A NEWTON-TYPE METHOD WITH NON-EQUIVALENCE DEFLATION FOR NONLINEAR EIGENVALUE PROBLEMS ARISING IN PHOTONIC CRYSTAL MODELING

TSUNG-MING HUANG*AND WEN-WEI LIN†AND VOLKER MEHRMANN‡

Abstract. The numerical simulation of the band structure of three-dimensional dispersive metallic photonic crystals with face-centered cubic lattices leads to large-scale nonlinear eigenvalue problems, which are very challenging due to a high dimensional subspace associated with the eigenvalue zero and the fact that the desired eigenvalues (with smallest real part) cluster and close to the zero eigenvalues. For the solution of the nonlinear eigenvalue problem, a Newton-type iterative method is proposed and the nullspace-free method is applied to exclude the zero eigenvalues from the associated generalized eigenvalue problem. To find the successive eigenvalue/eigenvector pairs, we propose a new non-equivalence deflation method to transform converged eigenvalues to infinity, while all other eigenvalues remain unchanged. The deflated problem is then solved by the same Newton-type method, which is used as a hybrid method that combines with the Jacobi-Davidson and the nonlinear Arnoldi methods to compute the clustering eigenvalues. Numerical results illustrate that the proposed method is robust even for the case of computing many clustering eigenvalues in very large problems.

Key words. Maxwell equation, dispersive metallic photonic crystals, nonlinear eigenvalue problem, Newton-type method, non-equivalence deflation, Jacobi-Davidson method, shift-invert residual Arnoldi method, nonlinear Arnoldi method

AMS subject classifications. 15A18, 15A90, 65F15

1. Introduction.

The electromagnetic wave propagation through dispersive metallic photonic crystals (PCs) has been extensively studied over the past few decades [8, 9, 10, 34, 38, 47]. A standard model to study the electromagnetic effects in periodic structures and dispersive isotropic materials is the three-dimensional (3D) Maxwell equation

$$\nabla \times \nabla \times E(\mathbf{r}) = \omega^2 \varepsilon(\mathbf{r}, \omega) E(\mathbf{r}), \tag{1.1}$$

where $E(\mathbf{r})$ denotes the electric field at position $\mathbf{r} \in \mathbb{R}^3$ and

$$\varepsilon(\mathbf{r}, \omega) = \begin{cases} \varepsilon_d(\omega), & \mathbf{r} \text{ in the dispersive material domain,} \\ \varepsilon_n, & \mathbf{r} \text{ in the nondispersive material domain,} \end{cases}$$

denotes the permittivity, in which ε_n is a constant and $\varepsilon_d(\omega)$ is dependent on the frequency ω.

In the Drude model of a dispersive material [34, 38, 52, 53], the permittivity $\varepsilon_d(\omega)$ is modeled as

$$\varepsilon_d(\omega) = 1 - \frac{\omega_p^2}{\omega^2 + \imath \Gamma_p \omega}, \tag{1.2}$$

where $\imath = \sqrt{-1}$, ω_p is a plasma frequency, and Γ_p is the corresponding damping frequency. The more involved Drude-Lorentz model [10, 11, 34, 47] uses the permittivity

[1]Department of Mathematics, National Taiwan Normal University, Taipei 116, Taiwan. E-mail: min@ntnu.edu.tw

[2]Department of Applied Mathematics, National Chiao Tung University, Hsinchu 300, Taiwan. E-mail: wwlin@math.nctu.edu.tw

[3]Institut für Mathematik, MA 4-5, TU Berlin, Straße des 17. Juni 136, D-10623 Berlin, Germany. E-mail: mehrmann@math.tu-berlin.de

model

$$\varepsilon_d(\omega) = \varepsilon_\infty - \frac{\omega_p^2}{\omega^2 + \imath\Gamma_p\omega} + \sum_{j=1}^{2} \Omega_j A_j \left(\frac{e^{\imath\phi_j}}{\Omega_j - \omega - \imath\Gamma_j} + \frac{e^{-\imath\phi_j}}{\Omega_j + \omega + \imath\Gamma_j} \right), \qquad (1.3)$$

where ε_∞ is a high-frequency limit dielectric constant, ϕ_j, Ω_j, A_j and Γ_j are the parameters for the two pairs of poles in the Lorentz model. The first and second term in (1.3) correspond to the contributions of a Drude model with $\varepsilon_\infty = 1$. The third term arises from the interband transitions.

There are many approaches for the numerical simulation of wave propagation in photonic crystals. Very often, time-domain simulation methods, see e.g. [12], are used to calculate the band structures of 3D metallic PCs, but in order to obtain reasonable results extremely long simulation times are required. As an alternative, often Fourier transform and the discretization of the time-invariant system (1.1) is considered, which leads to a nonlinear eigenvalue problem (NLEVP) [8, 9, 34] rational in the frequency ω. To solve large scale NLEVPs, however, is also a nontrivial task [37, 46], and is particularly challenging for 3D metallic PCs. In [8, 9, 34], the rational eigenvalue problem is reformulated as a polynomial eigenvalue problems (PEPs) by multiplying with the common denominator. The PEP is then reformulated (linearized) [13, 35] as a generalized eigenvalue problem to which standard eigenvalue methods [2, 32, 39] can be applied. However, in this approach the order of the problem is highly enlarged and the sensitivity of the eigenvalues and eigenvectors may increase considerably, since the set of admissible perturbations for the linearized eigenvalue problem is larger than that of the PEP [45] and, even more of the rational problem.

A completely different eigenvalue method is to work directly with the NLEVP. One can use the polynomial Jacobi-Davidson method [21, 24, 41, 42, 50] for the PEPs or the rational Krylov method [26, 40], nonlinear Arnoldi method [48], nonlinear Jacobi-Davidson method [49], contour integrals [1, 4], and other methods in [7, 25, 29, 33, 41] to solve the general NLEVP. In [5, 7] a very promising new method for PEPs is suggested that computes full invariant pairs but in Jacobi-Davidson or Newton type methods, often only one eigenvalue/eigenvector pair is determined at a time, e.g. when the algorithms in [21, 24, 26, 33, 40, 42, 48, 49, 50] are applied to solve the NLEVP. One way then to compute several successive eigenpairs is to deflate converged eigenvalue/eigenvectors from the NLEVP. In [15], an explicit non-equivalence low-rank deflation method is proposed for computing the smallest real eigenvalues of a special quadratic eigenvalue problem. Once the smallest positive eigenvalue is obtained, it is then transformed to zero by the deflation scheme, while all other eigenvalues remain unchanged. The next successive eigenvalue thus becomes the smallest positive eigenvalue of the transformed problem, which is then again solved by the proposed method. The concept of non-equivalence deflation is also applied to solve special quadratic problems in [20, 23], and cubic polynomial eigenvalue problems [22, 50]. One of the differences of the non-equivalence deflation scheme in [15] and [20, 22, 23, 50] is that the convergent eigenvalue is transformed to infinity rather than 0.

Almost all currently available eigenvalue methods have major difficulties when multiple eigenvalues occur, because then the convergence of Newton-type methods deteriorates and in this case also the condition number of the problem typically is extremely large, so that small perturbations lead to large errors. In this case it is necessary to consider either block oriented methods [3, 5, 7, 36] which currently are designed for quadratic and polynomial problems, or in the more general nonlinear

case, one needs to employ specially designed deflation techniques as the one we will discuss in this paper.

We consider large, sparse NLEVPs of the form

$$A\mathbf{x} = \omega(\omega B_n + \omega\varepsilon_d(\omega)B_d)\mathbf{x}, \tag{1.4}$$

arising from a 3D dispersive metallic PC with a face-centered cubic lattice, where A, B_n, B_d are large, sparse matrices, and the subscripts n and d indicate the non-dispersive and the dispersive materials, respectively. This problem has a large number of zero eigenvalues and the eigenvalues with smallest positive real part are of interest. Moreover, most of the desired eigenvalues are clustered so that it becomes an extremely challenging problem. To deal with this challenge, we propose a Newton-type iterative method with a special non-equivalence deflation scheme to solve (1.4).

By sequential application of the proposed non-equivalence deflation, we formulate the deflated NLEVPs as

$$A\tilde{\mathbf{x}} = \omega\widetilde{B}(\omega)\tilde{\mathbf{x}}, \tag{1.5}$$

where the desired eigenpairs are sequentially computed from a sequence of NLEVP in (1.5). To solve (1.5), a novel Newton-type scheme is presented which solves in each step the generalized eigenvalue problem (GEP)

$$\beta A\tilde{\mathbf{x}} = \widetilde{B}(\omega_k)\tilde{\mathbf{x}}$$

starting from a sample frequency ω_k. We apply the nullspace-free method of [17, 19], and transform the GEP to a standard eigenvalue problem (SEP). Both, the SEP and the GEP have the same nonzero finite eigenvalues. Furthermore, a heuristic strategy is proposed to determine the initial value ω_0 of the Newton-type method so that the desired eigenvalue clusters can be successfully computed. The quadratic convergence of the method is illustrated via numerical examples. The SEP can be solved by the Jacobi-Davidson method [43] or the shift-and-invert residual Arnoldi method [19, 30, 31]. An efficient FFT-based preconditioner is derived in the eigensolver so that the SEP can be efficiently solved. To make this method more practical, we discuss other strategies for determining initial vectors and stopping tolerances for the solution of the SEP, which reduce the computational cost and accelerate the convergence. Intensive numerical experiments show that our proposed method is robust and efficient to sequently compute the desired eigenpairs even through the desired eigenvalues are clustered.

This paper is organized as follows. In Section 2, we briefly derive the NLEVP and the new non-equivalence low-rank deflation method. In Section 3, we introduce the Newton-type method. The Jacobi-Davidson, the shift-invert residual Arnoldi, the nonlinear Arnoldi and nonlinear Jacobi-Davidson methods to solve the SEP and NLEVP, respectively, are reviewed in Section 4. Some practical implementations to reduce the computational cost are proposed in Section 5. Numerical experiments to validate the robustness of the proposed schemes are demonstrated in Section 6. We conclude the paper in Section 7.

2. Nonlinear eigenvalue problems. In this section, we first introduce the resulting NLEVP by using the Yee scheme [51] for the discretization of the Maxwell equation (1.1). Then we propose a non-equivalence deflation scheme which allows to transform the resulting NLEVP with eigenvalue/eigenvector pair $(\mu, \mathbf{x})$ to a new

NLEVP with the same eigenvalues except that μ is replaced by infinity. Both the original and the deflated NLEVP can be written in the form

$$A\mathbf{x} = \omega \widetilde{B}(\omega)\mathbf{x} \tag{2.1}$$

In the next section, we will then develop a Newton-type method to solve (2.1) so that we can intertwine this method and the non-equivalence deflation scheme to compute the desired eigenvalue/eigenvector pairs.

Based on the Bloch Theorem [28], we aim to find the Bloch eigenfunctions $E(\mathbf{r})$ for (1.1) satisfying the following quasi-periodicity condition

$$E(\mathbf{r} + \mathbf{a}_\ell) = e^{\imath 2\pi \mathbf{k}\cdot\mathbf{a}_\ell} E(\mathbf{r}),$$

for $\ell = 1, 2, 3$. Here, $2\pi\mathbf{k}$ is the Bloch wave vector in the first Brillouin zone [27] and the vectors $\mathbf{a}_\ell$ are the lattice translation vectors that span the primitive cell, which are extended periodically to form the dispersive metallic photonic crystal. In this paper, we focus on the face-centered cubic lattice (FCC) vectors, i.e.,

$$\mathbf{a}_1 = \frac{a}{\sqrt{2}}[1,0,0]^\top,\ \mathbf{a}_2 = \frac{a}{\sqrt{2}}\left[\frac{1}{2}, \frac{\sqrt{3}}{2}, 0\right]^\top,\ \mathbf{a}_3 = \frac{a}{\sqrt{2}}\left[\frac{1}{2}, \frac{1}{2\sqrt{3}}, \sqrt{\frac{2}{3}}\right]^\top$$

with lattice constant a.

Let n_1, n_2, and n_3 with $n = n_1 n_2 n_3$ be the number of grid points in x, y, and z direction, respectively, and let δ_x, δ_y, and δ_z denote the associated grid lengths in x, y, and z axial direction, respectively.

The resulting matrix A arising from the discretized double-curl operator using the Yee scheme [51] on a primitive cell is then of the form [16, 17, 18]

$$A = C^*C \in \mathbb{C}^{3n\times 3n}, \tag{2.2}$$

where

$$C = \begin{bmatrix} 0 & -C_3 & C_2 \\ C_3 & 0 & -C_1 \\ -C_2 & C_1 & 0 \end{bmatrix} \in \mathbb{C}^{3n\times 3n},$$

with

$$C_1 = I_{n_2 n_3} \otimes K_1 \in \mathbb{C}^{n\times n},\ C_2 = I_{n_3} \otimes K_2 \in \mathbb{C}^{n\times n},\ C_3 = K_3 \in \mathbb{C}^{n\times n}. \tag{2.3}$$

Here $\otimes$ denotes the Kronecker product, see Appendix A.1 or [17] for the detailed definition of the pseudo periodical matrices K_1, K_2, and K_3. The resulting NLEVP then has the form

$$F(\omega)\mathbf{x} \equiv \left(A - \omega^2 B(\omega)\right)\mathbf{x} = 0 \tag{2.4}$$

with

$$B(\omega) = B_n + \varepsilon_d(\omega) B_d, \tag{2.5}$$

where A is defined in (2.2), B_n, B_d are diagonal and $\varepsilon_d(\omega)$ is the related permittivity of the dispersive material given in (1.2) and (1.3). The goal of the eigenvalue computation is to compute the eigenvalues of smallest positive real part and associated eigenvectors.

To conduct the analysis of the NLEVP (2.4) we have the following definition and lemma.

DEFINITION 2.1. *Let $F(\omega)$ be a rational matrix, i.e., all entries of $F(\omega)$ are rational in ω. Furthermore, suppose that $F(\omega)$ can be represented as*

$$F(\omega) = P(\omega) + R(\omega), \tag{2.6}$$

where $P(\omega)$ is a polynomial matrix of degree r and $R(\omega)$ is a rational matrix with entries being proper rational.

(a) *If $\omega_0 \in \mathbb{C}$ satisfies $\det(F(\omega_0)) = 0$, then ω_0 is an eigenvalue of $F(\omega)$. A nonzero vector $\mathbf{x}$ satisfying $F(\omega_0)\mathbf{x} = 0$ is then called associated eigenvector corresponding to ω_0.*

(b) *$F(\omega)$ has an eigenvalue at infinity if $\lim_{\omega\to\infty} \det(\omega^{-r}F(\omega)) = 0$. A nonzero vector $\mathbf{x}$ satisfying $\lim_{\omega\to\infty}(\omega^{-r}F(\omega))\mathbf{x} = 0$ is then called associated eigenvector corresponding to the infinite eigenvalue.*

LEMMA 2.2. *Let $\varepsilon_d(\omega)$ be defined as in (1.2) or (1.3). Then $\overline{\varepsilon_d(\omega)} = \varepsilon_d(-\bar{\omega})$.*

Proof. From (1.2) and (1.3), we directly have

$$\overline{\varepsilon_d(\omega)} = 1 - \frac{\omega_p^2}{\bar{\omega}^2 - \imath\Gamma_p\bar{\omega}} = 1 - \frac{\omega_p^2}{(-\bar{\omega})^2 + \imath\Gamma_p(-\bar{\omega})} = \varepsilon_d(-\bar{\omega})$$

and

$$\begin{aligned}\overline{\varepsilon_d(\omega)} &= \varepsilon_\infty - \frac{\omega_p^2}{(\bar{\omega})^2 - \imath\Gamma_p\bar{\omega}} + \sum_{j=1}^{2} \Omega_j A_j \left(\frac{e^{-\imath\phi_j}}{\Omega_j - \bar{\omega} + \imath\Gamma_j} + \frac{e^{\imath\phi_j}}{\Omega_j + \bar{\omega} - \imath\Gamma_j}\right) \\ &= \varepsilon_d(-\bar{\omega}),\end{aligned}$$

respectively. □

With the help of Lemma 2.2 we have the following Theorem.

THEOREM 2.3. *The NLEVP (2.4) has the following properties.*

(a) *$F(\omega)$ has n zero eigenvalues and no eigenvalue at infinity.*

(b) *$F^*(\omega) = F(-\bar{\omega})$, i.e., ω and $-\bar{\omega}$ are eigenvalues of $F(\omega)$.*

(c) *If $\mathbf{y}$ is a left eigenvector of $F(\omega)$ corresponding to the eigenvalue ω, then $\mathbf{y}$ is a right eigenvector corresponding to the eigenvalue $-\bar{\omega}$.*

Proof. From (2.5) and the definition of $\varepsilon_d(\omega)$ in (1.2) or (1.3), it follows that $(\omega B(\omega))|_{\omega=0} = -\frac{\omega_p^2}{\imath\Gamma_p}B_d$. This implies that the eigenvectors of A corresponding to the zero eigenvalues are also the eigenvectors of $F(\omega)$ corresponding to the zero eigenvalues. Therefore, by Theorem 3.7 in [17], $F(\omega)$ has a semi-simple zero eigenvalue with multiplicity n.

On the other hand, $F(\omega)$ can be represented as the form in (2.6) with degree r of $P(\omega)$ being equal to 2. Because $\lim_{\omega\to\infty}\omega^{-2}F(\omega)$ is equal to the nonsingular matrix $B_n + B_d$ or $B_n + \varepsilon_\infty B_d$ for $\varepsilon_d(\omega)$ in (1.2) or (1.3), respectively, by the definition, $F(\omega)$ has no eigenvalue at infinity.

By the definition of $B(\omega)$ in (2.5) and Lemma 2.2, it follows that

$$\begin{aligned}F^*(\omega) &= A^* - (\bar{\omega})^2\left(B_n + \overline{\varepsilon_d(\omega)}B_d\right) \\ &= A - (-\bar{\omega})^2\left(B_n + \varepsilon_d(-\bar{\omega})B_d\right) \\ &= F(-\bar{\omega}).\end{aligned}$$

This means that if $\mathbf{y}$ is the left eigenvector of $F(\omega)$ corresponding to ω, then

$$0 = F^*(\omega)\mathbf{y} = F(-\bar{\omega})\mathbf{y},$$

and, therefore, $\mathbf{y}$ is the right eigenvector corresponding to $-\bar{\omega}$. □

Let us assume that each of the pairwise different nonzero eigenvalues μ_j has equal algebraic and geometric multiplicity m_j, $j = 1, \ldots, \ell$, and let

$$X = \begin{bmatrix} X_1 & X_2 & \cdots & X_\ell \end{bmatrix} \tag{2.7}$$

with $X_j \in \mathbb{C}^{3n \times m_j}$, $j = 1, \ldots, \ell$, satisfy $X^*X = I_m$ with $m = m_1 + \cdots + m_\ell$. The following non-equivalence deflation allows us to transform the original problem (2.4) to a new NLEVP with the same eigenvalues, except that μ_i is replaced by infinity with multiplicity m_i for $i = 1, \ldots, \ell$. With

$$\widetilde{F}(\omega)\tilde{\mathbf{x}} := \left(F(\omega) \prod_{j=1}^{\ell} \left(I - \frac{\omega}{\omega - \mu_j} X_j X_j^* \right) \right) \tilde{\mathbf{x}}, \tag{2.8}$$

we have the following theorem.

THEOREM 2.4. *Let $F(\omega)$ and $\widetilde{F}(\omega)$ be defined as in* (2.4) *and* (2.8)*, respectively. Then,*

$$\begin{aligned} &\left\{ \omega |\ \widetilde{F}(\omega)\tilde{\mathbf{x}} = 0,\ \tilde{\mathbf{x}} \neq 0 \right\} \\ &\quad = \ \{ \omega |\ F(\omega)\mathbf{x} = 0, \mathbf{x} \neq 0 \} \setminus \{ \mu_1, \cdots, \mu_1, \cdots, \mu_\ell, \cdots, \mu_\ell \} \cup \{ \infty \}. \end{aligned}$$

Furthermore, if $(\mu, \tilde{\mathbf{x}})$ is an eigenvalue/eigenvector pair of $\widetilde{F}(\omega)$, then $(\mu, \mathbf{x})$ is an eigenvalue/eigenvector pair of $F(\omega)$ with

$$\mathbf{x} = \prod_{j=1}^{\ell} \left(I - \frac{\mu}{\mu - \mu_j} X_j X_j^* \right) \tilde{\mathbf{x}}. \tag{2.9}$$

Proof. Since $X_i^* X_i = I_{m_i}$ for $i = 1, \ldots, \ell$, we have

$$(\lim_{\omega \to \infty} \omega^{-2} \widetilde{F}(\omega)) X_i = \left((B_n + \alpha B_d) \prod_{j=1}^{\ell} \left(I - X_j X_j^* \right) \right) X_i = 0,$$

where $\alpha = 1$ or $\alpha = \varepsilon_\infty$. This implies that $\widetilde{F}(\omega)$ has eigenvalue at infinity and the columns of X_i are the associated eigenvectors. Moreover, using the determinant identity $\det(I_n + RS) = \det(I_m + SR)$, where R and S^* are $n \times m$ matrices, we get

$$\begin{aligned} \det\left(\widetilde{F}(\omega) \right) &= \det\left(F(\omega) \right) \prod_{j=1}^{\ell} \det\left(I - \frac{\omega}{\omega - \mu_j} X_j X_j^* \right) \\ &= \det\left(F(\omega) \right) \prod_{j=1}^{\ell} \left(1 - \frac{\omega}{\omega - \mu_j} \right)^{m_j} \\ &= \det\left(F(\omega) \right) \prod_{j=1}^{\ell} \left(\frac{-\mu_j}{\omega - \mu_j} \right)^{m_j}. \end{aligned} \tag{2.10}$$

Because $\omega = \mu_j$ is a root with multiplicity m_j of $\det(F(\omega))$ by assumption, (2.10) implies that μ_j is NOT a root of $\det(\widetilde{F}(\omega))$. Hence, the nonlinear eigenvalue problem (2.8) has the same eigenvalues as (2.4) except that m_j copies of the eigenvalue μ_j are replaced by infinite eigenvalues.

From the definition of $\widetilde{F}(\omega)$ in (2.8), it is obvious to see that the vector $\mathbf{x}$ in (2.9) is an eigenvector of $F(\omega)$. □

REMARK 1. *In fact, in the non-equivalence deflation* (2.8)*, the matrix* X *as in* (2.7) *can be a randomly constructed orthonormal matrix. However, the locking scheme used later in the nonlinear Arnoldi algorithm (NAr) needs to lock the convergent eigenvectors into the search subspace. So, we prefer to use the set of convergent eigenvectors to generate such a matrix* X*. If one of the convergent eigenvectors,* $\mathbf{x}_i$*, is linearly dependent to others, then we can randomly construct a vector to replace* $\mathbf{x}_i$*.*

Using the fact that $X^*X = I_m$, we obtain

$$\prod_{j=1}^{\ell}\left(I - \frac{\omega}{\omega-\mu_j}X_jX_j^*\right) = I - \sum_{j=1}^{\ell}\frac{\omega}{\omega-\mu_j}X_jX_j^* = I - \omega X D(\omega) X^*, \tag{2.11}$$

where

$$D(\omega) = \operatorname{diag}\left((\omega-\mu_1)^{-1}I_{m_1}, (\omega-\mu_2)^{-1}I_{m_2}, \cdots, (\omega-\mu_\ell)^{-1}I_{m_\ell}\right).$$

Plugging $F(\omega)$ of (2.4) and (2.11) into (2.8), $\widetilde{F}(\omega)$ can be reformulated in the following simple form

$$\widetilde{F}(\omega) = A - \omega\left[\omega B(\omega) + (A - \omega^2 B(\omega))XD(\omega)X^*\right]. \tag{2.12}$$

Define

$$\widetilde{B}(\omega) = \begin{cases} \omega B(\omega) & \text{for } F(\omega), \\ \omega B(\omega) + (A - \omega^2 B(\omega))XD(\omega)X^* & \text{for } \widetilde{F}(\omega). \end{cases} \tag{2.13}$$

We then have that the NLEVPs (2.4) and (2.8) can both be represented in the form (2.1).

Note that, from the definition of $B(\omega)$ in (2.5) and Lemma 2.2, $B(\omega)$ is a complex diagonal matrix and $\omega B(\omega) + (A - \omega^2 B(\omega))XD(\omega)X^*$ is a complex diagonal plus a non-Hermitian low rank matrix.

Therefore, the desired eigenvalue/eigenvector pairs of $F(\omega)$, i.e. the ones with smallest positive real part, can be found by repeatedly solving deflated NLEVPs in (2.12) and using (2.9) in Theorem 2.4 to recover the eigenvector of $F(\omega)$. We summarize the computational process in Algorithm 1.

3. Newton-type methods. Based on the Newton-type method suggested in [15], in this section, we will develop a Newton-type method for the computation of the desired eigenvalue/eigenvector pair for general NLEVPs of the form $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$ in Line 3 of Algorithm 1. In (2.1) for a given ω, we consider the GEP

$$\beta A\mathbf{x} = \widetilde{B}(\omega)\mathbf{x}, \tag{3.1}$$

where the eigenvalues β depend on the chosen value of ω. To determine an eigenvalue of (2.1), it is sufficient to find a value ω_* such that the eigenvalue $\beta(\omega_*)$ of (3.1)

Algorithm 1 Non-equivalence deflated method for solving $A\mathbf{x} = \omega^2 B(\omega)\mathbf{x}$.

Input: Coefficient matrices A (Hermitian) and $B(\omega)$.
Output: The desired eigenvalue/eigenvector pair $(\mu_d, \mathbf{x}_d)$ for $d = 1, \ldots, \ell$.

Set $X = [\,]$ and $\widetilde{B}(\omega) = \omega B(\omega)$.
for $d = 1, \ldots, \ell$ **do**
  Compute the desired eigenvalue/eigenvector pair $(\mu_d, \mathbf{x}_d)$ of $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$;
  % Retrieve the eigenvector of $Ax = \omega^2 B(\omega)x$ by Theorem 2.4
  for $i = 1, \ldots, d-1$ **do**
    Compute $\mathbf{x}_d = \left(I - \frac{\mu_d}{\mu_d - \mu_i}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^*\right)\mathbf{x}_d$;
  end for
  % Compute the orthonormal matrix X from the convergent eigenvectors
  Set $\tilde{\mathbf{x}}_d = \mathbf{x}_d$; Orthogonalize $\tilde{\mathbf{x}}_d$ against X and normalize $\tilde{\mathbf{x}}_d$;
  Expand $X = [X, \tilde{\mathbf{x}}_d]$;
  % Create the coefficient matrix of the new deflated nonlinear eigenvalue problem
  Set

$$\widetilde{B}(\omega) = \omega B(\omega) + (A - \omega^2 B(\omega))XD(\omega)X^*,$$

  where $D(\omega) = \mathrm{diag}\left((\omega - \mu_1)^{-1}, \cdots, (\omega - \mu_d)^{-1}\right)$;
end for

satisfies the condition $\beta(\omega_*) = \omega_*^{-1}$, which is equivalent to determine a root of the nonlinear equation

$$\beta(\omega) = \omega^{-1}. \tag{3.2}$$

The simplest method to solve this equation is to use a fixed-point iteration $\omega_{k+1} = \beta(\omega_k)^{-1}$, so that when it has converged to a value ω_*, then an eigenvalue $\beta(\omega_k)$ of (3.1) with $\omega = \omega_*$ has been computed. But the convergence of fixed-point iterations is typically linear. In this paper, we apply the Newton method

$$\omega_{k+1} = \omega_k - \left(\beta'(\omega_k) + \omega_k^{-2}\right)^{-1}\left(\beta(\omega_k) - \omega_k^{-1}\right) \tag{3.3}$$

to (3.2) to accelerate the convergence.

3.1. Nullspace-free method. The success of Newton's method (3.3) is primarily based on reliable computation of $\beta(\omega_k)$ and $\beta'(\omega_k)$. However, such a reliable computation may not be guaranteed if one computes them directly using the original form of the eigenproblem $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$, because the zero eigenvalue of multiplicity n severely interferes with our search for the eigenvalues of smallest positive real part. To resolve this difficulty, we propose a null-space free method [17] that transforms the GEP (3.1) into an SEP of smaller dimension with the zero eigenvalue deflated. To this end, we present the following theorems and shall make use of them for computing $\beta'(\omega)$ reliably.

THEOREM 3.1 ([17]). *Let C_ℓ ($\ell = 1, 2, 3$) be as in (2.3). Then, $C_i^* C_j = C_j C_i^*$, $C_i C_j = C_j C_i$, for $i, j = 1, 2, 3$ and all three matrices C_ℓ can be diagonalized by the same unitary matrix T, i.e.,*

$$C_1 T = T\Lambda_1,\ C_2 T = T\Lambda_2, \text{ and } C_3 T = T\Lambda_3, \tag{3.4}$$

where Λ_1*,* Λ_2*, and* Λ_3 *are diagonal matrices* (*See Appendix A.2 for the definition of* T).

THEOREM 3.2 ([17]). *Let* A *and* $(\Lambda_1, \Lambda_2, \Lambda_3, T)$ *be defined as in* (2.2) *and* (3.4)*, respectively. Then there exists a unitary matrix*

$$\begin{bmatrix} Q_0 & Q \end{bmatrix} := (I_3 \otimes T) \begin{bmatrix} \Pi_0 & \Pi_1 \end{bmatrix} \equiv (I_3 \otimes T) \left[\begin{array}{c|cc} \Pi_{0,1} & \Pi_{1,1} & \Pi_{1,2} \\ \Pi_{0,2} & \Pi_{1,3} & \Pi_{1,4} \\ \Pi_{0,3} & \Pi_{1,5} & \Pi_{1,6} \end{array}\right],$$

where $\Pi_{0,i}$*,* $\Pi_{1,j} \in \mathbb{C}^{n\times n}$*,* $i = 1, 2, 3$*,* $j = 1, \ldots, 6$*, are diagonal and* $\Pi_1 \in \mathbb{C}^{3n\times 2n}$ *such that* A *has a unitary eigendecomposition as the form*

$$\begin{bmatrix} Q_0 & Q \end{bmatrix}^* A \begin{bmatrix} Q_0 & Q \end{bmatrix} = \operatorname{diag}(0, \Lambda_q, \Lambda_q) \equiv \operatorname{diag}(0, \Lambda), \tag{3.5}$$

where $\Lambda_q = \Lambda_1^*\Lambda_1 + \Lambda_2^*\Lambda_2 + \Lambda_3^*\Lambda_3$.

In Theorem 3.2 it has been shown that with $n = n_1n_2n_3$ in the discretization (2.2), the matrix A has n zero eigenvalues. Because we are interested in finding the eigenvalues of (2.1) with smallest positive real parts, it means that the eigenvalues β^{-1} of the GEP $A\mathbf{x} = \beta^{-1}\widetilde{B}(\omega)\mathbf{x}$ with smallest positive real parts are of interest. In this respect, the large dimension of the invariant space corresponding to zero eigenvalues in the above GEP leads to several numerical difficulties, see [17]. To tackle this issue, the nullspace-free technique in Theorem 3.3 can be applied for solving (3.1) efficiently.

THEOREM 3.3 ([19]). *Let* A *be as in* (2.2)*, let* (Q, Λ) *be as in Theorem 3.2, and let* ω *satisfy* $\widetilde{B}(\omega)$ *in* (3.1) *being nonsingular. Then (denoting by span of a matrix the span of its columns),*

$$\operatorname{span}\widetilde{B}(\omega)^{-1}Q\Lambda^{1/2} = \operatorname{span}\left\{\mathbf{x} | A\mathbf{x} = \lambda\widetilde{B}(\omega)\mathbf{x},\ \lambda \neq 0\right\},$$

and furthermore,

$$\left\{\lambda \neq 0 \,|\, A\mathbf{x} = \lambda\widetilde{B}(\omega)\mathbf{x}\right\} = \left\{\lambda \,|\, \Lambda^{1/2}Q^*\widetilde{B}(\omega)^{-1}Q\Lambda^{1/2}\mathbf{u} = \lambda\mathbf{u}\right\}.$$

Using Theorem 3.3, we can transform the $3n \times 3n$ GEP (3.1) to the $2n \times 2n$ SEP

$$K(\omega)^{-1}\mathbf{u} = \beta\mathbf{u}, \tag{3.6}$$

where

$$K(\omega) = \Lambda^{1/2}Q^*\widetilde{B}(\omega)^{-1}Q\Lambda^{1/2}, \tag{3.7}$$

and the GEP (3.1) and the SEP (3.6) have the same nonzero eigenvalues. The SEP can be solved by the eigensolver *without* being affected by zero eigenvalues.

3.2. Computing $\beta'(\omega)$**.** To evaluate the derivative $\beta'(\omega)$ in (3.3) we can use the following method. Let $\mathbf{u}(\omega)$ and $\mathbf{v}(\omega)$ with $\mathbf{v}(\omega)^*\mathbf{u}(\omega) = 1$ be the right and the left eigenvectors of $K(\omega)^{-1}$, respectively, corresponding to the eigenvalue $\beta(\omega)$, i.e., $K(\omega)^{-1}\mathbf{u}(\omega) = \beta(\omega)\mathbf{u}(\omega)$ and $\mathbf{v}(\omega)^*K(\omega)^{-1} = \beta(\omega)\mathbf{v}(\omega)^*$. Then

$$\beta(\omega) = \mathbf{v}(\omega)^*K(\omega)^{-1}\mathbf{u}(\omega) \tag{3.8}$$

and

$$(\mathbf{v}(\omega)^*)'\mathbf{u}(\omega) + \mathbf{v}(\omega)^*\mathbf{u}(\omega)' = 0. \tag{3.9}$$

Using (3.8) and (3.9), and the fact that $(K(\omega)^{-1})' = -K(\omega)^{-1}K(\omega)'K(\omega)^{-1}$, we obtain

$$\begin{aligned}
\beta'(\omega) &= \mathbf{v}(\omega)^* \left(K(\omega)^{-1}\right)' \mathbf{u}(\omega) + (\mathbf{v}(\omega)^*)'K(\omega)^{-1}\mathbf{u}(\omega) + \mathbf{v}(\omega)^*K(\omega)^{-1}\mathbf{u}(\omega)' \\
&= -\mathbf{v}(\omega)^*K(\omega)^{-1}K(\omega)'K(\omega)^{-1}\mathbf{u}(\omega) + \beta(\omega)(\mathbf{v}(\omega)^*)'\mathbf{u}(\omega) + \beta(\omega)\mathbf{v}(\omega)^*\mathbf{u}(\omega)' \\
&= -\mathbf{v}(\omega)^*K(\omega)^{-1}K(\omega)'K(\omega)^{-1}\mathbf{u}(\omega) \\
&= -\beta(\omega)^2\mathbf{v}(\omega)^*K(\omega)'\mathbf{u}(\omega) \\
&= -\beta(\omega)^2\mathbf{v}(\omega)^*\Lambda^{1/2}Q^* \left[\widetilde{B}(\omega)^{-1}\right]' Q\Lambda^{1/2}\mathbf{u}(\omega) \\
&= \beta(\omega)^2\mathbf{v}(\omega)^*\Lambda^{1/2}Q^*\widetilde{B}(\omega)^{-1}\widetilde{B}(\omega)'\widetilde{B}(\omega)^{-1}Q\Lambda^{1/2}\mathbf{u}(\omega). \qquad (3.10)
\end{aligned}$$

Algorithm 2 Newton-type method for computing eigenvalues of $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$.

Input: Coefficient matrices A (Hermitian), $\widetilde{B}(\omega)$, an initial value ω_0 and a stopping tolerance tol.

Output: An eigenvalue/eigenvector pair $(\mu_d, \mathbf{x}_d)$.

1: Set $k = 0$.
2: **repeat**
3: Compute the eigenvalue β_k^{-1} with the smallest positive real part and the associated eigenvector $\mathbf{u}_k$ of

$$\beta^{-1}\mathbf{u} = K(\omega_k)\mathbf{u} \equiv (\Lambda^{1/2}Q^*\widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2})\mathbf{u} \qquad (3.11)$$

by JD or SIRA method (see Section 4 for details);
4: Compute the left eigenvector $\mathbf{v}_k$ of (3.11) corresponding to β_k;
5: Compute $\beta'(\omega_k)$ by

$$\beta'(\omega_k) = \beta_k^2\mathbf{v}_k^*\Lambda^{1/2}Q^*\widetilde{B}(\omega_k)^{-1}\widetilde{B}(\omega_k)'\widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2}\mathbf{u}_k;$$

6: Compute ω_{k+1} by

$$\omega_{k+1} = \omega_k - \left(\beta'(\omega_k) + \omega_k^{-2}\right)^{-1}\left(\beta_k - \omega_k^{-1}\right);$$

7: Set $k = k + 1$;
8: **until** $|\omega_k - \omega_{k-1}| < tol$.
9: Set $\mu_d = \omega_k$;
10: Compute the eigenvector $\mathbf{x}_d = \widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2}\mathbf{u}_k$.

We summarize the Newton-type method in Algorithm 2. For the calculation of $\beta'(\omega)$ in (3.10) and for solving SEP (3.6), it is required to compute $Q^*\tilde{\mathbf{p}}$, $Q\tilde{\mathbf{q}}$, and $\widetilde{B}(\omega)^{-1}\mathbf{d}$ for given vectors $\tilde{\mathbf{p}}$, $\tilde{\mathbf{q}}$ and $\mathbf{d}$. For computing $Q^*\tilde{\mathbf{p}}$ and $Q\tilde{\mathbf{q}}$, the matrix Q itself does not need to be formed explicitly because the matrix-vector products $T^*\mathbf{p}$ and $T\mathbf{q}$ can be evaluated by the fast Fourier transform efficiently (see Algorithms 5 and 6 in Appendix for details). On the other hand, if $\widetilde{B}(\omega)$ is as in (2.12), then $\widetilde{B}(\omega)$ can be represented as

$$\widetilde{B}(\omega) = \omega B(\omega) + Y(\omega)X^*,$$

where

$$Y(\omega) = (A - \omega^2 B(\omega))XD(\omega).$$

Using the Sherman-Morrison-Woodbury formula [14] we get

$$\widetilde{B}(\omega)^{-1} = \omega^{-1}\left\{I - B(\omega)^{-1}Y(\omega)\left(\omega I + X^* B(\omega)^{-1}Y(\omega)\right)^{-1}X^*\right\}B(\omega)^{-1}. \quad (3.12)$$

By (3.12), to compute $\widetilde{B}(\omega)^{-1}\mathbf{d}$, it is necessary to evaluate $B(\omega)^{-1}\mathbf{d}$ and $B(\omega)^{-1}Y(\omega)$. Because $B(\omega)$ is diagonal and $Y(\omega)$ is low rank, $\widetilde{B}(\omega)^{-1}\mathbf{d}$ can be obtained at low cost.

3.3. Enhanced Newton-type method. It is well-known that the convergence of Newton's method is heavily dependent on the choice of the initial value. In particular, when the target roots of the nonlinear equation (3.2) are clustered, i.e., the NLEVP (2.4) has clustered eigenvalues, then the convergence is very sensitive to the choice of the initial value ω_0. Hence, it is very important to provide a good initial value to guarantee convergence. There are two problems in the choice of the initial value ω_0: (i) how to detect ω_0 which is good enough (i.e., for the case of well-separated eigenvalues); (ii) how to provide a new better initial value when ω_0 is not good enough (i.e., for the case of clustered eigenvalues).

In the kth Newton step, we need to solve the SEP (3.11). If the eigenvalue problem $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$ has clustered eigenvalues which are of interest, then the convergence for solving (3.11) becomes very slow for a randomly chosen ω_0. Based on this observation, we propose in Algorithm 3 a heuristic strategy to tackle problems (i) and (ii) above. If the desired eigenvalue of (3.11) converges in a suitable iteration number m, it means that the provided initial value ω_0 is good enough. Otherwise, we switch to the approximate computation of a new approximate eigenpair $(\omega_0, \mathbf{x}_0)$ of NLEVP (2.4) by the nonlinear eigensolver, e.g. the nonlinear Arnoldi method (see the `if-endif` block in Lines 5-10 of Algorithm 3) and re-solve (3.11) with this new approximate eigenvalue ω_0. We repeat this process until $\{\omega_k\}$ converges (see the `while-endwhile` block in Lines 3-11 of Algorithm 3).

After proposing the Newton-type method it remains to describe eigenvalue solvers used in Algorithm 3. This is done in the next section.

4. Eigenvalue Solvers. In this section, we present eigenvalue algorithms for solving the SEP (3.11) in Line 3 of Algorithm 2 and (3.13) in Line 4 of Algorithm 3. Recall from Section 3 that these SEPs are linearized and deflated variants of the original nonlinear eigenproblem $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$. We shall also briefly discuss the nonlinear Arnoldi method (NAr) [48] and the nonlinear Jacobi-Davidson method (NJD) [49] as competitors for solving $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$ directly.

4.1. Jacobi-Davidson Method for the SEP (3.11)**.** The Jacobi-Davidson method (JD) [43] is an inexact eigenvalue solver for solving the SEP. In each iteration of JD, the correction equation

$$(I - \mathbf{u}\mathbf{u}^*)\,(K(\omega_k) - \theta I)\,(I - \mathbf{u}\mathbf{u}^*)\,\mathbf{t} = -\mathbf{r}, \quad \mathbf{t}\perp\mathbf{u} \quad (4.1)$$

is solved approximately by an iterative solver, where $K(\omega_k)$ is defined as in (3.11), $(\theta, \mathbf{u})$ is the Ritz pair of $K(\omega_k)$ and $\mathbf{r} = (K(\omega_k) - \theta I)\mathbf{u}$. Here $\mathbf{t}\perp\mathbf{u}$ means that $\mathbf{t}$ is orthogonal to $\mathbf{u}$. In each iteration for solving (4.1) with a giving ω_k, we need to solve a linear system of the form

$$M_p\mathbf{z} = \mathbf{d}, \quad \mathbf{z}\perp\mathbf{u}, \quad (4.2)$$

Algorithm 3 Enhanced Newton-type method for computing smallest positive real part eigenvalues of $A\mathbf{x} = \omega\widetilde{B}(\omega)\mathbf{x}$.

Input: Coefficient matrices A (Hermitian), $\widetilde{B}(\omega)$, initial value ω_0, shift value σ, maximal iteration number m, the stopping tolerance τ_0, τ_a and tol.

Output: The target eigenvalue/eigenvector pairs $(\mu_d, \mathbf{x}_d)$.

1: Set $k = 0$.
2: **repeat**
3: **while** ($\|\mathbf{r}_h\| \geq \tau_k$) **do**
4: Compute the eigenvalue β_k^{-1} with the smallest positive real part, the associated eigenvector $\mathbf{u}_k$ of

$$\beta^{-1}\mathbf{u} = (\Lambda^{1/2}Q^*\widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2})\mathbf{u} \tag{3.13}$$

and the corresponding residual vector $\mathbf{r}_h$ by JD or SIRA method with maximal iteration number m and the stopping tolerance τ_k;
5: **if** ($\|\mathbf{r}_h\| \geq \tau_k$) **then**
6: Use $\mathbf{u}_k$ to compute an approximate eigenvector $\mathbf{x}_0$ of NLEVP (2.4) (refer to (5.6)).
7: Use the nonlinear Arnoldi method (NAr) with initial vector $\mathbf{x}_0$ and suitable stopping tolerance τ_a to compute the approximate eigenvalue/eigenvector pair $(\omega_a, \mathbf{x}_a)$ of the NLEVP (2.4), where ω_a is the closest eigenvalue to σ.
8: Use $\mathbf{x}_a$ to compute an approximate eigenvector $\mathbf{u}_0$ of (3.13) (refer to (5.8) and (5.9)).
9: Set $\omega_k = \omega_a$. % Use $(\omega_k, \mathbf{u}_0)$ as the new initial data to re-solve $\beta^{-1}\mathbf{u} = K(\omega_k)\mathbf{u}$.
10: **end if**
11: **end while**
12: Compute the left eigenvector $\mathbf{v}_k$ of (3.13) corresponding to β_k;
13: Compute $\beta'(\omega_k)$ via

$$\beta'(\omega_k) = \beta_k^2\mathbf{v}_k^*\Lambda^{1/2}Q^*\widetilde{B}(\omega_k)^{-1}\widetilde{B}(\omega_k)'\widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2}\mathbf{u}_k;$$

14: Compute ω_{k+1} by

$$\omega_{k+1} = \omega_k - \left(\beta'(\omega_k) + \omega_k^{-2}\right)^{-1}\left(\beta_k - \omega_k^{-1}\right);$$

15: Set $k = k + 1$ and determine stopping tolerance τ_k;
16: **until** $|\omega_k - \omega_{k-1}| < tol$.
17: Set $\mu_d = \omega_k$ and compute the eigenvector $\mathbf{x}_d = \widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2}\mathbf{u}_k$.

where $\mathbf{d}$ is a given vector and

$$M_p \equiv (I - \mathbf{u}\mathbf{u}^*)\, M_K\, (I - \mathbf{u}\mathbf{u}^*)\,,$$

with M_K being the preconditioner of $K(\omega_k) - \theta I$.

For the deflated NLEVP (2.8), one has from (3.7) and (3.12) that

$$K(\omega_k) - \theta I = \left(\Lambda^{1/2}Q^*(\omega_k^{-1}B(\omega_k)^{-1})Q\Lambda^{1/2} - \theta I\right) - U(\omega_k)\Psi(\omega_k)^{-1}V(\omega_k)^*,$$

where

$$
\begin{aligned}
U(\omega_k) &= \omega_k^{-1}\Lambda^{1/2}Q^*B(\omega_k)^{-1}(A-\omega_k^2B(\omega_k))X,\\
V(\omega_k) &= \left[\omega_k^{-1}X^*B(\omega_k)^{-1}Q\Lambda^{1/2}\right]^*,\\
\Psi(\omega_k) &= D(\omega_k)^{-1}+\omega_k^{-1}X^*B(\omega_k)^{-1}(A-\omega_k^2B(\omega_k))X.
\end{aligned}
$$

Therefore, we take as preconditioner

$$
\begin{aligned}
M_K &= \left(\Lambda^{1/2}Q^*(\alpha_{a,k}I)Q\Lambda^{1/2}-\theta I\right) - U(\omega_k)\Psi(\omega_k)^{-1}V(\omega_k)^*\\
&:= \Omega_k - U(\omega_k)\Psi(\omega_k)^{-1}V(\omega_k)^*,
\end{aligned}
\tag{4.3}
$$

where $\alpha_{a,k}$ is the arithmetic average of the diagonal elements of $\omega_k^{-1}B(\omega_k)^{-1}$ and

$$
\Omega_k = \Lambda^{1/2}Q^*(\alpha_{a,k}I)Q\Lambda^{1/2}-\theta I = \alpha_{a,k}\Lambda-\theta I
$$

is a complex diagonal matrix. By the Sherman-Morrison-Woodbury formula, we get

$$
M_K^{-1} = \Omega_k^{-1}\left\{I+U(\omega_k)\left(\Psi(\omega_k)-V(\omega_k)^*\Omega_k^{-1}U(\omega_k)\right)^{-1}V(\omega_k)^*\Omega_k^{-1}\right\} \tag{4.4}
$$

and the linear system (4.2) can be easily solved via

$$
\mathbf{z} = M_K^{-1}\mathbf{d}+\eta M_K^{-1}\mathbf{u} \quad \text{with} \quad \eta = -\frac{\mathbf{u}^*M_K^{-1}\mathbf{d}}{\mathbf{u}^*M_K^{-1}\mathbf{u}}.
$$

4.2. Shift-Invert Residual Arnoldi Method for the SEP (3.11)**.** The shift-and-invert residual Arnoldi method (SIRA) [19, 30, 31] is another inexact eigenvalue solver. In each iteration of SIRA, the linear residual system

$$
(K(\omega_k)-\sigma I)\,\mathbf{t} = \mathbf{r} \tag{4.5}
$$

is solved approximately by an iterative solver, where $\mathbf{r} = K(\omega_k)\mathbf{u}-\theta\mathbf{u}$ is the residual vector and σ is a fixed shift value. From (3.12), we have

$$
\Lambda^{1/2}Q^*\widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2} = \omega_k^{-1}\Lambda^{1/2}Q^*B(\omega_k)^{-1}Q\Lambda^{1/2} - U(\omega_k)\Psi(\omega_k)^{-1}V(\omega_k)^*.
$$

Using the construction of the preconditioner M_K in (4.3), we take

$$
M_S = (\alpha_{a,k}\Lambda-\sigma I) - U(\omega_k)\Psi(\omega_k)^{-1}V(\omega_k)^*
$$

as a preconditioner and rewrite the linear system (4.5) as

$$
A_M\mathbf{t} = M_S^{-1}\mathbf{r}, \tag{4.6}
$$

where

$$
\begin{aligned}
A_M &= M_S^{-1}\left[\Lambda^{1/2}Q^*\widetilde{B}(\omega_k)^{-1}Q\Lambda^{1/2}-\sigma I\right]\\
&= M_S^{-1}\left\{M_S+\Lambda^{1/2}Q^*\left[\omega^{-1}B(\omega_k)^{-1}-\alpha_{a,k}I\right]Q\Lambda^{1/2}\right\}\\
&= I+M_S^{-1}\Lambda^{1/2}Q^*\left[\omega_k^{-1}B(\omega_k)^{-1}-\alpha_{a,k}I\right]Q\Lambda^{1/2}.
\end{aligned}
$$

We summarize the JD and SIRA methods for solving the SEP (3.11) in Algorithm 4. Note that, in practice, a restart with search subspace contraction [16] is used in Algorithm 4.

Algorithm 4 JD/SIRA method for solving $K\mathbf{x} = \lambda\mathbf{x}$

Input: matrix K, an initial matrix $\mathbb{V}_1$, real shift σ, tolerance τ and maximal number of iterations maxit. (A restarting scheme is used if it needed.)

Output: The desired eigenvalue/eigenvector pair $(\lambda, \mathbf{x})$ of K where λ is closest to σ and the associated residual vector $\mathbf{r}$.

Set 'solver' = 'JD' or 'SIRA'.
Set $j = 1$ and $\mathbf{r}_0 = \mathbf{e}_1$.
repeat
  Compute $\mathbb{W}_j = K\mathbb{V}_j$ and $\mathbb{M}_j = \mathbb{V}_j^*\mathbb{W}_j$.
  while ($j \le$ maxit and $\|\mathbf{r}_{j-1}\|_2 \ge \tau$) **do**
    Compute the eigenvalue/eigenvector pairs $(\theta_i, \mathbf{s}_i)$ of $\mathbb{M}_j\mathbf{s} = (\mathbb{V}_j^*K\mathbb{V}_j)s = \theta\mathbf{s}$ with $\|\mathbf{s}_i\|_2 = 1$ and $|\theta_1 - \sigma| \le |\theta_2 - \sigma| \le \cdots$.
    Compute $\mathbf{u}_j = \mathbb{V}_j\mathbf{s}_1$ and $\mathbf{r}_j = (K - \theta_1 I)\mathbf{u}_j$.
    if ($\|\mathbf{r}_j\|_2 \ge \tau$) **then**
      if ('solver' = 'SIRA') **then**
        Compute (approximate) solution $\mathbf{t}_j$ for

$$(K - \sigma I)\mathbf{t}_j = \mathbf{r}_j.$$

      else if ('solver' = 'JD') **then**
        Compute (approximate) solution $\mathbf{t}_j \perp \mathbf{u}_j$ for

$$\left(I - \mathbf{u}_j\mathbf{u}_j^*\right)(K - \theta_1 I)\left(I - \mathbf{u}_j\mathbf{u}_j^*\right)\mathbf{t}_j = -\mathbf{r}_j.$$

      end if
      Orthogonalize $\mathbf{t}_j$ against $\mathbb{V}_j$; set $\mathbf{v}_{j+1} = \mathbf{t}_j/\|\mathbf{t}_j\|$.
      Compute $\mathbf{w}_{j+1} = K\mathbf{v}_{j+1}$, $\mathbb{M}_{j+1} = \begin{bmatrix} \mathbb{M}_j & \mathbb{V}_j^*\mathbf{w}_{j+1} \\ \mathbf{v}_{j+1}^*\mathbb{W}_j & \mathbf{v}_{j+1}^*\mathbf{w}_{j+1} \end{bmatrix}$.
      Expand $\mathbb{V}_{j+1} = [\mathbb{V}_j, \mathbf{v}_{j+1}]$ and $\mathbb{W}_{j+1} = [\mathbb{W}_j, \mathbf{w}_{j+1}]$.
    end if
    Set $j := j + 1$.
  end while
  if ($\|\mathbf{r}_{j-1}\|_2 \ge \tau$) **then**
    Set $\mathbb{V}_m = \mathbb{V}_{j-1}\begin{bmatrix} s_1 & \cdots & s_m \end{bmatrix}$, $\mathbf{r}_{m-1} = \mathbf{r}_{j-1}$ and $j = m$.
  end if
until desired eigenvalue/eigenvector is convergent
Set $\lambda = \theta_1$, $\mathbf{x} = \mathbf{u}_j$, $\mathbf{r} = \mathbf{r}_j$ and $\mathbb{V}_1 = \mathbb{V}_j[\mathbf{s}_1, \cdots, \mathbf{s}_p]$.

4.3. Nonlinear eigensolvers for the NLEVP (2.4)**.** The NLEVP (2.4) can be solved by the nonlinear Arnoldi method (NAr) or nonlinear Jacobi-Davidson method (NJD) directly. For a given search subspace V, let $(\tilde{\omega}, \tilde{\mathbf{z}})$ be an eigenvalue/eigenvector pair of the projected problem $V^*(A - \omega^2 B(\omega))V\mathbf{z} = 0$ and let $\tilde{\mathbf{x}} = V\tilde{\mathbf{z}}$ be the associated Ritz vector, i.e. the eigenvector lifted to the large space. The new search direction $\mathbf{v}$ in the NAr and NJD methods is chosen as

$$\mathbf{v} = \left(A - \sigma^2 B(\sigma)\right)^{-1}\mathbf{r} \tag{4.7}$$

and

$$\left(I - \frac{(2\tilde{\omega}B(\tilde{\omega}) + \tilde{\omega}^2 B(\tilde{\omega})')\tilde{\mathbf{x}}\tilde{\mathbf{x}}^*}{\tilde{\mathbf{x}}^*(2\tilde{\omega}B(\tilde{\omega}) + \tilde{\omega}^2 B(\tilde{\omega})')\tilde{\mathbf{x}}}\right)(A - \tilde{\omega}^2 B(\tilde{\omega}))\left(I - \frac{\tilde{\mathbf{x}}\tilde{\mathbf{x}}^*}{\tilde{\mathbf{x}}^*\tilde{\mathbf{x}}}\right)\mathbf{v} = -\mathbf{r}, \quad \mathbf{v} \perp \tilde{\mathbf{x}}, \quad (4.8)$$

respectively, where $\mathbf{r} = (A - \tilde{\omega}^2 B(\tilde{\omega}))\tilde{\mathbf{x}}$ is the residual vector for $(\tilde{\omega}, \tilde{\mathbf{x}})$ and σ is a given shift value. After re-orthogonalizing $\mathbf{v}$ against V, the vector is appended to V and one repeats this process until $(\tilde{\omega}, \tilde{\mathbf{x}})$ converges to the desired eigenvalue/eigenvector pair.

The major cost of the NAr (NJD) method arises in the solving (4.7) ((4.8)). This cost can be significantly reduced by using a technique suggested in [19]. Since $B(\sigma)$ in (2.4) is diagonal, we employ a preconditioner

$$M = A - \sigma^2 \alpha_\sigma I$$

and

$$M_J = \left(I - \frac{(2\tilde{\omega}B(\tilde{\omega}) + \tilde{\omega}^2 B(\tilde{\omega})')\tilde{\mathbf{x}}\tilde{\mathbf{x}}^*}{\tilde{\mathbf{x}}^*(2\tilde{\omega}B(\tilde{\omega}) + \tilde{\omega}^2 B(\tilde{\omega})')\tilde{\mathbf{x}}}\right)(A - \tilde{\omega}^2 \alpha_\sigma I)\left(I - \frac{\tilde{\mathbf{x}}\tilde{\mathbf{x}}^*}{\tilde{\mathbf{x}}^*\tilde{\mathbf{x}}}\right)$$

for solving (4.7) and (4.8), respectively, where α_σ is the arithmetic average of the diagonal elements of $B(\sigma)$. The associated linear system with coefficient matrix $(A - \gamma I)$ can be efficiently solved by applying the spectral decompositions of the matrices C_ℓ, $\ell = 1, 2, 3$, see [19] or Appendix A.3. Furthermore, we can apply the left-preconditioning M^{-1} to (4.7) and obtain the system

$$\left[I + \sigma^2 M^{-1}(\alpha_\sigma I - B(\sigma))\right]\mathbf{v} = M^{-1}\mathbf{r},$$

which only requires to compute $\mathbf{d} + \sigma^2 M^{-1}(\alpha_\sigma I - B(\sigma))\mathbf{d}$ in each iteration for a given vector $\mathbf{d}$. There is no need to compute a matrix-vector multiplication with A.

5. Practical implementation. Suppose that we want to find ℓ smallest positive real part eigenvalues $\mu_1, \ldots, \mu_\ell$ of the NLEVP (2.4). In previous sections, we propose a Newton-type method with non-equivalence deflation to sequentially compute the desired eigenvalues μ_d, $d = 1, \ldots, \ell$. Now, we briefly summarize it to further clarify the relation among the proposed algorithms for the computation of the dth eigenvalue/eigenvector of (2.4).

In Algorithm 1, sequentially applying the non-equivalence deflation, we formulate the deflated eigenvalue problems as

$$A\mathbf{x} = \omega \widetilde{B}(\omega)\mathbf{x}, \quad (5.1)$$

which is of the same form as the original eigenvalue problem. The Newton-type method in Algorithm 2 is then applied to solve (5.1) by successively solving the GEP

$$\beta A\mathbf{x} = \widetilde{B}(\omega_k^{(d)})\mathbf{x} \quad \text{for } k = 0, 1, 2, \ldots. \quad (5.2)$$

Each $3n \times 3n$ GEP in (5.2) is reduced into an $2n \times 2n$ SEP

$$\mathbf{u} = \beta\left(\Lambda^{1/2} Q^* \widetilde{B}(\omega_k^{(d)})^{-1} Q \Lambda^{1/2}\right)\mathbf{u} \quad (5.3)$$

by deflating out all zero eigenvalues of A. When the desired eigenvalues of (5.1) are clustered, an enhanced Newton-type method in Algorithm 3 by using the nonlinear

Arnoldi method for finding a good initial guess. The sequence of SEPs are then solved by JD or SIRA in Algorithm 4.

In above, we develop a sequence of Algorithms 1-4 for solving the NLEVP (2.4). In this section, we focus on the practical implementations including (i) the setting of the initial values and vectors, and (ii) the stopping tolerance of the associated solvers. These practical issues can significantly improve the efficiency and robustness of our proposed methods.

5.1. Initial value ω_0 and initial vector for Algorithms 2 and 4. As above, for finding the d-th eigenpair in Line 3 of Algorithm 1, a sequence of SEPs (5.3) arising from Line 3 of Algorithm 2 or Line 4 of Algorithm 3 is solved by Algorithm 4. In solving (5.3) with a given $\omega_k^{(d)}$, usually more than one Ritz pair is chosen at each jth iteration. Suppose that $(\beta_{1,j}^{(d)}, \mathbf{u}_{1,j}^{(d)})$, ..., $(\beta_{m_1,j}^{(d)}, \mathbf{u}_{m_1,j}^{(d)})$ with $|(\beta_{1,j}^{(d)})^{-1} - \sigma| \le \cdots \le |(\beta_{m_1,j}^{(d)})^{-1} - \sigma|$ in Line 6 of Algorithm 4 are chosen. When $\{\beta_{1,j}^{(d)}\}$ is converging to the desired eigenvalue of SEP (5.3) at the j_kth iteration, $\beta_{1,j_k}^{(d)}$ is applied to compute the next $\omega_{k+1}^{(d)}$ and the subspace $\mathrm{span}\{\mathbf{u}_{1,j_k}^{(d)}, \ldots, \mathbf{u}_{m_1,j_k}^{(d)}\}$ is used as the initial subspace $\mathbb{V}_1$ in Algorithm 4 for solving (5.3) with new $\omega_{k+1}^{(d)}$.

After d steps in Algorithm 1, the d eigenvalues/eigenvectors $(\mu_i, \mathbf{x}_i)$ for $i = 1, \ldots, d$ of the original problem $A\mathbf{x} = \omega^2 B(\omega)\mathbf{x}$ have been computed, and we now start the $(d+1)$th step in Algorithm 1, invoking Algorithm 2 for solving the new deflated NLEVP

$$\widetilde{F}(\omega)\tilde{\mathbf{x}} = F(\omega)\prod_{i=1}^{d}\left(I - \frac{\omega}{\omega - \mu_i}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^*\right)\tilde{\mathbf{x}}, \tag{5.4}$$

where $\{\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_d\}$ is an orthonormal basis for the subspace $\mathrm{span}\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$. Applying the Newton-type method to solve (5.4), we need to give an initial value $\omega_0^{(d+1)}$. In here, we take $\omega_0^{(d+1)} = (\beta_{2,j_{k_d}}^{(d)})^{-1}$, the Ritz value obtained in Algorithm 4 that is subsequent to the Ritz value $(\beta_{1,j_{k_d}}^{(d)})^{-1}$ that has just converged to $\omega_{k_d}^{(d)} \approx \mu_d$. For the given initial $\omega_0^{(d+1)}$, it needs to solve the SEP

$$\mathbf{u} = \beta\left(\Lambda^{1/2}Q^*\widetilde{B}(\omega_0^{(d+1)})^{-1}Q\Lambda^{1/2}\right)\mathbf{u} \tag{5.5}$$

by Algorithm 4 with a given initial subspace. Now, we propose a method to construct the initial subspace from $\{\mathbf{u}_{2,j_{k_d}}^{(d)}, \ldots, \mathbf{u}_{m_1,j_{k_d}}^{(d)}\}$ as follows.

For convenience, we use $\mathbf{u}_{2,j_{k_d}}^{(d)}$ as an example. Because $\mathbf{u}_{2,j_{k_d}}^{(d)}$ is a Ritz vector of

$$\mathbf{u} = \beta\left(\Lambda^{1/2}Q^*\widetilde{B}(\omega_{k_d}^{(d)})^{-1}Q\Lambda^{1/2}\right)\mathbf{u},$$

from Theorem 3.3, we define

$$\hat{\mathbf{x}}_0^{(d+1)} \equiv \widetilde{B}(\omega_{k_d}^{(d)})^{-1}Q\Lambda^{1/2}\mathbf{u}_{2,j_{k_d}}^{(d)} = \widetilde{B}(\mu_d)^{-1}Q\Lambda^{1/2}\mathbf{u}_{2,j_{k_d}}^{(d)}$$

and use it to approximate the eigenvector of $A\mathbf{x} = \omega\widetilde{B}(\mu_d)\mathbf{x}$. From Theorem 2.4, we set

$$\mathbf{x}_0^{(d+1)} := \prod_{i=1}^{d-1}\left(I - \frac{\mu_d}{\mu_d - \mu_i}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^*\right)\hat{\mathbf{x}}_0^{(d+1)} \tag{5.6}$$

and take the pair $(\omega_0^{(d+1)}, \mathbf{x}_0^{(d+1)})$ as an approximate eigenvalue/eigenvector pair of NLEVP (2.4). The associated residual vector $\mathbf{r}_2$ is equal to

$$\begin{aligned}\mathbf{r}_2 &= F(\omega_0^{(d+1)})\mathbf{x}_0^{(d+1)} = \widetilde{F}(\omega_0^{(d+1)})\prod_{i=1}^{d}\left(I - \frac{\omega_0^{(d+1)}}{\omega_0^{(d+1)} - \mu_i}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^*\right)^{-1}\mathbf{x}_0^{(d+1)} \\ &:= \widetilde{F}(\omega_0^{(d+1)})\tilde{\mathbf{x}}_0^{(d+1)}, \end{aligned} \tag{5.7}$$

where

$$\begin{aligned}\tilde{\mathbf{x}}_0^{(d+1)} &= \prod_{i=1}^{d}\left(I - \frac{\omega_0^{(d+1)}}{\omega_0^{(d+1)} - \mu_i}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^*\right)^{-1}\mathbf{x}_0^{(d+1)} = \prod_{i=1}^{d}\left(I - \frac{\omega_0^{(d+1)}}{\mu_i}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^*\right)\mathbf{x}_0^{(d+1)} \\ &= \mathbf{x}_0^{(d+1)} - \sum_{i=1}^{d}\frac{\omega_0^{(d+1)}}{\mu_i}\left(\tilde{\mathbf{x}}_i^*\mathbf{x}_0^{(d+1)}\right)\tilde{\mathbf{x}}_i. \end{aligned} \tag{5.8}$$

Therefore, we may take $(\omega_0^{(d+1)}, \tilde{\mathbf{x}}_0^{(d+1)})$ as an approximate eigenvalue/eigenvector pair for (5.4) and use

$$\mathbf{u}_0^{(d+1)} = \Lambda^{-1/2}Q^*\widetilde{B}(\omega_0^{(d+1)})\tilde{\mathbf{x}}_0^{(d+1)} \tag{5.9}$$

as an initial vector for solving (5.5). The numerical results in Figure 6.6 (see Section 6) show that $\mathbf{u}_0^{(d+1)}$ is a good initial vector if $\omega_0^{(d+1)}$ is good enough.

5.2. Stopping tolerances. The sequence $\{\omega_k^{(d)}\}$ of index k is constructed by a sequence of eigenvalues of SEPs (5.3). In the Newton-type method, we need the accurate eigenvalue/eigenvector pair of (5.3) for each $\omega_k^{(d)}$. Actually, the concept of an inexact Newton method [44] can be applied to adaptively control the accuracy of the eigenvalue/eigenvector pairs to reduce the computational cost. This concept gives us that $\omega_k^{(d)}$ is closer to μ_d, the more accurate solution of (5.3) is needed. Based on this concept, we propose a heuristic strategy for the construction of the stopping tolerance τ_k for $k > 0$, in solving (5.3) of Algorithm 3 with

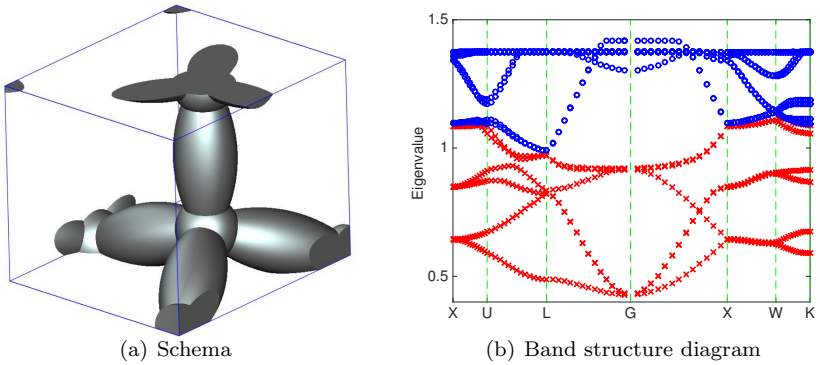
$$\begin{aligned}\tau_k = \max\Bigg\{&\max\left\{5\cdot 10^{-12}, \frac{10^4\mathrm{eps}}{2\sqrt{\delta_x^{-2} + \delta_y^{-2} + \delta_z^{-2}}}\right\}, \\ &\min\left\{5\cdot 10^{-4}, 0.1\cdot|\omega_k^{(d)} - \omega_{k-1}^{(d)}|^2\right\}\Bigg\}, \end{aligned} \tag{5.10}$$

where δ_x, δ_y, and δ_z denote the grid lengths in x, y, and z axial direction, respectively. The stopping tolerance τ_0 for $\omega_0^{(d)}$ is set to be

$$\tau_0 = \begin{cases} 10^{-3}, & \text{for } d = 1; \\ \min(10^{-3}, 0.1\cdot\|\mathbf{r}_2\|_2), & \text{for } d > 1, \end{cases} \tag{5.11}$$

where $\mathbf{r}_2$ is the residual vector in (5.7).

If a maximal number of JD/SIRA iterations is achieved but the Ritz vector does not converge, then NAr is applied to obtain a better initial data. In order to get useful initial data, we determine the stopping tolerance of NAr according the concept that the closer to the eigenvalue/eigenvector pair of (5.3) for the Ritz value/vector pair



(a) Schema (b) Band structure diagram

FIG. 5.1. *(a) A schematic view of a dispersive metallic photonic crystal structure with a FCC lattice within a single primitive cell. (b) The computed band structure diagram of the Drude model with matrix dimension* $3 \cdot 48^3 = 331,776$. *There are* 16 *eigenvalues in the diagram. The six smallest real part nonzero eigenvalues* $\mu_1, \ldots, \mu_6$ *are denoted by (red)* $\times$.

by JD/SIRA, the smaller the stopping tolerance for NAr is needed. According this concept, the stopping tolerance of the residual norm for NAr at Line 7 in Algorithm 3 is taken as

$$\tau_a = \max\left\{\min\left\{\|\mathbf{r}_h\|, 10^{-3}\right\}, 5 \times 10^{-8}\right\}, \tag{5.12}$$

where $\mathbf{r}_h$ is the residual vector of the approximate eigenvalue/eigenvector of (5.3) at Line 4 in Algorithm 3. The stopping tolerance for solving the linear system (4.7) in NAr is set to be 10^{-4}. The heuristic strategies in [16] are applied to determine the maximal iteration number for solving the correction equation in JD and NJD. The associated stopping tolerance for solving the correction equation is set to 10^{-3}. As the results in [19] suggested, we set the stopping tolerance by 5×10^{-4} for (4.5) in SIRA.

6. Numerical Results. To study the numerical performance of the described methods for solving the (2.4) arising in the 3D dispersive metallic PCs, we consider the setup described in [6, 17, 18]. The lattice in Figure 5.1(a) consists of spheres with a connecting spheroid. The radius r of the spheres is $r = 0.08a$ and the connecting spheroid has a minor axis length $s = 0.06a$ with $a = 2\pi$. The perimeter of the irreducible Brillouin zone for the lattice is formed by the corners $X = \frac{2\pi}{a}\Omega[0,1,0]^\top$, $U = \frac{2\pi}{a}\Omega\left[\frac{1}{4},1,\frac{1}{4}\right]^\top$, $L = \frac{2\pi}{a}\Omega\left[\frac{1}{2},\frac{1}{2},\frac{1}{2}\right]^\top$, $G = [0,0,0]^\top$, $W = \frac{2\pi}{a}\Omega\left[\frac{1}{2},1,0\right]^\top$, and $K = \frac{2\pi}{a}\Omega\left[\frac{3}{4},\frac{3}{4},0\right]^\top$, where

$$\Omega = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 & 0 \\ -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{3}} \\ \frac{2}{\sqrt{6}} & -\frac{2}{\sqrt{6}} & \frac{2}{\sqrt{6}} \end{bmatrix}.$$

The permittivity ε_n of the non-dispersive material is set to be 1. Parameters in the Drude and Drude-Lorentz models are $\omega_p = \frac{10\pi}{a}$, $\Gamma_p = \frac{2\pi}{14500}$, $\Omega_1 = \frac{2\pi}{470}$, $\Omega_2 = \frac{2\pi}{325}$, $\Gamma_1 = \frac{2\pi}{1900}$, $\Gamma_2 = \frac{2\pi}{1060}$, $\varepsilon_\infty = 1.54$, $A_1 = 1.27$, $A_2 = 1.1$, $\phi_1 = -\frac{\pi}{4}$ and $\phi_2 = -\frac{\pi}{4}$ [34]. The associated band structure for the Drude model is shown in Figure 5.1(b). The band structure for the Drude-Lorentz model is similar to Figure 5.1(b).

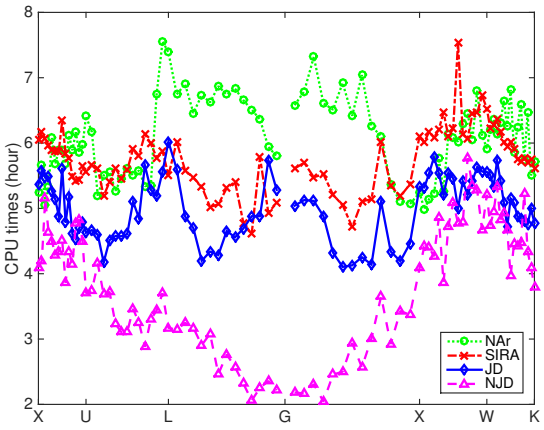


FIG. 6.1. *CPU times for computing the six smallest real part nonzero eigenvalues (and associated eigenvectors) denoted by (red) $\times$ in Figure 5.1(b). The matrix dimension is* $2,654,208$.

All computations in this section are carried out in MATLAB 2013b, and some implementation details are addressed as follows. The MATLAB function `bicgstabl` is used to solve the linear systems in Algorithm 4, NJD and NAr. On the other hand, the MATLAB functions `fft` and `ifft` are applied to compute the products $T^*\mathbf{p}$ and $T\mathbf{q}$, respectively.

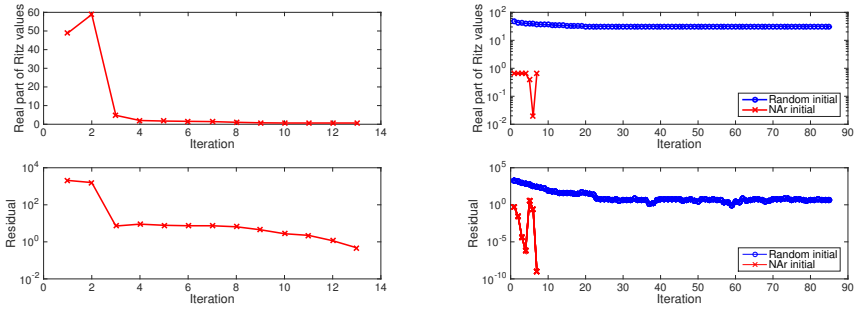
For the hardware configuration, we use a HP DL360p Gen8 workstation that is equipped with two Intel Quad-Core Xeon E5-2643 3.33GHz CPUs, 96 GB of main memory, and the RedHat Enterprise Linux 6 operating system.

6.1. Comparison between Newton-type method and nonlinear eigensolvers. We demonstrate the efficiency of the new Newton-type method, NJD and NAr in computing the six smallest positive real part eigenvalues $\mu_1, \ldots, \mu_6$ of the Drude model (1.2). The maximal dimension of the search subspace V in NJD and NAr is set to 35. When the dimension of V is larger than 35, then we take three Ritz vectors with associated Ritz values that are closest to the shift value as an initial subspace and restart the iteration.

The real parts of $\mu_1, \ldots, \mu_6$ are denoted by red $\times$ in Figure 5.1(b). As shown in the figure, these eigenvalues are well-separated so that they can be computed by the Newton-type method without using NAr to get the initial data. The CPU times for computing these six eigenvalue/eigenvector pairs by

- JD: Algorithm 1 + Algorithm 2 with solving (5.3) by JD
- SIRA: Algorithm 1 + Algorithm 2 with solving (5.3) by SIRA
- NAr: nonlinear Arnoldi method with fixed shift value σ for solving (2.4)
- NJD: nonlinear Jacobi-Davidson method for solving (2.4)

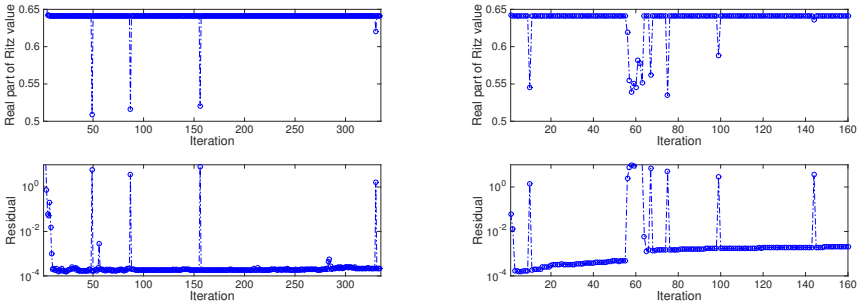
are depicted in Figure 6.1, which shows that NJD obviously outperforms JD, SIRA and NAr. Except for NJD, JD is much better than SIRA and NAr. Note that, in practice, NAr is used to compute the initial vector of NJD for computing the first desired eigenvalue/eigenvector. If a random initial vector is used, then NJD would usually not converge for our benchmark problems as shown in Figure 6.2.



(a) Using NAr to compute an initial vector of NJD

(b) NJD with random initial vector and initial vector produced by NAr in Fig. 6.2(a)

FIG. 6.2. *Convergence behaviour of the NJD with using NAr for the initial vector and random initial vector, respectively, for computing the first desired eigenpair. The matrix dimension is* $2,654,208$.



(a) NAr

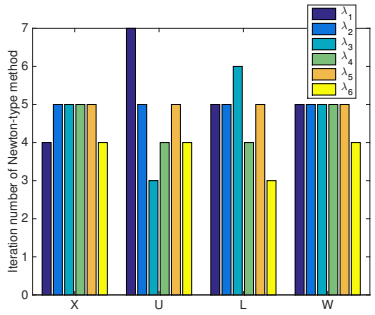
(b) NJD

FIG. 6.3. *Convergence behaviour of the NAr and the NJD for computing the first desired eigenpair* $\mu_1 \approx 0.6407596 - 2.377323 \times 10^{-4}\imath$ *of the Drude-Lorentz model at the wave vector* $\frac{4}{7}X + \frac{3}{7}W$. *The matrix dimension is* $2,654,208$.

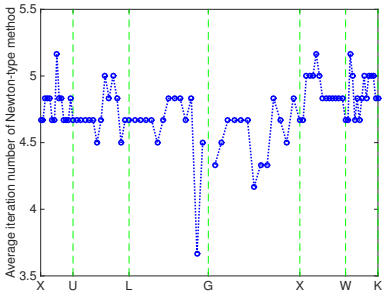
The results in Figure 6.1 show that NAr and NJD can be successfully applied to compute the well-separated six eigenvalues of the Drude model (1.2). However, for the Drude-Lorentz model (1.3) whose nonlinearity is more complicated than Drude model, Figure 6.3 shows that the Ritz values produced by the NAr and the NJD can not converge to the first eigenvalue/eigenvector satisfying the stopping tolerance 10^{-10}.

Note that some of these Ritz values in Figure 6.3 are dragged toward 0.5 during the subspace iteration. This effect of dragging is also reflected in the convergence history of the residual. Figure 6.5 also shows the effect of dragging for the Ritz values. This effect is produced by the zero eigenvalue of multiplicity n. Due to this huge nullity, the Ritz vector associated with the Ritz value in Figure 6.3 or 6.5 easily contains the component of the null vectors. Such component results in dragging the Ritz value away from the desired eigenvalue as shown in Figures 6.3 and 6.5.

6.2. Convergence of the Newton-type method. In this subsection, we illustrate the convergence of the Newton-type method in Algorithm 3 without using NAr for the estimation of the initial values of $\mu_1, \ldots, \mu_6$ discussed in the previous



(a) Number of iterations k of Newton-type method for computing each eigenvalue at wave vectors X, U, L, W and K.



(b) Average number of iterations of Newton-type method for computing six eigenvalues.

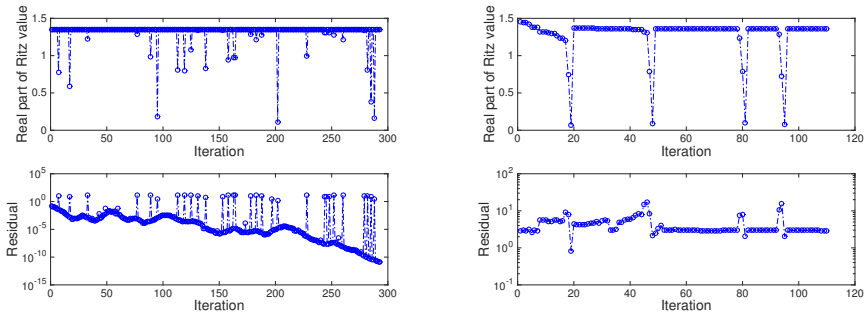
FIG. 6.4. *Number of iterations of Newton-type method with JD eigensolver for computing the six smallest real part nonzero eigenvalues denoted by (red) $\times$ in Figure 5.1(b). The matrix dimension is* $2,654,208$.

	Drude model	Drude-Lorentz model
μ_7	$1.352760915 - 2.15717754 \times 10^{-4}\imath$	$1.326911260 - 2.11350594 \times 10^{-3}\imath$
μ_8	$1.352771023 - 2.15790978 \times 10^{-4}\imath$	$1.326915939 - 2.11375183 \times 10^{-3}\imath$
μ_9	$1.352771589 - 2.15790991 \times 10^{-4}\imath$	$1.326916471 - 2.11375357 \times 10^{-3}\imath$
μ_{10}	$1.352774278 - 2.15790186 \times 10^{-4}\imath$	$1.326919090 - 2.11375510 \times 10^{-3}\imath$
μ_{11}	$1.354710739 - 2.15785421 \times 10^{-4}\imath$	$1.328746727 - 2.11897302 \times 10^{-3}\imath$
μ_{12}	$1.354711852 - 2.15790561 \times 10^{-4}\imath$	$1.328747433 - 2.11899196 \times 10^{-3}\imath$
μ_{13}	$1.354711871 - 2.15790691 \times 10^{-4}\imath$	$1.328747439 - 2.11899260 \times 10^{-3}\imath$
μ_{14}	$1.354711899 - 2.15790684 \times 10^{-4}\imath$	$1.328747467 - 2.11899263 \times 10^{-3}\imath$

TABLE 6.1
Eigenvalues $\mu_7, \ldots, \mu_{14}$ of (2.4) with the wave vector $\frac{3}{7}X$ and the matrix dimension $2,654,208$.

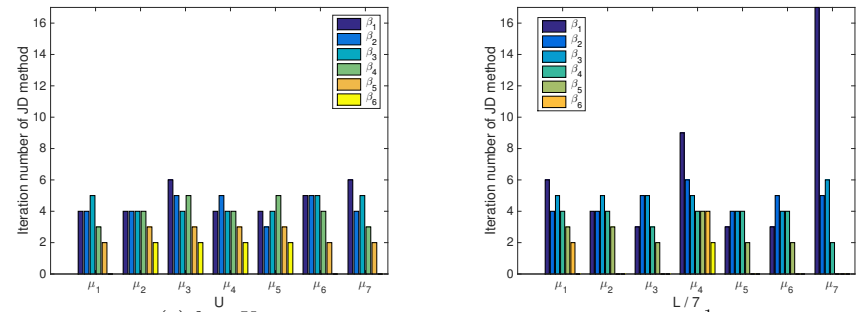
subsection. Using JD to solve the eigenvalue problem (3.11), the number of iterations k for computing each μ_i at wave vectors X, U, L, W and K in the FCC lattice is depicted in Figure 6.4(a). We can see from this figure that only 3 to 7 iterations are needed for computing each eigenvalue. Figure 6.4(b) shows the average number of iterations for computing $\mu_1, \ldots, \mu_6$ with various wave vectors $\mathbf{k}$. The numerical experience indicates that the average ranges from 3.6 to 5.2 for all benchmark problems with the matrix dimension $3 \times 96^3 = 2,654,208$. This convergence behaviour coincides with that of Newton's method for solving general nonlinear equations.

6.3. Clustered eigenvalues. From the band structure diagram in Figure 5.1(b) we see that the eigenvalues are clustered near 1.35 and 1.32 for the Drude and Drude-Lorentz models, respectively. Table 6.1 shows the clustering eigenvalues $\mu_7, \ldots, \mu_{14}$ of (2.4) (with the wave vector $\frac{3}{7}X$) of the Drude and Drude-Lorentz models, respectively. These clustering eigenvalues not only significantly increase the number of iterations for the NAr and lead to the non-convergence of NJD as shown in Figures 6.5, but also lead to a challenge for the Newton-type method: how to detect the clustering eigenvalues. In Figure 6.6, we depict the number of iterations of JD for computing eigenvalue/eigenvector pairs $(\beta_k, \mathbf{u}_k)$ of (3.11) in finding each $\mu_1, \ldots, \mu_7$ for the Drude model. For the specific wave vector U, $\mu_1, \ldots, \mu_7$ are well separated. Figure 6.6(a)



(a) NAr (b) NJD

FIG. 6.5. *Convergence behaviour of the Ritz values (vs number of iterations), produced by NAr and NJD, for computing* μ_7 *(at the wave vector* $\frac{3}{7}X$*) for the Drude model. The matrix dimension is* $2,654,208$.

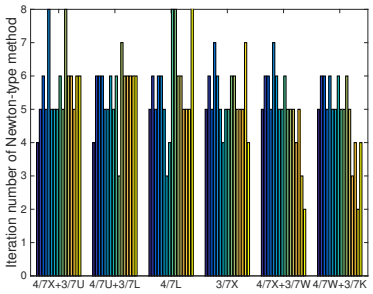


(a) $\mathbf{k} = U$ (b) $\mathbf{k} = \frac{1}{7}L$

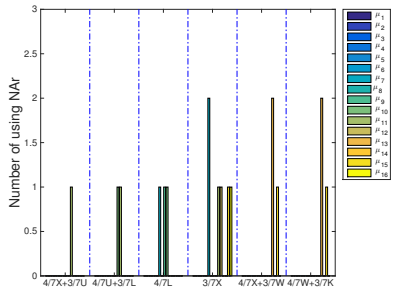
FIG. 6.6. *Number of iterations of JD for computing eigenvalue* β_k *of* (3.11) *in Algorithm 3 at the wave vector* $\mathbf{k}$. *The matrix dimension is* $2,654,208$.

shows that all numbers of iterations of the JD method are less than 7. However, for wave vector $\frac{1}{7}L$, μ_7 is close to the clustering eigenvalues. The results in Figure 6.6(b) indicate that the number of iterations for computing β_1 in μ_7 is 17 which is obviously larger than that for other eigenvalues. This means that the number of iterations of the JD method are a crucial indicator for detecting clustering eigenvalues. In this example, we set the maximal number m of iterations to be 15 in Algorithm 3 and if JD is not convergent within 15 iterations, then the eigenvalues are regarded to be clustering and the NAr is used to provide a good initial data.

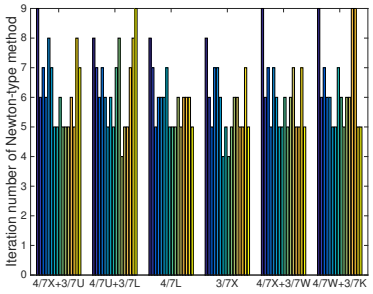
6.4. Efficiency of Algorithm 1 combined with Algorithm 3. Combining Algorithm 1 with the enhanced Newton-type method in Algorithm 3 with $m = 15$, the band structure diagram can be produced as in Figure 5.1(b). We consider six different wave vectors $\frac{4}{7}X + \frac{3}{7}U$, $\frac{4}{7}U + \frac{3}{7}L$, $\frac{4}{7}L$, $\frac{3}{7}X$, $\frac{4}{7}X + \frac{3}{7}W$, and $\frac{4}{7}W + \frac{3}{7}K$, and take $n_1 = n_2 = n_3 = 96$, i.e., the matrix dimension is $2,654,208$. For each wave vector, 16 eigenvalues $\mu_1, \ldots, \mu_{16}$ are computed. For each eigenvalue μ_i, we depict the associated number of iterations k_i of the Newton-type method and the number p_i of using NAr to get the initial data. The values of (k_i, p_i) for the Drude and the Drude-Lorentz models are shown in Figures (6.7(a), 6.7(b)) and (6.7(c), 6.7(d)),



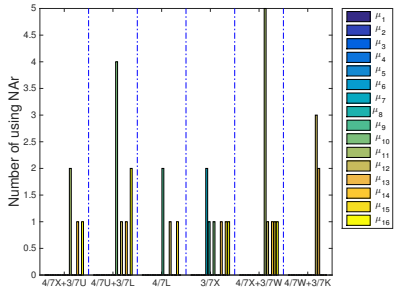
(a) Number of iterations of Newton-type method.



(b) Number of using NAr to get initial data.



(c) Number of iterations of Newton-type method.



(d) Number of using NAr to get initial data.

FIG. 6.7. *Number of iterations of Newton-type method for computing eigenvalues* $\mu_1, \ldots, \mu_{16}$*, and number of steps of the NAr method for computing the initial value and vector for each eigenvalue/eigenvector pair at wave vectors* $\frac{4}{7}X+\frac{3}{7}U$*,* $\frac{4}{7}U+\frac{3}{7}L$*,* $\frac{4}{7}L$*,* $\frac{3}{7}X$*,* $\frac{4}{7}X+\frac{3}{7}W$*, and* $\frac{4}{7}W+\frac{3}{7}K$*. The matrix dimension is* $3 \times 96^3 = 2,654,208$*. (a) and (b) are the results for the Drude model, (c) and (d) are the results for the Drude-Lorentz model.*

respectively. In our proposed method, we not only use the NAr to provide an initial data for the eigenvalue/eigenvector pair, but also use the strategies in Subsection 5.1 to confirm convergence. Therefore, even if the eigenvalues are strongly clustered as shown in Table 6.1, we can still compute the desired eigenvalue/eigenvector pairs within a reasonable (k_i, p_i) as shown in Figure 6.7.

7. Conclusions. Solving the nonlinear eigenvalue problem (NLEVP) arising from Yee's discretization of a three-dimensional dispersive metallic photonic crystal is a computational challenge. We have proposed a Newton-type method to compute one desired eigenvalue/eigenvector pair of the NLEVP at a time. Once the desired eigenvalue is converged, it is then transformed to infinity by the proposed non-equivalence deflation scheme, while all other eigenvalues remain unchanged. The next successive eigenvalue thus becomes the smallest nonzero real part eigenvalue of the transformed NLEVP which is then again solved by the Newton-type method. Furthermore, some heuristic strategies for the determination of initial data and stopping tolerances of the iterative eigenvalue methods are introduced to accelerate the convergence. In order to compute the clustering eigenvalues of the NLEVP, we propose a hybrid method by using the Jacobi-Davidson or the shift-invert residual Arnoldi method to solve the standard eigenvalue problems in the Newton-type method and the NAr to com-

pute the initial data. The numerical results demonstrate that our proposed method is robust and outperforms the NAr and NJD for solving both of well-separated and clustering eigenvalues of the NLEVP for the Drude-Lorentz model.

Appendix.

A.1 The matrices K_i, $i = 1, 2, 3$, in (2.3) are defined as

$$K_1 = \frac{1}{\delta_x}\begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ e^{\imath 2\pi \mathbf{k}\cdot\mathbf{a}_1} & & & -1 \end{bmatrix} \in \mathbb{C}^{n_1\times n_1},$$

$$K_2 = \frac{1}{\delta_y}\begin{bmatrix} -I_{n_1} & I_{n_1} & & \\ & \ddots & \ddots & \\ & & -I_{n_1} & I_{n_1} \\ e^{\imath 2\pi \mathbf{k}\cdot\mathbf{a}_2} J_2 & & & -I_{n_1} \end{bmatrix} \in \mathbb{C}^{(n_1n_2)\times(n_1n_2)},$$

$$K_3 = \frac{1}{\delta_z}\begin{bmatrix} -I_{n_1n_2} & I_{n_1n_2} & & \\ & \ddots & \ddots & \\ & & -I_{n_1n_2} & I_{n_1n_2} \\ e^{\imath 2\pi \mathbf{k}\cdot\mathbf{a}_3} J_3 & & & -I_{n_1n_2} \end{bmatrix} \in \mathbb{C}^{n\times n},$$

with

$$J_2 = \begin{bmatrix} 0 & e^{-\imath 2\pi \mathbf{k}\cdot\mathbf{a}_1} I_{n_1/2} \\ I_{n_1/2} & 0 \end{bmatrix} \in \mathbb{C}^{n_1\times n_1}, \text{ and}$$

$$J_3 = \begin{bmatrix} 0 & e^{-\imath 2\pi \mathbf{k}\cdot\mathbf{a}_2} I_{\frac{1}{3}n_2} \otimes I_{n_1} \\ I_{\frac{2}{3}n_2} \otimes J_2 & 0 \end{bmatrix} \in \mathbb{C}^{(n_1n_2)\times(n_1n_2)}.$$

A.2 Unitary matrix T in (3.4) is defined as

$$T = \frac{1}{\sqrt{n_1n_2n_3}}\begin{bmatrix} T_1 & T_2 & \cdots & T_{n_1} \end{bmatrix} \in \mathbb{C}^{n\times n}$$

with $T_i = \begin{bmatrix} T_{i,1} & T_{i,2} & \cdots & T_{i,n_2} \end{bmatrix} \in \mathbb{C}^{n\times(n_2n_3)}$ and

$$T_{i,j} = \begin{bmatrix} \mathbf{z}_{i,j,1}\otimes\mathbf{y}_{i,j}\otimes\mathbf{x}_i & \mathbf{z}_{i,j,2}\otimes\mathbf{y}_{i,j}\otimes\mathbf{x}_i & \cdots & \mathbf{z}_{i,j,n_3}\otimes\mathbf{y}_{i,j}\otimes\mathbf{x}_i \end{bmatrix} \in \mathbb{C}^{n\times n_3}$$

for $i = 1, \ldots, n_1$, $j = 1, \ldots, n_2$, and $k = 1, \ldots, n_3$, where

$$\begin{aligned} \mathbf{x}_i &= E_{\mathbf{x}}\begin{bmatrix} 1 & e^{\theta_{\mathbf{x},i}} & \cdots & e^{(n_1-1)\theta_{\mathbf{x},i}} \end{bmatrix}^\top \equiv E_{\mathbf{x}}\mathbf{u}_{\mathbf{x},i}, \\ \mathbf{y}_{i,j} &= E_{\mathbf{y},i}\begin{bmatrix} 1 & e^{\theta_{\mathbf{y},j}} & \cdots & e^{(n_2-1)\theta_{\mathbf{y},j}} \end{bmatrix}^\top \equiv E_{\mathbf{y},i}\mathbf{u}_{\mathbf{y},j}, \\ \mathbf{z}_{i,j,k} &= E_{\mathbf{z},i+j}\begin{bmatrix} 1 & e^{\theta_{\mathbf{z},k}} & \cdots & e^{(n_3-1)\theta_{\mathbf{z},k}} \end{bmatrix}^\top \equiv E_{\mathbf{z},i+j}\mathbf{u}_{\mathbf{z},k}, \end{aligned}$$

and

$$\theta_i = \frac{\imath 2\pi i}{n_1} + \frac{\imath 2\pi \mathbf{k}\cdot\mathbf{a}_1}{n_1} \equiv \theta_{\mathbf{x},i} + \varepsilon_{\mathbf{x}},$$
$$\theta_{i,j} = \frac{\imath 2\pi j}{n_2} + \frac{\imath 2\pi}{n_2}\left\{\mathbf{k}\cdot(\mathbf{a}_2 - \frac{1}{2}\mathbf{a}_1) - \frac{i}{2}\right\} \equiv \theta_{\mathbf{y},j} + \varepsilon_{\mathbf{y},i},$$
$$\theta_{i,j,k} = \frac{\imath 2\pi k}{n_3} + \frac{\imath 2\pi}{n_3}\left\{\mathbf{k}\cdot(\mathbf{a}_3 - \frac{1}{3}(\mathbf{a}_1+\mathbf{a}_2)) - \frac{1}{3}(i+j)\right\} \equiv \theta_{\mathbf{z},k} + \varepsilon_{\mathbf{z},i+j}.$$

Here $E_{\mathbf{x}} = \mathrm{diag}\left(1, e^{\varepsilon_{\mathbf{x}}}, \cdots, e^{(n_1-1)\varepsilon_{\mathbf{x}}}\right)$, $E_{\mathbf{y},i} = \mathrm{diag}\left(1, e^{\varepsilon_{\mathbf{y},i}}, \cdots, e^{(n_2-1)\varepsilon_{\mathbf{y},i}}\right)$ and $E_{\mathbf{z},i+j} = \mathrm{diag}\left(1, e^{\varepsilon_{\mathbf{z},i+j}}, \cdots, e^{(n_3-1)\varepsilon_{\mathbf{z},i+j}}\right)$. Denote

$$U_{\mathbf{x}} = \begin{bmatrix}\mathbf{u}_{\mathbf{x},1} & \mathbf{u}_{\mathbf{x},2} & \cdots & \mathbf{u}_{\mathbf{x},n_1}\end{bmatrix},$$
$$U_{\mathbf{y}} = \begin{bmatrix}\mathbf{u}_{\mathbf{y},1} & \mathbf{u}_{\mathbf{y},2} & \cdots & \mathbf{u}_{\mathbf{y},n_2}\end{bmatrix},$$
$$U_{\mathbf{z}} = \begin{bmatrix}\mathbf{u}_{\mathbf{z},1} & \mathbf{u}_{\mathbf{z},2} & \cdots & \mathbf{u}_{\mathbf{z},n_3}\end{bmatrix}.$$

Then the matrix-vector products of $T^*\mathbf{p}$ and $T\mathbf{q}$ can be efficiently computed by Algorithms 5 and 6, respectively.

Algorithm 5 Forward FFT-based matrix-vector product $T^*\mathbf{p}$ [17].

Input: Any vector $\mathbf{p} = \begin{bmatrix}\mathbf{p}_1^\top & \cdots & \mathbf{p}_{n_3}^\top\end{bmatrix}^\top \in \mathbb{C}^n$ with $\mathbf{p}_k = \begin{bmatrix}\mathbf{p}_{1,k}^\top & \cdots & \mathbf{p}_{n_2,k}^\top\end{bmatrix}^\top$ and $\mathbf{p}_{j,k} \in \mathbb{C}^{n_1}$ for $j = 1, \ldots, n_2$, $k = 1, \ldots, n_3$.
Output: The vector $\mathbf{f} \equiv T^*\mathbf{p}$.

for $k = 1, \ldots, n_3$ **do**
  Compute $P_{\mathbf{x}}(:,:,k) = \begin{bmatrix}\mathbf{p}_{1,k} & \cdots & \mathbf{p}_{n_2,k}\end{bmatrix}^* E_{\mathbf{x}} U_{\mathbf{x}}$.
end for
for $i = 1, \ldots, n_1$ **do**
  Compute $P_{\mathbf{y}} = \begin{bmatrix}P_{\mathbf{x}}(:,i,1) & P_{\mathbf{x}}(:,i,2) & \cdots & P_{\mathbf{x}}(:,i,n_3)\end{bmatrix}^\top E_{\mathbf{y},i} U_{\mathbf{y}}$.
  Compute $P_{\mathbf{z}} = U_{\mathbf{z}}^* \begin{bmatrix}E_{\mathbf{z},i+1}^* \bar{P}_{\mathbf{y}}(:,1) & E_{\mathbf{z},i+2}^* \bar{P}_{\mathbf{y}}(:,2) & \cdots & E_{\mathbf{z},i+n_2}^* \bar{P}_{\mathbf{y}}(:,n_2)\end{bmatrix}$.
  Set $\mathbf{f}((i-1)n_2n_3 + 1 : in_2n_3) = \frac{1}{\sqrt{n_1n_2n_3}}\mathrm{vec}(P_{\mathbf{z}})$.
end for

Algorithm 6 Backward FFT-based matrix-vector product $T\mathbf{q}$ [17].

Input: Any vector $\mathbf{q} = \begin{bmatrix}\mathbf{q}_1^\top & \cdots & \mathbf{q}_{n_1}^\top\end{bmatrix}^\top \in \mathbb{C}^n$ with $\mathbf{q}_i = \begin{bmatrix}\mathbf{q}_{i,1}^\top & \cdots & \mathbf{q}_{i,n_2}^\top\end{bmatrix}^\top$ and $\mathbf{q}_{i,j} \in \mathbb{C}^{n_3}$ for $i = 1, \ldots, n_1$, $j = 1, \ldots, n_2$.
Output: The vector $\mathbf{g} \equiv T\mathbf{q}$.

for $i = 1, \ldots, n_1$ **do**
  Compute $Q_{\mathbf{z},i} = U_{\mathbf{z}} \begin{bmatrix}\mathbf{q}_{i,1} & \mathbf{q}_{i,2} & \cdots & \mathbf{q}_{i,n_2}\end{bmatrix}$.
  Update $Q_{\mathbf{z},i} := \begin{bmatrix}E_{\mathbf{z},i+1} Q_{\mathbf{z},i}(:,1) & E_{\mathbf{z},i+2} Q_{\mathbf{z},i}(:,2) & \cdots & E_{\mathbf{z},i+n_2} Q_{\mathbf{z},i}(:,n_2)\end{bmatrix}^\top$.
  Compute $Q_{\mathbf{y}}(:,:,i) = E_{\mathbf{y},i}\left(U_{\mathbf{y}} Q_{\mathbf{z},i}\right)$
end for
for $k = 1, \ldots, n_3$ **do**
  Compute $Q_{\mathbf{x}} = E_{\mathbf{x}} U_{\mathbf{x}} \begin{bmatrix}Q_{\mathbf{y}}(:,k,1) & Q_{\mathbf{y}}(:,k,2) & \cdots & Q_{\mathbf{y}}(:,k,n_1)\end{bmatrix}^\top$.
  Set $\mathbf{g}((k-1)n_1n_2 + 1 : kn_1n_2) = \frac{1}{\sqrt{n_1n_2n_3}}\mathrm{vec}(Q_{\mathbf{x}})$.
end for

A.3 Using Theorem 3.1, it holds that

$$A = I_3 \otimes (G^*G) - GG^*,$$

where $G = [C_1^\top, C_2^\top, C_3^\top]^\top$. Applying the eigendecompositions of C_ℓ's in Theorem 3.1 and the fact $CG = 0$, the solution of the linear system

$$(A - \tau I)\mathbf{y} = \mathbf{d}$$

can be computed by

$$(I_3 \otimes \Lambda_q - \tau I)\,\tilde{\mathbf{y}} = \left(I - \tau^{-1} \begin{bmatrix} \Lambda_1 \\ \Lambda_2 \\ \Lambda_3 \end{bmatrix} \begin{bmatrix} \Lambda_1^* & \Lambda_2^* & \Lambda_3^* \end{bmatrix} \right) (I_3 \otimes T)^*\,\mathbf{d}$$

and

$$\mathbf{y} = (I_3 \otimes T)\,\tilde{\mathbf{y}}.$$

Acknowledgments. The authors thank the anonymous referees for their useful comments and suggestions. The first author's work was partially supported by the Ministry of Science and Technology (MoST), National Centre of Theoretical Sciences (NCTS) in Taiwan. The second author's work was partially supported by MoST, NCTS and ST Yau Centre in Taiwan, and the DAAD "Deutscher Akademischer Austausch Dienst" in Germany. The research of the third author was carried out in the framework of MATHEON project *D-OT3, Adaptive finite element methods for nonlinear parameter-dependent eigenvalue problems in photonic crystals* supported by Einstein Foundation Berlin.

REFERENCES

[1] J. Asakura, T. Sakurai, H. Tadano, T. Ikegami, and K. Kimura. A numerical method for nonlinear eigenvalue problems using contour integrals. *JSIAM Lett.*, 1:52–55, 2009.

[2] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, 2000.

[3] C. G. Baker, U. L. Hetmaniuk, R. B. Lehoucq, and H. K. Thornquis. Anasazi software for the numerical solution of large-scale eigenvalue problems. *ACM Trans. Math. Software*, 36:Article 13, 2009.

[4] W.-J. Beyn. An integral method for solving nonlinear eigenvalue problems. *Linear Alg. Appl.*, 436:3839–3863, 2012.

[5] W.-J. Beyn, C. Effenberger, and D. Kressner. Continuation of eigenvalues and invariant pairs for parameterized nonlinear eigenvalue problems. *Numer. Math.*, 119:489–516, 2011.

[6] R.-L. Chern, C.-C. Chang, C.-C. Chang, and R.-R. Hwang. Numerical study of three-dimensional photonic crystals with large band gaps. *J. Phys. Soc. Japan*, 73:727–737, 2004.

[7] C. Effenberger. *Robust solution methods for nonlinear eigenvalue problems*. PhD thesis, Ecole polytechnique federale de Lausanne, Lausanne, CH, 2013.

[8] C. Engström, C. Hafner, and K. Schmidt. Computations of lossy Bloch waves in two-dimensional photonic crystals. *J. Comput. Theor. Nanosci.*, 6:1–9, 2009.

[9] C. Engström and M. Wang. Complex dispersion relation calculations with the symmetric interior penalty method. *Int. J. Numer. Meth. Engng.*, 84:849–863, 2010.

[10] P. G. Etchegoin, E. C. Le Ru, and M. Meyer. An analytic model for the optical properties of gold. *J. Chem. Phys.*, 125:164705, 2006.

[11] P. G. Etchegoin, E. C. Le Ru, and M. Meyer. Erratum: "An analytic model for the optical properties of gold" [J. Chem. Phys. 125, 164705 (2006)]. *J. Chem. Phys.*, 127:189901, 2007.

[12] S. Fan, P. R. Villeneuve, and J. D. Joannopoulos. Large omnidirectional band gaps in metallodielectric photonic crystals. *Phys. Rev. B*, 54:11245–11251, 1996.

[13] I. Gohberg, P. Lancaster, and L. Rodman. *Matrix Polynomials*. Academic Press, New York, 1982.
[14] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, 1996.
[15] J.-S. Guo, W.-W. Lin, and C.-S. Wang. Numerical solutions for large sparse quadratic eigenvalue problems. *Linear Alg. Appl.*, 225:57–89, 1995.
[16] T.-M. Huang, W.-J. Chang, Y.-L. Huang, W.-W. Lin, W.-C. Wang, and W. Wang. Preconditioning bandgap eigenvalue problems in three dimensional photonic crystals simulations. *J. Comput. Phys.*, 229:8684–8703, 2010.
[17] T.-M. Huang, H.-E. Hsieh, W.-W. Lin, and W. Wang. Eigendecomposition of the discrete double-curl operator with application to fast eigensolver for three dimensional photonic crystals. *SIAM J. Matrix Anal. Appl.*, 34:369–391, 2013.
[18] T.-M. Huang, H.-E. Hsieh, W.-W. Lin, and W. Wang. Matrix representation of the double-curl operator for simulating three dimensional photonic crystals. *Math. Comput. Model.*, 58:379–392, 2013.
[19] T.-M. Huang, H.-E. Hsieh, W.-W. Lin, and W. Wang. Eigenvalue solvers for three dimensional photonic crystals with face-centered cubic lattice. *J. Comput. Appl. Math.*, 272:350–361, 2014.
[20] T.-M. Huang and W.-W. Lin. A novel deflation technique for solving quadratice eigenvalue problems. *Bulletin of the Institute of Mathematics, Academia Sinica (New Series)*, 9:57–84, 2014.
[21] F. Hwang, Z. Wei, T.-M. Huang, and W. Wang. A parallel additive Schwarz preconditioned Jacobi-Davidson algorithm for polynomial eigenvalue problems in quantum dot simulation. *J. Comput. Phys.*, 229:2932–2947, 2010.
[22] T.-M. Hwang, W.-W. Lin, J.-L. Liu, and W. Wang. Jacobi–Davidson methods for cubic eigenvalue problems. *Numer. Linear Algebr. Appl.*, 12:605–624, 2005.
[23] T.-M. Hwang, W.-W. Lin, and V. Mehrmann. Numerical solution of quadratic eigenvalue problems with structure-preserving methods. *SIAM J. Sci. Comput.*, 24:1283–1302, 2003.
[24] T.-M. Hwang, W.-W. Lin, W.-C. Wang, and W. Wang. Numerical simulation of three dimensional pyramid quantum dot. *J. Comput. Phys.*, 196:208–232, 2004.
[25] E. Jarlebring, W. Michiels, and K. Meerbergen. A linear eigenvalue algorithm for the nonlinear eigenvalue problem. *Numer. Math.*, 122:169–195, 2012.
[26] E. Jarlebring and H. Voss. Rational Krylov for nonlinear eigenvalues, an iterative projectioon method. *Appl. Math.*, 50:543–554, 2005.
[27] J. D. Joannopoulos, S. G. Johnson, J. N. Winn, and R. D. Meade. *Photonic Crystals: Molding the Flow of Light*. Princeton University Press, Princeton, NJ, 2008.
[28] C. Kittel. *Introduction to solid state physics*. Wiley, New York, 2005.
[29] D. Kressner. A block Newton method for nonlinear eigenvalue problems. *Numer. Math.*, 114:355–372, 2009.
[30] C. Lee. *Residual Arnoldi method: theory, package and experiments*. PhD thesis, TR-4515, Department of Computer Science, University of Maryland at College Park, 2007.
[31] C. Lee and G. W. Stewart. Analysis of the residual Arnoldi method. Technical report, TR-4890, Department of Computer Science, University of Maryland at College Park, 2007.
[32] R. Lehoucq, D. Sorensen, and C. Yang. *ARPACK USERS GUIDE: Solution of large scale eigenvalue problems with implicitely restarted Arnoldi methods*. SIAM, Philadelphia, 1998.
[33] B.-S. Liao, Z. Bai, L.-Q. Lee, and K. Ko. Nonlinear Rayleigh-Ritz iterative method for solving large scale nonlinear eigenvalue problems. *Taiwan. J. Math.*, 14:869–883, 2010.
[34] M. Luo and Q. H. Liu. Three-dimensional dispersive metallic photonic crystals with a bandgap and a high cutoff frequency. *J. Opt. Soc. Am. A Opt. Image Sci. Vis.*, 27:1878–1884, 2010.
[35] D. Mackey, N. Mackey, C. Mehl, and V. Mehrmann. Vector spaces of linearizations for matrix polynomials. *SIAM J. Matrix Anal. Appl.*, 28:971–1004, 2006.
[36] V. Mehrmann and C. Schröder. Nonlinear eigenvalue and frequency response problems in industrial practice. *J. Math. in Industry*, 1:7, 2011. http://www.mathematicsinindustry.com/.
[37] V. Mehrmann and H. Voss. Nonlinear eigenvalue problems: A challenge for modern eigenvalue methods. *GAMM Mitt. Ges. Angew. Math. Mech.*, 27:121–152, 2004.
[38] E. Moreno, D. Erni, and C. Hafner. Band structure computations of metallic photonic crystals with the multiple multipole method. *Phys. Rev. B*, 65:155120, 2002.
[39] B. N. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall, Englewood Cliffs, NJ, 1980.
[40] A. Ruhe. The rational Krylov algorithm for nonlinear matrix eigenvalue problems. *Zap. Nauchn. Semin. POMI*, 268:176–180, 2000.
[41] K. Schreiber. *Nonlinear eigenvalue problems: Newton type methods and nonlinear Rayleigh functionals*. PhD thesis, Inst. f. Mathematik, TU Berlin, Berlin, FRG, 2008.

[42] G. L. G. Sleijpen, A. G. L. Booten, D. R. Fokkema, and H. A. van der Vorst. Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems. *BIT*, 36:595–633, 1996.

[43] G. L. G. Sleijpen and H. A. van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 17:401–425, 1996.

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

[45] F. Tisseur. Backward error and condition of polynomial eigenvalue problems. *Linear Alg. Appl.*, 309:339–361, 2000.

[46] F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43:235–286, 2001.

[47] A. Vial. Implementation of the critical points model in the recursive convolution method for modelling dispersive media with the finite-difference time domain method. *J. Opt. A: Pure Appl. Opt.*, 9:745–748, 2007.

[48] H. Voss. An Arnoldi method for nonlinear eigenvalue problems. *BIT*, 44:387–401, 2004.

[49] H. Voss. A Jacobi-Davidson method for nonlinear and nonsymmetric eigenproblems. *Comput. Struct.*, 85:1284–1292, 2007.

[50] W. Wang, T.-M. Hwang, W.-W. Lin, and J.-L. Liu. Numerical methods for semiconductor heterostructures with band nonparabolicity. *J. Comput. Phys.*, 190:141–158, 2003.

[51] K. Yee. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Trans. Antennas and Propagation*, 14:302–307, 1966.

[52] Y. Zhao, C. Argyropoulos, and Y. Hao. Full-wave finite-difference time-domain simulation of electromagnetic cloaking structures. *Opt. Express*, 16:6717–6730, 2008.

[53] R. Ziolkowski. Pulsed and CW Gaussian beam interactions with double negative metamaterial slabs. *Opt. Express*, 11:662–681, 2003.