SIAM J. Sci. Comput.
Vol. 41, No. 2, pp. A989–A1012

BROYDEN'S METHOD FOR NONLINEAR EIGENPROBLEMS[*]

ELIAS JARLEBRING[†]

Abstract. Broyden's method is a general method commonly used for nonlinear systems of equations when very little information is available about the problem. We develop an approach based on Broyden's method for the structure appearing in nonlinear eigenvalue problems. Our approach is designed for problems where the evaluation of a matrix vector product is computationally expensive, essentially as expensive as solving the corresponding linear system of equations. We show how the structure of the Jacobian matrix can be incorporated into the algorithm to improve convergence. The algorithm exhibits local superlinear convergence for simple eigenvalues, and we characterize the convergence. We show how deflation can be integrated and combined such that the method can be used to compute several eigenvalues. A specific problem in machine tool milling, coupled with a PDE, is used to illustrate the approach. The simulations were carried out using the Julia programming language, and the simulation software is provided publicly for reproducibility.

Key words. nonlinear eigenproblems, Broyden methods, Newton methods, time-delay systems, eigenvalue problems

AMS subject classifications. 65Fxx, 65F15, 65H17, 65Nxx

DOI. 10.1137/18M1173150

1. Introduction.

We consider the nonlinear eigenvalue problem (NEP) defined by

$$M(\lambda)v = 0, \tag{1.1}$$

where $M : \mathbb{C} \to \mathbb{C}^{n\times n}$ is an analytic function of λ. This problem can equivalently be written as a system of nonlinear equations

$$F\left(\begin{bmatrix} v \\ \lambda \end{bmatrix}\right) = 0, \tag{1.2}$$

where

$$F\left(\begin{bmatrix} v \\ \lambda \end{bmatrix}\right) := \begin{bmatrix} M(\lambda)v \\ c^H v - 1 \end{bmatrix} \tag{1.3}$$

under the assumption that c is not orthogonal to the eigenvector. The normalization condition $c^H v = 1$ is selected such that F is analytic and therefore complex differentiable, which is not the case for the Euclidean normalization condition $\|v\|_2^2 = v^H v = 1$.

This class of NEPs has been studied for decades, as can be seen in summary references [37, 33, 47] and the benchmark collection [2]. Several standard approaches for NEPs of the type (1.1) are based on Newton's method. The Newton approach for NEPs was proposed in 1950 [44] and later developed further in [35, 37]. The residual inverse iteration [34] is an implicit Newton method [25] and forms the basis of a state-of-the-art algorithm for NEPs, the nonlinear Arnoldi method [46]. More recently,

[*] Submitted to the journal's Methods and Algorithms for Scientific Computing section February 27, 2018; accepted for publication (in revised form) January 23, 2019; published electronically April 3, 2019.
http://www.siam.org/journals/sisc/41-2/M117315.html

[†] Department of Mathematics, KTH Royal Institute of Technology, SeRC Swedish e-Science Research Center, Stockholm 10044, Sweden (eliasj@kth.se).

block variants of Newton's method have been developed [28]. There is a summary of many methods in [18] of which many are Newton methods or can be interpreted as flavors of Newton's method. The QR-approach for banded matrices in [14] is based on Kublanovskaya's approach [29], which is also a Newton method applied to the (n, n)-element of the R-matrix in the QR-factorization of $M(\lambda)$. Two-sided Newton approaches and Jacobi–Davidson approaches have been studied in [38]. Considerable convergence theory and specialization of Newton-type approaches can be found in the literature; e.g., for convergence theory see [41, 42, 43], and for related topics, such as inexact solves and preconditioning, see [40].

These Newton-approaches depend on explicit access to the matrix $M(\lambda)$, in ways which are not always easily available in applications. Most methods depend on direct access to $M(\lambda)$ and/or that the NEP can be expressed in an affine form as

$$M(\lambda) = M_1 f_1(\lambda) + \cdots + M_m f_m(\lambda), \tag{1.4}$$

where $f_1, \ldots, f_m$ are analytic functions. Note that an affine form always exists in theory since if $m = n^2$, we can select $M_1, \ldots, M_m$ to be the standard basis of $\mathbb{C}^{n\times n}$. In many approaches, the affine form is required to be short in the sense that $m \ll n$, since parts of the algorithms require $1, \ldots, m$ operations, and would be computationally very demanding if, e.g., $m = n^2$. Our approach does not require a short affine form but only that we have access to the matrix vector product $M(\mu)v$ (and at one evaluation of the matrix $M(\mu)$). In the example in section 7, there is no short affine form.

In addition, the availability of a short affine form usually implies that the projected NEP $V^T M(\lambda)Wz = 0$ can be solved in a computationally cheap way, since the projected matrices can be formed individually as $V^T M_1 W, \ldots, V^T M_m W$ before running an NEP-solver for dense problems. In this way the projected problem is of the same type as the original problem. In the example in section 7 we essentially only have access to a matrix vector product, which is computationally very expensive. An iterative solution method for the projected problem would, in every step need evaluation of the $V^T M(\lambda)Wz$, require one evaluation of a matrix vector product. The projected solve would therefore be computationally dominating and unsuitable for that problem. Our approach does not require a projected problem.

The approach presented here is based on Broyden's method for nonlinear systems of equations; see [7], and for more recent summaries see [10, 1]. Broyden's method is also based on Newton's method, but the Jacobian approximation is updated (typically with a rank-one matrix) in order to avoid the computation of the Jacobian matrix. An attractive feature of Broyden's method is that it requires only one function evaluation per iteration. In the context of NEPs, this leads to the property of not needing either an affine form or direct accurate access to the Jacobian matrix.

Similarly to many other structured iterative methods, application of a general purpose approach to a specific problem leads to structures which can be exploited in the algorithm. We show in section 3 that Broyden's method iterates for (1.2) have a particular structure, which allow us to improve the approach by exploiting the Jacobian structure. In section 4 we show how this can be integrated with a deflation technique. In this context we also show how restarting can be carried out in a natural way. A local convergence characterization is provided in section 5. We show how the convergence is related to the Jordan structure in the sense of [16, 19]. More precisely, we show how the convergence is given by the Jordan chains defined as the existence of solutions to the equation

$$\sum_{i=0}^{a} \frac{M^{(i)}(\lambda)}{i!} v_{a-i} = 0,$$

where v_0 is the eigenvector corresponding to $M(\lambda)$.

We present numerical results and simulations for several problems in sections 6.1 and 7 in order to illustrate the properties of the method and its competitiveness for a time-periodic time-delay system. The software in the simulations is provided publicly online and the algorithm is incorporated NEP-PACK [24], which is a software package for NEP-methods.

2. Background and basic algorithm. We briefly summarize the specific version of Broyden's method which will be the basis of our algorithm. We use a damped version of Broyden's method, as described, e.g., in [1, section 7]. The algorithm is derived from the Newton-like update equation

$$J_k \Delta x_k = -F(x_k), \tag{2.1}$$

where the next approximation is computed with a damped update equation

$$x_{k+1} = x_k + \gamma_k \Delta x_k. \tag{2.2}$$

The choice of the damping parameter γ_k will be tuned to our setting, essentially to increase the convergence basin by avoiding steps that are too large (as we further describe in Remark 3.3). The next matrix J_{k+1} will satisfy (what is commonly called) the secant condition

$$J_{k+1}(x_{k+1} - x_k) = F(x_{k+1}) - F(x_k), \tag{2.3}$$

where J_{k+1} is a rank-one modification of J_k. We will focus on updates of the form

$$J_{k+1} = J_k + \frac{1}{\|\Delta x_k\|^2} z_{k+1} \Delta x_k^H. \tag{2.4}$$

By combining (2.1), (2.3), and (2.4), it is clear that z_{k+1} can be directly computed from

$$z_{k+1} = \frac{1}{\gamma_k}(F(x_{k+1}) - (1-\gamma_k)F(x_k)). \tag{2.5}$$

In the literature on Broyden's method (without damping), e.g., the original work [7], the relation (2.4) with choice (2.5) is typically viewed as the minimizer of the update matrix $J_{k+1} - J_k$ with respect to the Frobenius norm and maintaining the secant condition (2.3).

Equations (2.1), (2.2), (2.5), and (2.4) form an explicit algorithm, where the state consists of a vector x_k and a matrix J_k, which takes the role of a Jacobian matrix. This algorithm is called Broyden's good method. (Our results can be modified for the bad Broyden's method. We focus on the good Broyden's method for simplicity.) An unfavorable aspect from a computational perspective is that the linear system in (2.1) needs to be solved in every step. There are several ways to avoid this. Instead of storing J_k, we can store its inverse

$$H_k = J_k^{-1}$$

and state the algorithm in terms of H_k instead of J_k. We see immediately that (2.1) becomes

$$\Delta x_k = -H_k F(x_k). \tag{2.6}$$

Similarly, the update equation (2.4) can be reformulated in terms of H_k. More precisely, by applying the Sherman–Morrison–Woodbury formula [17, section 2.1.4], we obtain

$$H_{k+1} = J_{k+1}^{-1} = \left(J_k + \frac{1}{\|\Delta x_k\|^2} z_{k+1}\Delta x_k^H\right)^{-1} \tag{2.7a}$$

$$= J_k^{-1} - \frac{J_k^{-1} z_{k+1}\Delta x_k^H J_k^{-1}}{\|\Delta x_k\|^2 + \Delta x_k^H J_k^{-1} z_{k+1}} \tag{2.7b}$$

$$= H_k - \frac{H_k z_{k+1}\Delta x_k^H H_k}{\|\Delta x_k\|^2 + \Delta x_k^H H_k z_{k+1}}. \tag{2.7c}$$

By using (2.5) and (2.3) we see that $H_k z_{k+1} = \frac{1}{\gamma_k}(H_k F(x_{k+1}) + (1-\gamma_k)\Delta x_k)$, and we get the following equivalent alternative relation for H_{k+1}:

$$H_{k+1} = H_k - \frac{(H_k F(x_{k+1}) + (1-\gamma_k)\Delta x_k)\Delta x_k^H H_k}{\Delta x_k^H (H_k F(x_{k+1}) + \Delta x_k)}. \tag{2.8}$$

Example 2.1. In order to illustrate the differences between the two versions of Broyden's method in terms of round-off error, in our setting we carry out simulations on the following small example. We consider the quadratic eigenvalue problem

$$M(\lambda) = A_0 + A_1\lambda + A_2\lambda^2,$$

where A_0, A_1, and A_2 were randomly generated. We carried out the simulation for both the H-version and the J-version in single precision as well as in sufficiently high precision such that the resulting iteration can be treated as exact. The residual norm history is given in Figure 1. We see that the H-version follows the exact error history (computed with high-precision arithmetic) in a worse way than the J-version. The algorithm presented in the next section follows the trajectory even better (T-variant). Although the differences between the methods are small in this example, they illustrate what can be seen in longer simulations, e.g., those in section 6.1.

3. Structure exploiting Broyden's method.

3.1. Structure of the iterates.

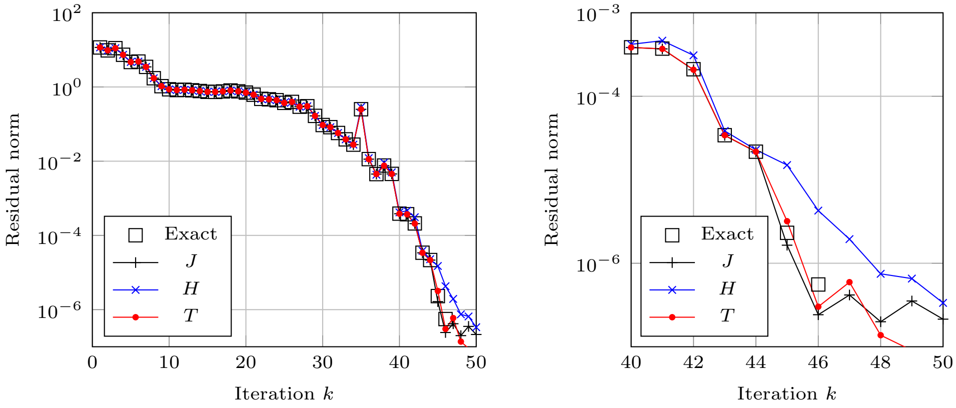
We now consider the following nonlinear systems of equations with a particular structure:

$$F\left(\lambda, \begin{bmatrix} v \\ u \end{bmatrix}\right) = \begin{bmatrix} M(\lambda) & U(\lambda) \\ C^H & 0 \end{bmatrix}\begin{bmatrix} v \\ u \end{bmatrix} - b \in \mathbb{C}^{n+p+1}, \tag{3.1}$$

where $C^H \in \mathbb{C}^{(p+1)\times n}$, $U(\lambda) \in \mathbb{C}^{n\times p}$, $v \in \mathbb{C}^n$, and $u \in \mathbb{C}^p$. We will also consistently partition b as $b^T = \begin{bmatrix} b_1^T & b_2^T \end{bmatrix}$.

This structure includes the nonlinear equation formulation in (1.3) as the special case where $p = 0$ and $b = e_{n+1}$. We take this more general approach in order to incorporate deflation in a natural way, as we will describe in section 4. The Jacobian of this problem can be derived explicitly as follows:

$$J\left(\lambda, \begin{bmatrix} v \\ u \end{bmatrix}\right) = \begin{bmatrix} M(\lambda) & U(\lambda) & M'(\lambda)v + U'(\lambda)u \\ C^H & 0 & 0 \end{bmatrix} \in \mathbb{C}^{(n+p+1)\times(n+p+1)}. \tag{3.2}$$



(a) Iteration history.

(b) Zoomed iteration history.

FIG. 1. *Round-off error illustration.*

We first note that the structures of the Jacobian and the iterates are preserved in Broyden's method when we denote

$$x_k = \begin{bmatrix} v_k \\ u_k \\ \lambda_k \end{bmatrix}. \tag{3.3}$$

More precisely, if we initialize the Jacobian in Broyden's method with the structure

$$J_1 = \begin{bmatrix} M_1 & W_1 \\ C^H & 0 \end{bmatrix} \in \mathbb{C}^{(n+p+1)\times(n+p+1)}, \tag{3.4}$$

where $W_1 = \begin{bmatrix} U_1 & f_1 \end{bmatrix} \in \mathbb{C}^{n\times(p+1)}$, $M_1 \in \mathbb{C}^{n\times n}$, and $C^H \in \mathbb{C}^{(p+1)\times n}$, then this structure is preserved in the sense of the following theorem, where we have given names to the blocks of the Jacobian approximation J_k consistent with (3.4). Since normally J_k is an approximation of the Jacobian matrix (3.2), we expect

$$M_k \approx M(\lambda_k), \tag{3.5a}$$

$$W_k \approx \begin{bmatrix} U(\lambda_k) & M'(\lambda_k)v_k + U'(v_k)u_k \end{bmatrix}. \tag{3.5b}$$

THEOREM 3.1 (structured iterates of Broyden's method). *Let (v_1, u_1, λ_1) be such that*

$$C^H v_1 = b_2, \tag{3.6}$$

and let J_1 be set to (3.4)*. Suppose Broyden's method initiated with (v_1, u_1, λ_1) and J_1 applied to* (3.1) *does not break down, and let (v_k, u_k, λ_k) and J_k, for $k = 2, \dots,$ be the iterates. Then, the structures* (3.6) *and* (3.4) *are preserved for all k; i.e., for $k = 2, \dots,$ we have*

$$J_k = \begin{bmatrix} M_k & W_k \\ C^H & 0 \end{bmatrix} \tag{3.7}$$

and

$$C^H v_k = b_2. \tag{3.8}$$

Proof. The proof is by induction. We suppose (3.7) and (3.8) hold for a specific k and prove the corresponding equations for $k+1$. It is clear from (2.1) that Δx_k satisfies

$$\begin{bmatrix} M_k & W_k \\ C^H & 0 \end{bmatrix} \Delta x_k = \begin{bmatrix} M(\lambda_k)v_k + U(\lambda_k)u_k - b_1 \\ C^H v_k - b_2 \end{bmatrix} = \begin{bmatrix} M(\lambda_k)v_k + U(\lambda_k)u_k - b_1 \\ 0 \end{bmatrix},$$

whose last block row implies that $C^H \Delta v_k = 0$. Since $v_{k+1} = v_k + \gamma_k \Delta v_k$, we have

$$C^H v_{k+1} = C^H(v_k + \gamma_k \Delta v_k) = C^H v_k = b_2,$$

which shows (3.8) for $k+1$. Therefore, the vector z_{k+1} defined in (2.5) has the structure

$$\begin{aligned} z_{k+1} &= \frac{1}{\gamma_k}\left(\begin{bmatrix} M(\lambda_{k+1})v_{k+1} + U(\lambda_{k+1}) - b_1 \\ C^H v_{k+1} - b_2 \end{bmatrix} - (1-\gamma_k)\begin{bmatrix} M(\lambda_k)v_k + U(\lambda_k) - b_1 \\ C^H v_k - b_2 \end{bmatrix}\right) \\ &= \frac{1}{\gamma_k}\begin{bmatrix} M(\lambda_{k+1})v_{k+1} + U(\lambda_{k+1}) - b_1 - (1-\gamma_k)(M(\lambda_k)v_k + U(\lambda_k) - b_1) \\ 0 \end{bmatrix}. \end{aligned}$$

The matrix J_k is updated according to (2.4). The last block row of the update in (2.4) is zero, since the last block row of z_{k+1} is zero. Therefore, we can define some J_{k+1} and W_{k+1} such that (3.7) is satisfied for $k+1$. □

3.2. Structured Broyden. With the objective to improve Broyden's method for nonlinear systems of equations of the form (3.1), we now show how the structure proved in Theorem 3.1 can be implicitly preserved. The J-version is straightforward to modify to incorporate the structure by consideration of the blocks of (2.1) in $J_k \Delta x_k = -F(x_k)$ as follows. We multiply the first block row of (2.1) from the left with $C^H M_k^{-1}$, i.e.,

$$C^H M_k^{-1}\left(M_k \Delta v_k + W_k \begin{bmatrix} \Delta u_k \\ \Delta\lambda_k \end{bmatrix}\right) = -C^H M_k^{-1} r_k, \tag{3.9}$$

where the residual r_k is defined as

$$r_k = M(\lambda_k)v_k + U(\lambda_k)u_k - b_1. \tag{3.10}$$

By using that $C^H \Delta v_k = C^H(v_{k+1} - v_k)/\gamma_k = (b_1 - b_1)/\gamma_k = 0$ due Theorem 3.1, we conclude from (3.9) that the following linear system for Δu_k and $\Delta\lambda_k$ is satisfied:

$$-C^H M_k^{-1} r_k = (C^H M_k^{-1} W_k)\begin{bmatrix} \Delta u_k \\ \Delta\lambda_k \end{bmatrix}. \tag{3.11}$$

Subsequently, Δv_k can be found from the first block row of (2.1), i.e.,

$$\Delta v_k = -M_k^{-1}\left(W_k \begin{bmatrix} \Delta u_k \\ \Delta\lambda_k \end{bmatrix} + r_k\right). \tag{3.12}$$

Hence, the solution of the linear system in (2.1) can be replaced by first solving (3.11) and then computing (3.12). This procedure can be implemented with $p+2$ linear solves with the matrix M_k.

At first sight, nothing is gained since we need even more linear solves than the J-version. However, similarly to the H-version, we can now formulate the algorithm by representing an inverse. More precisely, instead of storing M_k, we store

$$T_k := M_k^{-1}.$$

The reasoning with exploitation of the Jacobian in the J-version can be translated as follows. Equation (3.11) can be replaced by computing

$$Z_k = T_k W_k, \tag{3.13}$$

which allows us to compute the corresponding linear system in $p+1$ unknowns,

$$\begin{bmatrix} \Delta u_k \\ \Delta \lambda_k \end{bmatrix} = -(C^H Z_k)^{-1}(C^H T_k r_k), \tag{3.14}$$

from which we can form

$$\Delta v_k = -Z_k \begin{bmatrix} \Delta u_k \\ \Delta \lambda_k \end{bmatrix} - T_k r_k. \tag{3.15}$$

For notational convenience we now set $\underline{I}^H := \begin{bmatrix} I & 0 \end{bmatrix} \in \mathbb{R}^{n \times (n+p+1)}$. After updating the iterates

$$v_{k+1} = v_k + \gamma_k \Delta v_k, \tag{3.16a}$$

$$u_{k+1} = u_k + \gamma_k \Delta u_k, \tag{3.16b}$$

$$\lambda_{k+1} = \lambda_k + \gamma_k \Delta \lambda_k, \tag{3.16c}$$

we compute a new residual corresponding to r_{k+1} using (3.10) and define $\tilde{z}_{k+1}$ as

$$\tilde{z}_{k+1} = \underline{I}^T z_{k+1} = \frac{1}{\gamma_k}(r_{k+1} - (1-\gamma_k) r_k). \tag{3.17}$$

By again applying the Sherman–Morrison–Woodbury formula, we see that we can directly update T_k as follows:

$$\begin{aligned} T_{k+1} = M_{k+1}^{-1} &= \left(M_k + \frac{1}{\|\Delta x_k\|^2} \underline{I}^T z_{k+1} \Delta x_k^H \underline{I} \right)^{-1} \\ &= T_k - \frac{1}{\|\Delta x_k\|^2 + \Delta x_k^H \underline{I} T_k \underline{I}^T z_{k+1}} T_k \underline{I}^T z_{k+1} \Delta x_k^H \underline{I} T_k, \end{aligned}$$

which can be further simplified to not contain x-dependence,

$$T_{k+1} = T_k + T_k \tilde{z}_{k+1} a_{k+1}^H, \tag{3.18}$$

where $a_{k+1}^H := -\Delta v_k^H T_k / (\|\Delta v_k\|^2 + \|\Delta u_k\|^2 + |\Delta \lambda_k|^2 + \Delta v_k^H T_k \tilde{z}_{k+1})$. We can subsequently update W_k with

$$W_{k+1} = W_k + \tilde{z}_{k+1} b_{k+1}^H, \tag{3.19}$$

where $b_{k+1}^H = \begin{bmatrix} \Delta u_k^H & (\Delta \lambda_k)^H \end{bmatrix} / (\|\Delta v\|^2 + \|\Delta u\|^2 + |\Delta \lambda|^2)$. Finally, as a consequence of the fact that T_k and W_k are updated with rank-one matrices, we can also compute Z_{k+1} by a rank-one update of Z_k rather than by using the definition (3.13). By combining (3.18) and (3.19) we find that

$$Z_{k+1} = T_{k+1} W_{k+1} = Z_k + T_k \tilde{z}_{k+1}(a_{k+1}^H W_k + (1 + a_{k+1}^H \tilde{z}_{k+1}) b_{k+1}^H). \tag{3.20}$$

We now note that the above equations form an algorithm, which does not contain explicitly J_k or x_k and implicitly preserves the Jacobian structure in Theorem 3.1. The algorithm is summarized in Algorithm 1. For further implementation details, such as how to update T_k, W_k, and Z_k by using only two vector operations, we refer the reader to the publicly available software, described in section 6.1. As a consequence of the derivation, we have the following equivalence.

Algorithm 1. Structured Broyden's method.

```
Input: Starting values:
    Vectors: $v_1 \in \mathbb{C}^n$, $u_1 \in \mathbb{C}^p$, $\lambda_1 \in \mathbb{C}$ approximating solution to (3.1)
    Matrices: $T_1 = M_1^{-1} \in \mathbb{C}^{n\times n}$ and $W_1 \in \mathbb{C}^{n\times(p+1)}$ approximating (3.5)
    Input must satisfy $C^H v_1 = b_2$, where $C^H \in \mathbb{C}^{(p+1)\times n}$
Output: $v_m$, $u_m$, $\lambda_m$, $T_m$, $W_m$
Compute $r_1$ according to (3.10)
Compute $Z_1$ according to (3.13)
while $k = 1, 2, \ldots$ until convergence do
    Compute $\Delta u_k$, $\Delta\lambda_k$ by solving the linear system (3.14) in $p+1$ variables
    Compute $\Delta v_k$ with (3.15)
    Select the damping parameter $\gamma_k$, e.g., as in Remark 3.3
    Update the iterates by computing $v_{k+1}$, $u_{k+1}$, and $\lambda_{k+1}$ with (3.16)
    Compute $r_{k+1}$ according to (3.10)
    Compute $\tilde{z}_{k+1}$ using $r_k$ and $r_{k+1}$ and (3.17)
    Compute $Z_{k+1}$ with (3.20)
    Compute $W_{k+1}$ with (3.19)
    Compute $T_{k+1}$ with (3.18)
end while
```

THEOREM 3.2 (equivalence Broyden's methods). *The J-version of Broyden's method applied to* (3.1)*, i.e., the iteration defined by* (2.1)*,* (2.2)*, and* (2.3)*, is equivalent to the structured Broyden's method, i.e., the iteration defined by* (3.14)*,* (3.15)*,* (3.16)*,* (3.18)*,* (3.19)*, and* (3.20)*. Moreover, the states of the algorithms are related by* (3.3) *and*

$$J_k = \begin{bmatrix} T_k^{-1} & W_k \\ C^H & 0 \end{bmatrix}. \tag{3.21}$$

Remark 3.3 (selection of damping). The damping parameter is used to prevent the algorithm from taking too big steps in a pre-asymptotic phase, which can otherwise lead to divergence or convergence to an (undesired) solution far away. In practice, we observed that the λ-approximation in the beginning of the iteration often generated new approximations far away from the true solution. Therefore, we capped the step by selecting

$$\gamma_k = \min(1, t/|\Delta x_k|), \tag{3.22}$$

where t is a threshold parameter.

This implies that $\|x_{k+1} - x_k\| < t$ and, in particular, that $|\lambda_{k+1} - \lambda_k| \leq t$. This choice was determined based on numerical simulations. Another option would be the Armijo-steplength, as used, e.g., in the context of Newton's method for NEPs in [28]. In contrast to (3.22), the standard implementation of the Armijo-steplength involves function evaluations and is not competitive in our situation. We note that there is very little general conclusive theoretical analysis concerning how the damping parameter is best chosen in a Broyden setting, as, e.g., pointed out in [1].

4. Deflation.

4.1. A deflated NEP. The structured Broyden's method can be directly applied to (1.3) to compute an eigenpair of (1.1), as was illustrated in Example 2.1. In order to provide the possibility of computing several eigenvalues in a robust way, here we develop a deflation technique, which can be integrated with the structured Broyden's method. Our reasoning is inspired by the work on invariant pairs for NEPs in [28] and on deflation [11]. These works, in turn, are inspired by ideas for quadratic eigenvalue problems [31, 32, 3].

The essential conclusion of our reasoning provided below is that we can define an augmented NEP as

$$G(\lambda) = \begin{bmatrix} M(\lambda) & U(\lambda) \\ X^H & 0 \end{bmatrix}, \tag{4.1}$$

whose eigenvalues are essentially the same as the original NEP except for some eigenvalues which are removed. The matrices X^H and $U(\lambda)$ are defined in terms of invariant pairs, as we formalize in the discussion and Theorem 4.2 below. Note that if we add an orthogonalization constraint to $G(\lambda)w = 0$ as in (1.2) with a particular vector $c^H = [c_1^H \;\; 0]$, we obtain a nonlinear system with the structure of the previous section, i.e., (3.1). Our construction is based on applying Algorithm 1 to this problem.

For the derivation of this approach we need the concepts of invariant pairs, orthogonalization conditions, and augmented invariant pairs, which we briefly summarize. See [28, 11, 12] for a detailed characterization. Without loss of generality, let M in (1.1) be decomposed as a sum of products of matrices and functions as in (1.4). This decomposition always exists, although in computation it does not always lead to efficient algorithms if m is large. We will only use this decomposition for theoretical purposes and not in the final algorithm. An invariant pair of (1.1) is defined as a pair $(X, S) \in \mathbb{C}^{n\times p} \times \mathbb{C}^{p\times p}$ which satisfies

$$0 = M_1 X f_1(S) + \cdots + M_m X f_m(S),$$

where $f_i(S) \in \mathbb{C}^{m\times m}$ are matrix functions of f_i, $i = 1, \ldots, m$. By computing a Schur decomposition of S, it is possible to show that the eigenvalues of S are eigenvalues of (1.1). For standard eigenvalue problems, we usually require that the columns of X (which form a basis of an invariant subspace) be linearly independent. This is done in order to prevent the same eigenspace from appearing several times in the invariant pair. In practice (still within the setting of linear eigenvalue problems) this is usually achieved by requiring that the columns of X be orthonormal. The concept of minimality formalizes this reasoning. The minimality concept is slightly different in the nonlinear case, due to the fact that several eigenvalues can have the same eigenvector (or, correspondingly, for invariant subspaces). The generalization is not expressed in terms of the column span X but instead in terms of the column span of

$$\begin{bmatrix} X \\ \vdots \\ XS^{\ell-1} \end{bmatrix}. \tag{4.2}$$

If there exists $\ell \in \mathbb{N}$ such that (4.2) has full column rank, then the pair is called minimal, and the smallest ℓ such that (4.2) has full column rank is called the minimality index of the pair (X, S). As pointed out in [12], for minimal invariant pairs, $\ell = 1$ is generic.

The concept of invariant pairs was used in a natural way to construct a deflation technique for a (simplified) Newton method and a Jacobi–Davidson method in [12, 11]. The main idea is to compute invariant pairs one column at a time. Given an invariant pair (X, S), vectors v, u, and λ are computed such that the extended pair

$$(\hat{X}, \hat{S}) = \left(\begin{bmatrix} X & v \end{bmatrix}, \begin{bmatrix} S & u \\ & \lambda \end{bmatrix} \right) \tag{4.3}$$

is also an invariant pair. In [11, Lemma 6.1.3] the minimality is guaranteed by imposing orthogonality to the columns of (4.2),

$$\begin{bmatrix} X \\ \vdots \\ XS^{\ell-1} \end{bmatrix}^H \begin{bmatrix} \hat{X} \\ \vdots \\ \hat{X}\hat{S}^{\ell-1} \end{bmatrix} e_{p+1} = 0. \tag{4.4}$$

In this way, we avoid *reconvergence*; i.e., if an eigenvalue is contained in S, the algorithm will not find this eigenvalue again unless it has multiplicity greater than one.

The condition that the extended pair (4.3) is invariant is equivalent to a more explicit condition, shown in the following lemma.

LEMMA 4.1 (Lemma 6.1.1 of [11]). *Let (X, S) be an invariant pair of the nonlinear eigenvalue problem* (1.1)*. Then, the extended pair* (4.3) *is an invariant pair if and only if*

$$M(\lambda)v + U(\lambda)u = 0, \tag{4.5}$$

where

$$U(\lambda) = \frac{1}{2\pi i} \oint_\Gamma M(\xi)(\xi I - S)^{-1}(\xi - \lambda)^{-1}\, d\xi. \tag{4.6}$$

If $\lambda \notin \lambda(S)$, we have additionally (as formalized in [11, Lemma 6.2.2])

$$U(\lambda) = M(\lambda)X(\lambda I - S)^{-1}. \tag{4.7}$$

In this work we will in practice extensively use (4.7) rather than the slightly more general definition (4.6). The assumption $\lambda \notin \lambda(S)$, which we require to use (4.7), is of theoretical character and is rarely violated in practice. However, the matrix $\lambda I - S$ can become close to singular if λ is almost an eigenvalue of S. The algorithm we derive does not seem to suffer from this in practice. We conjecture that it is for similar reasons the Rayleigh quotient iteration for standard eigenvalue problems does not suffer from the fact that one needs to solve a linear system with an almost singular matrix if a backward stable linear solver is used.

We now focus on the case $\ell = 1$; see Remark 4.3 for discussion of the general case. By combining (4.5) and (4.4) we reach the nonlinear eigenvalue problem corresponding to (4.1) where $\begin{bmatrix} v^T & u^T \end{bmatrix}^T$ is an eigenvector of the NEP $G(\lambda)$ given by (4.1).

This reasoning is formalized in the following theorem, which can be interpreted as a complement to [12, Theorem 3.6]; in our theorem we also stress that imposing orthogonality does not restrict the set of minimal invariant pairs. We state the theorem in terms of similarity transformations. The pair (X, S) is a minimal invariant pair if and only if $(XZ, Z^{-1}SZ)$ is a minimal invariant pair where $Z \in \mathbb{C}^{p\times p}$ is invertible [11, Lemma 3.2.3]. We say that (X, S) and $(XZ, Z^{-1}SZ)$ are *equivalent by similarity transformation*. Essentially, the theorem states that without loss of generality we can extend a minimal invariant pair in such a way that the new vector satisfies (4.8).

THEOREM 4.2 (index one extensions). *Suppose (X,S) is a minimal invariant pair with index one. The following sets are equivalent by similarity transformation:*

- *Index one minimal invariant pair extensions of (X,S) of the form (4.3).*
- *Index one minimal invariant pair extensions of (X,S) of the form (4.3), where the extensions satisfy*

$$\begin{bmatrix} M(\lambda) & U(\lambda) \\ X^H & 0 \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} = 0, \tag{4.8}$$

where $\|v\| + \|u\| \neq 0$.

Proof. Let $Z = R^{-1}P$ be the similarity transformation defined by the QR-factorization of $X = QR$ (where $Q \in \mathbb{C}^{n\times p}$ and where $R \in \mathbb{C}^{p\times p}$ is invertible since the columns of X are linearly independent) and by the Schur factorization $RSR^{-1} = PRP^H$. From this transformation we see that (X,S) is equivalent by similarity transformation to $(XZ, Z^{-1}SZ)$, where XZ is orthogonal and $Z^{-1}SZ$ is upper triangular. By a change of variable, the condition (4.8) is unmodified by the transformation. Hence, without loss of generality we can assume that X is orthogonal and S is upper triangular.

Suppose $(\hat{X}, \hat{S})$ is an augmented minimal invariant pair (with extensions that do not necessarily satisfy (4.8)). With the similarity transformation

$$Z = \begin{bmatrix} I & -X^H v \\ 0 & 1 \end{bmatrix} \in \mathbb{C}^{(p+1)\times(p+1)},$$

by using Lemma 4.1 we can verify that (4.8) is satisfied by selecting vectors corresponding to (v,u,λ) as $(\hat{X}Ze_{p+1}, [I_p \quad 0]Z^{-1}\hat{S}Ze_{p+1}, \lambda)$. The converse holds due to the fact that a solution satisfying (4.8) forms a vector v which is orthogonal to X and nonzero; otherwise, the extension would be nonminimal. □

Remark 4.3 (minimality index greater than one). The generalization of the above reasoning to a higher minimality index can be seen as follows. The orthogonality condition (4.4) with $\ell = 2$ implies that v, u, λ must satisfy

$$(X^H + \lambda S^H X^H)v + S^H X^H X u = 0.$$

Unlike the case $\ell = 1$, this expression depends on both u and λ. The NEP analogous to (4.8) becomes

$$\begin{bmatrix} M(\lambda) & U(\lambda) \\ X^H + \lambda S^H X^H & S^H X^H X \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} = 0.$$

Unfortunately, when we include a normalization condition as in (1.2), this problem does not lead to a nonlinear equation of the form (3.1) which we need for the structured Broyden's method. It includes more blocks and more λ-dependence,

$$0 = \begin{bmatrix} M(\lambda) & U(\lambda) \\ C_1^H(\lambda) & C_2^H \\ c_1^H & 0 \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} - b \in \mathbb{C}^{n+p+1}, \tag{4.9}$$

where $C_1^H(\lambda) = X^H + \lambda S^H X^H$ and $C_2^H = S^H X^H X$. This prevents us from using structured Broyden in the same way. However, we can apply Broyden's method (without exploiting the same amount of structure) as illustrated in the simulations in section 6.1. More precisely, Algorithm 1 can be applied to the problem (4.9) since the required structure (3.1) is satisfied (for the matrices denoted with a tilde) if we change variables by

$$\begin{bmatrix} M(\lambda) & U(\lambda) \\ C_1^H(\lambda) & C_2^H \\ c_1^H & 0 \end{bmatrix} = \begin{bmatrix} \tilde{M}(\lambda) & \tilde{u}(\lambda) \\ c_1^H & 0 \end{bmatrix},$$

where $\tilde{M}(\lambda) \in \mathbb{C}^{n+p}$ and $\tilde{u}(\lambda) \in \mathbb{C}^{n+p}$. In this sense, our algorithm presented in the next section can in principle be constructed with a higher minimality index, but we only impose the structure on one row rather than $p+1$ rows. In this paper we develop an efficient algorithm for $\ell = 1$ and propose to use the slower variant without structure exploitation for problems where eigenpairs share eigenvectors. For most problems stemming from PDEs, $\ell = 1$ is generic.

4.2. Combining structured Broyden's with deflation. The previous section showed that, given an index one invariant pair, we can compute an extension of that invariant pair by solving the NEP (4.1). Since (4.1) combined with the normalization condition with $c^H = [c_1^H \;\; 0]$ leads to a nonlinear system of equations of the structure (3.1), we can apply Algorithm 1.

This structured extension of the invariant pair can be further combined with the algorithm. Algorithm 2 shows this combination, including handling of invariant pairs and starting values. We now provide further details and justification of the algorithm and show how restarting can be incorporated.

Recall that our method is mainly intended for problems where the matrix vector product $M(\lambda)z$ is computationally expensive. At step 8 of Algorithm 1 we need to compute the residual (3.10), which in our setting contains terms $M(\lambda)v$ and $U(\lambda)u$; each involves one matrix vector product with $M(\lambda)$. When we use Algorithm 1, we can combine M with the formula for U in (4.7) and compute the residual (3.10) directly by using only one matrix vector product. This is due to the identity

$$r_{k+1} = M(\lambda_{k+1})v_{k+1} + U(\lambda_{k+1})u_{k+1} = M(\lambda_{k+1})(v_{k+1} + X(\lambda_{k+1}I - S)^{-1}u_{k+1}).$$

Our algorithm requires starting values for each extension of the invariant pair. Although starting values are usually tuned to the applications—and this is also a possibility in our case—here we propose a quite general application-independent procedure to select starting values. We base the starting values on previously computed information, which can be viewed as a restarting procedure. Starting values are required for $M_1 = T_1^{-1}$, W_1, v_1, u_1, and λ_1. If we are interested in eigenvalues close to a target σ, we propose (step 1) to use $M_1 \approx M(\sigma)$ (or $M_1 = M(\sigma)$ if it can be computed cheaply) and set $\lambda_1 = \sigma$.

The eigenvector approximation (consisting of $v_1 \in \mathbb{C}^n$ and $u_1 \in \mathbb{C}^p$) is computed in step 6 following an approximation of one step of the method called safeguarded iteration [33, Algorithm 4]. Eigenvector approximations in safeguarded iteration are extracted by selecting the eigenvector corresponding to a small eigenvalue of the matrix $M(\lambda)$. We select v_1 and u_1 in this way but applied to the extended deflated NEP (4.1) by replacing the blocks of the matrix with approximations $M(\lambda_1) \approx M_1$ and $U(\lambda_1) \approx U_1$, where U_1 is computed directly from (4.7) by using p matrix vector products.

We see by comparing (3.21) and (3.2) that W_1 should be an approximation of $[U(\lambda_0) \;\; M'(\lambda_0)v_0 + U'(\lambda_0)u_0]$. The approximation of $U(\lambda_0)$ is chosen as the already computed U_1. The formula for $U(\lambda)$ in (4.7) gives us directly

$$U'(\lambda) = -M(\lambda)X(\lambda I - S)^{-2} + M'(\lambda)X(\lambda I - S)^{-1}. \tag{4.10}$$

Algorithm 2. Deflated Broyden's method.

```
Input: Target $\sigma$ and normalization vector $c \in \mathbb{C}^n \backslash \{0\}$
Output: A standard minimal invariant pair $(X, S) \in \mathbb{C}^{n\times p} \times \mathbb{C}^{p\times p}$ of (1.1)
Compute $M_1 \approx M(\sigma)$ and $T_1 \approx M(\sigma)^{-1}$
Set $X =$ empty matrix and $S =$ empty matrix
Set $k = 1$
while $k < p$ do
    Compute $U_1 \approx U(\sigma) \in \mathbb{C}^{n\times(k-1)}$ where $U$ is given by (4.7)
    Compute the smallest (in modulus) eigenvalue of the matrix
        $\begin{bmatrix} M_1 & U_1 \\ X^H & 0 \end{bmatrix} \in \mathbb{C}^{(n+k-1)\times(n+k-1)}$
    and let $\begin{bmatrix} v_1^T & u_1^T \end{bmatrix}^T$ be the corresponding eigenvector normalized such that $c^H v_1 = 1$
    Impose orthogonalization (3.6) on $v_1$ by updating $v_1$ and $u_1$
    Compute $f_1$ as approximation of (4.11), e.g., with finite difference for $M'(\sigma)v_1$
    Run structured Broyden for NEPs (Algorithm 1) with $C = \begin{bmatrix} X & c \end{bmatrix}$, starting value $(\sigma, v_1, u_1)$, and Jacobian approximation $(T_1, W_1) = (T_1, [U_1 \;\; f_1])$. Save output in $(\lambda, v, u)$ and $(T_N, [U_N \;\; f_N])$
    Expand invariant pair according to (4.3)
    $k = k + 1$
    If NEP has conjugate pair symmetry, expand also with conjugate eigenpair
end while
```

In order to compute a starting value of the last column of W_1, we use the fact that the chain-rule for differentiation applied to $U(\lambda)$ implies

$$\begin{aligned} M'(\lambda_1)v_1 + U'(\lambda_1)u_1 &= M'(\lambda_1)(v_1 + X(\lambda_1 I - S)^{-1}u_1) - M(\lambda_1)X(\lambda_1 I - S)^{-2}u_1 \\ &= M'(\lambda_1)(v_1 + X(\lambda_1 I - S)^{-1}u_1) - U(\lambda_1)(\lambda_1 I - S)^{-1}u_1. \end{aligned} \tag{4.11}$$

Unless the matrix vector action of the derivative M is explicitly available, the first term can be approximated by central finite difference, $M'(\lambda_1)z \approx \frac{1}{2h}(M(\lambda_1 + h)z - M(\lambda_1 - h)z)$, where $z = (v_1 + X(\lambda_1 I - S)^{-1}u_1)$. The second term can be approximated by using the already computed U_1, i.e., $U(\lambda_1)(\lambda_1 I - S)^{-1}u_1 \approx U_1(\lambda_1 I - S)^{-1}u_1$. This is done in step 8.

In step 12, we expand the invariant pair again if the problem exhibits symmetry. It is straightforward to show that if $\overline{M(\lambda)} = M(\overline{\lambda})$ for all λ, then an eigenpair (v, λ) implies that $(\overline{v}, \overline{\lambda})$ is an eigenpair which can be included in the invariant pair if $\lambda \notin \mathbb{R}$. The new complex conjugate pair $(\overline{v}, \overline{\lambda})$ is included by carrying out a Gram–Schmidt orthogonalization against X and storing the Gram–Schmidt coefficients in the new column of S.

5. Convergence theory.

Due to its equivalence with Broyden's method, the convergence of our approach can be characterized with more general results. It is well known that Broyden's method has asymptotic local superlinear convergence in general [15, 6]. However, the theory for superlinear convergence only holds under the assumption that the Jacobian at the solution is invertible. If this is not satisfied, we can invoke theory for Broyden's method of singular Jacobians [9], which implies (in general) linear convergence with a convergence factor equal to the reciprocal golden ratio. We characterize the singularity of the Jacobian of our particular problem.

The singularity of the Jacobian of many iterative methods for NEPs is directly given from the multiplicity (or Jordan chain structure) of the solution to the NEP; cf. [42, 27, 23, 43]. Our construction is equivalent to applying Broyden's method to the augmented NEP (4.1). Therefore, the Jacobian singularity of the augmented system (3.1) can be characterized with the multiplicity of the augmented NEP (4.1). The Jacobian singularity is characterized in terms of the eigenvalue multiplicity of the original NEP (1.1) as follows.

We first note that the Jacobian of (3.1) with M selected as the right-hand side of (4.1) is given by

$$J_* = \begin{bmatrix} M(\lambda) & U(\lambda) & M'(\lambda)v + U'(\lambda)u \\ X^H & & \\ c^H & & \end{bmatrix}. \tag{5.1}$$

It turns out that the condition that the vector c should not be orthogonal to the eigenvector generalizes to the condition that the matrix

$$\begin{bmatrix} X^H \\ c^H \end{bmatrix} \begin{bmatrix} X & v_1 \end{bmatrix} \tag{5.2}$$

needs to be nonsingular. This condition is needed in the following theorem, which gives a precise condition for the extended Jacobian (5.1) to be singular.

Theorem 5.1 (Jacobian singularity). *Suppose (X,S) is a minimal index one invariant pair of (1.1), where the columns of X are orthonormal and S is upper triangular. Suppose λ_1, v_1 is an eigenpair of (1.1) such that (5.2) is nonsingular, and suppose $M(\lambda_1)$ has null space of dimension one. Moreover, assume $\lambda_1 \notin \lambda(S)$ and $v_1 \notin \operatorname{Range}(X)$. Then*

$$\begin{bmatrix} v \\ u \\ \lambda \end{bmatrix} = \begin{bmatrix} (I - XX^H)v_1 \\ (\lambda_1 I - S)X^H v_1 \\ \lambda_1 \end{bmatrix} \tag{5.3}$$

is a solution to (4.8). Moreover, the Jacobian (5.1) corresponding to this solution is singular if and only if there exists a Jordan chain of length two (or more), i.e., there exists a vector v_2 such that

$$M(\lambda_1)v_2 + M'(\lambda_1)v_1 = 0. \tag{5.4}$$

Proof. We verify (5.3) directly by inserting it into (4.8) and using the formula for U in (4.7) and the fact that $M(\lambda_1)v_1 = 0$. In order to establish when the Jacobian is singular, we give necessary and sufficient conditions for the existence of the nontrivial singular vector $\begin{bmatrix} z_1^T & z_2^T & z_3^T \end{bmatrix}^T$ such that

$$\begin{bmatrix} M(\lambda) & U(\lambda) & M'(\lambda)v + U'(\lambda)u \\ X^H & & \\ c^H & & \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = 0. \tag{5.5}$$

By using the formula for $U'(\lambda_1)$ in (4.10) and the formula for the solution vector (5.3), the first block equation becomes

$$M(\lambda_1)(z_1 + X(\lambda I - S)^{-1}z_2 + X(\lambda I - S)^{-1}X^H v_1 z_3) + M'(\lambda_1)v_1 z_3 = 0. \tag{5.6}$$

We separate the rest of the proof into the following two cases:

- Suppose $z_3 = 0$, such that (5.6) reduces to $M(\lambda)(z_1 + X(\lambda I - S)^{-1}z_2) = 0$. Since $M(\lambda_1)$ has a one-dimensional null space, we must have $z_1 + X(\lambda I - S)^{-1}z_2 = \beta v_1$. By multiplying from left with $\begin{bmatrix} X & c \end{bmatrix}^H$, combining this with the last rows in (5.5), and using the assumption (5.2), we see that $\beta = 0$ and $z_2 = 0$. Consequently, $z_1 = 0$, such that z_1, z_2, z_3 are identically zero and do not form a nontrivial singular vector. Hence, any nontrivial singular vector must satisfy $z_3 \neq 0$.
- If we assume that $z_3 \neq 0$, without loss of generality we can assume that $z_3 = 1$. Clearly, (5.6) can only be zero if there exists a vector v_2 such that (5.4) is satisfied. Moreover, z_1 and z_2 must satisfy, for some value β,

$$z_1 + X(\lambda I - S)^{-1}z_2 + X(\lambda I - S)^{-1}X^H v_1 = v_2 + \beta v_1.$$

We multiply this relation by $\begin{bmatrix} X & c \end{bmatrix}^H$ and use the fact that $\begin{bmatrix} X & c \end{bmatrix}^H z_1 = 0$ (due to the second and third rows of (5.5)) to obtain

$$\left(\begin{bmatrix} X^H \\ c^H \end{bmatrix} \begin{bmatrix} X & v_1 \end{bmatrix} \right) \begin{bmatrix} (\lambda I - S)^{-1}z_2 \\ -\beta \end{bmatrix} = \begin{bmatrix} X^H \\ c^H \end{bmatrix} \left(-X(\lambda I - S)^{-1}X^H v_1 + v_2 \right).$$

This linear system has a solution since (5.2) is invertible by assumption, and it directly gives us a singular vector from a vector v_2 satisfying (5.4). The vector is nontrivial since $z_3 = 1$. □

Example 5.2 (double eigenvalue). The convergence properties for a singular Jacobian matrix can be observed in practice, and we illustrate this with the NEP presented in [26] (and also analyzed in [23, 27]). The problem is a delay eigenvalue problem $M(\lambda) = -\lambda I + A_0 + A_1 e^{-\tau\lambda}$ constructed such that it has a double non–semisimple eigenvalue at $\lambda_* = 3\pi i$. The error history of Algorithm 2 with $\sigma = 0$ and $M_0 = M(\sigma)$ is given in Figure 2. We clearly see that we have linear convergence the first time the iteration converges to λ_*. The second time the iteration converges to λ_*, we have superlinear convergence, consistent with the fact that the eigenvalue has multiplicity two (and not three). Once one of the double eigenvalues has been deflated, the Jacobian is regular, i.e., the convergence behaves as the convergence for simple eigenvalue.

We also observe (consistent with convergence theory [9]) that the linear convergence has a convergence factor equal to the reciprocal golden ratio, i.e., approximately 0.618.

Example 5.3 (multiplicity three). We also wish to illustrate that Theorem 5.1 gives information about not only double eigenvalues but also triple. The following NEP has an eigenvalue in $\lambda = -2$:

$$M(\lambda) = \begin{bmatrix} \lambda & -1 & 0 \\ 1/2 + \sqrt{-1-\lambda} & 7/8 & 1/8 \\ 0 & -\lambda - 1 & 1 \end{bmatrix}$$

with eigenvector $v_1 = \begin{bmatrix} 1 & -2 & 2 \end{bmatrix}^T$. Moreover, this eigenvalue has a Jordan chain of length three, which implies that (5.4) is satisfied for some vector v_2. We apply

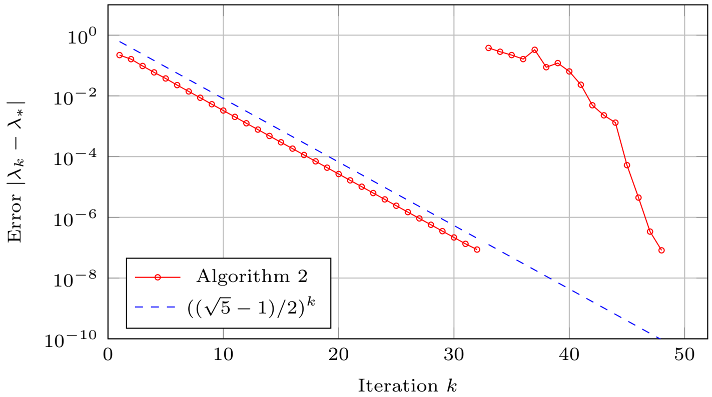


FIG. 2. *Convergence for the problem with a double eigenvalue at $\lambda_* = 3\pi i$ in Example* 5.2. *We clearly observe linear convergence the first time the iteration converges to λ_* and observe fast superlinear convergence the second time it converges to λ_*.*

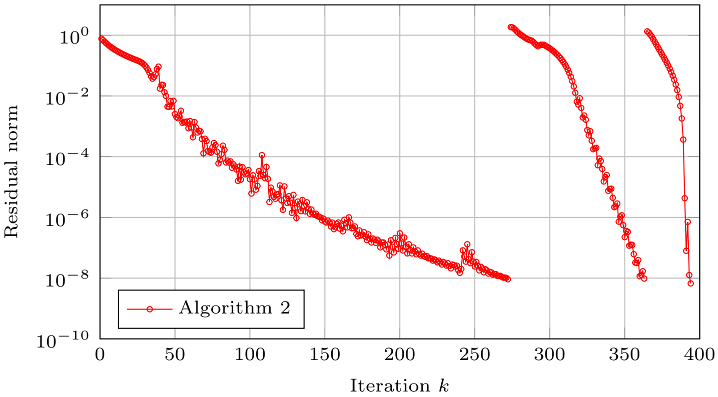


FIG. 3. *Convergence for the problem with an eigenvalue having multiplicity three in Example* 5.3.

Algorithm 2 with $\lambda_0 = 0$ to this problem, and the convergence is visualized in Figure 3. We clearly see that we do not have superlinear convergence for the first eigenvalue (as predicted by Theorem 5.1). In fact, the first convergence appears sublinear and very slow. The second reconvergence appears linear, and the third appears superlinear.

6. Comparison with standard NEP-solvers.

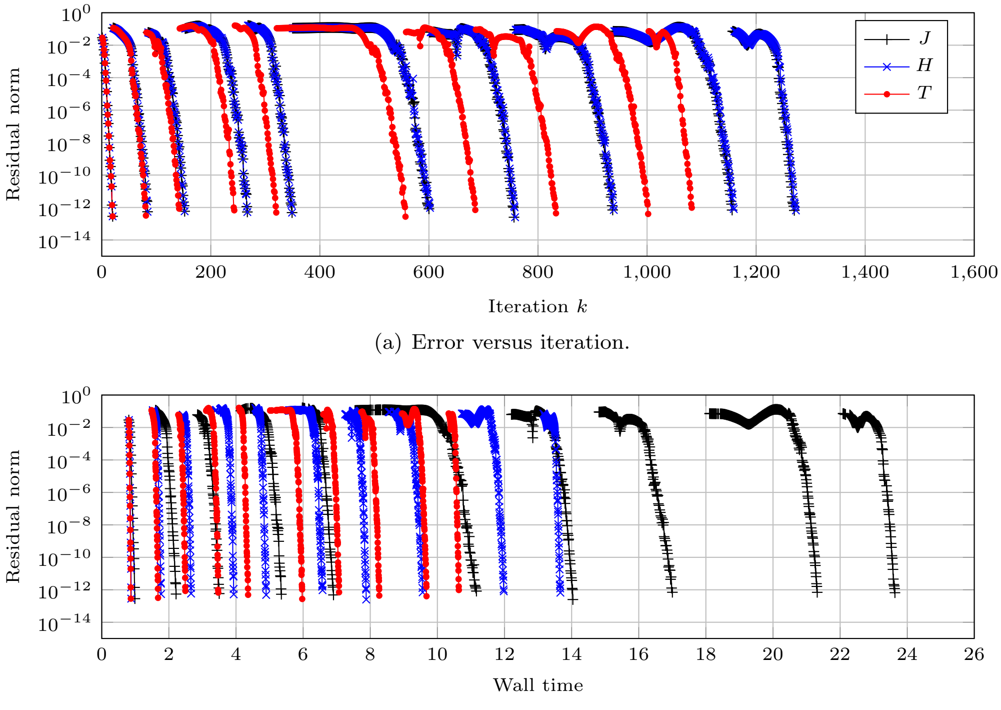
6.1. Simulations for quadratic time-delay system. We provide results of simulations for various NEPs. Our implementation is in the Julia programming language [4], version 0.6.2, using a computer with a 2.10 GHz Intel (quad-core) i7-4600U CPU and 16 GB RAM.[1]

In order to show the properties of our approach, we now apply the algorithm to the problem

$$M(\lambda) = -\lambda^2 I + A_0 + A_1 e^{-\lambda},$$

where the matrices are chosen as in [13] where $n = 400$ and the starting value is $\sigma = 0.1i$. The matrix $M(\sigma)$ has 3620 nonzero elements (2.2% of all elements). We

[1] The simulations are publicly available online at http://www.math.kth.se/~eliasj/src/broyden.



(a) Error versus iteration.

(b) Error versus wall time.

FIG. 4. *Comparison of structure exploiting and nonexploiting Broyden's method.*

applied the algorithm with termination criteria, residual norm less than 10^{-12}. The simulations of this section are intended to illustrate method properties, and we do not claim that this method is the best method for this type of problem.

We first illustrate the structure exploitation. In Figure 4 we see the convergence of the discussed versions of Broyden's method. Figures 4a and 4b show the same simulation but with different x-axis. The structure exploiting Broyden's method (Algorithm 1) converges (slightly) faster in terms of iterations, although they are equivalent in exact arithmetic. The structure exploiting Broyden's method is considerably faster than the other variants in terms of computation time.

The relevance of the damping is illustrated in Figure 6. No damping (or a very large t) typically leads to faster convergence, but robustness is lost as the solution can start diverging. The parameter t can be viewed as a trade-off parameter between robustness and convergence speed.

In order to illustrate the value of superlinear convergence, we compare the algorithm residual inverse iteration as described in [34], which is a very well established method. Residual inverse iteration is an implicit quasi-Newton method [25] and exhibits linear convergence. We see in Figure 7 that our proposed method is faster in terms of iterations. In the residual inverse iteration, we have precomputed an LU-factorization in order to speed up the computation of the linear solves.

In order to illustrate that a higher minimality index can allow us to compute more than n eigenvalues, we adapted the idea from Remark 4.3. A comparison with $\ell = 2$ can be seen in Figure 5 with $n = 5$. Minimality index $\ell = 2$ provides the possibility to compute $2n = 10$ eigenvalues.

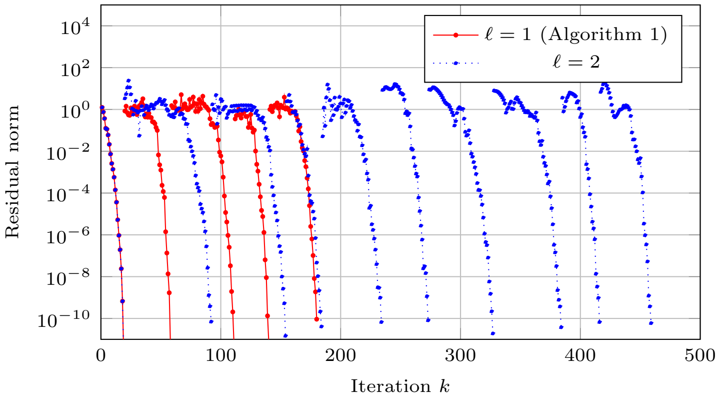


Fig. 5. *Comparison with minimality index one or two.*

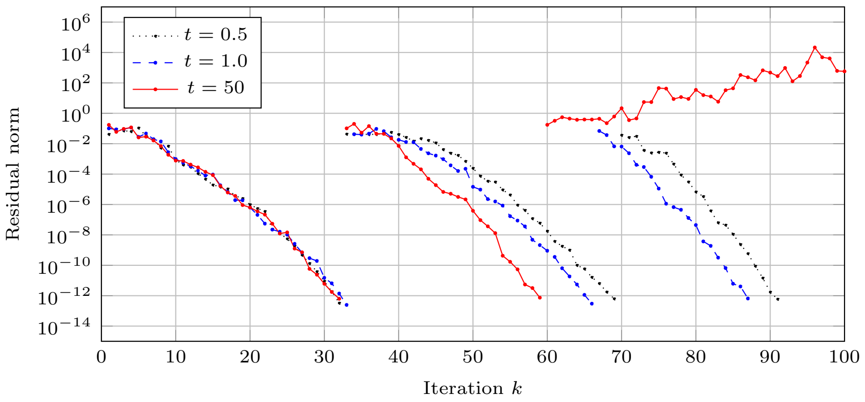
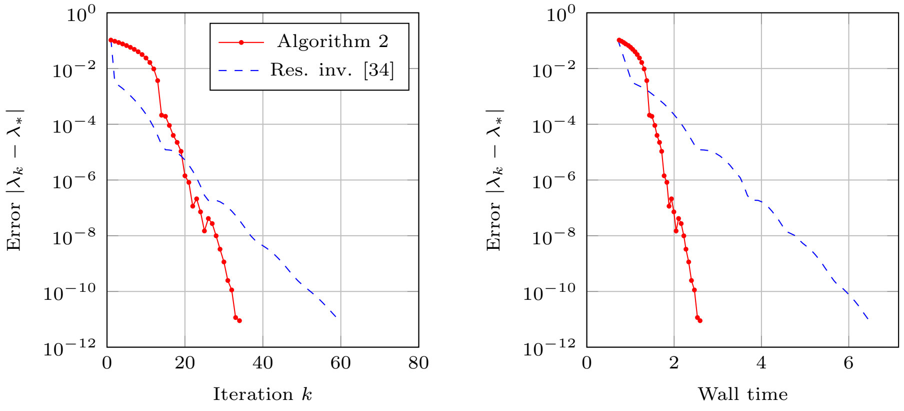


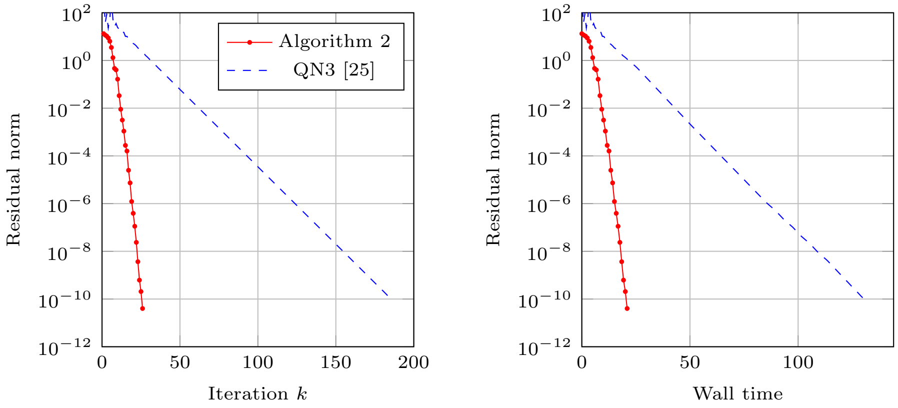
Fig. 6. *Comparison for different threshold parameters.*



(a) Error versus iteration.

(b) Error versus wall time.

Fig. 7. *Comparison with residual inverse iteration.*



(a) Error versus iteration.

(b) Error versus wall time.

FIG. 8. *Comparison with the quasi-Newton method for the problem in section* 6.2.

6.2. Comparison with many nonlinear terms. The proposed method is intended for problems where the computation of the matrix $M(\lambda)$ and the corresponding matrix vector product is computationally expensive. This occurs if we can only decompose the sum in (1.4) with many terms, i.e., $m \gg 1$. Such problems arise naturally in applications, e.g., when the NEP corresponds to the discretization of a PDE-eigenvalue problems with frequency-dependent absorbing boundary conditions and the frequency dependency can only be taken into account using several modes in the exterior of the domain. We let $n = 500$ and consider the following NEP which is a sum of $m = 1000$ different sine functions with different frequencies:

$$M(\lambda) = \lambda A_0 + \sum_{k=1}^{m} \cos(k^{1/4}\lambda) A_k.$$

The matrices are randomly generated full matrices. A simulation comparison with the quasi-Newton approach in [25] is given in Figure 8 with initial value $\lambda_0 = 1.5 + 0.1i$. The figure does not include initialization, which for Algorithm 2 was 6.6 seconds and for quasi-Newton was 2.5 seconds. The iteration converged to a residual norm of 10^{-10} in 26 and 186 iterations, respectively. The proposed algorithm has faster asymptotic convergence speed and also converges faster in practice for this parameter setup. Note that both [25] and Algorithm 2 highly depend on the starting values, and in some cases the quasi-Newton method was faster. However, we expect Algorithm 2 to win if m is sufficiently large, since Algorithm 2 requires fewer matrix vector products per iteration, and our algorithm has superlinear convergence, whereas the quasi-Newton method has linear convergence.

7. Simulations for time-periodic delay-differential equation.

The following problem is called a time-periodic delay-differential equation. We consider a linear (time-varying) delay-differential equation

$$\dot{y}(t) = A(t)y(t) + B(t)y(t-\tau), \tag{7.1}$$

where $A(t), B(t) \in \mathbb{C}^{n\times n}$ are periodic functions with period τ. We briefly summarize a stability characterization which leads to an NEP. For details, see the descriptions of certain applications in [30] and references therein and a number of numerical methods in [39, 21, 22, 20, 5]. The observation that (7.1) can be characterized with an NEP was also used in [36]. We consider the ODE (without delay) associated with (7.1),

$$\dot{p}(t) = C(t,\lambda)p(t), \tag{7.2}$$

where

$$C(t,\lambda) := A(t) + B(t)e^{-\lambda\tau} - \lambda I.$$

We define an NEP $M(\lambda)$ by the action on a vector as

$$M(\lambda)v = p(\tau) - v, \tag{7.3}$$

where $p(\tau)$ is the solution $p(t)$ of (7.2) at $t = \tau$ with initial condition

$$p(0) = v.$$

The solutions of the NEP defined by (7.3) correspond to (λ, v) such that $p(\tau) = v = p(0)$; i.e., the starting value and final vector of p are the same, and p can be viewed as a periodic function. From Floquet theory, one can show that the stability of (7.1) is determined from the rightmost solution λ. The value $\mu = e^{\tau\lambda}$ is called the characteristic multiplier, which is greater than one for right half-plane solutions to the NEP.

Note that the NEP given by (7.3) has an action defined by a solution to an ODE; i.e., the action is computationally expensive and is of the type we consider in this work. In particular, accurate computation of the matrix $M(\lambda)$ requires the solution of n ODEs of dimension n.

7.1. Benchmark problem.

Time-periodic time-delay systems have been considerably used in models and studied in specific applications in the literature. Certain vibrations in machine tool milling can be modeled with time-periodic time-delay systems, where dominant modes correspond to the undesirable machine tool chatter. The delay in this case stems from the fact that the cut of the previous lap has an influence on the current lap. The periodicity stems from the periodicity in the force and from modeling of the cutting tooth which is periodic in time due to the rotation. We consider a specific setup used as a benchmark in several papers; see [21, 22, 20] and references therein. The equations of motion are second order but can be reformulated into a first-order time-periodic time-delay system

$$\dot{y} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 - \frac{a_p w(t)}{m} & -2\zeta\omega_0 \end{bmatrix} y(t) + \begin{bmatrix} 0 & 0 \\ \frac{a_p w(t)}{m} & 0 \end{bmatrix} y(t-\tau).$$

By consideration of the projection of the application of the force (as described in [36]), the time-periodic coefficient becomes

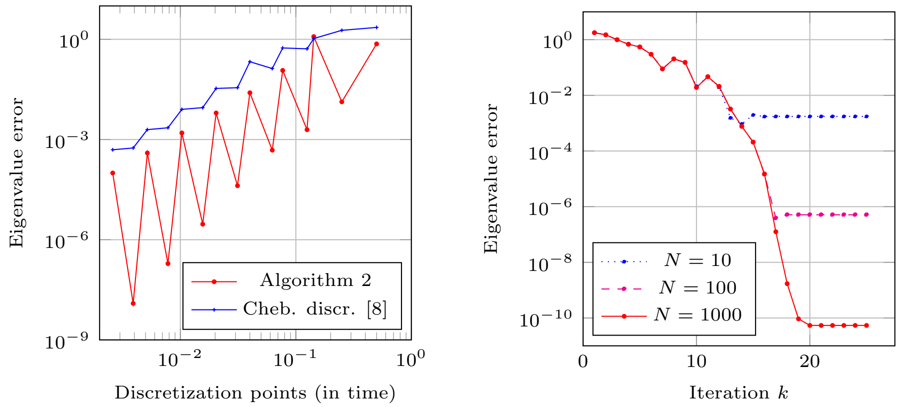
$$w(t) = H(t - \tau/2)(\sin^2(\phi(t))K_R + \cos(\phi(t))\sin(\phi(t))K_T),$$

where $H(t)$ is the Heaviside function and $\phi(t) = 2\pi t/\tau$. We see that if the force modeling is not considered, $w(t)$ is constant, and the problem reduces to the (easier) standard time-delay system.

We carried out simulations with parameters $a_p = m = \tau = \omega_0 = \zeta = 1$ and solved the time-dependent ODE with the Runge–Kutta fourth-order method with N

discretization points. The convergence as a function of iteration is given in Figure 9b. Note that the problem $M(\lambda) \in \mathbb{C}^{2\times 2}$ is a full matrix.

One of the most successful numerical approaches for this problem corresponds to discretizations of operator formulations, e.g., a spectral discretization of the monodromy operator in [8, 5]. In practice, this approach involves the solution of a large (linear) eigenvalue problem. A comparison with the approach in [8] is shown in Figure 9a. We clearly see that the discretization in [8] and our approach lead to algebraic convergence of similar order. That is, although a spectral discretization is used in [8], the observed convergence with respect to ODE-discretization is not exponential but only algebraic. This is expected since $A(t)$ and $B(t)$ have discontinuous derivatives, and in general one cannot expect exponential convergence for PDEs which have discontinuous derivatives.



(a) Continuous problem error versus time-discretization.

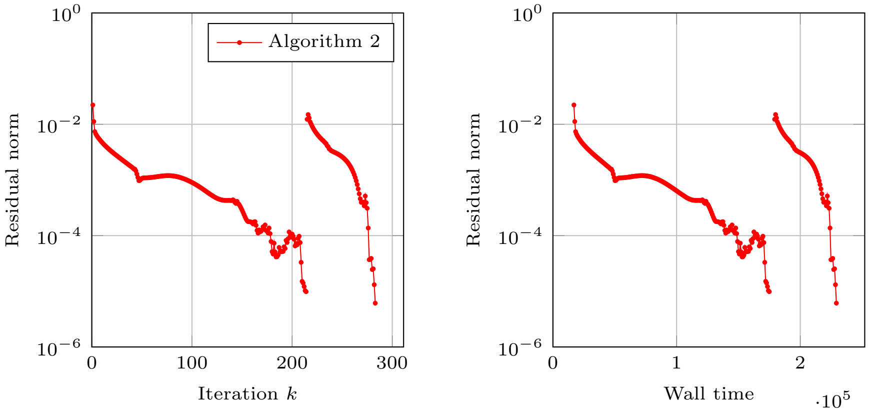
(b) Continuous problem error in comparison with the continuous solution.

FIG. 9. *Simulation with benchmark problem in section* 7.1 *for different time-discretizations.*

7.2. Benchmark problem with PDE coupling. In order to also take into account vibrations in the workpiece in the milling, in [36] we presented a model which couples with a PDE. A discretization of the PDE leads to the following problem. Let $D_{xx} = \frac{1}{h}\,\mathrm{tridiag}(1, -2, 1) \in \mathbb{R}^{N\times N}$, where $h = 1/N$. The identity operator in the finite-element basis is denoted P^{-1}, and $p_n = Pe_n$. The time-periodic time-delay system is now given by

$$\dot{y}(t) = \begin{bmatrix} & & I & \\ & & & 1 \\ -\epsilon P D_{xx} - \frac{a_p w(t)}{A} p_N e_N^T & -\frac{a_p w(t)}{A} e_N & -dPD_{xx} & \\ -\frac{a_p w(t)}{m} e_N^T & -\omega_0^2 - \frac{a_p w(t)}{m} & & -2\zeta\omega_0 \end{bmatrix} y(t) + \begin{bmatrix} & & & \\ & & & \\ \frac{a_p w(t)}{A} p_N e_N^T & \frac{a_p w(t)}{A} p_N & & \\ \frac{a_p w(t)}{m} e_N^T & \frac{a_p w(t)}{m} & & \end{bmatrix} y(t-\tau).$$

We carried out simulations for a discretization with $N = 5000$ on a computer with 64 GB of RAM. Note that $M(\sigma)$ is a full $n \times n$-matrix with $n = 10002$. The results are presented in Figure 10. The action of the ODE was discretized with $N = 15$, whereas the approximation of $M(\sigma)$ was computed with $N = 7$. The problem is stiff, and we therefore used an implicit time-stepping scheme. The inverse of the identity was treated in a way that avoids computing a full matrix. Profiling has determined that most of the computational effort lies in the solution of the ODE associated with the matrix vector—in particular the linear system to be solved in every time-step. This problem is of such size that our implementation of the approach of [8] was not applicable due to the high demand of memory resources.



(a) Error versus iteration.

(b) Error versus wall time (sec).

FIG. 10. *PDE coupling milling simulation.*

8. Conclusions and outlook.

Broyden's method, a standard approach for nonlinear systems of equations, has here been developed and turned into a useful algorithm for certain types of NEPs. Broyden's method has been developed and studied considerably in the literature in various general settings. Several techniques, such as limited memory versions [10, 45], seem to carry over directly, and variations, such as the (so-called) bad Broyden's method [7], can be specialized completely analogously to our approach. In order to maintain generality we intentionally have not pursued a detailed study of the ODE-solver used in the specific application in section 7.2. A more specialized result using the structure of the matrices would probably lead to even further efficiency but would be beyond the scope of this paper, where we focus on NEP-methods rather than the specific problem in section 7.2.

Acknowledgments. The author is grateful for the valuable discussions about Broyden's method with David Ek and Anders Forsgren of the mathematics department, KTH Royal Institute of Technology. The author is also grateful to the referees for constructive comments.

REFERENCES

[1] M. Al-Baali, E. Spedicato, and F. Maggioni, *Broyden's quasi-Newton methods for a nonlinear system of equations and unconstrained optimization: A review and open problems*, Optim. Methods Softw., 29 (2014), pp. 937–954.

[2] T. Betcke, N. J. Higham, V. Mehrmann, C. Schröder, and F. Tisseur, *NLEVP: A collection of nonlinear eigenvalue problems*, ACM Trans. Math. Softw., 39 (2013), 7.

[3] W. J. Beyn and V. Thümmler, *Continuation of invariant subspaces for parameterized quadratic eigenvalue problems*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1361–1381, https://doi.org/10.1137/080723107.

[4] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, SIAM Rev., 59 (2017), pp. 65–98, https://doi.org/10.1137/141000671.

[5] D. Breda, S. Maset, and R. Vermiglio, *Numerical computation of characteristic multipliers for linear time periodic coefficients delay differential equations*, IFAC Proc., 39 (2006), pp. 163–168.

[6] C. G. Broyden, J. E. Dennis, Jr., and J. J. Moré, *On the local and superlinear convergence of quasi-Newton methods*, IMA J. Appl. Math., 12 (1973), pp. 223–245.

[7] C. G. Broyden, *A class of methods for solving nonlinear simultaneous equations*, Math. Comp., 19 (1965), pp. 577–593.

[8] E. Bueler, *Error bounds for approximate eigenvalues of periodic-coefficient linear delay differential equations*, SIAM J. Numer. Anal., 45 (2007), pp. 2510–2536, https://doi.org/10.1137/050633330.

[9] D. W. Decker and C. T. Kelley, *Broyden's method for a class of problems having singular Jacobian at the root*, SIAM J. Numer. Anal., 22 (1985), pp. 566–574, https://doi.org/10.1137/0722034.

[10] P. Deuflhard, *Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms*, Springer, 2004.

[11] C. Effenberger, *Robust Solution Methods for Nonlinear Eigenvalue Problems*, Ph.D. thesis, EPF Lausanne, 2013.

[12] C. Effenberger, *Robust successive computation of eigenpairs for nonlinear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 1231–1256, https://doi.org/10.1137/120885644.

[13] S. W. Gaaf and E. Jarlebring, *The infinite bi-Lanczos method for nonlinear eigenvalue problems*, SIAM J. Sci. Comput., 39 (2017), pp. S898–S919, https://doi.org/10.1137/16M1084195.

[14] C. K. Garrett, Z. Bai, and R.-C. Li, *A nonlinear QR algorithm for banded nonlinear eigenvalue problems*, ACM Trans. Math. Softw., 43 (2016), 4.

[15] D. M. Gay, *Some convergence properties of Broyden's method*, SIAM J. Numer. Anal., 16 (1979), pp. 623–630, https://doi.org/10.1137/0716047.

[16] I. Gohberg, P. Lancaster, and L. Rodman, *Matrix Polynomials*, Academic Press, 1982.

[17] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 2007.

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

[19] R. Hryniv and P. Lancaster, *On the perturbation of analytic matrix functions*, Integral Equations Oper. Theory, 34 (1999), pp. 325–338.

[20] T. Insperger, *Stability Analysis of Periodic Delay-Differential Equations Modeling Machine Tool Chatter*, Ph.D. thesis, Budapest University of Technology and Economics, 2002.

[21] T. Insperger and G. Stépán, *Stability chart for the delayed Mathieu equation*, Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci., 458 (2002), pp. 1989–1998.

[22] T. Insperger and G. Stépán, *Updated semi-discretization method for periodic delay-differential equations with discrete delay*, Internat. J. Numer. Methods Engrg., 61 (2004), pp. 117–141.

[23] E. Jarlebring, *Convergence factors of Newton methods for nonlinear eigenvalue problems*, Linear Algebra Appl., 436 (2012), pp. 3943–3953.

[24] E. Jarlebring, M. Bennedich, G. Mele, E. Ringh, and P. Upadhyaya, *NEP-PACK: A Julia Package for Nonlinear Eigenproblems*, preprint, https://arxiv.org/abs/1811.09592, 2018; software available at https://github.com/nep-pack.

[25] E. Jarlebring, A. Koskela, and G. Mele, *Disguised and new quasi-Newton methods for nonlinear eigenvalue problems*, Numer. Algorithms, 79 (2018), pp. 331–335.

[26] E. Jarlebring and W. Michiels, *Invariance properties in the root sensitivity of time-delay systems with double imaginary roots*, Automatica, 46 (2010), pp. 1112–1115.

[27] E. Jarlebring and W. Michiels, *Analyzing the convergence factor of residual inverse iteration*, BIT, 51 (2011), pp. 937–957.

[28] D. Kressner, *A block Newton method for nonlinear eigenvalue problems*, Numer. Math., 114 (2009), pp. 355–372.

[29] V. N. Kublanovskaya, *On an approach to the solution of the generalized latent value problem for λ-matrices*, SIAM J. Numer. Anal., 7 (1970), pp. 532–537, https://doi.org/10.1137/0707043.

[30] B. P. Mann, T. Insperger, P. V. Bayly, and G. Stépán, *Stability of up-milling and down-milling, part* 2: *Experimental verification*, International Journal of Machine Tools and Manufacture, 43 (2003), pp. 35–40.

[31] K. Meerbergen, *Locking and restarting quadratic eigenvalue solvers*, SIAM J. Sci. Comput., 22 (2001), pp. 1814–1839, https://doi.org/10.1137/S106482759935174X.

[32] K. Meerbergen, *The quadratic Arnoldi method for the solution of the quadratic eigenvalue problem*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1463–1482, https://doi.org/10.1137/07069273X.

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

[34] A. Neumaier, *Residual inverse iteration for the nonlinear eigenvalue problem*, SIAM J. Numer. Anal., 22 (1985), pp. 914–923, https://doi.org/10.1137/0722055.

[35] G. Peters and J. H. Wilkinson, *Inverse iteration, ill-conditioned equations and Newton's method*, SIAM Rev., 21 (1979), pp. 339–360, https://doi.org/10.1137/1021052.

[36] O. Rott and E. Jarlebring, *An iterative method for the multipliers of periodic delay-differential equations and the analysis of a PDE milling model*, IFAC Proc., 43 (2010), pp. 120–125.

[37] A. Ruhe, *Algorithms for the nonlinear eigenvalue problem*, SIAM J. Numer. Anal., 10 (1973), pp. 674–689, https://doi.org/10.1137/0710059.

[38] K. Schreiber, *Nonlinear Eigenvalue Problems: Newton-Type Methods and Nonlinear Rayleigh Functionals*, Ph.D. thesis, TU Berlin, 2008.

[39] R. Szalai, G. Stépán, and S. J. Hogan, *Continuation of bifurcations in periodic delay-differential equations using characteristic matrices*, SIAM J. Sci. Comput., 28 (2006), pp. 1301–1317, https://doi.org/10.1137/040618709.

[40] D. B. Szyld and F. Xue, *Efficient preconditioned inner solves for inexact Rayleigh quotient iteration and their connections to the single-vector Jacobi–Davidson method*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 993–1018, https://doi.org/10.1137/100807922.

[41] D. B. Szyld and F. Xue, *Local convergence analysis of several inexact Newton-type algorithms for general nonlinear eigenvalue problems*, Numer. Math., 123 (2013), pp. 333–362.

[42] D. B. Szyld and F. Xue, *Local convergence of Newton-like methods for degenerate eigenvalues of nonlinear eigenproblems.* I. *Classical algorithms*, Numer. Math., 129 (2015), pp. 353–381.

[43] D. B. Szyld and F. Xue, *Local convergence of Newton-like methods for degenerate eigenvalues of nonlinear eigenproblems:* II. *Accelerated algorithms*, Numer. Math., 129 (2015), pp. 383–403.

[44] H. Unger, *Nichtlineare Behandlung von Eigenwertaufgaben*, Z. Angew. Math. Mech., 30 (1950), pp. 281–282; English translation available at http://www.math.tu-dresden.de/~schwetli/Unger.html.

[45] B. A. van de Rotten and S. M. Verduyn Lunel, *A memory-efficient Broyden method to compute fixed points of non-linear maps arising in periodically forced processes*, IMA J. Appl. Math., 80 (2015), pp. 585–607.

[46] H. Voss, *An Arnoldi method for nonlinear eigenvalue problems*, BIT, 44 (2004), pp. 387–401.

[47] H. Voss, *Nonlinear eigenvalue problems*, in Handbook of Linear Algebra, 2nd ed., L. Hogben, ed., CRC Press, 2014, Chapter 60.