NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2005; **12**:605–624
Published online 6 September 2004 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nla.423

Jacobi–Davidson methods for cubic eigenvalue problems

Tsung-Min Hwang[1], Wen-Wei Lin[2,*,†], Jinn-Liang Liu[3] and Weichung Wang[4]

[1]*Department of Mathematics, National Taiwan Normal University, Taipei 116, Taiwan*
[2]*Department of Mathematics, National Tsing Hua University, Hsinchu 300, Taiwan*
[3]*Department of Applied Mathematics, National Chiao Tung University, Hsinchu 300, Taiwan*
[4]*Department of Applied Mathematics, National University of Kaohsiung, Kaohsiung 811, Taiwan*

SUMMARY

Several Jacobi–Davidson type methods are proposed for computing interior eigenpairs of large-scale cubic eigenvalue problems. To successively compute the eigenpairs, a novel explicit non-equivalence deflation method with low-rank updates is developed and analysed. Various techniques such as locking, search direction transformation, restarting, and preconditioning are incorporated into the methods to improve stability and efficiency. A semiconductor quantum dot model is given as an example to illustrate the cubic nature of the eigenvalue system resulting from the finite difference approximation. Numerical results of this model are given to demonstrate the convergence and effectiveness of the methods. Comparison results are also provided to indicate advantages and disadvantages among the various methods. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: cubic eigenvalue problem; cubic Jacobi–Davidson method; non-equivalence deflation; 3D Schrödinger equation

1. INTRODUCTION

A cubic eigenvalue problem of order n can be defined as

$$\mathbf{A}(\lambda)\mathbf{F} \equiv (\lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0)\mathbf{F} = 0 \tag{1}$$

where $\lambda \in \mathbb{C}$, $\mathbf{F} \in \mathbb{C}^n$, and $A_i \in \mathbb{R}^{n\times n}$ for $i = 0, 1, 2, 3$. In applications, a set of the eigenvalues embedded in the interior of the spectrum of a large-scale eigenvalue problem are often of interest. For example, a semiconductor quantum dot model with non-parabolic band structure described by the three-dimensional (3D) Schrödinger equation [1–3] can result in a cubic eigenvalue problem of (1) with order up to 211 400 from the finite difference approximation

*Correspondence to: W.-W. Lin, Department of Mathematics, National Tsing Hua University, Hsinchu 300, Taiwan.
†E-mail: wwlin@am.nthu.edu.tw

Contract/grant sponsor: National Science Council
Contract/grant sponsor: National Center for Theoretical Sciences

Received 16 April 2003
Revised 8 February 2004

(see Section 3). And we are concerned only with several smallest positive real eigenvalues (energy states) and their associated eigenvectors (wave functions). Motivated by this model, various methods based on the Jacobi–Davidson (JD) and explicit deflation techniques are proposed here for calculating the interior eigenpairs of the cubic eigenvalue problem (1).

A classical approach that can be used for computing the solutions of (1) is to consider the linearization of (1),

$$\begin{bmatrix} 0 & I & 0 \\ 0 & 0 & I \\ A_0 & A_1 & A_2 \end{bmatrix} \begin{bmatrix} \mathbf{F} \\ \lambda\mathbf{F} \\ \lambda^2\mathbf{F} \end{bmatrix} = \lambda \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & -A_3 \end{bmatrix} \begin{bmatrix} \mathbf{F} \\ \lambda\mathbf{F} \\ \lambda^2\mathbf{F} \end{bmatrix} \tag{2}$$

This enlarged linear eigenvalue problem can then be solved by various Lanczos or Arnoldi methods [4]. These methods are well established in many aspects of numerical algorithms, convergence properties, and stability analysis [5–7]. However, disadvantages of such an approach still exist. First of all, the order of the matrix is tripled and its condition number may increase significantly since the set of admissible perturbations for (2) is larger than that of (1) [8]. Secondly, the performance of these methods may be reduced for the enlarged problem in terms of convergence, efficiency, and accuracy. Thirdly, Lanczos and Arnoldi methods require the use of the shift-and-invert technique for such a large sparse eigenvalue problem since the desired eigenpairs are located in the interior of the spectrum of the problem. Consequently, the computational cost for solving linear system is excessive.

Another approach is a direct solution of (1) by means of the JD method. Although this method has been developed for linear and quadratic eigenvalue problems [4, 9–12], it is far less studied than its classical counterpart. To our knowledge, there appears no numerical algorithms or computational experiences being reported in the literature for the cubic eigenvalue problems. In this paper, we extend the JD method presented in References [4, 9–12] to solve the cubic problems and propose various forms of the method to improve stability and efficiency in calculating the interior eigenpairs.

In order to compute the interior eigenpairs successively, it is necessary to incorporate the JD method with some deflation techniques. For linear eigenvalue problems, it is well known that a combination of JD and implicit deflation techniques based on the Schur form can lead to effective algorithms (see e.g. Reference [4, Sections 4.7 and 8.4]). For quadratic eigenvalue problems, Meerbergen [13] proposes a JD method by using the locking and restarting scheme based on the Schur form of the linearized problem. This method illustrates the essential ingredients for the extension of the JD method from the linear case to the quadratic case. Furthermore, Guo *et al.* develop a deflation method for large sparse quadratic eigenvalue problems [14] and examine several deflation strategies for analytic non-defective matrix function [15]. Ruhe [16] suggests using the smallest eigenvalue as an initial guess for computing the second eigenvalue and using the sum of the first two eigenvalues as an initial guess for the third eigenvalue in Newton's method.

However, it is not clear how to incorporate an implicit deflation scheme with the JD method for the cubic eigenvalue problems since the Schur form is not defined for a cubic matrix pencil in general. We propose here a cubic version of the JD method and an explicit non-equivalence deflation method with low-rank updates to deal with these problems. Several algorithms are then given to illustrate various modifications of these two methods. The main

procedure of the algorithms is as follows. The standard cubic JD (CJD) method is first used to find the first smallest eigenpair. The current eigenvalue is then deflated to infinity and a new (deflated) cubic eigenvalue problem is subsequently formed. The CJD method itself or its variant is applied again for the next eigenpair. This procedure is repeated until all the desired eigenpairs are found.

The main results of this paper are briefly summarized as follows:

- The explicit non-equivalence deflation method is proved to deflate the computed eigenvalues to infinity while all other unknown eigenvalues remain unchanged.
- Several variants of the CJD method are developed for the deflated cubic eigenproblem to improve the stability and efficiency of the method in cases that the two consecutive eigenvalues are too close to each other and that the computational cost is expensive due to the accumulative low-rank updates as deflations increase.
- Among all the CJD methods, we find that the combination of the CJD, the locking, and the explicit deflation ε_o (see Section 2) is shown numerically to be most robust and efficient in terms of accuracy and computational cost.

This paper is organized as follows. In Section 2, we first present the cubic Jacobi–Davidson method and a primitive locking technique based on Reference [13] for computing the desired eigenvalues. An explicit non-equivalence deflation method is then given and analysed for the rest of the desired eigenpairs. The variants of the CJD method for the deflated cubic eigenproblem are also given in this section. In Section 3, the quantum dot model is described and discretized by the finite difference method using non-uniform grids. A brief derivation of the resulting cubic eigenvalue problem (1) from the discretization then follows. Numerical results are given in Section 4. Some concluding remarks are made in Section 5. Note that throughout the paper, when we specify the order of an eigenpair such as the smallest (first) positive eigenpair, we mean the smallest positive eigenvalue and the associated eigenvector.

2. CUBIC JACOBI–DAVIDSON AND EXPLICIT DEFLATION METHODS

In this section, we first present the CJD method incorporated with a locking technique for the desired eigenpairs in Section 2.1. The explicit non-equivalence deflation method is presented and analysed in Section 2.2. The deflation method is then generalized to deal with more practical situations to improve its stability and efficiency in Section 2.3. We summarize and compare all the proposed algorithms in Section 2.4.

2.1. A CJD method for desired eigenpairs

We first propose a CJD method incorporated with a simple locking technique in Algorithm 2.1. The algorithm adopts the same framework of the quadratic JD method presented in References [17, 18]. The locking technique used here is similar to the techniques suggested in Reference [13] for quadratic eigenvalue problems. However, our locking scheme is rather primitive in the following sense. Unlike the schemes in Reference [13], we do not perform the reordering of the partial Schur form. We simply append the convergent eigenvectors into the trial subspace $V = [V_{\text{ini}}]$ as shown in Steps (2.3) and (2.4) of Algorithm 2.1.

Algorithm 2.1 (*CJD-Lk*)**.**

CJD method with locking for cubic eigenproblem.

(0) Given $\mathbf{A}(\lambda)=\sum_{i=0}^{3}\lambda^i A_i$ and the number k of desired eigenvalues.
(1) Choose an n-by-m orthonormal matrix $V=[V_{\text{ini}}]$ and set $V_F=[]$.
(2) For $\ell=1,\ldots,k$
  (2.1) Compute $W_i=A_iV$ and $M_i=V^*W_i$ for $i=0,\ldots,3$.
  (2.2) Iterate until convergence
    (i) Compute the eigenpairs (θ,s) of $(\sum_{i=0}^{3}\theta^i M_i)s=0$ by using QZ algorithm [6] for solving the generalized linear eigenproblem

$$\begin{bmatrix} 0 & I & 0 \\ 0 & 0 & I \\ M_0 & M_1 & M_2 \end{bmatrix}\begin{bmatrix} s \\ \theta s \\ \theta^2 s \end{bmatrix}=\theta\begin{bmatrix} I & & \\ & I & \\ & & -M_3 \end{bmatrix}\begin{bmatrix} s \\ \theta s \\ \theta^2 s \end{bmatrix}$$

    (ii) Select the desired (target) eigenpair (θ,s) with $\| s \|_2 =1$.
    (iii) Compute $u=Vs$, $p=\mathbf{A}'(\theta)u$, $r=\mathbf{A}(\theta)u$.
    (iv) If $(\|r\|_2<\varepsilon)$, Set $\lambda_\ell=\theta$, $\mathbf{F}_\ell=u$, Goto Locking Steps (2.3) and (2.4).
    (v) Solve (approximately) for $t\perp u$ from

$$\left(I-\frac{pu^*}{u^*p}\right)\mathbf{A}(\theta)(I-uu^*)t=-r$$

    (vi) Orthogonalize t against V, $v=t/\|t\|_2$.
    (vii) Compute $w_i=A_iv$, $M_i=\begin{bmatrix} M_i & V^*w_i \\ v^*W_i & v^*w_i \end{bmatrix}$ for $i=0,\ldots,3$.
    (viii) Expand $V=[V,v]$ and $W_i=[W_i,w_i]$
  (2.3) Orthogonalize $\mathbf{F}_\ell$ against V_F; Compute $\mathbf{F}_\ell=\mathbf{F}_\ell/\| \mathbf{F}_\ell \|$; Update $V_F=[V_F,\mathbf{F}_\ell]$.
  (2.4) Choose an orthonormal matrix $V_{\text{ini}}\perp V_F$; Set $V=[V_F,V_{\text{ini}}]$.
  End for
(3) Output the approximated eigenpairs $(\lambda_\ell,\mathbf{F}_\ell)$ for $\ell=1,\ldots,k$.

It is worth mentioning following practical considerations. As suggested in References [11, 12], the correction equation

$$\left(I-\frac{pu^*}{u^*p}\right)\mathbf{A}(\theta)(I-uu^*)t=-r \tag{3}$$

needs to be solved. Since the vector t is supposed to be orthogonal to the vector u, Equation (3) can be rewritten as

$$\mathbf{A}(\theta)t=-r+\varepsilon p \tag{4}$$

with

$$\varepsilon = \frac{u^* \mathbf{A}(\theta)^{-1} r}{u^* \mathbf{A}(\theta)^{-1} p}$$

In Step (2.2.v) of Algorithm 2.1, the correction equation (3) is solved approximately by choosing a preconditioner $M_{\mathbf{A}} \approx \mathbf{A}(\theta)$ so that the vector t is approximated by

$$t \approx -M_{\mathbf{A}}^{-1} r + \varepsilon M_{\mathbf{A}}^{-1} p \tag{5}$$

Since t is ideally orthogonal to the vector u, the scalar ε can be obtained by

$$\varepsilon = \frac{u^* M_{\mathbf{A}}^{-1} r}{u^* M_{\mathbf{A}}^{-1} p} \tag{6}$$

In Section 4, we give some suggestions on how to choose the preconditioner $M_{\mathbf{A}}$ for the model problem. The numerical results show that the algorithm can be very efficient if the preconditioner is suitably chosen.

2.2. An explicit non-equivalence deflation method

After the smallest, or a few smallest, positive eigenpairs have been computed, we proceed to compute the rest of eigenpairs by an explicit non-equivalence deflation method in a consecutive manner. This method is modified from that of Reference [14].

Let $(\Lambda, V_F) \in \mathbb{R}^{r\times r} \times \mathbb{R}^{n\times r}$ be an eigenmatrix pair of $\mathbf{A}(\lambda)$ with $V_F^{\mathrm{T}} V_F = I_r$ and $0 \notin \sigma(\Lambda)$, where $\sigma(\Lambda)$ denotes the spectrum of Λ. In other words, we have

$$A_3 V_F \Lambda^3 + A_2 V_F \Lambda^2 + A_1 V_F \Lambda + A_0 V_F = 0 \tag{7}$$

Now we define a new deflated cubic eigenvalue problem by

$$\tilde{\mathbf{A}}(\lambda)\mathbf{F} = (\lambda^3 \tilde{A}_3 + \lambda^2 \tilde{A}_2 + \lambda \tilde{A}_1 + \tilde{A}_0)\mathbf{F} = 0 \tag{8}$$

where

$$\begin{aligned}
\tilde{A}_0 &= A_0 \\
\tilde{A}_1 &= A_1 - (A_1 V_F V_F^{\mathrm{T}} + A_2 V_F \Lambda V_F^{\mathrm{T}} + A_3 V_F \Lambda^2 V_F^{\mathrm{T}}) \\
\tilde{A}_2 &= A_2 - (A_2 V_F V_F^{\mathrm{T}} + A_3 V_F \Lambda V_F^{\mathrm{T}}) \\
\tilde{A}_3 &= A_3 - A_3 V_F V_F^{\mathrm{T}}
\end{aligned} \tag{9}$$

Note that the superscript tilde is used to denote the variant coefficient matrices associated with the deflated cubic eigenvalue problem. In the following we first prove a useful lemma and then, in Theorem 2, we show that the computed eigenvalues Λ are deflated to infinity in the new deflated cubic eigenproblem $\tilde{\mathbf{A}}(\lambda)$ while the rest of the unknown eigenvalues remain unchanged.

Lemma 1
Let $\mathbf{A}(\lambda)$ and $\tilde{\mathbf{A}}(\lambda)$ be cubic pencils given by (1) and (8), respectively. Then it holds

$$\tilde{\mathbf{A}}(\lambda) = \mathbf{A}(\lambda)(I_n - \lambda V_F (\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}}) \tag{10}$$

Proof
Using (9) and (7), and the fundamental matrix calculation, we have

$$\begin{aligned}
\tilde{\mathbf{A}}(\lambda) &= \mathbf{A}(\lambda) - \lambda(\lambda^2 A_3 V_F V_F^{\mathrm{T}} + \lambda A_2 V_F V_F^{\mathrm{T}} + \lambda A_3 V_F \Lambda V_F^{\mathrm{T}} + A_1 V_F V_F^{\mathrm{T}} \\
&\quad + A_2 V_F \Lambda V_F^{\mathrm{T}} + A_3 V_F \Lambda^2 V_F^{\mathrm{T}}) \\
&= \mathbf{A}(\lambda) - \lambda(A_3 V_F(\lambda I_r - \Lambda)^3(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}} \\
&\quad + 3A_3 V_F \Lambda(\lambda I_r - \Lambda)^2(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}} + 3A_3 V_F \Lambda^2(\lambda I_r - \Lambda)(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}} \\
&\quad + A_2 V_F(\lambda I_r - \Lambda)^2(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}} + 2A_2 V_F \Lambda(\lambda I_r - \Lambda)(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}} \\
&\quad + A_1 V_F(\lambda I_r - \Lambda)(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}}) \\
&= \mathbf{A}(\lambda) - \lambda\{[A_3 V_F(\lambda^3 I_r - \Lambda^3) + A_2 V_F(\lambda^2 I_r - \Lambda^2) + A_1 V_F(\lambda I_r - \Lambda) + A_0 V_F \\
&\quad - A_0 V_F](\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}}\} \\
&= \mathbf{A}(\lambda) - \lambda[\mathbf{A}(\lambda) V_F(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}}] \\
&= \mathbf{A}(\lambda)[I_n - \lambda V_F(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}}]
\end{aligned}$$

□

Theorem 2
Let $(\Lambda, V_F) \in \mathbb{R}^{r\times r} \times \mathbb{R}^{n\times r}$ be an eigenmatrix pair of $\mathbf{A}(\lambda)$ as in (7) with $V_F^{\mathrm{T}} V_F = I_r$. Then

(i) the new deflated cubic pencil $\tilde{\mathbf{A}}(\lambda)$ in (8) has the same eigenvalues as those of $\mathbf{A}(\lambda)$ except that the r eigenvalues of Λ are replaced by infinity, i.e. $(\sigma(\mathbf{A}(\lambda)) \setminus \sigma(\Lambda)) \cup \{\infty\} = \sigma(\tilde{\mathbf{A}}(\lambda))$.
(ii) Let (μ, z) be an eigenpair of $\mathbf{A}(\lambda)$ with $\| z \|_2 = 1$ and $\mu \notin \sigma(\Lambda)$. Define

$$\tilde{z} = (I_n - \mu V_F \Lambda^{-1} V_F^{\mathrm{T}}) z \equiv T(\mu) z \tag{11}$$

Then $(\mu, \tilde{z})$ is an eigenpair of $\tilde{\mathbf{A}}(\lambda)$.

Proof
(i) Using the identity (see e.g. Reference [19, pp. 53])

$$\det(I_n + RS) = \det(I_m + SR)$$

and Lemma 1, we have

$$\begin{aligned}
\det(\tilde{\mathbf{A}}(\lambda)) &= \det(\mathbf{A}(\lambda))\det(I_n - \lambda V_F(\lambda I_r - \Lambda)^{-1} V_F^{\mathrm{T}}) \\
&= \det(\mathbf{A}(\lambda))\det(I_n - \lambda(\lambda I_r - \Lambda)^{-1}) \\
&= \det(\mathbf{A}(\lambda))\det(\lambda I_r - \Lambda)^{-1}\det(-\Lambda)
\end{aligned}$$

Since $0 \notin \sigma(\Lambda)$, $\det(-\Lambda) \neq 0$. Thus, $\tilde{\mathbf{A}}(\lambda)$ and $\mathbf{A}(\lambda)$ have the same finite spectrum except the eigenvalues in $\sigma(\Lambda)$. Furthermore, dividing Equation (8) by λ^3 and using the fact that

$$\tilde{\mathbf{A}}_3 V_F = (A_3 - A_3 V_F V_F^{\mathrm{T}}) V_F = 0$$

we see that $(\mathrm{diag}_r\{\infty, \ldots, \infty\}, V_F)$ is an eigenmatrix pair of $\tilde{\mathbf{A}}(\lambda)$ corresponding to infinite eigenvalues.

(ii) Since $\mu \notin \sigma(\Lambda)$, the matrix $T(\mu) = (I - \mu V_F \Lambda^{-1} V_F^{\mathrm{T}})$ in (11) is invertible with the inverse

$$T(\mu)^{-1} = I_n - \mu V_F (\mu I_r - \Lambda)^{-1} V_F^{\mathrm{T}} \tag{12}$$

From Lemma 1, we have

$$\tilde{\mathbf{A}}(\mu)\tilde{z} = \mathbf{A}(\mu)[I_n - \mu V_F(\mu I_r - \Lambda)^{-1} V_F^{\mathrm{T}}][I_n - \mu V_F \Lambda^{-1} V_F^{\mathrm{T}}]z = 0$$

This completes the proof. □

Theorem 2 suggests that the explicit non-equivalence deflation scheme can be applied repeatedly to compute all desired interior eigenpairs. To be specific, Algorithm 2.1 is modified to achieve the goal as follows. We refer this modified algorithm as CJD-Dfl.

1. The locking steps (2.3) and (2.4) in Algorithm 2.1 are replaced with the following two updating steps. Note that in (2.4), the convergent eigenvectors in V_F are not appended to the trial subspace V.

   (2.3) Orthonormalize $\mathbf{F}_\ell$ against current V_F. Update $V_F = [V_F, \mathbf{F}_\ell]$ and Λ by the upper triangular matrix in the Gram–Schmidt process.
   (2.4) Choose an orthonormal matrix $V_{\mathrm{ini}} \perp V_F$. Set $V = V_{\mathrm{ini}}$.

2. In the first iteration on Step (2), the matrices $\mathbf{A}(\theta)$ in (2.2.iii) and (2.2.v) are defined by the original eigenvalue problem (1). Starting from the second iteration, the matrices are defined by the deflated system (8).

However, there are some drawbacks with the deflation transformation matrix $T(\mu)$ in (11). For example, if μ is close to the eigenvalue of Λ, the matrix $T(\mu)$ may be ill-conditioned and hence the transformation (11) may be inaccurate. Moreover, the computational cost for solving the deflated cubic eigenproblem (8) becomes more expensive when the number of columns of V_F in (9) is getting larger. Fortunately, the drawbacks can be avoided by the observations in the next subsection.

2.3. Variants of the CJD method for deflated cubic eigenproblems

To overcome these disadvantages, the main idea is to avoid the use of the deflated cubic eigenproblem $\tilde{\mathbf{A}}(\lambda)$ in (8) and the deflation transformation $T(\mu)$ in (11), directly. The goal can be achieved by rewriting the correction equation in the CJD-Dfl method involving the matrices $\tilde{\mathbf{A}}(\lambda)$ and $T(\mu)$ so that the new equivalent correction equation depends only on the original vectors and matrices. Consequently, the computational cost can be reduced significantly and the scheme becomes more stable.

Using the CJD-Dfl method for solving the deflated cubic eigenproblem (8), we first note that it is required to compute

$$\tilde{r} = \tilde{\mathbf{A}}(\theta)\tilde{u} \quad \text{and} \quad \tilde{p} = \tilde{\mathbf{A}}'(\theta)\tilde{u} \tag{13}$$

where θ is a Ritz value. By the definition of $T(\theta) \equiv (I - \theta V_F \Lambda^{-1} V_F^{\mathrm{T}})$ and (12), Lemma 1 implies

$$\tilde{\mathbf{A}}(\theta)T(\theta) = \mathbf{A}(\theta) \tag{14}$$

Differentiating $\tilde{\mathbf{A}}(\theta)$ with respect to θ and using (14), we get

$$\begin{aligned}
\tilde{\mathbf{A}}'(\theta)T(\theta) &= \mathbf{A}'(\theta) - \mathbf{A}(\theta)T^{-1}(\theta)T'(\theta) \\
&= \mathbf{A}'(\theta) - \mathbf{A}(\theta)\{V_F[-\Lambda^{-1} + \theta(\theta I_r - \Lambda)^{-1}\Lambda^{-1}]V_F^{\mathrm{T}}\} \quad \text{(from (12))} \\
&= \mathbf{A}'(\theta) - \mathbf{A}(\theta)V_F(\theta I_r - \Lambda)^{-1}V_F^{\mathrm{T}} \\
&= \mathbf{A}'(\theta) - (\theta^3 A_3 + \theta^2 A_2 + \theta A_1)V_F(\theta I_r - \Lambda)^{-1}V_F \quad \text{(from (7))} \\
&\quad + (A_3 V_F \Lambda^3 + A_2 V_F \Lambda^2 + A_1 V_F \Lambda)(\theta I_r - \Lambda)^{-1}V_F \\
&= \mathbf{A}'(\theta) - [A_3 V_F(\Lambda^2 + \theta\Lambda + \theta^2 I_r)V_F^{\mathrm{T}} + A_2 V_F(\Lambda + \theta I_r)V_F^{\mathrm{T}} + A_1 V_F V_F^{\mathrm{T}}]
\end{aligned} \tag{15}$$

By defining

$$\bar{u} = T(\theta)^{-1}\tilde{u} \tag{16}$$

Theorem 2 (ii) shows that if $(\theta, \tilde{u})$ is an eigenpair of $\tilde{\mathbf{A}}(\theta)$, then the vector $(\theta, \bar{u})$ is an eigenpair of $\mathbf{A}(\theta)$. Furthermore, from (14) and (16) the residual $\tilde{r}$ of the eigenpair $(\theta, \tilde{u})$ for the deflated cubic eigenproblem can be rewritten as

$$\tilde{r} = \tilde{\mathbf{A}}(\theta)\tilde{u} = \tilde{\mathbf{A}}(\theta)T(\theta)\bar{u} = \mathbf{A}(\theta)\bar{u} = r \tag{17}$$

which is also the residual of the eigenpair $(\theta, \bar{u})$ of the original cubic eigenproblem. Moreover, by (15) and (16), the skew orthogonalization vector $\tilde{p}$ in (13) for CJD-Dfl method can then be computed by

$$\tilde{p} = \mathbf{A}'(\theta)\bar{u} - [A_3 V_F(\Lambda^2 + \theta\Lambda + \theta^2 I_r)V_F^{\mathrm{T}} + A_2 V_F(\Lambda + \theta I_r)V_F^{\mathrm{T}} + A_1 V_F V_F^{\mathrm{T}}]\bar{u} \tag{18}$$

In other words, by using (17) and (18) rather than (13), we can achieve significant saving on computing $\tilde{r}$ and $\tilde{p}$ as the size of Λ and V_F becomes large.

We can further reduce the cost of computation of the vector

$$\tilde{t} = \tilde{\mathbf{A}}^{-1}(\theta)\tilde{r} + \tilde{\varepsilon}\tilde{\mathbf{A}}^{-1}(\theta)\tilde{p} \tag{19}$$

by defining

$$\bar{t} = T(\theta)^{-1}\tilde{t} \tag{20}$$

and using (14) such that

$$\bar{t} = -\mathbf{A}^{-1}(\theta)\tilde{r} + \tilde{\varepsilon}\mathbf{A}^{-1}(\theta)\tilde{p}$$

The vector $\bar{t}$ can therefore be approximated by

$$\bar{t} \approx -M_A^{-1}\tilde{r} + \tilde{\varepsilon}M_A^{-1}\tilde{p} \tag{21}$$

with a preconditioner $M_A \approx \mathbf{A}(\theta)$, which is preferable since it is in general more cost efficient than the matrix $M_{\tilde{A}} \approx \tilde{\mathbf{A}}(\theta)$.

We next note that, by neglecting the low-rank updates in the deflated matrix $\tilde{\mathbf{A}}(\theta)$, the matrices $\tilde{W}_i$ and the vectors $\tilde{w}_i$ can also be computed by using the original matrices, i.e.

$$\tilde{W}_i = A_i V \quad \text{and} \quad \tilde{w}_i = A_i v \tag{22}$$

for $i = 0, 1, 2, 3$. This heuristic scheme results in that the Ritz vector $\bar{u}$ of $\mathbf{A}(\theta)$ can be obtained without using the transformation in (16). In other words, with Equations (16), (20) and (22), there is no need to explicitly compute $\tilde{u}$ and $\tilde{t}$ when applying Algorithm 2.1 to the deflated cubic eigenproblem.

Finally, based on the previous observation, we consider two different choices of the parameter $\tilde{\varepsilon}$ for approximating the vector $\bar{t}$ in (21) for the deflated cubic eigenproblem.

1. The vector $\tilde{t}$ defined in (19) should be orthogonal to the vector $\tilde{u} = T(\theta)\bar{u}$, i.e. $\tilde{u}^*\tilde{t} = 0$. Consequently, $\tilde{\varepsilon}$ can be chosen as

$$\tilde{\varepsilon} = \tilde{\varepsilon}_D = \frac{\bar{u}^* T^*(\theta)T(\theta)M_A^{-1}\tilde{r}}{\bar{u}^* T^*(\theta)T(\theta)M_A^{-1}\tilde{p}} \tag{23}$$

   where

$$T^*(\theta)T(\theta) = I_n - \theta V_F(\Lambda^{-\mathrm{T}} + \Lambda^{-1} - \theta\Lambda^{-\mathrm{T}}\Lambda^{-1})V_F^{\mathrm{T}}$$

   Here, the subscript 'D' in (23) is used to indicate that the vectors $\tilde{u}$ and $\tilde{t}$ involve the deflation transformation $T(\theta)$.
2. Since we have simplified the computation of $\tilde{t}$ by replacing it with $\bar{t}$, it is natural to require $\bar{t}$ defined in (21) be orthogonal to the vector $\bar{u}$, i.e. $\bar{u}^*\bar{t} = 0$. We can thus choose

$$\tilde{\varepsilon} = \tilde{\varepsilon}_O = \frac{\bar{u}^* M_A^{-1}\tilde{r}}{\bar{u}^* M_A^{-1}\tilde{p}} \tag{24}$$

   By doing so, we further relax the need of computing $T^*(\theta)T(\theta)$. The subscript 'O' in (24) is used to emphasize that the computation of $\bar{u}$ and $\bar{t}$ involve only the original cubic eigenproblem.

In short, by introducing the vectors $\bar{u}$ and $\bar{t}$, we have shown that the computation of $\tilde{r}$, $\tilde{W}_i$ and $\tilde{w}_i$ in the process of applying the CJD-Dfl method to the deflated eigenproblem can involve only the original system $\mathbf{A}(\theta)$. The vector $\tilde{p}$ computed by (18) still depends on the matrices Λ and V_F, but not on the transformation matrix $T(\theta)$.

We summarize previous discussions in the following algorithm for the computation of all desired eigenpairs of the deflated cubic eigenproblem.

Algorithm 2.2 (*CJD-Lk-Dfl-ε_D and CJD-Lk-Dfl-ε_O*).
CJD method with locking and variant explicit deflations.

(0) Given $\mathbf{A}(\lambda)=\sum_{i=0}^{3}\lambda^i A_i$ and the number k of desired eigenvalues.
(1) Choose an n-by-m orthonormal matrix $V=[V_{\text{ini}}]$; Set $V_F=[]$ and $\Lambda=[]$.
(2) For $\ell=1,\ldots,k$
  (2.1) Compute $W_i=A_iV$ and $M_i=V^*W_i$ for $i=0,\ldots,3$.
  (2.2) Iterate until convergence
    (i) Compute the eigenpairs (θ,s) of $(\sum_{i=0}^{3}\theta^i M_i)s=0$ by using QZ algorithm for solving the enlarged generalized linear eigenproblem as in (2.2.i) of Algorithm 2.1.
    (ii) Select the desired (target) eigenpair (θ,s) with $\|s\|_2=1$.
    (iii) Compute $u=Vs$, $r=\mathbf{A}(\theta)u$ and p by Equation (18).
    (iv) If $(\|r\|_2<\varepsilon)$, Set $\lambda_\ell=\theta$, $\mathbf{F}_\ell=u$, Goto Locking Steps (2.3), (2.4).
    (v) Compute $t=-M_A^{-1}r+\varepsilon M_A^{-1}p$ by
$$\varepsilon=\varepsilon_D=\frac{u^*T(\theta)^*T(\theta)M_A^{-1}r}{u^*T(\theta)^*T(\theta)M_A^{-1}p}\quad\text{or}\quad\varepsilon=\varepsilon_O=\frac{u^*M_A^{-1}r}{u^*M_A^{-1}p}$$
    (vi) Orthogonalize t against V, $v=t/\|t\|_2$.
    (vii) Compute $w_i=A_iv$, $M_i=\begin{bmatrix} M_i & V^*w_i \\ v^*W_i & v^*w_i\end{bmatrix}$ for $i=0,\ldots,3$.
    (viii) Expand $V=[V,v]$ and $W_i=[W_i,w_i]$.
  (2.3) Update Λ and V_F by the Gram–Schmidt process. That is, orthonormalize $\mathbf{F}_\ell$ against current V_F, expand $V_F=[V_F,\mathbf{F}_\ell]$, update Λ by the upper triangular matrix in the Gram–Schmidt process.
  (2.4) Choose an orthonormal matrix $V_{\text{ini}}\perp V_F$; Set $V=[V_F,V_{\text{ini}}]$.
  End for
(3) Output the approximated eigenpairs (λ_ℓ,F_ℓ) for $\ell=1,\ldots,k$.

2.4. A summary of the algorithms

We have proposed several ideas for computing all desired interior eigenpairs of the cubic eigenvalue problems. These ideas have led to the following four algorithms. We discuss the advantages and disadvantages of the methods, which elaborate the motivations regarding the developments of the methods. Furthermore, these considerations will be verified by the numerical experiments in Section 4.

CJD-Lk (proposed in Section 2.1):

This method is described in Algorithm 2.1, which includes the primitive locking technique.

In general, a Schur form does not exist for a cubic eigenvalue problem. Two *spurious* Ritz values (which have no meaning) thus will be obtained when the convergent eigenvectors are appended to the subspace V and the small cubic eigenvalue problems in step (2.2.i) of Algorithm 2.1 are solved. These two spurious Ritz values could affect the choice of the next desired eigenvalue. An incorrect choice of the Ritz value will slow down the overall convergence. Neglecting this disadvantage, however, CJD-Lk needs less computational cost.

CJD-Dfl (proposed in Section 2.2):

Without the locking steps, this scheme solves the original cubic eigenvalue problem (1) and the deflated cubic eigenvalue problems (8)–(9).

In this case, the convergent eigenpairs have been deflated to the infinity (Theorem 2). Therefore, the method would not produce any spurious Ritz value to affect the convergence. However, the computational cost for solving (8) becomes more and more expensive when the number of the desired eigenpairs is getting larger.

CJD-Lk-Dfl-ε_D and CJD-Lk-Dfl-ε_O (proposed in Section 2.3):

Aiming to improve the performance of CJD-Dfl, these two methods are described in Algorithm 2.2. The primitive locking technique is used.

Two variant choices of $\varepsilon = \varepsilon_D$ in (23) or $\varepsilon = \varepsilon_O$ in (24) are derived to predict the new search direction $\bar{t}$ in (21). The only difference between (21) and (5) is the choice of the skew orthogonalization vector $\tilde{p}$ in (18). This new search direction $\tilde{p}$ involves only the *original* cubic eigenvalue problem matrices, A_i, but not $\tilde{A}_i$. Consequently, we can achieve significant saving on the computation of $\tilde{r}$ and $\tilde{p}$ as the size of the desired eigenpair becomes large by using (17), (18), (21), (23) and (24). We would like to emphasize that the choice of $\tilde{p}$ and then $\bar{t}$ (by ε_D or ε_O) does share the same concept with the explicit non-equivalence deflation in the *deflated* cubic eigenvalue problem, the choice further gains the saving on computation.

On the other hand, performing the locking steps will also benefit Algorithm 2.2. Since the new search direction $\bar{t}$ in (21) is solved approximately by a suitable chosen preconditioner M_A, the inexact search direction $\bar{t}$ might slow down the convergence of the rest desired eigenpairs. Furthermore, neglecting the low-rank updates in (22) leads to slow convergences. We therefore suggest applying the locking technique to yield better overall performance in Algorithm 2.2.

3. A QUANTUM DOT MODEL PROBLEM

Semiconductor quantum dot (QD) is a structure in which the carriers are confined in all three dimensions. In many physics and engineering applications, it is essential to estimate the discrete energy states (eigenvalues) and wave functions (eigenvectors) of the QD structure. Specifically, we consider that a single electron is confined by a cylindrical InAs QD embedded in the centre of a cylindrical GaAs matrix with the same rotation axis. Figure 1 illustrates the schema of the QD structure. Moreover, the model is based on the effective-mass envelope-function approximation with one conduction band, the BenDaniel–Duke boundary conditions, and non-parabolic effective mass depending on both energy and position [1–3]. On the boundary of the QD, the finite hard-wall 3D confinement potential is induced by real discontinuity of the conduction band.

The QD model can be described by the following time-independent Schrödinger equation [1, 2] in the cylindrical co-ordinate (r, θ, z)

$$\frac{-\hbar^2}{2m_\ell(\lambda)}\left[\frac{\partial^2 F}{\partial r^2}+\frac{1}{r}\frac{\partial F}{\partial r}+\frac{1}{r^2}\frac{\partial^2 F}{\partial \theta^2}+\frac{\partial^2 F}{\partial z^2}\right]+c_\ell F=\lambda F \tag{25}$$

where $\hbar$ is the reduced Plank constant, λ is the total electron energy, and $F = F(r, \theta, z)$ is a wave function. The index ℓ depends on r and z and is used to make a distinction

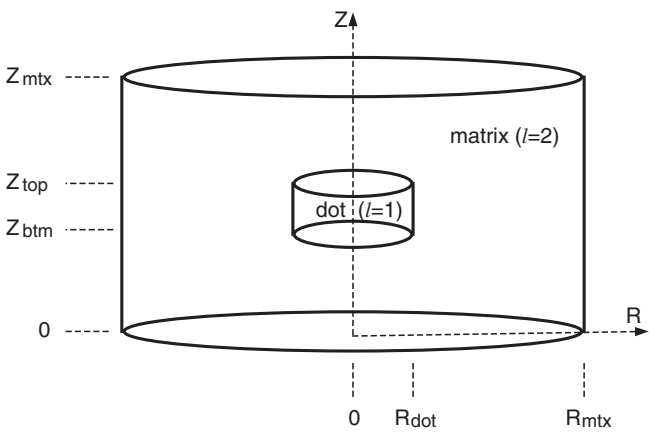

Figure 1. The quantum dot structure schema showing that a cylindrical quantum dot is embedded in the hetero-structure matrix.

between the region of the dot ($\ell = 1$) and the matrix ($\ell = 2$). Here the effective mass $m_\ell(\lambda)$ is given as

$$\frac{1}{m_\ell(\lambda)} = \frac{P_\ell^2}{\hbar^2}\left(\frac{2}{\lambda + g_\ell - c_\ell} + \frac{1}{\lambda + g_\ell - c_\ell + \delta_\ell}\right) \tag{26}$$

where P_ℓ, g_ℓ, c_ℓ, and δ_ℓ are momentum element, energy gap, confinement potential, and spin-orbit splitting in the ℓth region, respectively. Equation (25) is equipped with Dirichlet boundary conditions

$$F(r, \theta, Z_{\text{mtx}}) = F(r, \theta, 0) = F(R_{\text{mtx}}, \theta, z) = 0 \tag{27}$$

and the interface conditions

$$\begin{aligned}
\frac{-\hbar^2}{m_1(\lambda)}\frac{\partial F}{\partial r}\bigg|_{R_{\text{dot}}^-} &= \frac{-\hbar^2}{m_2(\lambda)}\frac{\partial F}{\partial r}\bigg|_{R_{\text{dot}}^+} \\
\frac{-\hbar^2}{2m_2(\lambda)}\frac{\partial F}{\partial z}\bigg|_{Z_{\text{btm}}^-} &= \frac{-\hbar^2}{2m_1(\lambda)}\frac{\partial F}{\partial z}\bigg|_{Z_{\text{btm}}^+} \\
\frac{-\hbar^2}{2m_1(\lambda)}\frac{\partial F}{\partial z}\bigg|_{Z_{\text{top}}^-} &= \frac{-\hbar^2}{2m_2(\lambda)}\frac{\partial F}{\partial z}\bigg|_{Z_{\text{top}}^+}
\end{aligned} \tag{28}$$

where Z_{mtx}, Z_{top}, and Z_{btm} denote the co-ordinate of the top of the matrix, the top of the dot, and the bottom of the dot, respectively. The radii of the dot and the matrix are denoted as R_{dot} and R_{mtx}, respectively.

To discretize the 3D cylindrical model (25), we choose non-uniform mesh points with fine meshes around the heterojunction (interface). Furthermore, the mesh points are shifted with a half mesh width in the radial direction, so that no pole conditions need to be imposed [20]. Based on the mesh points, we use the standard centred seven-point finite difference

method and two-point finite difference method to approximate Equation (25) and the interface conditions (28), respectively.

Due to the non-parabolic effective mass (26), the discretization results in a large sparse cubic eigenvalue problem of (1) with a matrix size $\rho\eta\zeta$-by-$\rho\eta\zeta$, where ρ, η, and ζ denote the mesh point numbers in the radial (r), azimuthal (θ), and axial (z) direction, respectively. However, by exploring the periodicity in the azimuth direction and applying suitable permutations and the Fourier transformation, the 3D eigenvalue problem can be decoupled into η independent $\rho\zeta$-by-$\rho\zeta$ 2D eigenproblems as

$$\begin{bmatrix} \tilde{\mathbf{G}}_1(\lambda) & & \\ & \ddots & \\ & & \tilde{\mathbf{G}}_\eta(\lambda) \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{F}}_1 \\ \vdots \\ \tilde{\mathbf{F}}_\eta \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{D}}_1(\lambda) & & \\ & \ddots & \\ & & \tilde{\mathbf{D}}_\eta(\lambda) \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{F}}_1 \\ \vdots \\ \tilde{\mathbf{F}}_\eta \end{bmatrix} \tag{29}$$

where $\tilde{\mathbf{G}}_j(\lambda)$ and $\tilde{\mathbf{D}}_j(\lambda)$ are $\rho\zeta$-by-$\rho\zeta$ matrices for $j = 1, \ldots, \eta$. Note that the mesh points associated with a certain azimuthal index number j (i.e. with the unknown vector $\tilde{F}_j$) have the same θ value. That is, these mesh points are all located on a certain vertical 2D half-plane. It is worth pointing out that only several 2D eigenproblems associated with the first j-indices need to be solved to obtain the smallest eigenvalues which are of interest in application.

The decoupled 2D eigenproblems in (29) can be straightforwardly formulated as a non-linear eigenvalue problem

$$\mathbf{G}(\lambda)\mathbf{F} = \lambda\mathbf{D}\mathbf{F} \tag{30}$$

where $\mathbf{G}(\lambda)$ is a $\rho\zeta$-by-$\rho\zeta$ matrix with entries containing λ in rational form (see (26)), $\mathbf{D}$ is the corresponding diagonal matrix, and $\mathbf{F}$ is the jth part of the associated eigenvector. By multiplying the common denominator of (26) and then simplifying the equation, we obtain a reduced version of (1), i.e.

$$\mathbf{A}(\lambda)\mathbf{F} = (\lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0)\mathbf{F} = 0 \tag{31}$$

where A_0, A_1, A_2, and A_3 are $n \times n$ real coefficient matrices.

The decoupling scheme dramatically reduces computational cost without losing accuracy. For an example as will be used in Section 4.2, a partition of the domain with 755, 280, and 360 grid points in the radial, axial, and azimuthal direction, respectively, results in a 3D system with the matrix size about 76 million. It is then reduced to several (three, for instance, in the next section) decoupled cubic eigenvalue systems (29) with the size of 211 400. The reduction from the 3D formulation to the 2D formulation (29) and full description of the matrices in these formulations are rather complicated and tedious. We refer readers to Reference [21] for more details. Nevertheless, we present the sparsity patterns of the matrices A_0, A_1, A_2, and A_3 for $\rho = 8$ and $\zeta = 12$ in Figure 2 to provide more characteristic insights about the cubic eigenvalue problems. Furthermore, the spectrum of a specific cubic eigenvalue problem with the matrices $A_i \in \mathbb{R}^{169\times 169}$, $i = 0, 1, 2, 3$, is illustrated in Figure 3. All the computed eigenvalues are plotted on the complex plane with the plus symbol. For this specific example, the target eigenvalues are located within the interval $[0, 0.35]$, and they are emphasized by the symbol $\oplus$. It is clear that the target eigenvalues are embedded in the interior of the spectrum. In the

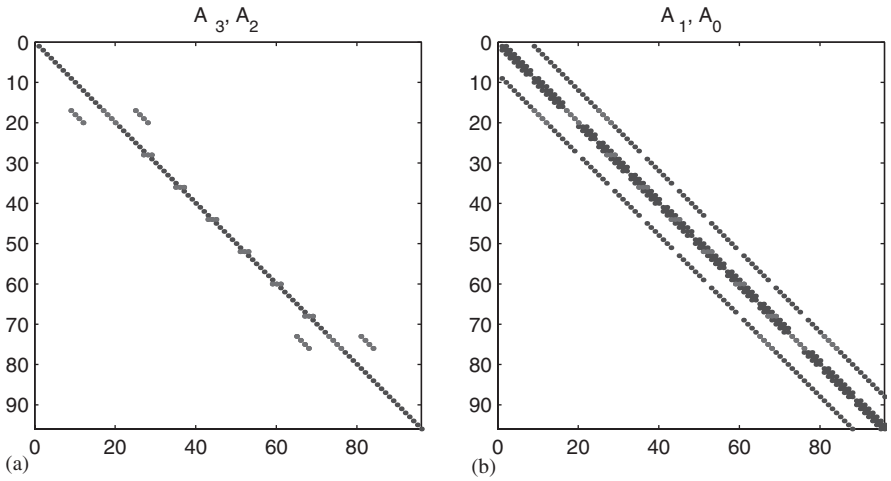

Figure 2. Sparsity patterns of the matrices A_3 and A_2, as well as A_1 and A_0 are shown in (a) and (b), respectively. Note that the rows containing non-zeros in the off-diagonal of A_3 are associated with the interface of the hetero-structure. For other A_i's, the corresponding rows have the same property.

next section, we explore the performance of the algorithms for solving the cubic eigenvalue problem (31) with more details.

4. NUMERICAL EXPERIMENTS

We implemented the proposed algorithms by Fortran 90 for the numerical experiments. All the numerical tests were performed on a Linux (Red Hat release 7.3) based workstation equipped with 2.2 GHz Xeon CPU and four gigabytes main memory. Absoft Pro Fortran [22] compiler was used to compile the programs. The timing results are in seconds.

The diameter and the height of the cylindrical QD considered here are 15 and 2.5 nm, respectively, whereas that of the matrix are 75 and 12.5 nm, respectively. The QD size is chosen so that it is approximately comparable with that of the experimental model presented in Reference [23] and the non-parabolic effect of the band structure is significant [3]. Furthermore, the semiconductor band structure parameters used in the numerical computations are $c_1 = 0.000$, $g_1 = 0.235$, $\delta_1 = 0.81$, $P_1 = 0.2875$, $c_2 = 0.350$, $g_2 = 1.590$, $\delta_2 = 0.80$, and $P_2 = 0.1993$.

4.1. Choice of the parameters

The first part of the numerical experiments shows that the timing performance can be significantly improved by tuning the following two parameters:

- The first one is the number of Ritz vectors used to span the initial search subspace whenever restarting occurs in Step (2.2.viii) of Algorithm 2.1 or 2.2. We perform the restarting scheme to keep the matrix V in reasonable sizes. The Ritz vectors extracted

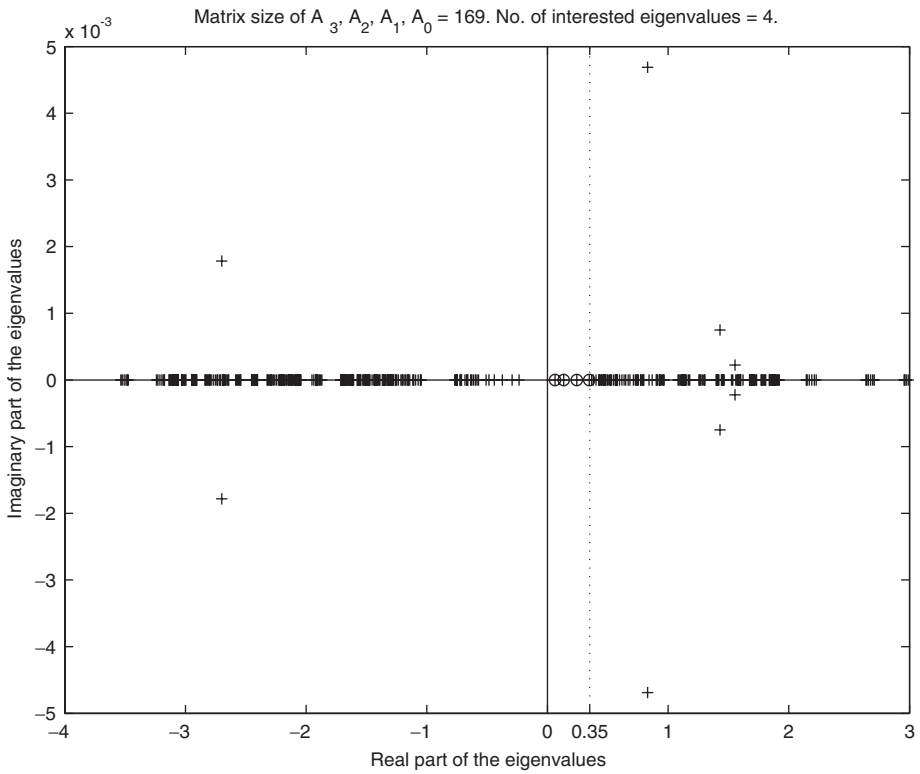

Figure 3. The spectrum of a cubic eigenvalue problem with $A_i \in \mathbb{R}^{169\times 169}$, for $i = 0, 1, 2, 3$. The desired eigenvalues marked by $\oplus$ are located in the interior of the spectrum (namely, the interval $[0, 0.35]$).

to form the new V are those associated with the Ritz values that are closest to the target eigenvalue.

- The second one is a parameter involved in the preconditioner. To compute $M_{\mathbf{A}}^{-1} r$ and $M_{\mathbf{A}}^{-1} p$ in Equation (5) or (21), we use SSOR(ω) as a preconditioner, i.e. we set

$$M_{\mathbf{A}} := \mathrm{SSOR}(\omega) = (D - \omega L)D^{-1}(D - \omega U), \quad \omega \in (0, 2)$$

where D, L, and U are the diagonal, strictly lower triangular, and strictly upper triangular matrices of $\mathbf{A}(\theta)$, respectively.

To explore the effect of the parameters, we solve the three eigenvalue problems in the form of (31) that are corresponding to the first three azimuthal indices $j = 1, 2,$and 3. We compute the smallest positive eigenvalues by running through the cases for ω chosen to be 0.1 to 1.9. The number of Ritz vectors in the initial search subspace is set to 1 to 5. The matrix size of the eigenvalue problems are 107 055.

From the computational results illustrated in Figure 4, we observe that the best choice for ω is around 1.6. Convergence is slow if we restart with only one Ritz vector even for better

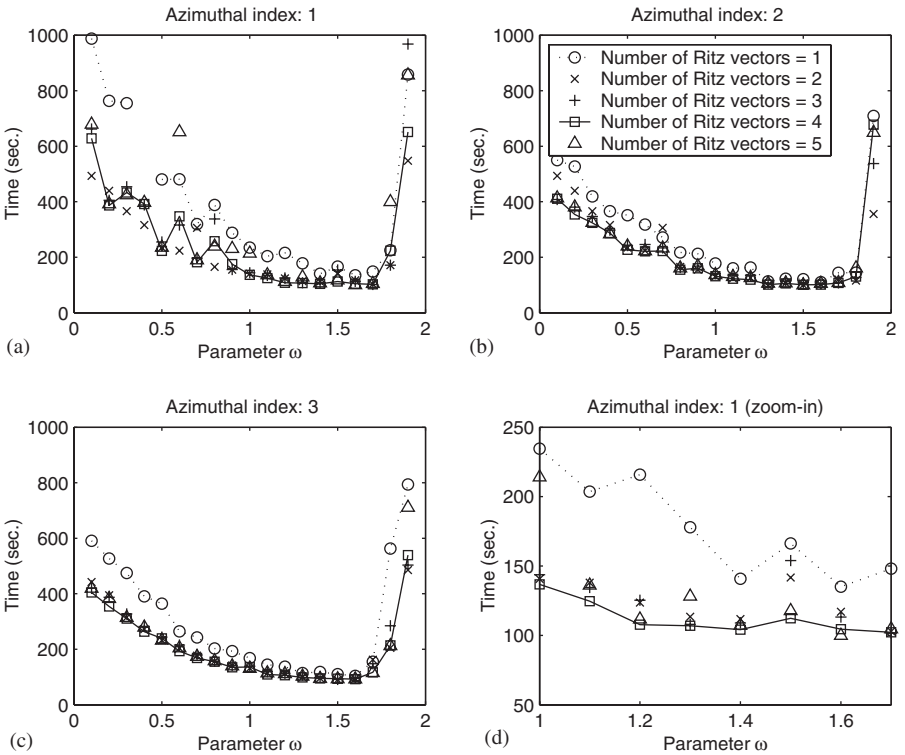

Figure 4. Comparison results of Algorithm 2.1 with various relaxation values and Ritz vectors. The timing results are marked by pluses, $\times$-marks, triangles, and squares for the algorithms CJD-Dfl, CJD-Lk, CJD-Lk-Dfl-ε_D, and CJD-Lk-Dfl-ε_O, respectively.

ω. The results obtained by using two to five Ritz vectors are quite similar. However, a closer look at part (d) shows that the case of using four Ritz vectors is most efficient for all three eigenvalue problems. In summary, to span the initial subspace when restarting, our numerical results suggest the use of four Ritz vectors associated with the four Ritz values that are closest to the target eigenvalue (which is equal to zero here).

4.2. Variants of the CJD methods for cubic eigenvalue problems

We solve the cubic eigenvalue problems by CJD-Dfl, CJD-Lk, CJD-Lk-Dfl-ε_D, and CJD-Lk-Dfl-ε_O. Comparison results of the four variants are given in Figures 5 and 6. All programmes are terminated if the residual is less than 5.0×10^{-12} or the iteration number is greater than 6000. The matrix size of the cubic eigenvalue problems (31) are 211 400.

Parts (a)–(c) of Figure 5 show the results of the slices with the index of azimuthal angle $j = 1$, 2, and 3, respectively. The timing results are marked by pluses, $\times$-marks, triangles, and squares for CJD-Dfl, CJD-Lk, CJD-Lk-Dfl-ε_D, and CJD-Lk-Dfl-ε_O, respectively. Results are not shown in the figure if the method fails to converge in a reasonable time

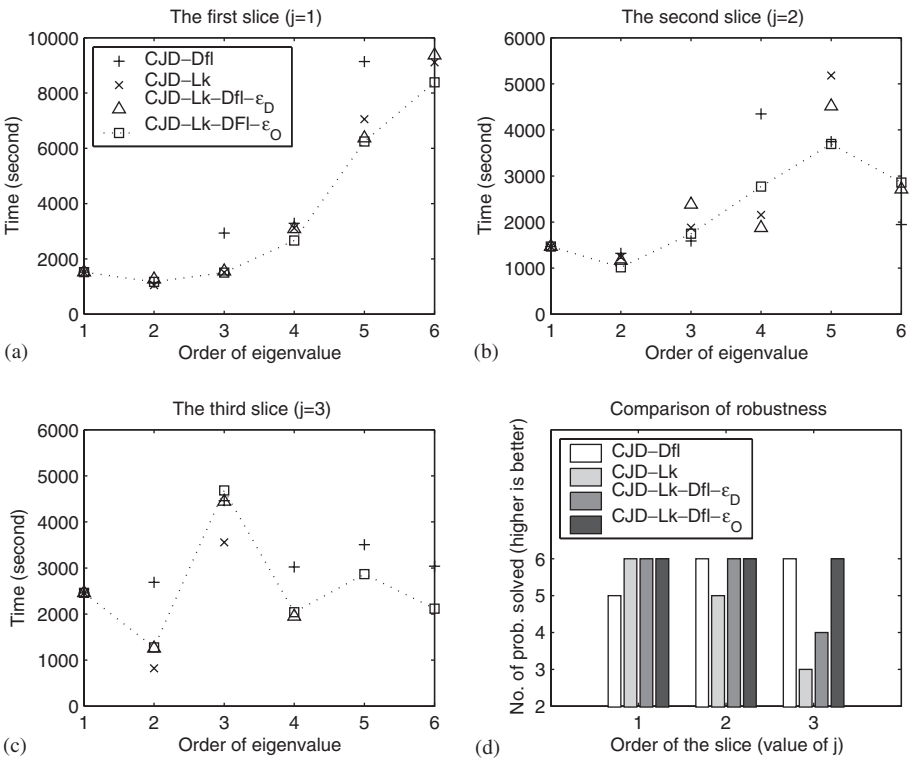

Figure 5. Comparison results in times of four methods are shown in parts (a)–(c). Numbers of eigenvalue subproblems solved successfully in the three eigenvalue problems for $j = 1, 2, 3$ are shown in part (d).

(10 000 s for the case $j = 1$ and 6000 s for the cases $j = 2, 3$). The timing marks of CJD-Lk-Dfl-ε_O are connected by dotted lines to serve as a base line. Parts (a)–(c) in Figure 5 shows that, in almost all of the cases, the CJD-Lk-Dfl-ε_O method is the quickest one among the four methods. Moreover, it can be observed from part (d) that the CJD-Lk-Dfl-ε_O method is the most robust in the sense that it converges within the iteration limit for all six eigenpairs in all three cases. Part (d) also suggests that the methods based on the explicit deflation scheme (CJD-Dfl, CJD-Lk-Dfl-ε_D, and CJD-Lk-Dfl-ε_O) are more robust than the CJD-Lk method that no explicit deflation scheme is involved.

In order to further explore the overall performances among the different methods, the 'average' timing results are presented in Figure 6. The average times are calculated by the following ways. In part (a), for each one of the three cubic eigenproblems (31) corresponding to $j = 1, 2, 3$, we consider only the computing times for the eigenpairs that all four methods converge. That is, we take the arithmetic mean of the times for the first five, five, and three eigenpairs corresponding to the problems for $j = 1$, 2, and 3, respectively. In part (b), we take the arithmetic mean of the six computing times for the first six eigenpairs as the average time if the method converges. Otherwise, the computing time is taken as the maximum

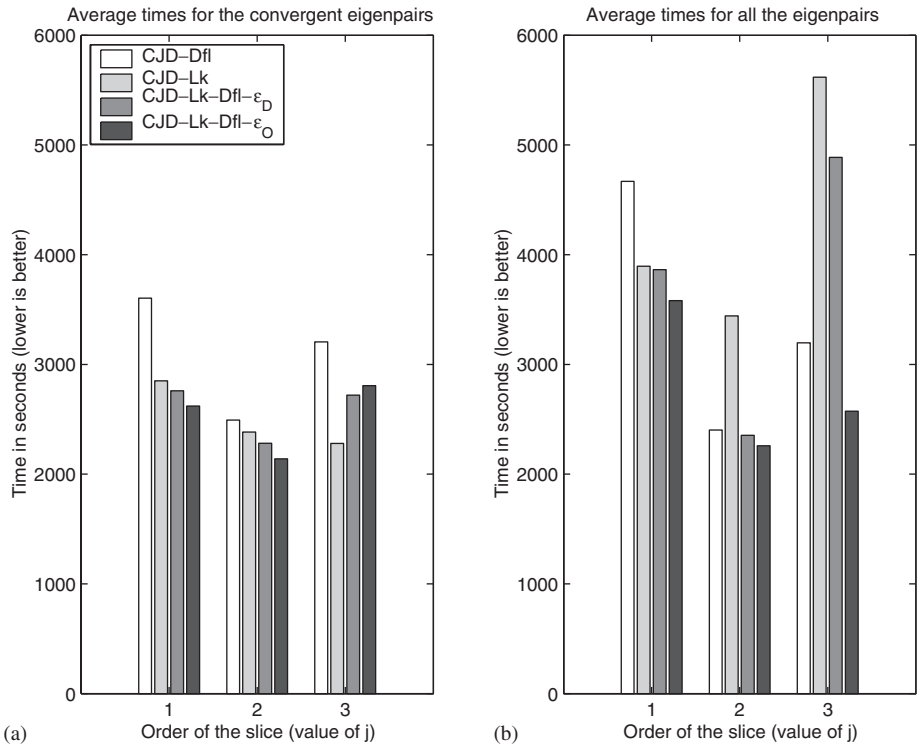

Figure 6. Comparison of efficiency. Average timing results calculated by two ways are compared for the four methods.

(i.e. 6000) iteration time for each failed eigenpair, i.e. the method fails to converge for this eigenpair. Based on Figure 6, we highlight following observations. First, if we ignore the results beyond the iteration limit, the CJD-Lk,CJD-Lk-Dfl-ε_D, and CJD-Lk-Dfl-ε_O methods are basically comparable to each other, while CJD-Lk-Dfl-ε_O is the fastest one in almost all cases. Second, without using the transformation $T(\theta)$ in (23), the CJD-Lk-Dfl-ε_O method is better than the CJD-Lk-Dfl-ε_D method. Considering the overall performance, we recommend using CJD-Lk-Dfl-ε_O for the target eigenvalue problems due to its efficiency and robustness.

We finally demonstrate the computational results of the desired eigenpairs by CJD-Lk-Dfl-ε_O. The decoupled system (29) allows us to solve several cubic eigenvalue problems independently to obtain all bound states. Table I shows all the computed eigenvalues that are less than 0.35, which is the difference between the confinement potentials c_1 and c_2. The table also presents the values of azimuthal index j, the order of the smallest eigenvalues of each slice (denoted as Ord.), and the convergent residuals of the eigensystems. The mesh size is so chosen that at least three significant digits of the computed eigenvalues remain unchanged whenever the domain is further refined.

Table I. Computational results of the discrete eigenvalues, the azimuthal indices j, the order of the smallest eigenvalues of the slices (denoted as Ord.), and the convergent residuals of the eigensystems.

λ	j	Ord.	Residual
0.0873	1	1	2.84e-12
0.1101	2	1	3.38e-12
0.1386	3	1	4.36e-12
0.1503	1	2	4.88e-12
0.1708	4	1	4.61e-12
0.1931	2	2	3.01e-12
0.2054	5	1	4.56e-12
0.2370	3	2	4.57e-12
0.2412	6	1	4.78e-12
0.2459	1	3	4.18e-12
0.2777	7	1	4.81e-12
0.2811	4	2	4.25e-12
0.2971	2	3	4.82e-12
0.3141	8	1	4.08e-12
0.3245	5	2	4.99e-12
0.3305	1	4	4.82e-12
0.3384	2	4	4.46e-12
0.3454	3	3	4.43e-12
0.3485	3	4	4.00e-12
0.3495	9	1	4.27e-12

5. CONCLUSION

Numerical methods that can be used to effectively compute multiple eigenvalues embedded in the interior of the spectrum of an eigenvalue system together with their associated eigenvectors are of great interests in a wide range of engineering and scientific areas. And yet many challenging issues in this regard remain to be explored, especially for large-scale and non-linear problems. Based on the framework of the Jacobi–Davidson method, we propose and compare several numerical algorithms for the cubic eigenvalue problems in this article. Moreover, an explicit non-equivalence deflation method for computing successive eigenpairs is developed and analysed. Several improvements by using effective preconditioners and locking and restarting techniques on these methods are also provided to yield better performance.

All numerical results are generated by using a semiconductor quantum dot model which exhibits both non-linear and large-scale properties in the resulting eigenvalue systems from the finite difference approximation in cylindrical co-ordinates. These systems are decoupled into 2D subsystems by rotational symmetry of the model problem. The order of energy levels (eigenvalues) depends critically on the number of subsystems, i.e. on the partition number in the azimuthal direction.

Based on our intensive numerical investigation, we conclude that the cubic Jacobi–Davidson method combined with the explicit deflation method and the primitive locking technique, but without using the transformation matrix $T(\theta)$ (i.e. CJD-Lk-Dfl-ε_O) is the most favourable for the QD model in terms of robustness and efficiency. This method is most robust in the sense that it converges for all tested subsystems and for all desired eigenpairs. It is most efficient in the sense that the average computing time required for all the eigenpairs is minimal.

ACKNOWLEDGEMENTS

The authors thank O. Voskoboynikov for motivating their attention to the quantum dot model discussed in this paper. The authors are grateful for the anonymous referees' comments. This work is partially supported by the National Science Council and the National Center for Theoretical Sciences in Taiwan.

REFERENCES

1. de Andrada e Silva EA, La Rocca GC, Bassani F. Spin-split subbands and magneto-oscillations in III-V asymmetric heterostructures. *Physical Review B* 1994; **50**(12):8523–8533.
2. Voskoboynikov O, Liu SS, Lee CP. Spin-dependent electronic tunnelling at zero magnetic field. *Physical Review B* 1998; **58**(23):15397–15400.
3. Li Y, Liu J-L, Voskoboynikov O, Lee CP, Sze SM. Electron energy level calculations for cylindrical narrow gap semiconductor quantum dot. *Computer Physics Communications* 2001; **140**:399–404.
4. Bai Z, Demmel J, Dongarra J, Ruhe A, van der Vorst H. *Templates for the Solution of Algebraic Eigenvalue Problems*: *A Practical Guide*. SIAM: Philadelphia, 2000.
5. Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J, Du Croz J, Greenbaum A, Hammarling S, McKenney A, Sorensen D. *LAPACK Users' Guide* (3rd edn). SIAM: Philadelphia, 1999.
6. Golub GH, Van Loan CF. *Matrix Computations* (3rd edn). Johns Hopkins University Press: Baltimore, 1996.
7. Stewart GW, Sun J-G. *Matrix Perturbation Theory*. Academic Press: New York, 1990.
8. Tisseur F. Backward error analysis of polynomial eigenvalue problems. *Linear Algebra and its Applications* 2000; **309**:339–361.
9. Fokkema DR, Sleijpen GLG, van der Vorst HA. Jacobi–Davidson style QR and QZ algorithms for the reduction of matrix pencils. *SIAM Journal on Scientific Computing* 1998; **20**(1):94–125.
10. Hochstenbach ME, van der Vorst HA. Alternatives to the Rayleigh quotient for the quadratic eigenvalue problem. *Technical Report* 1212, Department of Mathematics, University of Utrecht, P.O. Box 80.010, NL-3508 TA Utrecht, The Netherlands, 2001.
11. Sleijpen GLG, Booten AGL, Fokkema DR, van der Vorst HA. Jacobi–Davidson type methods for generalized eigenproblems and polynomial eigenproblems. *BIT* 1996; **36**(3):595–633.
12. Sleijpen GLG, van der Vorst HA. A Jacobi–Davidson iteration method for linear eigenvalue problems. *SIAM Journal on Matrix Analysis and Applications* 1996; **17**(2):401–425.
13. Meerbergen K. Locking and restarting quadratic eigenvalue solvers. *SIAM Journal on Scientific Computing* 2001; **22**(5):1814–1839.
14. Guo J-S, Lin W-W, Wang C-S. Numerical solutions for large sparse quadratic eigenvalue problems. *Linear Algebra and its Applications* 1995; **225**:57–89.
15. Guo J-S, Lin W-W, Wang C-S. Nonequivalence deflation for the solution of matrix latent value problems. *Linear Algebra and its Applications* 1995; **231**:15–45.
16. Ruhe A. Algorithms for the non-linear eigenvalue problem. *SIAM Journal on Numerical Analysis* 1973; **10**: 674–689.
17. Bai Z, Sleijpen G, van der Vorst H. Nonlinear eigenvalue problems. In *Templates for the Solution of Algebraic Eigenvalue Problems*: *A Practical Guide*, Chapter 9, Bai Z, Demmel J, Dongarra J, Ruhe A, van der Vorst H (eds), SIAM: Philadelphia, 2000.
18. Sleijpen GLG, van der Vorst HA, van Gijzen M. Quadratic eigenproblems are no problem. *SIAM News* 1996; **29**:8–9.
19. Horn RA, Johnson CA. *Matrix Analysis*. Cambridge University Press: Cambridge, 1985.
20. Lai M-C. A note on finite difference discretizations for Poisson equation on a disk. *Numerical Methods for Partial Differential Equations* 2001; **17**(3):199–203.
21. Wang W, Hwang T-M, Lin W-W, Liu J-L. Numerical methods for semiconductor heterostructures with band non-parabolicity. *Journal of Computational Physics* 2003; **190**(1):141–158.
22. Absoft Corporation. *Pro Fortran Linux User Guide*. Rochester Hills: MI, U.S.A., 2001.
23. Schoenfeld WV. Spectroscopy of the electronic structure of coupled quantum dots systems. *Ph.D. Thesis*, Materials Department, University of California, Santa Barbara, July 2000.

