj.si

1 IMFM and Department of Mathematics, University of Ljubljana, Jadranska 19, 1000 Ljubljana, Slovenia

We are searching for nonzero vectors x_1, x_2 and values λ, μ such that (1.1) is satisfied. In such case we say that the pair (λ, μ) is an eigenvalue and the tensor product $x_1 \otimes x_2$ is the corresponding (right) eigenvector. Similarly, if y_1 and y_2 are nonzero vectors such that $y_1^H T_1(\lambda, \mu) = 0$ and $y_2^H T_2(\lambda, \mu) = 0$, then $y_1 \otimes y_2$ is the corresponding left eigenvector.

The N2EP can be seen as a generalization of both the nonlinear eigenvalue problem (NEP) and the two-parameter eigenvalue problem (2EP). The eigenvalues of (1.1) are the solutions of the following system of bivariate characteristic functions

$$\begin{aligned} f_1(\lambda, \mu) &:= \det(T_1(\lambda, \mu)) = 0, \\ f_2(\lambda, \mu) &:= \det(T_2(\lambda, \mu)) = 0. \end{aligned} \tag{1.2}$$

Similar as the NEP, the problem (1.1) can have zero, finite, or infinite number of eigenvalues. We assume that the set of eigenvalues is zero-dimensional, i.e., each eigenvalue (λ, μ) is an isolated point, so that the problem of numerical computation of some or all of the eigenvalues is well defined.

The paper if organized as follows. In Sect. 2, some motivating problems are presented. In Sect. 3, some basic facts about a related linear two-parameter eigenvalue problem are given. In Sect. 4 we give some theoretical results on N2EPs with simple eigenvalues. The main part of the paper are numerical methods for the N2EP in Sect. 5, which are followed by numerical examples in Sect. 6.

2 Motivating problems

An example of a N2EP is the quadratic two-parameter eigenvalue problem (Q2EP) [15,22] of the form

$$\begin{aligned} Q_1(\lambda, \mu)\, x_1 &:= (A_{00} + \lambda A_{10} + \mu A_{01} + \lambda^2 A_{20} + \lambda\mu A_{11} + \mu^2 A_{02})\, x_1 = 0, \\ Q_2(\lambda, \mu)\, x_2 &:= (B_{00} + \lambda B_{10} + \mu B_{01} + \lambda^2 B_{20} + \lambda\mu B_{11} + \mu^2 B_{02})\, x_2 = 0, \end{aligned} \tag{2.1}$$

where A_{ij}, B_{ij} are given $n \times n$ complex matrices. A Q2EP of a simpler form, with some of the quadratic terms λ^2, $\lambda\mu$, and μ^2 missing, appears in delay-differential equations (DDEs) with the single delay [15].

The eigenvalues of the Q2EP (2.1) are the roots of the system of bivariate characteristic polynomials $\det(Q_i(\lambda, \mu)) = 0$ for $i = 1, 2$. It follows from Bézout's theorem (see, e.g., [7]) that in the generic case the Q2EP (2.1) has $4n^2$ eigenvalues.

As a Q2EP can be linearized as a singular linear two-parameter eigenvalue problem [10,22], we cannot consider it entirely as a genuine nonlinear example. This is not true for the following N2EP

$$\begin{aligned} (A_0 - \lambda I + \mu A_1 + \mu^\alpha A_2)x_1 &= 0, \\ (A_1 + \lambda\mu I + \mu A_0 + \mu^{1-\alpha} A_2)x_2 &= 0, \end{aligned} \tag{2.2}$$

where $\alpha > 0$ is not an integer. Problem (2.2) occurs in the study of critical delays of DDEs with two or more non-commensurate delays. For more details about the problem and its numerical solution, see Example 6.2.

3 Linear problem

A (*linear*) *two-parameter eigenvalue problem* (*2EP*) has the form

$$\begin{aligned} A_1x_1 &= \lambda B_1x_1 + \mu C_1x_1, \\ A_2x_2 &= \lambda B_2x_2 + \mu C_2x_2, \end{aligned} \tag{3.1}$$

where A_i, B_i, and C_i are $n \times n$ complex matrices. We can study (3.1) in the tensor product space $\mathbb{C}^n \otimes \mathbb{C}^n$ by defining the so-called *operator determinants*

$$\begin{aligned} \Delta_0 &= B_1 \otimes C_2 - C_1 \otimes B_2, \\ \Delta_1 &= A_1 \otimes C_2 - C_1 \otimes A_2, \\ \Delta_2 &= B_1 \otimes A_2 - A_1 \otimes B_2 \end{aligned}$$

(for details see, e.g., [3]). We say that the 2EP (3.1) is *nonsingular* when Δ_0 is nonsingular. In this case the matrices $\Delta_0^{-1}\Delta_1$ and $\Delta_0^{-1}\Delta_2$ commute and (3.1) is equivalent to a coupled pair of generalized eigenvalue problems

$$\begin{aligned} \Delta_1 z &= \lambda \Delta_0 z, \\ \Delta_2 z &= \mu \Delta_0 z \end{aligned} \tag{3.2}$$

for a decomposable tensor $z = x_1 \otimes x_2$.

For an overview of numerical methods for 2EPs, see, e.g., [9]. If n is small, we can solve the coupled pair (3.2). An algorithm of this kind, based on the QZ algorithm, is presented in [9]. Its complexity is $\mathcal{O}(n^6)$ because the Δ-matrices are of size $n^2 \times n^2$. Therefore, when n is large it is not feasible to compute all eigenpairs. Instead, we can compute few eigenpairs with iterative methods. The Jacobi–Davidson type method [9,13] with harmonic Ritz values can compute eigenvalues (λ, μ) that are close to a given target (λ_T, μ_T), while the Sylvester–Arnoldi type method [19] gives very good results in applications from mathematical physics where we need the eigenvalues with the smallest $|\mu|$.

There are some iterative methods that can be applied to compute a solution close to a good initial approximation. One such method is the tensor Rayleigh quotient iteration (TRQI) from [24], which is a generalization of the standard Rayleigh quotient iteration (see, e.g., [8]) and computes one eigenpair at a time.

4 Simple eigenvalues

In this section we give some theoretical results on simple eigenvalues of a N2EP and discuss similarities and differences with the NEP.

For a start we recall the situation in the one-parameter case. Let λ_* be an eigenvalue of a NEP $A(\lambda)x = 0$, where the elements of matrix A are analytic functions of λ. The geometric multiplicity $m_g(\lambda_*)$ of λ_* is equal to the nullity of $A(\lambda_*)$, and the algebraic multiplicity $m_a(\lambda_*)$ is equal to the multiplicity of λ_* as a root of $\det(A(\lambda)) = 0$. We know that $m_g(\lambda_*) \le m_a(\lambda_*)$ holds for each eigenvalue λ_*.

If (λ_*, μ_*) is an eigenvalue of the N2EP (1.1), then its geometric multiplicity is

$$m_g(\lambda_*, \mu_*) = \text{nullity}(T_1(\lambda_*, \mu_*)) \cdot \text{nullity}(T_2(\lambda_*, \mu_*)).$$

Algebraic multiplicity $m_a(\lambda_*, \mu_*)$ is the multiplicity of (λ_*, μ_*) as a root of the system (1.2). The Jacobian of (1.2) is nonsingular in an algebraically simple eigenvalue. Following a proof for the NEP in [27, Proposition 2.1], we can show that also for the N2EP the algebraic simplicity of an eigenvalue implies the geometric simplicity.

Proposition 4.1 *Each algebraically simple eigenvalue of the N2EP* (1.1) *is geometrically simple.*

Proof Let (λ_*, μ_*) be an algebraically simple eigenvalue of (1.1) and suppose that its geometric multiplicity is $m_g \ge 2$. Without loss of generality we can assume that in this case $k := \text{nullity}(T_1(\lambda_*, \mu_*)) \ge 2$. Then there exist permutation matrices P_ℓ and P_r such that the $(n-k) \times (n-k)$ block $A_{11}(\lambda_*, \mu_*)$ is nonsingular, where

$$P_\ell T_1(\lambda, \mu) P_r^T = \begin{bmatrix} A_{11}(\lambda, \mu) & A_{12}(\lambda, \mu) \\ A_{21}(\lambda, \mu) & A_{22}(\lambda, \mu) \end{bmatrix} =: A(\lambda, \mu).$$

For (λ, μ) close to (λ_*, μ_*), block $A_{11}(\lambda, \mu)$ is nonsingular and we can write

$$L(\lambda, \mu) A(\lambda, \mu) R(\lambda, \mu) = \begin{bmatrix} A_{11}(\lambda, \mu) & 0 \\ 0 & S(\lambda, \mu) \end{bmatrix} =: D(\lambda, \mu),$$

where

$$L(\lambda, \mu) = \begin{bmatrix} I_{n-k} & 0 \\ -A_{21}(\lambda, \mu) A_{11}^{-1}(\lambda, \mu) & I_k \end{bmatrix}, \quad R(\lambda, \mu) = \begin{bmatrix} I_{n-k} & -A_{11}^{-1}(\lambda, \mu) A_{12}(\lambda, \mu) \\ 0 & I_k \end{bmatrix},$$

and

$$S(\lambda, \mu) = A_{22}(\lambda, \mu) - A_{21}(\lambda, \mu) A_{11}^{-1}(\lambda, \mu) A_{12}(\lambda, \mu)$$

is the Schur complement of $A_{11}(\lambda, \mu)$.

Since $\text{rank}(D(\lambda_*, \mu_*)) = n - k$ and $A_{11}(\lambda_*, \mu_*)$ is nonsingular, $S(\lambda_*, \mu_*) = 0$. The determinant $f_1(\lambda, \mu) := \det(T_1(\lambda, \mu))$ agrees up to the sign with $\det(D(\lambda, \mu)) = \det(A_{11}(\lambda, \mu)) \det(S(\lambda, \mu))$. Since $S(\lambda_*, \mu_*) = 0$ and the size of the block $S(\lambda, \mu)$ is at least 2×2, it follows that $\frac{\partial}{\partial \lambda} \det(S(\lambda_*, \mu_*)) = \frac{\partial}{\partial \mu} \det(S(\lambda_*, \mu_*)) = 0$ and thus $\frac{\partial f_1}{\partial \lambda}(\lambda_*, \mu_*) = \frac{\partial f_1}{\partial \mu}(\lambda_*, \mu_*) = 0$. The Jacobian of $f_1(\lambda, \mu)$ and $f_2(\lambda, \mu)$ at (λ_*, μ_*) is then singular and (λ_*, μ_*) is a multiple eigenvalue, which is a contradiction. □

If λ_* is an algebraically simple eigenvalue of the NEP $A(\lambda)x = 0$, then $A(\sigma)$ is nonsingular for $\sigma \neq \lambda_*$ sufficiently close to λ_*. On the contrary, if (λ_*, μ_*) is an algebraically simple eigenvalue of the N2EP (1.1) then there always exist points $(\sigma, \tau) \neq (\lambda_*, \mu_*)$ arbitrarily close to (λ_*, μ_*) such that one of the matrices $T_1(\sigma, \tau)$ or $T_2(\sigma, \tau)$ is singular. In fact, it follows from the nonsingularity of the Jacobian of (1.2) in (λ_*, μ_*) and the implicit function theorem that in a small neighbourhood of (λ_*, μ_*) there exists a parametric curve $(\lambda_i(t), \mu_i(t))$, where $t \in (-\varepsilon, \varepsilon)$, such that $\lambda_i(0) = \lambda_*$, $\mu_i(0) = \mu_*$, and $\det(T_i(\lambda_i(t), \mu_i(t))) = 0$ for $t \in (-\varepsilon, \varepsilon)$ for $i = 1, 2$. The eigenvalue (λ_*, μ_*) is the only intersection of curves $(\lambda_1(t), \mu_1(t))$ and $(\lambda_2(t), \mu_2(t))$.

If y and x are the left and the right eigenvector of an algebraically simple eigenvalue λ_* of a NEP $A(\lambda)x = 0$, then it is well-known that $y^H A'(\lambda_*)x \neq 0$, see, e.g., [2,23,27]. In [21] this relation is generalized to the following proposition for the N2EP.

Proposition 4.2 ([21, Proposition 3.2]) *Let (λ_*, μ_*) be an algebraically simple eigenvalue of the N2EP (1.1), and let $x_1 \otimes x_2$ and $y_1 \otimes y_2$ be the corresponding right and left eigenvector. Then the matrix*

$$\begin{bmatrix} y_1^H \frac{\partial T_1}{\partial \lambda}(\lambda_*, \mu_*)x_1 & y_1^H \frac{\partial T_1}{\partial \mu}(\lambda_*, \mu_*)x_1 \\ y_2^H \frac{\partial T_2}{\partial \lambda}(\lambda_*, \mu_*)x_2 & y_2^H \frac{\partial T_2}{\partial \mu}(\lambda_*, \mu_*)x_2 \end{bmatrix} \tag{4.1}$$

is nonsingular.

The above result is used in the next section to show the quadratic convergence of a generalization of the inverse iteration to the N2EP close to an algebraically simple eigenvalue. It is also a part of the selection criteria that enables us to compute more than one eigenvalue using the Jacobi–Davidson method [11].

5 Numerical methods

In this section we generalize some numerical methods for NEPs to N2EPs. For an overview of numerical methods for the NEP, see, e.g., [20,26]. All methods in this section can be applied to a truly N2EP, i.e, such that *cannot* be transformed into a polynomial one. The methods can of course be applied to the polynomial two-parameter eigenvalue problem (P2EP) as well, but let us remark that there are other specific methods for P2EPs, like the linearization to a singular 2EP [10,22] and the Jacobi–Davidson method [11].

5.1 Inverse iteration

First of the methods that can be generalized to N2EP is the inverse iteration, introduced by Ruhe [26]. In this method we choose nonzero vectors $v_1, v_2 \in \mathbb{C}^n$ and apply Newton's method to $F(x_1, x_2, \lambda, \mu) = 0$, where

$$F(x_1,x_2,\lambda,\mu):=\begin{bmatrix} T_1(\lambda,\mu)x_1\\ T_2(\lambda,\mu)x_2\\ v_1^H x_1-1\\ v_2^H x_2-1\end{bmatrix}.$$

The vector v_i is used to normalize the eigenvector x_i, so it should not be orthogonal to x_i for $i=1,2$. If $(x_1^{(k)}, x_2^{(k)}, \lambda_k, \mu_k)$ is the current approximation for the eigenpair, then we get the update from the linear system

$$JF(x_1^{(k)}, x_2^{(k)}, \lambda_k,\mu_k)\begin{bmatrix}\Delta x_1^{(k)}\\ \Delta x_2^{(k)}\\ \Delta\lambda_k\\ \Delta\mu_k\end{bmatrix} = -\begin{bmatrix} T_1(\lambda_k,\mu_k)x_1^{(k)}\\ T_2(\lambda_k,\mu_k)x_2^{(k)}\\ v_1^H x_1^{(k)}-1\\ v_2^H x_2^{(k)}-1\end{bmatrix} \tag{5.1}$$

with the Jacobian

$$JF(x_1^{(k)}, x_2^{(k)}, \lambda_k,\mu_k) = \begin{bmatrix} T_1(\lambda_k,\mu_k) & 0 & \frac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)x_1^{(k)} & \frac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)x_1^{(k)}\\ 0 & T_2(\lambda_k,\mu_k) & \frac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)x_2^{(k)} & \frac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)x_2^{(k)}\\ v_1^H & 0 & 0 & 0\\ 0 & v_2^H & 0 & 0\end{bmatrix}. \tag{5.2}$$

Let us assume that the matrices $T_1(\lambda_k,\mu_k)$ and $T_2(\lambda_k,\mu_k)$ are nonsingular. We can further assume that $x_1^{(k)}$ and $x_2^{(k)}$ are normalized, i.e., $v_1^H x_1^{(k)} = v_2^H x_2^{(k)} = 1$. It now follows from the bottom two rows of (5.1) that $v_1^H\Delta x_1^{(k)} = v_2^H \Delta x_2^{(k)} = 0$. By multiplying the top two rows of (5.1) by v_1^H and v_2^H, respectively, we obtain a 2×2 linear system

$$\begin{bmatrix} v_1^H T_1(\lambda_k,\mu_k)^{-1}\frac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)x_1^{(k)} & v_1^H T_1(\lambda_k,\mu_k)^{-1}\frac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)x_1^{(k)}\\ v_2^H T_2(\lambda_k,\mu_k)^{-1}\frac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)x_2^{(k)} & v_2^H T_2(\lambda_k,\mu_k)^{-1}\frac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)x_2^{(k)}\end{bmatrix}\begin{bmatrix}\Delta\lambda_k\\ \Delta\mu_k\end{bmatrix} = \begin{bmatrix}-1\\-1\end{bmatrix}$$

for $\Delta\lambda_k$ and $\Delta\mu_k$. Once $\Delta\lambda_k$ and $\Delta\mu_k$ are known, we get the corrections

$$\begin{aligned}\Delta x_1^{(k)} &= -T_1(\lambda_k,\mu_k)^{-1}\left(T_1(\lambda_k,\mu_k)+\Delta\lambda_k\frac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)+\Delta\mu_k\frac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)\right)x_1^{(k)},\\ \Delta x_2^{(k)} &= -T_2(\lambda_k,\mu_k)^{-1}\left(T_2(\lambda_k,\mu_k)+\Delta\lambda_k\frac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)+\Delta\mu_k\frac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)\right)x_2^{(k)}\end{aligned} \tag{5.3}$$

from the top two rows of (5.1). This gives the approximation for the eigenvector

$$\begin{aligned}x_1^{(k+1)} &= -\Delta\lambda_k T_1(\lambda_k,\mu_k)^{-1}\frac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)x_1^{(k)} - \Delta\mu_k T_1(\lambda_k,\mu_k)^{-1}\frac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)x_1^{(k)},\\ x_2^{(k+1)} &= -\Delta\lambda_k T_2(\lambda_k,\mu_k)^{-1}\frac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)x_2^{(k)} - \Delta\mu_k T_2(\lambda_k,\mu_k)^{-1}\frac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)x_2^{(k)}.\end{aligned}$$

The overall algorithm is presented in Algorithm 1. The complexity of one step is $\mathscr{O}(n^3)$. Note that Step 8 is just for precaution in numerical computation as in theory vectors $x_1^{(k+1)}$ and $x_2^{(k+1)}$ should already be normalized.

Algorithm 1 Inverse iteration (InvIter)

```
1: Start with $\lambda_0, \mu_0, v_1, v_2, x_1^{(0)}$, and $x_2^{(0)}$ such that $v_1^H x_1^{(0)} = v_2^H x_2^{(0)} = 1$.
2: for $k = 0, 1, 2, \ldots$ until convergence do
3:    Compute $a_1 = T_1(\lambda_k, \mu_k)^{-1} \frac{\partial T_1}{\partial \lambda}(\lambda_k, \mu_k) x_1^{(k)}$ and $a_2 = T_2(\lambda_k, \mu_k)^{-1} \frac{\partial T_2}{\partial \lambda}(\lambda_k, \mu_k) x_2^{(k)}$.
4:    Compute $b_1 = T_1(\lambda_k, \mu_k)^{-1} \frac{\partial T_1}{\partial \mu}(\lambda_k, \mu_k) x_1^{(k)}$ and $b_2 = T_2(\lambda_k, \mu_k)^{-1} \frac{\partial T_2}{\partial \mu}(\lambda_k, \mu_k) x_2^{(k)}$.
5:    Solve
      $$\begin{bmatrix} v_1^H a_1 & v_1^H b_1 \\ v_2^H a_2 & v_2^H b_2 \end{bmatrix} \begin{bmatrix} \Delta\lambda_k \\ \Delta\mu_k \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}.$$
6:    Update $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$ and $\mu_{k+1} = \mu_k + \Delta\mu_k$.
7:    Compute $x_1^{(k+1)} = -\Delta\lambda_k a_1 - \Delta\mu_k b_1$ and $x_2^{(k+1)} = -\Delta\lambda_k a_2 - \Delta\mu_k b_2$.
8:    Normalize $x_1^{(k+1)} = x_1^{(k+1)}/(v_1^H x_1^{(k+1)})$ and $x_2^{(k+1)} = x_2^{(k+1)}/(v_2^H x_2^{(k+1)})$.
9: end for
```

Steps 3 and 4 rely on the nonsingularity of $T_1(\lambda_k, \mu_k)$ and $T_2(\lambda_k, \mu_k)$. Let us show that in practice this does not present a difficulty. When we are close to the solution, $T_1(\lambda_k, \mu_k)$ and $T_2(\lambda_k, \mu_k)$ are nearly singular and we can expect that vectors a_1, a_2, b_1, and b_2 have huge norm, which then also applies to the matrix in Step 5. However, the condition number of this matrix stays bounded as in vicinity of the eigenpair it is close to a diagonally scaled matrix (4.1), where the scaling factors are expected to be of a similar magnitude. A rough justification is as follows.

Let $(x_1^{(k)}, x_2^{(k)}, \lambda_k, \mu_k)$ be close to $(x_1, x_2, \lambda_*, \mu_*)$, where (λ_*, μ_*) is an algebraically simple eigenvalue with the right and left eigenvectors $x_1 \otimes x_2$ and $y_1 \otimes y_2$. Then the smallest singular value σ_{in} of $T_i(\lambda_k, \mu_k)$ is close to zero and the corresponding right and left singular vectors are close to x_i and y_i, respectively. It follows that

$$a_i \approx \frac{y_i^H \frac{\partial T_i}{\partial \lambda}(\lambda_*, \mu_*) x_i}{\sigma_{in}} x_i, \quad b_i \approx \frac{y_i^H \frac{\partial T_i}{\partial \mu}(\lambda_*, \mu_*) x_i}{\sigma_{in}} x_i$$

for $i = 1, 2$ and

$$\begin{bmatrix} v_1^H a_1 & v_1^H b_1 \\ v_2^H a_2 & v_2^H b_2 \end{bmatrix} \approx \begin{bmatrix} \sigma_{1n}^{-1} & \\ & \sigma_{2n}^{-1} \end{bmatrix} \begin{bmatrix} y_1^H \frac{\partial T_1}{\partial \lambda}(\lambda_*, \mu_*) x_1 & y_1^H \frac{\partial T_1}{\partial \mu}(\lambda_*, \mu_*) x_1 \\ y_2^H \frac{\partial T_2}{\partial \lambda}(\lambda_*, \mu_*) x_2 & y_2^H \frac{\partial T_2}{\partial \mu}(\lambda_*, \mu_*) x_2 \end{bmatrix}.$$

In practice we do not have to worry if $T_i(\lambda_k, \mu_k)$ is singular, because, similar as in the inverse power method, rounding errors prevent the method from getting into trouble. A safe alternative with no worries is to solve (5.1) as a $(2n + 2) \times (2n + 2)$ linear system. However, this is approximately four times more expensive since we do not exploit the block structure of the Jacobian matrix.

Theorem 5.1 *Let (λ_*, μ_*) be an algebraically simple eigenvalue of the N2EP (1.1) and let $x_1 \otimes x_2$ be the corresponding right eigenvector such that $v_1^H x_1 = v_2^H x_2 = 1$. Then the inverse iteration has quadratic convergence close to the solution.*

Proof Since this is Newton's method, it is enough to show that the Jacobian (5.2) is nonsingular at $(x_1, x_2, \lambda_*, \mu_*)$. So, suppose that the Jacobian is singular. Then there exist vectors z_1, z_2 and scalars α_1, α_2, not all of them being zero, such that

$$JF(x_1, x_2, \lambda_*, \mu_*) \begin{bmatrix} z_1 \\ z_2 \\ \alpha_1 \\ \alpha_2 \end{bmatrix} = 0. \tag{5.4}$$

If we multiply the first and the second equation of (5.4) by y_1^H and y_2^H, respectively, where $y_1 \otimes y_2$ is the left eigenvector of (λ_*, μ_*), we obtain

$$\begin{bmatrix} y_1^H \frac{\partial T_1}{\partial \lambda}(\lambda_*, \mu_*)x_1 & y_1^H \frac{\partial T_1}{\partial \mu}(\lambda_*, \mu_*)x_1 \\ y_2^H \frac{\partial T_2}{\partial \lambda}(\lambda_*, \mu_*)x_2 & y_2^H \frac{\partial T_2}{\partial \mu}(\lambda_*, \mu_*)x_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = 0.$$

As we know from Proposition 4.2 that the above system is nonsingular for an algebraically simple eigenvalue, it follows that $\alpha_1 = \alpha_2 = 0$.

From the first and the second equation of (5.4) we now see that there should exist scalars β_1 and β_2 such that $z_1 = \beta_1 x_1$ and $z_2 = \beta_2 x_2$, as the null spaces of $T_1(\lambda_*, \mu_*)$ and $T_2(\lambda_*, \mu_*)$ have dimension 1. But then the last two equations of (5.4) read as $\beta_1 v_1^H x_1 = \beta_2 v_2^H x_2 = 0$, which yields $\beta_1 = \beta_2 = 0$ and we have a contradiction. □

The inverse iteration was applied to 2EPs in [16] and as a part of the continuation method in [24]. On both occasions the method was enhanced by a generalization of the Rayleigh quotient for the 2EP. In the next subsection we introduce several generalizations of the Rayleigh quotient for N2EPs that could be used for such purpose.

5.2 Residual inverse iteration

In each step of the inverse iteration we spend many operations for solving the linear systems with the matrices $T_1(\lambda_k, \mu_k)$ and $T_2(\lambda_k, \mu_k)$. The same difficulty appears in the inverse iteration for the NEP. Neumaier showed in [23] that in the NEP we can use a fixed matrix instead, if we are prepared to trade the quadratic convergence for a linear one.

This approach can be generalized to the N2EP as follows. Starting from (5.3) and following arguments from [23] we write

$$\begin{aligned} x_i^{(k)} - x_i^{(k+1)} &= T_i(\lambda_k, \mu_k)^{-1}\big(T_i(\lambda_k, \mu_k) + \Delta\lambda_k \tfrac{\partial T_i}{\partial \lambda}(\lambda_k, \mu_k) + \Delta\mu_k \tfrac{\partial T_i}{\partial \mu}(\lambda_k, \mu_k)\big)x_i^{(k)} \\ &= T_i(\lambda_k, \mu_k)^{-1}\big(T_i(\lambda_{k+1}, \mu_{k+1}) + \mathcal{O}\left((|\Delta\lambda_k| + |\Delta\mu_k|)^2\right)\big)x_i^{(k)} \\ &= T_i(\lambda_k, \mu_k)^{-1} T_i(\lambda_{k+1}, \mu_{k+1})x_i^{(k)} + \mathcal{O}\left((|\Delta\lambda_k| + |\Delta\mu_k|)^2\right) \end{aligned}$$

for $i = 1, 2$. We can approximate $T_i(\lambda_k, \mu_k)^{-1}$ by $T_i(\sigma, \tau)^{-1}$, where (σ, τ) is a fixed shift close to the eigenvalue, if we have an approximation for $(\lambda_{k+1}, \mu_{k+1})$. To obtain such approximation, we generalize the Rayleigh functional to N2EP.

If $x_1^{(k)} \otimes x_2^{(k)}$ is an approximation for the eigenvector, then the first generalization of the Rayleigh functional are solutions of the following, in general nonlinear, bivariate system

$$\begin{aligned} {x_1^{(k)}}^H T_1(\lambda_{k+1}, \mu_{k+1}) x_1^{(k)} &= 0, \\ {x_2^{(k)}}^H T_2(\lambda_{k+1}, \mu_{k+1}) x_2^{(k)} &= 0. \end{aligned} \tag{5.5}$$

For a new approximation $(\lambda_{k+1}, \mu_{k+1})$ we take the solution of (5.5) that is closest to (λ_k, μ_k). For the 2EP the system (5.5) has exactly one solution. It is called the tensor Rayleigh quotient and is equal to the pair (ρ_1, ρ_2), where

$$\rho_1(x_1^{(k)}, x_2^{(k)}) := \frac{({x_1^{(k)}}^H A_1 x_1^{(k)})({x_2^{(k)}}^H C_2 x_2^{(k)}) - ({x_1^{(k)}}^H C_1 x_1^{(k)})({x_2^{(k)}}^H A_2 x_2^{(k)})}{({x_1^{(k)}}^H B_1 x_1^{(k)})({x_2^{(k)}}^H C_2 x_2^{(k)}) - ({x_1^{(k)}}^H C_1 x_1^{(k)})({x_2^{(k)}}^H B_2 x_2^{(k)})},$$

$$\rho_2(x_1^{(k)}, x_2^{(k)}) := \frac{({x_1^{(k)}}^H B_1 x_1^{(k)})({x_2^{(k)}}^H A_2 x_2^{(k)}) - ({x_1^{(k)}}^H A_1 x_1^{(k)})({x_2^{(k)}}^H B_2 x_2^{(k)})}{({x_1^{(k)}}^H B_1 x_1^{(k)})({x_2^{(k)}}^H C_2 x_2^{(k)}) - ({x_1^{(k)}}^H C_1 x_1^{(k)})({x_2^{(k)}}^H B_2 x_2^{(k)})}.$$

For example, if we take the Q2EP, then (5.5) has 4 solutions in the general case. At least one of these four solutions is an eigenvalue if $x_1^{(k)} \otimes x_2^{(k)}$ is an exact eigenvector.

We can think of (5.5) as a one-sided Rayleigh functional, because we use the same vectors on the left and the right side. We could use the two-sided version instead, where we use approximation for the left eigenvector on the left side. If, similar to the NEP, we use $T_1(\sigma, \tau)^{-H} v_1 \otimes T_2(\sigma, \tau)^{-H} v_2$ for an approximation to the left eigenvector, we obtain the following system for the two-sided Rayleigh functional:

$$\begin{aligned} v_1^H T_1(\sigma, \tau)^{-1} T_1(\lambda_{k+1}, \mu_{k+1}) x_1^{(k)} &= 0, \\ v_2^H T_2(\sigma, \tau)^{-1} T_2(\lambda_{k+1}, \mu_{k+1}) x_2^{(k)} &= 0. \end{aligned} \tag{5.6}$$

Instead of solving (5.6) we perform one step of Newton's method using (λ_k, μ_k) as an initial approximation. As we show in Theorem 5.2, this is enough for convergence. This yields

$$\begin{bmatrix} v_1^H T_1(\sigma,\tau)^{-1} \frac{\partial T_1}{\partial \lambda}(\lambda_k,\mu_k) x_1^{(k)} & v_1^H T_1(\sigma,\tau)^{-1} \frac{\partial T_1}{\partial \mu}(\lambda_k,\mu_k) x_1^{(k)} \\ v_2^H T_2(\sigma,\tau)^{-1} \frac{\partial T_2}{\partial \lambda}(\lambda_k,\mu_k) x_2^{(k)} & v_2^H T_2(\sigma,\tau)^{-1} \frac{\partial T_2}{\partial \mu}(\lambda_k,\mu_k) x_2^{(k)} \end{bmatrix} \begin{bmatrix} \Delta\lambda_k \\ \Delta\mu_k \end{bmatrix} = -\begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix}, \tag{5.7}$$

where $\gamma_i = v_i^H T_i(\sigma, \tau)^{-1} T_i(\lambda_k, \mu_k) x_i^{(k)}$ for $i = 1, 2$. The values $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$ and $\mu_{k+1} = \mu_k + \Delta\mu_k$ present a generalization of Lancaster's one-parameter Rayleigh functional [18].

Hence, when we use the residual iteration, we first compute the new eigenvalue approximation $(\lambda_{k+1}, \mu_{k+1})$ from (5.7). The new eigenvector approximation before normalization is then

$$z_1 = x_1^{(k)} - T_1(\sigma,\tau)^{-1}T_1(\lambda_{k+1},\mu_{k+1})x_1^{(k)},$$
$$z_2 = x_2^{(k)} - T_2(\sigma,\tau)^{-1}T_2(\lambda_{k+1},\mu_{k+1})x_2^{(k)}.$$

Algorithm 2 Residual iteration (ResIter)

```
1: Start with $\lambda_0, \mu_0, v_1, v_2, x_1^{(0)}$, and $x_2^{(0)}$ such that $v_1^H x_0^{(1)} = v_2^H x_0^{(2)} = 1$.
2: for $k = 0, 1, 2, \ldots$ until convergence do
3:   Compute $a_1 = T_1(\lambda_0,\mu_0)^{-1}\frac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)x_1^{(k)}$ and $a_2 = T_2(\lambda_0,\mu_0)^{-1}\frac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)x_2^{(k)}$.
4:   Compute $b_1 = T_1(\lambda_0,\mu_0)^{-1}\frac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)x_1^{(k)}$ and $b_2 = T_2(\lambda_0,\mu_0)^{-1}\frac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)x_2^{(k)}$.
5:   Compute $\gamma_1 = v_1^H T_1(\lambda_0,\mu_0)^{-1}T_1(\lambda_k,\mu_k)x_1^{(k)}$ and $\gamma_2 = v_2^H T_2(\lambda_0,\mu_0)^{-1}T_2(\lambda_k,\mu_k)x_2^{(k)}$.
6:   Solve
       $\begin{bmatrix} v_1^H a_1 & v_1^H b_1 \\ v_2^H a_2 & v_2^H b_2 \end{bmatrix}\begin{bmatrix}\Delta\lambda_k \\ \Delta\mu_k\end{bmatrix} = \begin{bmatrix}-\gamma_1 \\ -\gamma_2\end{bmatrix}.$
7:   Update $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$ and $\mu_{k+1} = \mu_k + \Delta\mu_k$.
8:   $x_1^{(k+1)} = x_1^{(k)} - T_1(\lambda_0,\mu_0)^{-1}T_1(\lambda_{k+1},\mu_{k+1})x_1^{(k)}$
9:   $x_2^{(k+1)} = x_2^{(k)} - T_2(\lambda_0,\mu_0)^{-1}T_2(\lambda_{k+1},\mu_{k+1})x_2^{(k)}$
10:  Normalize $x_1^{(k+1)} = x_1^{(k+1)}/(v_1^H x_1^{(k+1)})$ and $x_2^{(k+1)} = x_2^{(k+1)}/(v_2^H x_2^{(k+1)})$.
11: end for
```

The above procedure is presented in Algorithm 2, where we use the initial eigenvalue approximation (λ_0, μ_0) as a fixed shift (σ, τ). The residual iteration has linear convergence if the initial approximation is close to the eigenpair.

Theorem 5.2 *Let (λ_*, μ_*) be an algebraically simple eigenvalue of the N2EP (1.1) and let $x_1 \otimes x_2$ be the corresponding right eigenvector such that $v_1^H x_1 = v_2^H x_2 = 1$. If (λ_0, μ_0) is sufficiently close to the eigenvalue, then the residual iteration has linear convergence close to the solution.*

Proof We know that the inverse iteration, being a Newton's method, has quadratic convergence close to the solution. If we replace the Jacobian (5.2) in the inverse iteration with the matrix

$$B(x_1^{(k)}, x_2^{(k)}, \lambda_k, \mu_k) := \begin{bmatrix} T_1(\lambda_0,\mu_0) & 0 & \frac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)x_1^{(k)} & \frac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)x_1^{(k)} \\ 0 & T_2(\lambda_0,\mu_0) & \frac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)x_2^{(k)} & \frac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)x_2^{(k)} \\ v_1^H & 0 & 0 & 0 \\ 0 & v_2^H & 0 & 0 \end{bmatrix},$$

where the top left diagonal blocks of the Jacobian are fixed at (λ_0, μ_0), then the new method has linear convergence when (λ_0, μ_0) is sufficiently close to (λ_*, μ_*) (see, e.g., [17, Theorem 5.4.1]). If we assume that $v_1^H x_1^{(k)} = v_2^H x_2^{(k)} = 1$, then the solution of the system

$$B(x_1^{(k)}, x_2^{(k)}, \lambda_k, \mu_k)^{-1}\begin{bmatrix}\Delta x_1^{(k)} \\ \Delta x_2^{(k)} \\ \Delta\lambda_k \\ \Delta\mu_k\end{bmatrix} = -\begin{bmatrix} T_1(\lambda_k,\mu_k)x_1^{(k)} \\ T_2(\lambda_k,\mu_k)x_2^{(k)} \\ v_1^H x_1^{(k)} - 1 \\ v_2^H x_2^{(k)} - 1 \end{bmatrix}$$

gives (5.7) for the corrections $\Delta\lambda_k$ and $\Delta\mu_k$, and we get

$$\begin{aligned}x_i^{(k+1)} &= x_i^{(k)} - T_i(\lambda_0,\mu_0)^{-1}\big(T_i(\lambda_k,\mu_k) + \Delta\lambda_k \tfrac{\partial T_i}{\partial\lambda}(\lambda_k,\mu_k) + \Delta\mu_k \tfrac{\partial T_i}{\partial\mu}(\lambda_k,\mu_k)\big)x_i^{(k)} \\ &= x_i^{(k)} - T_i(\lambda_0,\mu_0)^{-1}\left(T_i(\lambda_{k+1},\mu_{k+1}) + \mathscr{O}\left((|\Delta\lambda_k| + |\Delta\mu_k|)^2\right)\right)x_i^{(k)}\end{aligned}$$

for $i = 1, 2$. This differs only for $\mathscr{O}\left((|\Delta\lambda_k| + |\Delta\mu_k|)^2\right)$ from the vectors that are computed in Steps 8 and 9 in Algorithm 2. As this difference is too small to destroy the convergence, the residual iteration has linear convergence as well. □

5.3 Successive linear problems

This method was also introduced by Ruhe [26]. Its generalization to the N2EP is as follows. We expand

$$\begin{aligned}T_1(\lambda_k - \Delta\lambda_k, \mu_k - \Delta\mu_k)x_1 &= 0,\\ T_2(\lambda_k - \Delta\lambda_k, \mu_k - \Delta\mu_k)x_2 &= 0\end{aligned}$$

in the Taylor series as

$$\begin{aligned}\big(T_1(\lambda_k,\mu_k) - \Delta\lambda_k \tfrac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k) - \Delta\mu_k \tfrac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k) + \mathscr{O}\left((|\Delta\lambda_k| + |\Delta\mu_k|)^2\right)\big)x_1 &= 0,\\ \big(T_2(\lambda_k,\mu_k) - \Delta\lambda_k \tfrac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k) - \Delta\mu_k \tfrac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k) + \mathscr{O}\left((|\Delta\lambda_k| + |\Delta\mu_k|)^2\right)\big)x_2 &= 0.\end{aligned}$$

We discard higher order terms and take for $(\Delta\lambda_k, \Delta\mu_k)$ the eigenvalue closest to $(0, 0)$ of the 2EP

$$\begin{aligned}T_1(\lambda_k,\mu_k)x_1 &= \Delta\lambda_k \tfrac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)x_1 + \Delta\mu_k \tfrac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)x_1,\\ T_2(\lambda_k,\mu_k)x_2 &= \Delta\lambda_k \tfrac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)x_2 + \Delta\mu_k \tfrac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)x_2.\end{aligned} \tag{5.8}$$

The procedure is presented in Algorithm 3.

Algorithm 3 Method of successive linear problems (SuccLP)

```
1: Start with $\lambda_0$ and $\mu_0$.
2: for $k = 0, 1, 2, \ldots$ until convergence do
3:     Solve the linear two-parameter eigenvalue problem
           $T_1(\lambda_k,\mu_k)x_1 = \sigma \frac{\partial T_1}{\partial\lambda}(\lambda_k,\mu_k)x_1 + \tau \frac{\partial T_1}{\partial\mu}(\lambda_k,\mu_k)x_1,$
           $T_2(\lambda_k,\mu_k)x_2 = \sigma \frac{\partial T_2}{\partial\lambda}(\lambda_k,\mu_k)x_2 + \tau \frac{\partial T_2}{\partial\mu}(\lambda_k,\mu_k)x_2.$
4:     Select the eigenvalue $(\sigma, \tau)$ that is closest to $(0, 0)$.
5:     Update $\lambda_{k+1} = \lambda_k - \sigma$ and $\mu_{k+1} = \mu_k - \tau$.
6: end for
```

Numerical results show that the convergence is quadratic. One step has complexity $\mathcal{O}(n^6)$ when the algorithm from [9] is used to solve the 2EP in Step 3. But, since we only need one eigenvalue of (5.8), we can merge Steps 3 and 4 and apply an iterative method that can efficiently compute the closest eigenvalue. Here, the Jacobi–Davidson method from [13] or subspace methods from [19] could be applied.

5.4 Newton's method on determinants

Instead of (1.1) we can consider the system of determinants (1.2). In order to apply Newton's method we need an efficient numerical method for the partial derivatives $\frac{\partial f_i}{\partial \lambda}(\lambda,\mu)$ and $\frac{\partial f_i}{\partial \mu}(\lambda,\mu)$, where $f_i(\lambda,\mu) := \det(T_i(\lambda,\mu))$ for $i=1,2$. If $f_i(\lambda,\mu)\neq 0$ then Jacobi's formula for the derivative of the determinant yields

$$\begin{aligned}\frac{1}{f_i(\lambda,\mu)}\cdot\frac{\partial f_i}{\partial\lambda}(\lambda,\mu) &= \mathrm{tr}\left(T_i(\lambda,\mu)^{-1}\frac{\partial T_i}{\partial\lambda}(\lambda,\mu)\right),\\ \frac{1}{f_i(\lambda,\mu)}\cdot\frac{\partial f_i}{\partial\mu}(\lambda,\mu) &= \mathrm{tr}\left(T_i(\lambda,\mu)^{-1}\frac{\partial T_i}{\partial\mu}(\lambda,\mu)\right)\end{aligned} \tag{5.9}$$

for $i=1,2$. Using the above formulae we can compute the derivatives in $\mathcal{O}(n^3)$.

In [5] one can find an algorithm based on the LU factorization. To simplify the presentation, we describe the algorithm for one-parameter only. Suppose that $\det(A(\lambda))\neq 0$ and that $PA(\lambda)=LU$ is the result of the Gaussian elimination with partial pivoting, where P is a permutation matrix, L is a lower triangular matrix with ones on the diagonal and U is an upper triangular matrix. Then

$$f(\lambda)=\det(A(\lambda))=\det(P)\prod_{j=1}^{n}u_{jj}.$$

If we fix the permutation matrix P, then for each ν in a small neighbourhood of λ there exist matrices $L(\nu)$ and $U(\nu)$ such that

$$L(\nu)U(\nu)=PA(\nu) \tag{5.10}$$

is the LU factorization of $PA(\nu)$. By differentiating (5.10) at $\nu=\lambda$ we get

$$PA'(\lambda)=L'(\lambda)U(\lambda)+L(\lambda)U'(\lambda)=MU+LV, \tag{5.11}$$

where $M := L'(\lambda)$ is a lower triangular matrix with zeros on the diagonal and $V := U'(\lambda)$ is an upper triangular matrix. Matrices M and V of the proper form that satisfy (5.11) can be computed in $\mathcal{O}(n^3)$ flops from $A'(\lambda)$, P, L, and U, for details, see [5] or Algorithm 6.1 in [25]. It follows that

$$\frac{f'(\lambda)}{f(\lambda)}=\sum_{j=1}^{n}\frac{v_{jj}}{u_{jj}}.$$

The Newton's method combined with the above approach to compute the derivatives is presented in Algorithm 4. We assume that $T_1(\lambda_k, \mu_k)$ and $T_2(\lambda_k, \mu_k)$ are nonsingular. If not, we can use a slightly modified algorithm from [5] that is able to compute $f'(\lambda)$ by the LU factorization even when $f(\lambda) = 0$.

Algorithm 4 Newton's method on characteristic functions (NewtCF)

```
1: Start with $\lambda_0$ and $\mu_0$.
2: for $k = 0, 1, 2, \dots$ until convergence do
3:   Compute LU factorization with partial pivoting $P_i T_i(\lambda_k, \mu_k) = L_i U_i$ for $i = 1, 2$.
4:   Compute a lower triangular matrix $M_{1i}$ with zero diagonal and an upper triangular matrix $V_{1i}$ such
     that $P_i \frac{\partial T_i}{\partial \lambda}(\lambda_k, \mu_k) = M_{1i} U_i + L_i V_{1i}$ for $i = 1, 2$.
5:   Compute a lower triangular matrix $M_{2i}$ with zero diagonal and an upper triangular matrix $V_{2i}$ such
     that $P_i \frac{\partial T_i}{\partial \mu}(\lambda_k, \mu_k) = M_{2i} U_i + L_i V_{2i}$ for $i = 1, 2$.
6:   Compute $\alpha_i = \sum_{j=1}^n (V_{1i})_{jj}/(U_i)_{jj}$ and $\beta_i = \sum_{j=1}^n (V_{2i})_{jj}/(U_i)_{jj}$ for $i = 1, 2$.
7:   Solve
     $$\begin{bmatrix} \alpha_1 & \beta_1 \\ \alpha_2 & \beta_2 \end{bmatrix} \begin{bmatrix} \Delta\lambda_k \\ \Delta\mu_k \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}.$$
8:   Update $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$ and $\mu_{k+1} = \mu_k + \Delta\mu_k$.
9: end for
```

As this is Newton's method, the obtained method has quadratic convergence close to an algebraically simple eigenvalue. A variant of Algorithm 4 was already applied to the 2EP, see, e.g. [6].

5.5 Implicit determinant algorithm

A bottleneck of Algorithm 4 is the computation of partial derivatives of the determinants. An alternative is based on the implicit determinant algorithm, proposed in [29], see also [1], for the one-parameter nonlinear eigenvalue problem.

Lemma 5.3 *Let (λ_*, μ_*) be an algebraically simple eigenvalue of the N2EP (1.1) and let $x_1 \otimes x_2$ and $y_1 \otimes y_2$ be the corresponding right and left eigenvector. If vectors u_i and v_i are such that $u_i^H y_i \neq 0$ and $v_i^H x_i \neq 0$, then the bordered matrix*

$$M_i(\lambda, \mu) := \begin{bmatrix} T_i(\lambda, \mu) & u_i \\ v_i^H & 0 \end{bmatrix} \tag{5.12}$$

is nonsingular at (λ_*, μ_*) for $i = 1, 2$.

Proof Suppose that

$$M_i(\lambda_*, \mu_*) \begin{bmatrix} z \\ \alpha \end{bmatrix} = \begin{bmatrix} T_i(\lambda, \mu) & u_i \\ v_i^H & 0 \end{bmatrix} \begin{bmatrix} z \\ \alpha \end{bmatrix} = 0.$$

When we multiply the first equation by y_i^H we get $\alpha y_i^H u_i = 0$ and $y_i^H u_i \neq 0$ yields $\alpha = 0$. It follows that $z = \beta x_i$ for a scalar β. But $\beta v_i^H x_i = 0$ and $v_i^H x_i \neq 0$, therefore $\beta = 0$. □

Close to (λ_*, μ_*) we define vector $x_i(\lambda, \mu)$ and scalar $g_i(\lambda, \mu)$ as the solution of

$$M_i(\lambda, \mu) \begin{bmatrix} x_i(\lambda, \mu) \\ g_i(\lambda, \mu) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

for $i = 1, 2$. Then, by Cramer's rule,

$$g_i(\lambda, \mu) = \frac{\det(T_i(\lambda, \mu))}{\det(M_i(\lambda, \mu))}, \quad i = 1, 2, \tag{5.13}$$

and $f_1(\lambda, \mu) = f_2(\lambda, \mu) = 0$ has the same root (λ_*, μ_*) as $g_1(\lambda, \mu) = g_2(\lambda, \mu) = 0$. By differentiating (5.13) we get the linear system

$$M_i(\lambda, \mu) \begin{bmatrix} \frac{\partial x_i}{\partial \lambda}(\lambda, \mu) & \frac{\partial x_i}{\partial \mu}(\lambda, \mu) \\ \frac{\partial g_i}{\partial \lambda}(\lambda, \mu) & \frac{\partial g_i}{\partial \mu}(\lambda, \mu) \end{bmatrix} = - \begin{bmatrix} \frac{\partial T_i}{\partial \lambda}(\lambda, \mu) x_i & \frac{\partial T_i}{\partial \mu}(\lambda, \mu) x_i \\ 0 & 0 \end{bmatrix}$$

with the same matrix as in (5.13) and solve it to get the partial derivatives of $\frac{\partial g_i}{\partial \lambda}$ and $\frac{\partial g_i}{\partial \mu}$ for the Newton update. The overall method is presented in Algorithm 5.

In Algorithm 5 the matrix in Steps 3 and 4 is the same, so we have to compute the factorization only once. One step is faster than applying (5.9) or Algorithm 4. The algorithm depends on the vectors u_i and v_i. The optimal choice for u_i and v_i are left and right eigenvector components $u_i = y_i$ and $v_i = x_i$ for $i = 1, 2$ (see, e.g., Subsection 4.5 in [1]). If u_i and v_i differ much from y_i and x_i then Algorithm 5 can converge to a root different than (λ_*, μ_*).

Algorithm 5 Implicit determinant method (ImpDet)

```
1: Start with $\lambda_0, \mu_0$ and nonzero vectors $u_1, u_2, v_1, v_2$.
2: for $k = 0, 1, 2, \ldots$ until convergence do
3:    Solve $\begin{bmatrix} T_i(\lambda, \mu) & u_i \\ v_i^H & 0 \end{bmatrix} \begin{bmatrix} x_i \\ \gamma_i \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ for $i = 1, 2$.
4:    Solve $\begin{bmatrix} T_i(\lambda, \mu) & u_i \\ v_i^H & 0 \end{bmatrix} \begin{bmatrix} z_i & w_i \\ \alpha_i & \beta_i \end{bmatrix} = - \begin{bmatrix} \frac{\partial T_i}{\partial \lambda}(\lambda, \mu) x_i & \frac{\partial T_i}{\partial \mu}(\lambda, \mu) x_i \\ 0 & 0 \end{bmatrix}$ for $i = 1, 2$.
5:    Solve
          $\begin{bmatrix} \alpha_1 & \beta_1 \\ \alpha_2 & \beta_2 \end{bmatrix} \begin{bmatrix} \Delta\lambda_k \\ \Delta\mu_k \end{bmatrix} = \begin{bmatrix} -\gamma_1 \\ -\gamma_2 \end{bmatrix}.$
6:    Update $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$ and $\mu_{k+1} = \mu_k + \Delta\mu_k$.
7: end for
```

5.6 Jacobi–Davidson method

All the above-mentioned methods require a good initial approximation to converge to a wanted eigenvalue. In addition, they compute one eigenpair only. When we lack such approximation we can try a Jacobi–Davidson type method. This is also a method of choice when matrices are large and sparse. The first Jacobi–Davidson method for a 2EP was a one-sided version for a right-definite 2EP [12]. It was followed by the two-sided version for a general 2EP [9], while the latest version [13] uses harmonic Ritz values and works well for the interior eigenvalues. If we look at the NEP, then the Jacobi–Davidson method was applied to a polynomial eigenvalue problem in [28], while a version for a general NEP is presented in [4].

In Algorithm 6 we give a Jacobi–Davidson type method for a N2EP. Most of the details are omitted as they can be found in, e.g., [11] and references therein.

Algorithm 6 Jacobi–Davidson method

```
1: Start with nonzero vectors $s_1, s_2$, and $U_{10} = U_{20} = V_{10} = V_{20} = [\,]$.
2: for $k = 0, 1, 2, \dots$ until convergence do
3:   Expand the search space: $(U_{i,k-1}, s_i) \to U_{ik}$ for $i = 1, 2$.
4:   Update the appropriate test space: $(V_{i,k-1}) \to V_{ik}$ for $i = 1, 2$.
5:   Extract the Ritz pair $((\sigma, \tau), u_1 \otimes u_2)$, where $u_i = U_{ik}c_i$ for $i = 1, 2$, from the projected N2EP
         $V_{1k}^H T_1(\sigma, \tau) U_{1k} c_1 = 0$
         $V_{2k}^H T_2(\sigma, \tau) U_{2k} c_2 = 0.$
6:   Compute the residual $r_i = T_i(\sigma, \tau)u_i$ for $i = 1, 2$.
7:   Stop if $(\|r_1\|^2 + \|r_2\|^2)^{1/2} \le \varepsilon$.
8:   Solve approximately the correction equation, e.g.,
         $(I - u_i u_i^*) T_i(\sigma, \tau)(I - u_i u_i^*) s_i = -r_i,$
     to get a new direction $s_i \perp u_i$ for $i = 1, 2$.
9: end for
```

Let us give some comments on the algorithm. In Steps 3 and 4 we use the repeated Gram–Schmidt orthogonalization so that the columns of U_{ik} and V_{ik} are orthonormal for $i = 1, 2$. The choice of an appropriate test space depends on whether we are using one-sided variant, two-sided variant, or harmonic Ritz values. In the basic one-sided variant we take $V_{ik} = U_{ik}$ and then Step 4 is redundant.

In Step 5 we need a numerical method that can be applied to a projected N2EP with small matrices to compute the Ritz pairs of interest. In case of a P2EP we can apply the linearization, compute all Ritz pairs, and then choose the most appropriate one. For some other problems that are truly nonlinear, for example (2.2), we can apply the methods from the previous subsections to obtain a Ritz pair. In this case it is useful to have a good approximation for the eigenvalue, otherwise the methods might not converge. In a typical scenario we are looking for an eigenvalue close to a given target and the matrices are so large and sparse that the iterative iteration is too expensive. Instead, we apply the Jacobi–Davidson method and use the iterative iteration to solve the small projected problems.

In Step 8 we solve the correction equation only approximately by applying a few steps of the GMRES method or another appropriate subspace method. For a better convergence we should use preconditioning and restart the method when subspaces become too large.

The Jacobi–Davidson method was applied to the P2EP in [11]. Using a selection criteria that prevents the algorithm from selecting the already converged Ritz values it is possible to compute more eigenvalues.

6 Numerical examples

Two numerical examples are given. They were obtained on 64-bit Windows version of Matlab R2012b running on Intel 8700 processor and 8 GB of RAM. In the first example we compare the convergence of methods on a Q2EP. In the second example we apply inverse iteration on a N2EP related to the determination of the critical curve of a DDE.

Example 6.1 We take the Q2EP of the form (2.1), where A_{ij} and B_{ij} are random $n \times n$ matrices generated in Matlab as

```
rand('state', 0); k = 2;
for r = 0:k
    for c=0:(k-r)
        A{r+1, c+1} = rand(n)+i*rand(n);
        B{r+1, c+1} = rand(n)+i*rand(n);
    end
end
```

For $n = 250$, 500, and 1000 we first compute one eigenpair of the Q2EP by the Jacobi–Davidson method from [11], which is basically Algorithm 6 adjusted to Q2EP, and then use a perturbed solution for an initial approximation. We tested Algorithms 1, 2, 3, 4, and 5, to which we refer from now on by more descriptive names InvIter, ResIter, SuccLP, NewtCF, and ImpDet, respectively. For numerical experiments with the Jacobi–Davidson method (Algorithm 6) for Q2EP, see [11].

For $n = 250$ we give in Table 1 for all algorithms the norms of the eigenvalue updates $(|\Delta\lambda_k|^2 + |\Delta\mu_k|^2)^{1/2}$ in individual iterations. In addition, for InvIter and ResIter, where eigenvectors are computed as well, we give the norms of the residuals $(\|T_1(\lambda_k, \mu_k)x_1^{(k)}\|^2 + \|T_2(\lambda_k, \mu_k)x_2^{(k)}\|^2)^{1/2}$. It is clearly seen that ResIter has linear convergence while the other four algorithms have quadratic convergence.

In Table 2 we give the number of iterations and computational times. We iterate InvIter and ResIter until the norm of the residual drops below 10^{-10}, while in SuccLP, NewtCF, and ImpDet the same bound is used for the norm of the eigenvalue update.

NewtCF is slower than InvIter and ResIter. We must remark that in the numerical experiments we replaced the Steps 4 and 5 by (5.9). In theory, Steps 4 and 5 should be faster as they require 25 % less flops than (5.9). In practice, it is difficult to implement this part efficiently as all computations are done at most in Level 1 BLAS. For instance, Algorithm 4 using an implementation of Steps 4 and 5 in C using MEX (Matlab implementation is even much slower) requires 168 s for $n = 1000$.

In SuccLP we use subspace iteration with Ritz projections from [19] to compute the eigenvalue closest to zero, which is to our believe the fastest available option at the moment. Based on the properties of the algorithm from [19] it is clear that one step of SuccLP is always more expensive than one step of InvIter. As they both have quadratic convergence, the cheaper inverse iteration is preferred. ResIter is also

Table 1 Comparison of convergence of Algorithms 1 (InvIter), 2 (ResIter), 3 (SuccLP), 4 (NewtCF), and 5 (ImpDet) on a random Q2EP with matrices of size $n = 250$

Iteration	Norm of the residual		Norm of the eigenvalue update				
	InvIter	ResIter	InvIter	ResIter	SuccLP	NewtCF	ImpDet
1	1.1×10^{-2}	1.1×10^{-2}	5.9×10^{-3}	5.9×10^{-3}	6.0×10^{-3}	6.4×10^{-3}	6.3×10^{-3}
2	1.4×10^{-5}	2.7×10^{-4}	3.3×10^{-4}	2.9×10^{-4}	2.6×10^{-5}	3.2×10^{-4}	2.7×10^{-4}
3	7.3×10^{-11}	1.6×10^{-5}	6.1×10^{-7}	1.1×10^{-5}	4.6×10^{-10}	1.4×10^{-6}	4.7×10^{-7}
4	5.0×10^{-14}	1.0×10^{-6}	5.9×10^{-12}	5.7×10^{-7}	3.4×10^{-14}	3.6×10^{-11}	1.2×10^{-12}
5		5.8×10^{-8}		2.0×10^{-8}		1.8×10^{-15}	1.8×10^{-15}
6		3.7×10^{-9}		2.2×10^{-9}			
7		2.5×10^{-10}		1.5×10^{-10}			
8		1.6×10^{-11}		8.1×10^{-11}			
9		1.0×10^{-12}		3.3×10^{-12}			
10		7.2×10^{-14}		3.0×10^{-14}			

Table 2 Number of iterations and running times for Algorithms 1 (InvIter), 2 (ResIter), 3 (SuccLP), 4 (NewtCF), and 5 (ImpDet) applied to a random Q2EP with matrices of size $n = 250, 500$, and 1000

n	Number of iterations					Running time in seconds				
	InvIter	ResIter	SuccLP	NewtCF	ImpDet	InvIter	ResIter	SuccLP	NewtCF	ImpDet
250	3	8	4	4	4	0.13	0.19	0.97	0.32	0.17
500	4	9	4	5	4	0.66	0.74	3.01	2.43	0.76
1000	4	12	4	5	5	3.52	3.83	16.29	14.82	4.36

very competitive and has an advantage that by using the fixed matrix $T_i(\lambda_0, \mu_0)$ we avoid potential difficulty of $T_i(\lambda_k, \mu_k)$ being singular when (λ_k, μ_k) is close to an eigenvalue.

In ImpDet we choose the vectors u_i and v_i as the result of one step of standard inverse iteration on $T_i(\lambda_0, \mu_0)$ and $T_i(\lambda_0, \mu_0)^H$, respectively, for $i = 1, 2$. The method is competitive to InvIter and ResIter. Its advantage is that the bordered matrix (5.12) is nonsingular at exact eigenvalue (λ_*, μ_*).

Example 6.2 We consider a DDE with two delays of the form

$$\dot{x}(t) = A_0 x(t) + A_1 x(t - h_1) + A_2 x(t - h_2).$$

The corresponding characteristic equation is

$$M(\lambda)z := (-\lambda I + A_0 + A_1 e^{-h_1\lambda} + A_2 e^{-h_2\lambda})z = 0, \tag{6.1}$$

where nonzero z is an eigenvector and λ is the corresponding eigenvalue. In the critical delay, where the stability of the DDE changes, λ is purely imaginary. We would like to find the critical curve, i.e., we are interested in the curve (h_1, h_2) for $h_1, h_2 \geq 0$, where the stability changes.

A usual approach is to assume that $h_2 = \alpha h_1$ for different values of α. Then, as α goes from 0 to ∞, we compute the points on the critical curve. When we introduce $\mu = e^{-h_1\lambda}$ in (6.1) and write the conjugate equation, where, since λ is purely imaginary, $\overline{\lambda} = -\lambda$ and $\overline{\mu} = \mu^{-1}$, we obtain (2.2). This equation can be solved for some integer values of α, for instance $\alpha = 0, 1, 2$, if we transform the problem into a polynomial eigenvalue problem or into a P2EP, see, e.g., [15]. We could also take for α a rational number with small numerator and denominator and then solve the problem as a P2EP.

This is how we get some points on the critical curve that we can use as initial approximations for other values of α and thus follow the critical curve. For the missing values of α we can solve (2.2) using any of the proposed local methods.

For the numerical example we consider the DDE with two delays from [14]:

$$u_t = u_{xx} + a_0(x)u + a_1(x)u(x, t - h_1) + a_2(x)u(x, t - h_2), \; u(0, t) = u(\pi, t) = 0, \tag{6.2}$$

where the coefficients are $a_0(x) = 2 + 0.3\sin(x)$, $a_1(x) = -2 + 0.2x(1 - e^{x-\pi})$, and $a_2(x) = -2 - 0.3x(\pi - x)$. We discretize (6.2) by the finite differences with matrices of size 100×100 and thus obtain A_0, A_1, and A_2 in (2.2). For the initial point we assume $h_1 = h_2$ and compute the critical delay by the Q2EP formulation and the Jacobi–Davidson method as explained in [11].

It is important that (2.2) can be formulated as a Q2EP for $h_1 = h_2$. This enables us to apply the linearization in Step 5 of Algorithm 6 and compute all Ritz values of the projected problem. Only one eigenvalue of the related Q2EP corresponds to the critical delay and a special selection criteria (for details, see [11]) guides the Jacobi–Davidson method to this particular eigenvalue. This gives $\lambda = 4.2399286i$ and $h_1 = h_2 = 0.30266688$ for the initial value $\alpha_0 = 1$.

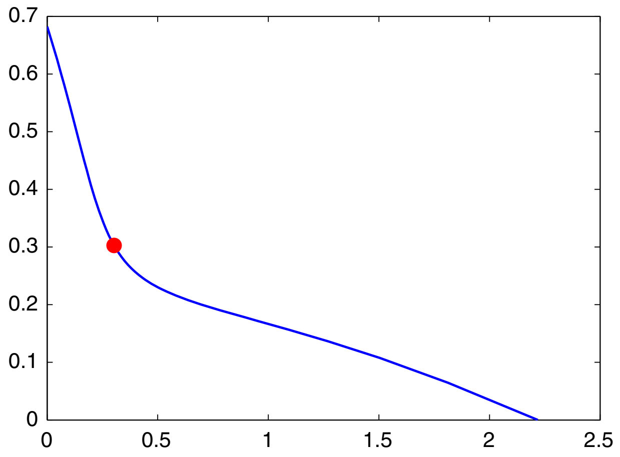

Fig. 1 The critical curve (h_1, h_2) for the DDE (6.2). We start in $h_1 = h_2 = 0.30266688$ and follow the curve with the inverse iteration method

Let us remark that for $\alpha \neq 1$ such that (2.2) cannot be formulated as a P2EP, it is difficult to compute the critical delay without a good approximation. Namely, if we apply the Jacobi–Davidson method, then in Step 5 of Algorithm 6 we get only one Ritz value and there is no guarantee that the computed eigenvalue corresponds to the critical delay.

In the second phase we set

$$\alpha_k = \tan\left(\frac{(m-k)\pi}{4m}\right)$$

for $k = 1, \ldots, m$. For each α_k we take the solution from step $k-1$ as an initial value for InvIter. In this way we get points where $h_2 < h_1$. For the other part of the critical curve we exchange the roles of h_1 and h_2 and repeat the procedure starting again from $\alpha_0 = 1$. The results for $m = 20$ are presented in Fig. 1, where the dot presents the initial delay at $h_1 = h_2$. The computation of all 41 points including the initial one takes just 1.5 s, which is much faster than reported in [14]. In addition, the results are more accurate as we use much finer mesh.

For large and sparse matrices A_0, A_1, and A_2 it would be more efficient to apply the Jacobi–Davidson method in the second phase instead of InvIter. But in our particular example, where the matrices A_0, A_1 and A_2 are tridiagonal, InvIter is very efficient and there is no need for the Jacobi–Davidson method.

7 Conclusions

We presented several numerical methods for nonlinear two-parameter eigenvalue problems. The most competitive local methods are the inverse iteration with quadratic convergence and the less expensive residual iteration with linear convergence. If we know how to solve the smaller projected problem, then we can use the Jacobi–Davidson method as a global method. As a practical application we presented the computation of critical curves of delay-differential equations with multiple delays.

Acknowledgments The author is grateful to the referee for the careful reading, the excellent remarks, and in particular for pointing out the implicit determinant method in [29]. The research was performed in part while the author was visiting the CASA group at the TU Eindhoven. The author wishes to thank the NWO for the visitor grant and the CASA group for the hospitality.

References

1. Akinola, R.O., Freitag, M.A., Spence, A.: The computation of Jordan blocks in parameter-dependent matrices. IMA J. Numer. Anal. **34**, 955–976 (2014)
2. Andrew, A.L., Eric Chu, K.-W., Lancaster, P.: Derivatives of eigenvalues and eigenvectors of matrix functions. SIAM J. Matrix Anal. Appl. **14**, 903–926 (1993)
3. Atkinson, F.V.: Multiparameter Eigenvalue Problems. Academic Press, New York (1972)
4. Betcke, T., Voss, H.: A Jacobi–Davidson-type projection method for nonlinear eigenvalue problems. Future Gener. Comput. Syst. **20**, 363–372 (2004)
5. Bohte, Z.: Calculation of the derivative of the determinant. Obzornik Mat. Fiz. **28**, 33–50 (1981)
6. Bohte, Z.: Numerical solution of some twoparameter eigenvalue problems. Anton Kuhelj Memorial Volume, Ljubljana, pp. 17–28 (1982)
7. Cox, D.A., Little, J.B., O'Shea, D.: Using Algebraic Geometry, 2nd edn. Springer, New York (2005)
8. Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. The Johns Hopkins University Press, Baltimore (1996)
9. Hochstenbach, M.E., Košir, T., Plestenjak, B.: A Jacobi–Davidson type method for the nonsingular two-parameter eigenvalue problem. SIAM J. Matrix Anal. Appl. **26**, 477–497 (2005)
10. Hochstenbach, M.E., Muhič, A., Plestenjak, B.: On linearizations of the quadratic two-parameter eigenvalue problems. Linear Algebr. Appl. **436**, 2725–2743 (2012)
11. Hochstenbach, M.E., Muhič, A., Plestenjak, B.: Jacobi–Davidson methods for polynomial two-parameter eigenvalue problems . J. Comput. Appl. Math. **288**, 251–263 (2015)
12. Hochstenbach, M.E., Plestenjak, B.: A Jacobi–Davidson type method for a right definite two-parameter eigenvalue problem. SIAM J. Matrix Anal. Appl. **24**, 392–410 (2002)
13. Hochstenbach, M.E., Plestenjak, B.: Harmonic Rayleigh–Ritz extraction for the multiparameter eigenvalue problem. Electron. Trans. Numer. Anal. **29**, 81–96 (2008)
14. Jarlebring, E.: The spectrum of delay-differential equations: numerical methods, stability and perturbation. PhD thesis, TU Braunschweig (2008)
15. Jarlebring, E., Hochstenbach, M.E.: Polynomial two-parameter eigenvalue problems and matrix pencil methods for stability of delay-differential equations. Linear Algebr. Appl. **431**, 369–380 (2009)
16. Ji, X.R., Jiang, H., Lee, B.H.K: A generalized Rayleigh quotient iteration for coupled eigenvalue problems. Technical Report 92-338, Department of Computing and Information Science, Queen's University (1992)
17. Kelley, C.T.: Iterative Methods for Linear and Nonlinear Equations. SIAM, Philadelphia (1995)
18. Lancaster, P.: Lambda-Matrices and Vibrating Systems. Pergamon Press, Oxford (1966)
19. Meerbergen, K., Plestenjak, B.: A Sylvester-Arnoldi type method for the generalized eigenvalue problem with two-by-two operator determinants. Report TW 653, Department of Computer Science, KU Leuven (2014)
20. Mehrmann, V., Voss, H.: Nonlinear eigenvalue problems: a challenge for modern eigenvalue methods. GAMM Mitt. Ges. Angew. Math. Mech. **27**, 121–152 (2004)
21. Muhič, A., Plestenjak, B.: On the singular two-parameter eigenvalue problem. Electron. J. Linear Algebr. **18**, 420–437 (2009)
22. Muhič, A., Plestenjak, B.: On the quadratic two-parameter eigenvalue problem and its linearization. Linear Algebr. Appl. **432**, 2529–2542 (2010)
23. Neumaier, A.: Residual inverse iteration for the nonlinear eigenvalue problem. SIAM J. Numer. Anal. **22**, 914–923 (1985)
24. Plestenjak, B.: A continuation method for a right definite two-parameter eigenvalue problem. SIAM J. Matrix Anal. Appl. **21**, 1163–1184 (2000)

25. Plestenjak, B.: Numerical methods for the tridiagonal hyperbolic quadratic eigenvalue problem. SIAM J. Matrix Anal. Appl. **28**, 1157–1172 (2006)
26. Ruhe, A.: Algorithms for the nonlinear eigenvalue problem. SIAM J. Numer. Anal. **10**, 674–689 (1973)
27. Schreiber, K.: Nonlinear eigenvalue problems: Newton-type methods and nonlinear Rayleigh functionals. PhD thesis, Department of Mathematics, TU Berlin (2008)
28. Sleijpen, G.L.G., Booten, A.G.L., Fokkema, D.R., van der Vorst, H.A.: Jacobi–Davidson type methods for generalized eigenproblems and polynomial eigenproblems. BIT **36**, 595–633 (1996)
29. Spence, A., Poulton, C.: Photonic band structure calculations using nonlinear eigenvalue techniques. J. Comput. Phys. **204**, 65–81 (2005)