East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.070816.131016a

Vol. **7**, No. 1, pp. 1-20
February 2017

Implicitly Restarted Refined Partially Orthogonal Projection Method with Deflation

Wei Wei and Hua Dai*

Department of Mathematics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China.

Received 7 August 2016; Accepted (in revised version) 13 October 2016.

Abstract. In this paper we consider the computation of some eigenpairs with smallest eigenvalues in modulus of large-scale polynomial eigenvalue problem. Recently, a partially orthogonal projection method and its refinement scheme were presented for solving the polynomial eigenvalue problem. The methods preserve the structures and properties of the original polynomial eigenvalue problem. Implicitly updating the starting vector and constructing better projection subspace, we develop an implicitly restarted version of the partially orthogonal projection method. Combining the implicit restarting strategy with the refinement scheme, we present an implicitly restarted refined partially orthogonal projection method. In order to avoid the situation that the converged eigenvalues converge repeatedly in the later iterations, we propose a novel explicit non-equivalence low-rank deflation technique. Finally some numerical experiments show that the implicitly restarted refined partially orthogonal projection method with the explicit non-equivalence low-rank deflation technique is efficient and robust.

AMS subject classifications: 65F15

Key words: Polynomial eigenvalue problem, partially orthogonal projection method, refinement, implicitly restarting, non-equivalence low-rank deflation.

1. Introduction

We consider the polynomial eigenvalue problem of finding a scalar $\lambda \in \mathbf{C}$ and nontrivial vectors $x, y \in \mathbf{C}^n$ such that

$$P(\lambda)x = 0, \quad y^H P(\lambda) = 0, \tag{1.1}$$

where $P(\lambda) = \lambda^d A_d + \lambda^{d-1} A_{d-1} + \cdots + \lambda A_1 + A_0$ with the coefficient matrices $A_i (0 \le i \le d)$ being $n \times n$ large and sparse. The scalar λ and the associated nonzero vectors x and y are called eigenvalue, right and left eigenvectors of the polynomial eigenvalue problem (1.1), respectively. Together, (λ, x) or (λ, x, y) is called an eigenpair of the polynomial eigenvalue problem (1.1). The problem is very general and includes the standard eigenvalue problem

*Corresponding author. *Email addresses:* `w.wei@nuaa.edu.cn` (W. Wei), `hdai@nuaa.edu.cn` (H. Dai)

$P(\lambda) = \lambda I - A$, the generalized eigenvalue problem $P(\lambda) = \lambda A - B$, the quadratic eigenvalue problem $P(\lambda) = \lambda^2 A + \lambda B + C$ (see, e.g., [28]) and the cubic eigenvalue problem $P(\lambda) = \lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0$ (see, e.g., [16]).

The polynomial eigenvalue problem arises from a remarkable variety of applications, such as vibration analysis of viscoelastic systems [1], structural dynamic analysis [9], stability analysis of control systems [12], numerical simulation of quantum dots [17] and so on. Considerable efforts have been devoted to the polynomial eigenvalue problem in the literature. Gohberg et al. [8] established the mathematical theory concerning matrix polynomials. Gohberg et al. [7], Higham and Tisseur et al. [5, 13], and Chu [4] developed the perturbation theory for the polynomial eigenvalue problem. Tisseur et al. [10, 27], Lawrence and Corless [20] analyzed backward error of the polynomial eigenvalue problem.

In this paper, we consider the computation of some eigenpairs with smallest eigenvalues in modulus of the polynomial eigenvalue problem (1.1). If the coefficient matrix A_0 is singular, then 0 is an eigenvalue of the polynomial eigenvalue problem (1.1), and therefore we assume that A_0 is nonsingular or, equivalently, 0 is not an eigenvalue of the polynomial eigenvalue problem (1.1). If some largest magnitude eigenvalues of the polynomial eigenvalue problem (1.1) are desired, we need only to invert the order of the coefficient matrices $A_i (0 \le i \le d)$ in $P(\lambda)$.

The classical approach for solving the polynomial eigenvalue problem is linearizing the problem (1.1) to produce an equivalent larger generalized eigenvalue problem (see, e.g., [8, 11, 21, 22]), solved using any appropriate eigensolver. The way of linearization is not unique. Using the second companion form of linearization [8], we may convert the polynomial eigenvalue problem (1.1) into the following generalized eigenvalue problem

$$Cy = \theta G y, \tag{1.2}$$

where $\theta = 1/\lambda$,

$$C = \begin{pmatrix} -A_1 & I & & \\ -A_2 & & \ddots & \\ \vdots & & & I \\ -A_d & & & 0 \end{pmatrix}, \qquad G = \begin{pmatrix} A_0 & & & \\ & I & & \\ & & \ddots & \\ & & & I \end{pmatrix},$$

$$y = \begin{pmatrix} x \\ -\lambda A_2 x - \cdots - \lambda^{d-1} A_d x \\ \vdots \\ -\lambda A_d x \end{pmatrix}. \tag{1.3}$$

Since A_0 is nonsingular, the generalized eigenvalue problem (1.3) may be further reduced to the following standard eigenvalue problem

$$\left(G^{-1}C\right) y = \theta y, \tag{1.4}$$

or

$$\left(CG^{-1}\right)z = \theta z, \tag{1.5}$$

where $z = Gy$. However, the linearization technique will enlarge the size of the problem, and matrix structures and spectral properties of the original polynomial eigenvalue problem will not be preserved, and more importantly, the linearized eigenvalue problem is more ill-conditioned [27]. In recent 20 years, many researchers have been developing numerical methods that work directly with the original data of the polynomial eigenvalue problem for avoiding these disadvantages. In these methods, the polynomial eigenvalue problem is projected onto a properly chosen low dimensional subspace to reduce directly a new polynomial eigenvalue problem with matrix dimension of low order. The reduced polynomial eigenvalue problem can then be solved by the linearization method. These methods include Jacobi-Davidson method [18,25] and Krylov-type subspace method [2,14]. Jacobi-Davidson method targets at one eigenvalue at one time with local convergence. However, if the desired eigenvalues of (1.1) form a cluster of nearby eigenvalues, then Jacobi-Davidson method has difficulties in detecting and resolving such a cluster. Bao et al. [2] proposed a generalized Arnoldi method for solving the polynomial eigenvalue problem. However, the generalized Arnoldi method is difficult to restart implicitly.

In the spirit of the Hessenberg-triangular decomposition of a matrix pencil, Huang et al. [15] proposed a semiorthogonal generalized Arnoldi procedure for the matrix pencil resulting from the second companion form of linearization for the quadratic eigenvalue problem, and developed a semiorthogonal generalized Arnoldi method for solving quadratic eigenvalue problem. Inspired by the semiorthogonal generalized Arnoldi method, Wei and Dai [29] presented a partially orthogonal projection method for solving the polynomial eigenvalue problem and derived its refinement scheme by using the refined projection principle [19]. In this paper, we further investigate the partially orthogonal projection method. Based on the implicitly shifted QZ iteration [26], we develop an implicitly restarted version of the partially orthogonal projection method. Combining the implicit restarting strategy with the refinement scheme, we present an implicitly restarted refined partially orthogonal projection method for solving the polynomial eigenvalue problem.

It is well known that implicit deflation technique based on Schur forms (see, e.g., [24]) combined with eigensolver performs well for the linear eigenvalue problem. However, in the polynomial eigenvalue problem, it is not clear how to incorporate an implicit deflation technique because a Schur form does not exist to the polynomial eigenvalue problem. Our another contribution in this paper is that we will develop an explicit non-equivalence low-rank deflation technique for the polynomial eigenvalue problem. Suppose that (λ_1, x_1, y_1) is a given eigenpair. The new deflation technique transforms the original polynomial eigenproblem (1.1) to a new deflated polynomial eigenproblem so that it has an infinite eigenvalue transformed from λ_1 and keeps the remaining eigenpairs invariant.

This paper is organized as follows. In Section 2, we give a brief description of the partially orthogonal decomposition and the explicitly restarted refined partially orthogonal projection method for the polynomial eigenvalue problem. In Section 3, we develop an implicitly restarted partially orthogonal projection method for the polynomial eigenvalue problem, and propose the implicitly restarted refined partially orthogonal projection method. In Section 4, we describe the explicit non-equivalence low-rank deflation technique for the polynomial eigenvalue problem, and present an implicitly restarted refined partially orthogonal projection method with the explicit non-equivalence low-rank deflation. Numerical examples are shown in Section 5 and the concluding remarks are given in Section 6.

Throughout this paper, we use the following notations. I_n denotes the $n \times n$ identity matrix, which can be written as I for short without confusion. e_j denotes the j-th column of the identity matrix. The superscript T and H denote the transpose and conjugate transpose, respectively, for a vector or a matrix. $\|\cdot\|_2$ denotes the Euclidean vector norm or the induced matrix norm, and $\|\cdot\|_F$ the Frobenius matrix norm. For $A \in \mathbf{C}^{n\times n}, v \in \mathbf{C}^n$ and a positive integer m, $K[A,v,m] \triangleq [v,Av,\cdots,A^{m-1}v]$ denotes the Krylov matrix, then $span(K[A,v,m])$ is the Krylov subspace spanned by A and v. We also adopt the following MATLAB notations: $A(i:j,k:l)$ denotes the submatrix of the matrix A that consists of the intersection of the rows i to j and the columns k to l, $A(i:j,:)$ and $A(:,k:l)$ select the rows i to j and the columns k to l of A, respectively.

2. Partially Orthogonal Projection Method and Its Refinement Scheme

In this section, we give a brief description of the partially orthogonal decomposition and the explicitly restarted partially orthogonal projection method with refinement technique for the polynomial eigenvalue problem, see [29] for details.

Definition 2.1 (cf. Ref. [29]). Given coefficient matrices $A_i (0 \le i \le d)$ and a positive integer $m \ll n$, the mth order partially orthogonal decomposition of the polynomial eigenvalue problem (1.1) is defined by the following relations

$$\begin{pmatrix} -A_1 & I & & \\ -A_2 & & \ddots & \\ \vdots & & & I \\ -A_d & & & 0 \end{pmatrix} \begin{pmatrix} Q_m \\ P_m^{(1)} \\ \vdots \\ P_m^{(d-1)} \end{pmatrix} = \begin{pmatrix} V_m \\ U_m^{(1)} \\ \vdots \\ U_m^{(d-1)} \end{pmatrix} H_m + \begin{pmatrix} g_m \\ f_m^{(1)} \\ \vdots \\ f_m^{(d-1)} \end{pmatrix} e_m^T, \tag{2.1}$$

$$\begin{pmatrix} A_0 & & & \\ & I & & \\ & & \ddots & \\ & & & I \end{pmatrix} \begin{pmatrix} Q_m \\ P_m^{(1)} \\ \vdots \\ P_m^{(d-1)} \end{pmatrix} = \begin{pmatrix} V_m \\ U_m^{(1)} \\ \vdots \\ U_m^{(d-1)} \end{pmatrix} R_m, \tag{2.2}$$

$$Q_m^H Q_m = V_m^H V_m = I_m, \qquad V_m^H g_m = 0, \tag{2.3}$$

where $Q_m, V_m, P_m^{(i)}, U_m^{(i)} \in \mathbf{C}^{n\times m}$, $g_m, f_m^{(i)} \in \mathbf{C}^n (1 \le i \le d-1)$, $H_m \in \mathbf{C}^{m\times m}$ is an upper Hessenberg matrix, and $R_m \in \mathbf{C}^{m\times m}$ is an upper triangular matrix.

Let

$$Z_m = \begin{pmatrix} Q_m \\ P_m^{(1)} \\ \vdots \\ P_m^{(d-1)} \end{pmatrix}, \qquad Y_m = \begin{pmatrix} V_m \\ U_m^{(1)} \\ \vdots \\ U_m^{(d-1)} \end{pmatrix}, \qquad \eta_m = \begin{pmatrix} g_m \\ f_m^{(1)} \\ \vdots \\ f_m^{(d-1)} \end{pmatrix}, \tag{2.4}$$

then the equations (2.1) and (2.2) can be written as

$$CZ_m = Y_m H_m + \eta_m e_m^T, \tag{2.5}$$

$$GZ_m = Y_m R_m. \tag{2.6}$$

Partially orthogonality means that Q_m and V_m satisfy the requirements in (2.3), which guarantees the linear independence of columns of Z_m and Y_m, respectively. Wei and Dai [29] discussed the existence and uniqueness of the partially orthogonal decomposition and presented the following algorithm for generating the partially orthogonal decomposition (POD).

Algorithm 2.1. POD(m): Partially orthogonal decomposition process.

Input: Coefficient matrices $A_i(0 \le i \le d)$, initial vectors $q_1, p_1^{(i)}(1 \le i \le d-1)$ and the dimension m of subspace.

Output: $Z_m, Y_m, H_m, R_m, \eta_m$ satisfying (2.5) and (2.6).

```
1. $q_1 = q_1/\|q_1\|_2$, $p_1^{(i)} = p_1^{(i)}/\|q_1\|_2$, $R_1 = \|A_0 q_1\|_2$, $v_1 = A_0 q_1/R_1$,
   $u_1^{(i)} = p_1^{(i)}/R_1$, $H_1 = v_1^H(-A_1 q_1 + p_1^{(1)})$, $g_1 = -A_1 q_1 + p_1^{(1)} - v_1 H_1$,
   $f_1^{(i)} = -A_{i+1} q_1 + p_1^{(i+1)} - u_1^{(i)} H_1 (1 \le i \le d-1)$;
2. for j = 2 : m
   if $g_j \neq 0$ then
   $\gamma = \|g_j\|_2$, $v = g_j/\gamma$, $u^{(i)} = f_j^{(i)}/\gamma$, $\rho = \|(I_j - Q_j Q_j^H) A_0^{-1} v\|_2^{-1}$,
   $r = -\rho R_j Q_j^H A_0^{-1} v$, $q = \rho(I_j - Q_j Q_j^H) A_0^{-1} v$, $p^{(i)} = U_j^{(i)} r + \rho u^{(i)}$,
   $h = V_j^H(-A_1 q + p^{(1)})$, $\alpha = v^H(-A_1 q + p^{(1)})$,
   $g_{j+1} = -A_1 q + p^{(1)} - V_j h - \alpha v$,
   $f_{j+1}^{(i)} = -A_{i+1} q + p^{(i+1)} - U_j^{(i)} h - \alpha u^{(i)} (1 \le i \le d-1)$.
   else
   if $f_j^{(k)} \in span\left\{ f_i^{(k)} \middle| g_i = 0, i < j \right\} (1 \le k \le d-1)$ then
   breakdown
   else
   $\gamma = 1$, $v = 0$, $u^{(i)} = f_j^{(i)}$, $\rho = 1$, $r = 0$, $q = 0$, $p^{(i)} = u^{(i)}$,
   $h = V_j^H p^{(1)}$, $\alpha = 0$, $g_{j+1} = p^{(1)} - V_j h$, $f_{j+1}^{(i)} = p^{(i+1)} - U_j^{(i)} h$
   $(1 \le i \le d-1)$.
   endif
   endif
```

$$Z_{j+1} = \left(Z_j, \begin{pmatrix} q \\ p^{(1)} \\ \vdots \\ p^{(d-1)} \end{pmatrix} \right), \qquad Y_{j+1} = \left(Y_j, \begin{pmatrix} v \\ u^{(1)} \\ \vdots \\ u^{(d-1)} \end{pmatrix} \right),$$

$$H_{j+1} = \begin{pmatrix} H_j & h \\ \gamma e_j^T & \alpha \end{pmatrix}, \qquad R_{j+1} = \begin{pmatrix} R_j & r \\ 0 & \rho \end{pmatrix}, \qquad \eta_{j+1} = \begin{pmatrix} g_{j+1} \\ f_{j+1}^{(1)} \\ \vdots \\ f_{j+1}^{(d-1)} \end{pmatrix}.$$

endfor

Remark 2.1. Computing $A_0^{-1}v$ or solving a linear system $A_0x = v$ is needed for three times in Algorithm 2.1. To make the computation more efficient, a LU factorization of A_0 should be made available outside of the first for-loop of the algorithm. From (2.2), we know that $P_m^{(i)}$ can be completely determined by $U_m^{(i)}$, hence we can use $U_m^{(i)}(:,1:j)R_m(1:j,j)$ to replace $p_j^{(i)}(1 \le i \le d-1)$ when they are needed. A breakdown of Algorithm 2.1 is discussed in [29]. We can use the modified Gram-Schmidt procedure or the QR factorization to check whether a breakdown occurs in Algorithm 2.1. The computational cost of the partially orthogonal decomposition process is slightly more expensive than the generalized Arnoldi procedure [2] since a partially orthogonalization is implemented at the each step.

Let $\mathcal{Q}_m = span\{Q_m\}$. Using the Rayleigh-Ritz projection technique on the subspace $\mathcal{Q}_m$, Wei and Dai [29] developed an orthogonal projection method for solving the polynomial eigenvalue problem (1.1). An approximation eigenpair (θ, z), where $\theta \in \mathbf{C}$ and $z \in \mathcal{Q}_m$, is seeked by imposing the following Galerkin condition

$$(\theta^d A_d + \theta^{d-1}A_{d-1} + \cdots + \theta A_1 + A_0)z \perp \mathcal{Q}_m. \tag{2.7}$$

Since $z \in \mathcal{Q}_m$, z can be expressed as $z = Q_m\xi$, and then (2.7) may be rewritten equivalently as

$$(\theta^d \hat{A}_d + \theta^{d-1}\hat{A}_{d-1} + \cdots + \theta \hat{A}_1 + \hat{A}_0)\xi = 0, \tag{2.8}$$

where

$$\hat{A}_i = Q_m^H A_i Q_m, \qquad 0 \le i \le d. \tag{2.9}$$

From (2.9), the new coefficient matrices $\hat{A}_i(0 \le i \le d)$ have the same structures and properties as $A_i(0 \le i \le d)$, like symmetry. For the projected polynomial eigenvalue problem (2.8), the linearization technique can be applied, and the transformed generalized eigenvalue problem can be solved by the QZ method [23]. A difficulty with the method is that both the computational cost and the storage become increasingly expensive as the dimension m of the projected subspace $\mathcal{Q}_m$ increases. To remedy the difficulty, an explicitly restarted version (ERPOP) of the method is presented in [29] as follows.

Algorithm 2.2. ERPOP(m): Explicitly restarted partially orthogonal projection method.

Input: Coefficient matrices $A_i(0 \le i \le d)$, initial vectors $q_1, p_1^{(i)}(1 \le i \le d-1)$, dimension m of the projected subspace, number k of the desired eigenpairs.

Output: k approximate eigenpairs and their relative residuals.

1. Run Algorithm 2.1 to get Q_m;
2. Compute $\hat{A}_i(0 \le i \le d)$ by (2.9);
3. Solve projected polynomial eigenvalue problem (2.8) for $(\theta_i, \xi_i)(1 \le i \le dm)$ with $|\theta_1| \le |\theta_2| \le \cdots \le |\theta_{dm}|$, and get k Ritz pairs (θ_i, x_i) with $x_i = Q_m\xi_i / \|Q_m\xi_i\|_2 (1 \le i \le k)$;
4. Compute k relative residuals $\alpha_i = \frac{\|(\theta_i^d A_d + \cdots + \theta_i A_1 + A_0)x_i\|_2}{|\theta_i|^d \|A_d\|_F + \cdots + |\theta_i| \|A_1\|_F + \|A_0\|_F}(1 \le i \le k)$, if all k relative residuals α_i are satisfied, then output (θ_i, x_i) and $\alpha_i(1 \le i \le k)$, and stop; otherwise set
$$\begin{pmatrix} q_1 \\ p_1^{(1)} \\ \vdots \\ p_1^{(d-1)} \end{pmatrix} = \sum_{i=1}^{k} \alpha_i \begin{pmatrix} x_i \\ -\theta_i A_2 x_i - \cdots - \theta_i^{d-1} A_d x_i \\ \vdots \\ -\theta_i A_d x_i \end{pmatrix}$$
as new initial vectors and go to 1.

The Ritz vectors obtained by the Rayleigh-Ritz projection cannot be guaranteed to converge even if the approximate eigenvalues or Ritz values do. In order to solve this problem, using the refined projection principle [19], Wei and Dai [29] derived the refined partially orthogonal projection method for solving polynomial eigenvalue problem.

Let $\theta_i(1 \le i \le k)$ be the Ritz values obtained by Algorithm 2.2. The refinement strategy seeks k unit vectors $\hat{x}_i \in \mathcal{Q}_m$, called refined Ritz vectors, such that
$$\hat{x}_i = \underset{x \in \mathcal{Q}_m, \|x\|_2 = 1}{\arg\min} \left\| (\theta_i^d A_d + \cdots + \theta_i A_1 + A_0)x \right\|_2. \tag{2.10}$$

It was shown that $\hat{x}_i = Q_m \hat{\xi}_i$ where $\hat{\xi}_i$ is the right singular vector corresponding to the smallest singular value σ_i of the matrix $T_i = \theta_i^d A_d Q_m + \cdots + \theta_i A_1 Q_m + A_0 Q_m \in \mathbf{C}^{n \times m}$. It follows from (2.1) and (2.2) that
$$T_i = [V_m, g_m, U_m^{(1)}, f_m^{(1)}, \cdots, U_m^{(d-1)}, f_m^{(d-1)}] \begin{pmatrix} -\theta_i H_m + R_m \\ -\theta_i e_m^T \\ -\theta_i^2 H_m + \theta_i R_m \\ \vdots \\ -\theta_i^d e_m^T \end{pmatrix}. \tag{2.11}$$

Since V_m is orthonormal, the QR factorization of the matrix $[V_m, g_m, U_m^{(1)}, f_m^{(1)}, \cdots, U_m^{(d-1)}, f_m^{(d-1)}]$ is of the form
$$[V_m, g_m, U_m^{(1)}, f_m^{(1)}, \cdots, U_m^{(d-1)}, f_m^{(d-1)}] = [V_m, \tilde{g}_m, \tilde{U}_m^{(1)}, \tilde{f}_m^{(1)}, \cdots, \tilde{U}_m^{(d-1)}, \tilde{f}_m^{(d-1)}]\tilde{R}, \tag{2.12}$$

where $[V_m, \tilde{g}_m, \tilde{U}_m^{(1)}, \tilde{f}_m^{(1)}, \cdots, \tilde{U}_m^{(d-1)}, \tilde{f}_m^{(d-1)}]$ is orthonormal, $\tilde{R} \in \mathbf{C}^{(m+1)d\times(m+1)d}$ is an upper triangular matrix. Let

$$\tilde{T}_i = \tilde{R}\begin{pmatrix} -\theta_i H_m + R_m \\ -\theta_i e_m^T \\ -\theta_i^2 H_m + \theta_i R_m \\ \vdots \\ -\theta_i^d e_m^T \end{pmatrix} \in \mathbf{C}^{(m+1)d\times m}, \tag{2.13}$$

then σ_i is the smallest singular value of $\tilde{T}_i$, and $\hat{\xi}_i$ is the corresponding right singular vector. The approximate eigenpair $(\theta_i, \hat{x}_i)$, called the refined Ritz pair, is better than the Ritz pair (θ_i, x_i) due its minimal property. The corresponding relative residual can be expressed by

$$\alpha_i = \frac{\sigma_i}{|\theta_i|^d\|A_d\|_F + \cdots + |\theta_i|\|A_1\|_F + \|A_0\|_F}, 1 \le i \le k. \tag{2.14}$$

Wei and Dai [29] presented the following explicitly restarted refined partially orthogonal projection (ERRPOP) method for solving polynomial eigenvalue problem (1.1).

Algorithm 2.3. ERRPOP(m): Explicitly restarted refined partially orthogonal projection method.

Input: Coefficient matrices $A_i (0 \le i \le d)$, initial vectors $q_1, p_1^{(i)} (1 \le i \le d-1)$, the dimension m of the projected subspace, and the number k of the desired eigenpairs.

Output: k refined Ritz pairs and their relative residuals.

```
1. Implement Steps 1 and 2 in Algorithm 2.2;
2. Compute all the eigenvalues $\{\theta_i\}_{i=1}^{dm}$ of the projected polynomial eigenvalue problem
   (2.8) with $|\theta_1| \le |\theta_2| \le \cdots \le |\theta_{dm}|$;
3. Compute the QR factorization (2.12) of $[V_m, g_m, U_m^{(1)}, f_m^{(1)}, \cdots, U_m^{(d-1)}, f_m^{(d-1)}]$;
4. for $i = 1, 2, \cdots, k$ do
     4.1. Compute the matrix $\tilde{T}_i$ as defined in (2.13);
     4.2. Compute the smallest singular value $\sigma_i$ of $\tilde{T}_i$ and the corresponding right singu-
          lar vector $\hat{\xi}_i$, the refined Ritz vector $\hat{x}_i = Q_m\hat{\xi}_i$ and the corresponding relative
          residual (2.14);
   endfor
5. If all $k$ relative residuals $\alpha_i$ are satisfied, then output $(\theta_i, \hat{x}_i)$ and $\alpha_i (1 \le i \le k)$, and
   stop; otherwise set
   $$\begin{pmatrix} q_1 \\ p_1^{(1)} \\ \vdots \\ p_1^{(d-1)} \end{pmatrix} = \sum_{i=1}^{k} \alpha_i \begin{pmatrix} \hat{x}_i \\ -\theta_i A_2 \hat{x}_i - \cdots - \theta_i^{d-1} A_d \hat{x}_i \\ \vdots \\ -\theta_i A_d \hat{x}_i \end{pmatrix}$$
   as new initial vectors and go to 1.
```

3. Implicitly Restarted Refined Partially Orthogonal Projection Method

Similar to the implicitly restarted refined semiorthogonal generalized Arnoldi method [15] for the quadratic eigenvalue problem, in this section we will develop the implicitly restarted refined partially orthogonal projection method for the polynomial eigenvalue problem by using the forward implicitly shifted QZ iteration [26].

Using Algorithm 2.1, we have computed the mth order partially orthogonal decomposition (2.3)-(2.6). For given shifts μ_i $(1 \le i \le s, s = m-k)$, the QR factorizations of the upper Hessenberg matrices $H_m - \mu_i R_m$ are computed as follows.

$$H_m - \mu_i R_m = E_i \hat{R}_i \quad (i = 1, 2, \cdots, s),$$

where E_i is a unitary matrix and $\hat{R}_i$ is an upper triangular matrix. It is straightforward to show that E_i is an upper Hessenberg matrix. Consider the triangular-orthogonal factorization

$$E_i^H R_m = \tilde{R}_i F_i,$$

where $\tilde{R}_i$ is an upper triangular matrix and F_i is a unitary matrix, and then F_i is a lower Hessenberg matrix [26].

Let

$$\begin{aligned}
&E = E_1 E_2 \cdots E_s, \qquad F = F_s F_{s-1} \cdots F_1,\\
&Q_m^+ = Q_m F^H = [Q_k^+, Q_s^+], \qquad V_m^+ = V_m E = [V_k^+, V_s^+],\\
&Z_m^+ = Z_m F^H = [Z_k^+, Z_s^+], \qquad Y_m^+ = Y_m E = [Y_k^+, Y_s^+] = [y_1^+, y_2^+, \cdots, y_m^+],\\
&H_m^+ = E^H H_m F^H = \begin{pmatrix} H_k^+ & H_{12}^+ \\ H_{21}^+ & H_{22}^+ \end{pmatrix}, \qquad R_m^+ = E^H R_m F^H = \begin{pmatrix} R_k^+ & R_{12}^+ \\ R_{21}^+ & R_{22}^+ \end{pmatrix},
\end{aligned} \tag{3.1}$$

where $Q_k^+, V_k^+ \in \mathbf{C}^{n\times k}$, $Z_k^+, Y_k^+ \in \mathbf{C}^{dn\times k}$, $H_k^+, R_k^+ \in \mathbf{C}^{k\times k}$, $y_j^+ \in \mathbf{C}^{dn} (1 \le j \le m)$. It follows from (2.5), (2.6) and (3.1) that

$$CZ_m^+ = Y_m^+ H_m^+ + \eta_m e_m^T F^H, \tag{3.2}$$

$$GZ_m^+ = Y_m^+ R_m^+. \tag{3.3}$$

It is easy to verify that H_m^+ is still an upper Hessenberg matrix, R_m^+ is an upper triangular matrix, and $(Q_m^+)^H Q_m^+ = (V_m^+)^H V_m^+ = I_m$.

Note that F_i is a lower Hessenberg matrix, and $e_m^T F_1^H = (0, \cdots, 0, \alpha, \beta)$. Then the first $k-1$ entries of $e_m^T F^H$ must be zeros. Since H_m^+ is the upper Hessenberg matrix, the entry $h_{k+1,k}^+$ in the top right corner of H_{21}^+ may be nonzero, and the others are zero. Let

$$e_m^T F^H = (0, \cdots, 0, \alpha_k^{(m)}, \alpha_{k+1}^{(m)}, \cdots, \alpha_m^{(m)}), \tag{3.4}$$

$$\eta_k^+ = h_{k+1,k}^+ y_{k+1}^+ + \alpha_k^{(m)} \eta_m = \begin{pmatrix} g_m^+ \\ f_m^+ \end{pmatrix}, \tag{3.5}$$

where $g_m^+ \in \mathbf{C}^n$.

Comparing the first k columns of (3.2) and (3.3), and using (3.1), (3.4) and (3.5), we obtain

$$CZ_k^+ = Y_k^+ H_k^+ + \eta_k^+ e_k^T, \tag{3.6}$$
$$GZ_k^+ = Y_k^+ R_k^+. \tag{3.7}$$

It follows from (2.3) and (3.1) that

$$(Q_k^+)^H Q_k^+ = (V_k^+)^H V_k^+ = I_k, \qquad (V_k^+)^H g_m^+ = 0. \tag{3.8}$$

Thus, (3.6), (3.7) and (3.8) constitute the kth order partially orthogonal decomposition.

A good selection of shift is a key for success of the implicit restart technique. A common choice of the shift value is to choose unwanted Ritz values. When we solve the projected polynomial eigenvalue problem (2.8) to get md eigenvalues $\{\theta_i\}_{i=1}^{dm}$ with $|\theta_1| \leq |\theta_2| \leq \cdots \leq |\theta_{dm}|$ and select k Ritz values $\{\theta_i\}_{i=1}^{k}$ as approximations to the desired eigenvalues, we may directly use the reciprocal values of the remaining unwanted Ritz values as shifts called exact shifts. We always take the reciprocal values of the s unwanted Ritz values which are farthest from the target as shifts, namely

$$\mu_i = 1/\theta_{(d-1)m+k+i}, \qquad 1 \leq i \leq s. \tag{3.9}$$

The refined partially orthogonal projection method can not only improve the accuracy of the Ritz vectors but also provide more accurate approximations to some of the unwanted eigenvalues. Suppose that ξ_i^+ is the right singular vector corresponding to the smallest singular value of the matrix $\tilde{T}_i (dm-s+1 \leq i \leq dm)$. Then $\hat{x}_i = Q_m \hat{\xi}_i (dm-s+1 \leq i \leq dm)$ are the refined Ritz vectors corresponding to the unwanted Ritz values $\{\theta_i\}_{i=dm-s+1}^{dm}$. Similar to the refined shifts suggested in [15], we can find better shifts based on the unwanted refined Ritz vectors $\{\hat{x}_i\}_{i=dm-s+1}^{dm}$. For the refined Ritz vector $\hat{x}_i$ of the polynomial eigenvalue problem (1.1), a popular method for deriving a more accurately approximate eigenvalue θ from $\hat{x}_i$ is to impose the Galerkin condition

$$(\theta^d A_d + \theta^{d-1} A_{d-1} + \cdots + \theta A_1 + A_0)\hat{x}_i \perp \hat{x}_i. \tag{3.10}$$

(3.10) is equivalent to the following polynomial equation

$$a_d \theta^d + \cdots + a_1 \theta + a_0 = 0, \tag{3.11}$$

where $a_j = (\xi_i^+)^H \hat{A}_j \xi_i^+$, and $\hat{A}_j$ is defined by (2.9). If ω_i is a root of the polynomial equation (3.11), then its reciprocal value would be better candidate for the implicit restart. Using MATLAB function *roots*, we can find d roots of the polynomial equation (3.11). Consequently, $\{\theta_i\}_{i=dm-s+1}^{dm}$ are s unwanted Ritz values that are farthest from our target and $\{\hat{x}_i\}_{i=dm-s+1}^{dm}$ are the corresponding refined Ritz vectors, we can obtain all sd roots, written as $\{\omega_i\}_{i=1}^{sd}$ with $|\omega_1| \leq |\omega_2| \leq \cdots \leq |\omega_{sd}|$, then we choose the s values from $\{\omega_i\}_{i=1}^{sd}$ that are farthest from our target and take their reciprocal values as the shifts for the restarting process, namely

$$\mu_i = 1/\omega_{sd-s+i}, \qquad 1 \leq i \leq s \tag{3.12}$$

and call them the refined shifts. Now, we describe the implicitly restarted refined partially orthogonal projection (IRRPOP) method as follows.

Algorithm 3.1. IRRPOP(m): Implicitly restarted refined partially orthogonal projection method.

Input: Coefficient matrices $A_i (0 \le i \le d)$, initial vectors $q_1, p_1^{(i)} (1 \le i \le d-1)$, dimension m of the projected subspace, and number k of the desired eigenpairs.

Output: k approximate eigenpairs and their relative residuals.

```
1. Implement Steps 1-4 in Algorithm 2.3;
2. If all $k$ relative residuals $\alpha_i$ are satisfied, then output $(\theta_i, \hat{x}_i)$ and $\alpha_i (1 \le i \le k)$, and
   stop; otherwise select $s := m-k$ refined shifts $\mu_i (1 \le i \le s)$ as (3.12), and set $\varepsilon := e_m^T$;
3. for $i = 1, 2, \cdots, s$ do
   3.1. Compute unitary matrices $E_i$ and $F_i$ by using the forward implicitly shifted QZ
        iteration for each shift $\mu_i$ so that $E_i^H H_m F_i^H$ and $E_i^H R_m F_i^H$ are upper Hessenberg
        and upper triangular matrices, respectively.
   3.2. Update $H_m := E_i^H H_m F_i^H$, $R_m := E_i^H R_m F_i^H$, $Z_m := Z_m F_i^H$, $Y_m := Y_m E_i$ and $\varepsilon :=$
        $\varepsilon F_i^H$;
   endfor
4. Set $\eta_k := H_m(k+1,k) Y_m(:,k+1) + \varepsilon(k) \eta_m$,
   $Z_k := Z_m(:,1:k)$,    $Y_k := Y_m(:,1:k)$,
   $H_k := H_m(1:k,1:k)$,    $R_k := R_m(1:k,1:k)$ and go to 1.
```

4. Non-Equivalence Low-Rank Deflation

In this section, we present a novel non-equivalence low-rank deflation technique for the polynomial eigenvalue problem. Suppose that $\lambda_1, \lambda_2, \cdots, \lambda_r (r \ge 1)$ are the converged eigenvalues of the polynomial eigenvalue problem (1.1), and $x_1, x_2, \cdots, x_r$ and $y_1, y_2, \cdots, y_r$ are the associated right and left eigenvectors, respectively. Let

$$\Lambda_1 = diag_r(\lambda_1, \lambda_2, \cdots, \lambda_r), \qquad X_1 = [x_1, x_2, \cdots, x_r], \qquad Y_1 = [y_1, y_2, \cdots, y_r].$$

Here the subindex r in $diag_r$ denotes the dimension of the diagonal matrix. Then the eigenmatrix pair $(\Lambda_1, X_1, Y_1) \in \mathbf{C}^{r\times r} \times \mathbf{C}^{n\times r} \times \mathbf{C}^{n\times r}$ satisfies

$$\begin{aligned} &A_d X_1 \Lambda_1^d + A_{d-1} X_1 \Lambda_1^{d-1} + \cdots + A_1 X_1 \Lambda_1 + A_0 X_1 = 0, \\ &\Lambda_1^d Y_1^H A_d + \Lambda_1^{d-1} Y_1^H A_{d-1} + \cdots + \Lambda_1 Y_1^H A_1 + Y_1^H A_0 = 0. \end{aligned} \tag{4.1}$$

We select the left eigenvectors $y_1, y_2, \ldots, y_r$ such that $Y_1^H X_1 = I_r$. We will construct a new non-equivalence low-rank deflated matrix polynomial $\tilde{P}(\lambda) = \lambda^d \tilde{A}_d + \lambda^{d-1} \tilde{A}_{d-1} + \cdots + \lambda \tilde{A}_1 + \tilde{A}_0$ such that the eigenmatrix pair (Λ_1, X_1, Y_1) of the matrix polynomial $P(\lambda) = \lambda^d A_d + \lambda^{d-1} A_{d-1} + \cdots + \lambda A_1 + A_0$ is replaced by $(diag_r(\infty, \cdots, \infty), X_1, Y_1)$, while the other eigenvalues and the associated eigenvectors are kept invariant.

Let

$$\begin{cases} \tilde{A}_0 = A_0, \\ \tilde{A}_1 = A_1 - (A_1X_1Y_1^H + A_2X_1\Lambda_1Y_1^H + \cdots + A_dX_1\Lambda_1^{d-1}Y_1^H), \\ \tilde{A}_2 = A_2 - (A_2X_1Y_1^H + A_3X_1\Lambda_1Y_1^H + \cdots + A_dX_1\Lambda_1^{d-2}Y_1^H), \\ \vdots \\ \tilde{A}_{d-1} = A_{d-1} - (A_{d-1}X_1Y_1^H + A_dX_1\Lambda_1Y_1^H), \\ \tilde{A}_d = A_d - A_dX_1Y_1^H. \end{cases} \tag{4.2}$$

We get the deflated matrix polynomial $\tilde{P}(\lambda) = \lambda^d\tilde{A}_d + \lambda^{d-1}\tilde{A}_{d-1} + \cdots + \lambda\tilde{A}_1 + \tilde{A}_0$. The relationship between the matrix polynomials $P(\lambda)$ and $\tilde{P}(\lambda)$ may be described in the following result.

Theorem 4.1. *Given an eigenmatrix pair $(\Lambda_1, X_1, Y_1) \in \mathbf{C}^{r\times r} \times \mathbf{C}^{n\times r} \times \mathbf{C}^{n\times r}$ of the matrix polynomial $P(\lambda) = \lambda^dA_d + \lambda^{d-1}A_{d-1} + \cdots + \lambda A_1 + A_0$ with $Y_1^HX_1 = I_r$, let*

$$\tilde{P}(\lambda) = \lambda^d\tilde{A}_d + \lambda^{d-1}\tilde{A}_{d-1} + \cdots + \lambda\tilde{A}_1 + \tilde{A}_0, \tag{4.3}$$

where $\tilde{A}_i (0 \le i \le d)$ are defined by (4.2). *Then*

$$\tilde{P}(\lambda) = P(\lambda)\big[I_n - \lambda X_1(\lambda I_r - \Lambda_1)^{-1}Y_1^H\big], \tag{4.4}$$

and the eigenvalues of $\tilde{P}(\lambda)$ are the same as those of $P(\lambda)$ except that the eigenvalues of Λ_1 are replaced by r infinities.

Proof. Using (4.1) and (4.2), we obtain

$$\begin{aligned} \tilde{P}(\lambda) =& P(\lambda) - \lambda(A_1X_1Y_1^H + A_2X_1\Lambda_1Y_1^H + \cdots + A_dX_1\Lambda_1^{d-1}Y_1^H) \\ & - \lambda^2(A_2X_1Y_1^H + A_3X_1\Lambda_1Y_1^H + \cdots + A_dX_1\Lambda_1^{d-2}Y_1^H) - \cdots \\ & - \lambda^{d-1}(A_{d-1}X_1Y_1^H + A_dX_1\Lambda_1Y_1^H) - \lambda^dA_dX_1Y_1^H \\ =& P(\lambda) - \lambda\big[A_dX_1(\lambda^{d-1}I_r + \lambda^{d-2}\Lambda_1 + \cdots + \Lambda_1^{d-1})Y_1^H \\ & + A_{d-1}X_1(\lambda^{d-2}I_r + \lambda^{d-3}\Lambda_1 + \cdots + \Lambda_1^{d-2})Y_1^H + \cdots \\ & + A_2X_1(\lambda I_r + \Lambda_1)Y_1^H + A_1X_1Y_1^H\big] \\ =& P(\lambda) - \lambda\big[A_dX_1(\lambda^dI_r - \Lambda_1^d) + A_{d-1}X_1(\lambda^{d-1}I_r - \Lambda_1^{d-1}) + \cdots \\ & + A_1X_1(\lambda I_r - \Lambda_1) + A_0X_1 - A_0X_1\big](\lambda I_r - \Lambda_1)^{-1}Y_1^H \\ =& P(\lambda) - \lambda P(\lambda)X_1(\lambda I_r - \Lambda_1)^{-1}Y_1^H \\ =& P(\lambda)\big[I_n - \lambda X_1(\lambda I_r - \Lambda_1)^{-1}Y_1^H\big]. \end{aligned}$$

By using the identity

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

where $R \in \mathbf{C}^{n\times m}$, $S \in \mathbf{C}^{m\times n}$, and (4.4), we have

$$\begin{aligned} det[\tilde{P}(\lambda)] =& det[P(\lambda)]det\left[I_n - \lambda X_1(\lambda I_r - \Lambda_1)^{-1}Y_1^H\right] \\ =& det[P(\lambda)]\frac{det(-\Lambda_1)}{det(\lambda I_r - \Lambda_1)}. \end{aligned} \tag{4.5}$$

Since 0 is not an eigenvalue of the polynomial eigenvalue problem (1.1), then $det(-\Lambda_1) \neq 0$. Therefore, $\tilde{P}(\lambda)$ has the same eigenvalues as $P(\lambda)$ except that r eigenvalues of Λ_1 are replaced by r infinities. $\square$

It is easy to verify that $(diag_r(\infty, \cdots, \infty), X_1, Y_1)$ is an eigenmatrix pair of the matrix polynomial $\tilde{P}(\lambda)$.

Now, we can use Theorem 4.1 to transform the converged eigenvalues to infinity so that the next desired eigenvalues become the closest eigenvalues to the target. Suppose that $(\lambda_1, x_1, y_1) \in \mathbf{C} \times \mathbf{C}^n \times \mathbf{C}^n$ is a computed eigenpair of the polynomial eigenvalue problem (1.1) with $\lambda_1 \neq 0$ and $y_1^H x_1 = 1$. Then (4.2) reduces to

$$\tilde{A}_0 = A_0, \qquad \tilde{A}_i = A_i - \sum_{j=i}^{d} \lambda_1^{j-i} A_j x_1 y_1^H \quad (1 \le i \le d). \tag{4.6}$$

If a non-equivalence low-rank deflation has been implemented, then the mth order partially orthogonal decomposition (2.3)-(2.6) will not hold any more with the new coefficient matrices $\tilde{A}_i (i = 0, 1, \cdots, d)$. How to choose new initial vectors to restart the partially orthogonal process is a key for success of the partially orthogonal projection method. A popular way is to find new initial vectors $q_1, p_1^{(i)} (i = 1, 2, \cdots, d-1)$ in the complementary subspace of the space spanned by the vectors that are built up by the converged eigenpairs according to (1.3). For the implicitly restarted refined partially orthogonal projection method, a good choice is to take the first column of Z_k^+ as the new initial vectors, namely,

$$\begin{pmatrix} q_1 \\ p_1^{(1)} \\ \vdots \\ p_1^{(d-1)} \end{pmatrix} = Z_k^+(:, 1). \tag{4.7}$$

We summarize the implicitly restarted refined partially orthogonal projection (DIRRPOP) method with the non-equivalence low-rank deflation in the following algorithm.

Algorithm 4.1. DIRRPOP(m): Implicitly restarted refined partially orthogonal projection method with the non-equivalence low-rank deflation.

Input: Coefficient matrices $A_i (0 \le i \le d)$, initial vectors $q_1, p_1^{(i)} (1 \le i \le d-1)$, dimension m of the projected subspace, and number k of the desired eigenpairs.

Output: k approximate eigenpairs and their relative residuals.

1. Implement Steps 1-4 in Algorithm 2.3;
2. Set $Idef = 0$. If all k relative residuals α_i are satisfied, output $(\theta_i, \hat{x}_i)$ and $\alpha_i (1 \le i \le k)$, and stop; if none of $\alpha_i (1 \le i \le k)$ is satisfied, go to 3; if $r (0 < r < k)$ relative residuals α_i are satisfied, output the r eigenpairs and the corresponding relative residuals, and set $k := k - r, Idef = 1$;
3. Select $s := m - k$ refined shifts $\mu_i (1 \le i \le s)$ as (3.12), and set $\varepsilon := e_m^T$;

```
4. for $i = 1, 2, \cdots, s$ do
    4.1. Compute unitary matrices $E_i$ and $F_i$ by using the forward implicitly shifted QZ
         iteration for each shift $\mu_i$ so that $E_i^H H_m F_i^H$ and $E_i^H R_m F_i^H$ are upper Hessenberg
         and upper triangular matrices, respectively.
    4.2. Update $H_m := E_i^H H_m F_i^H$, $R_m := E_i^H R_m F_i^H$, $Z_m := Z_m F_i^H, Y_m := Y_m E_i$ and $\varepsilon :=$
         $\varepsilon F_i^H$;
   endfor
5. Set $\eta_k := H_m(k+1,k) Y_m(:,k+1) + \varepsilon(k)\eta_m$,
   $Z_k := Z_m(:,1:k)$,    $Y_k := Y_m(:,1:k)$,
   $H_k := H_m(1:k,1:k)$,    $R_k := R_m(1:k,1:k)$.
   If $Idef = 0$, go to (1); otherwise go to 6;
6. Compute $r$ left eigenvectors of (1.1) corresponding to $r$ converged eigenvalues;
7. Implement the non-equivalence low-rank deflation by using (4.2), update new initial
   vectors as (4.7), and go to 1.
```

5. Numerical Experiments

In this section, we will present some numerical examples to illustrate the effectiveness of the proposed methods. Numerical results in [29] showed that the explicitly restarted refined partially orthogonal projection is superior to the explicitly restarted partially orthogonal projection. Here we compare the explicitly restarted refined partially orthogonal projection (ERRPOP) method, the implicitly restarted refined partially orthogonal projection (IRRPOP) method and the implicitly restarted refined partially orthogonal projection (DIRRPOP) method with the non-equivalence low-rank deflation with the explicitly restarted refined generalized Arnoldi (RGAR) method [2].

All numerical experiments are performed in MATLAB on an Intel Core 2.9 GHz PC with memory 4 GB. The initial vectors are randomly chosen. m denotes the dimension of the initially projected subspace. k denotes the number of the desired eigenpairs. $Iter$ and $Iter_{max}$ denote the number and the maximum number of restarting process, respectively. CPU denotes the CPU time (in seconds) for running an algorithm. The stopping tolerance for relative residuals is chosen to be tol, namely, $ERR \triangleq \max_i\{\alpha_i\} \leq tol$.

Example 5.1. Consider the following cubic eigenvalue problem

$$P(\lambda)x = (\lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0)x = 0, \tag{5.1}$$

where the coefficient matrices come from the "$plasma-drift$" problem in [3], and these matrices are 512×512. Using MATLAB function $polyeig(4)$, we find 4 smallest magnitude

Table 1: Numerical results for Example 5.1.

Methods	$Iter$	CPU	α_i	computed eigenvalues
ERRPOP(20)	4	0.38	0.1568e-19	0.027660094023661 + 0.003726041834763i
			0.1237e-19	-0.029277842413461 + 0.003704756021182i
			0.9554e-12	0.052045262986852 + 0.005176026939440i
			0.9930e-12	0.064135132835627 + 0.008905094382977i
IRRPOP(20)	2	0.22	0.1521e-18	0.027660094023662 + 0.003726041834763i
			0.1457e-17	-0.029277842413448 + 0.003704756021198i
			0.1121e-17	0.052045262881353 + 0.005176026761455i
			0.2808e-13	0.064135132824675 + 0.008905094381634i
RGAR(20)	100	11.01	0.3882e-13	0.027660094023662 + 0.003726041834762i
			0.5311e-13	-0.029277842413462 + 0.003704756021179i
			0.3584e-12	0.052045262828213 + 0.005176026827947i
			0.1610e-10	0.064135133167395 + 0.008905094387536i

eigenvalues of the cubic eigenvalue problem

$$\begin{aligned}
\lambda_1 &= 0.027660094023645 + 0.003726041834717i,\\
\lambda_2 &= -0.029277842413435 + 0.003704756021168i,\\
\lambda_3 &= 0.052045262881366 + 0.005176026761463i,\\
\lambda_4 &= 0.064135132831625 + 0.008905094377921i,
\end{aligned}$$

its CPU time is 35.04s.

Setting $m = 20$, $k = 4$, $tol = 10^{-12}$ and $Iter_{max} = 100$, we use the ERRPOP method, the IRRPOP method and the RGAR method to compute 4 eigenpairs with smallest eigenvalues in modulus of the problem. The numerical results are shown in Table 1.

Table 1 shows that both ERRPOP(20) and IRRPOP(20) converge, their CPU times are all far less than $polyeig(4)$, and IRRPOP(20) is superior to ERRPOP(20) in both the number of restarting process and the CPU time, while RGAR(20) does not converge even if the number of restarting process reaches the maximum number $Iter_{max}$. It is observed that only the relative residual α_4 computed by RGAR(20) does not satisfy the stopping tolerance.

Example 5.2. Consider the following quartic eigenvalue problem

$$P(\lambda)x = (\lambda^4 A_4 + \lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0)x = 0, \tag{5.2}$$

where A_4, A_3, A_2, A_1 are all $n \times n$ random sparse matrices with 0.1 nonzero entries and $A_0 = I_n$. Setting $n = 1000$ and using MATLAB function $polyeig(8)$, we find 8 smallest magnitude eigenvalues of the quartic eigenvalue problem

$$\begin{aligned}
\lambda_{1,2} &= -0.074100957207546 \pm 0.066805279567792i,\\
\lambda_{3,4} &= -0.080134503344701 \pm 0.059486778068479i,\\
\lambda_{5,6} &= -0.064774671319283 \pm 0.077153059698112i,\\
\lambda_{7,8} &= -0.097059310067706 \pm 0.028502193965644i,
\end{aligned}$$

Table 2: Numerical results for Example 5.2.

Methods	$Iter$	CPU	α_i	computed eigenvalues
ERRPOP(30)	200	165.87	0.5083e-03	-0.080039355327533 ± 0.059255892009508i
			0.3569e-03	-0.080293587858382 ± 0.059164331085948i
			0.2443e-03	-0.065045653611504 ± 0.077259581446338i
			0.1899e-03	-0.097118809138960 ± 0.028408120642559i
IRRPOP(30)	44	31.72	0.9241e-16	-0.074100957207546 ± 0.066805279567792i
			0.1383e-15	-0.080134503344702 ± 0.059486778068480i
			0.1561e-12	-0.064774671319307 ± 0.077153059698129i
			0.9324e-12	-0.097059310068078 ± 0.028502193965994i
RGAR(30)	200	234.22	0.6916e-02	-0.009841703564664 ± 0.103320210722644i
			0.7160e-02	0.049472898189088 ± 0.093242080954868i
			0.8164e-02	-0.082592418833363 ± 0.068553742731940i
			0.6758e-02	0.038083901993055 ± 0.100935234970110i

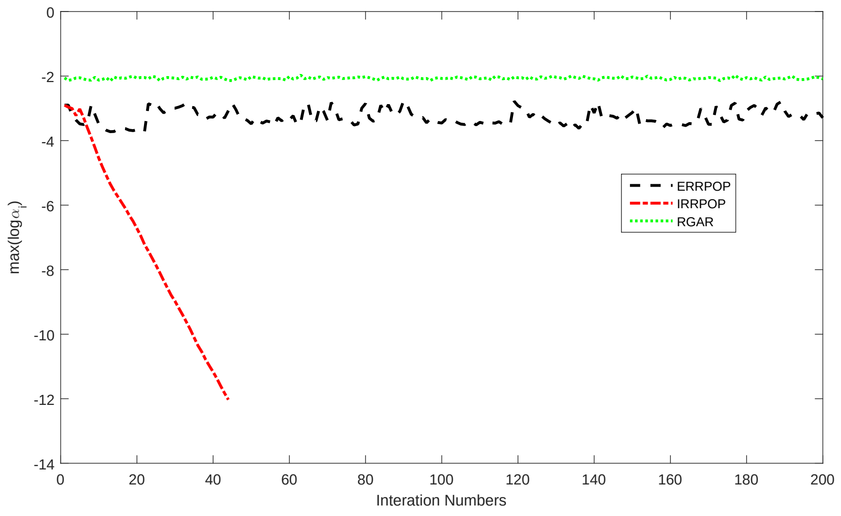

Figure 1: Convergence history of Algorithm 2.3, Algorithm 3.1 and RGAR for Example 5.2.

its CPU time is 928.73s.

Setting $m = 30$, $k = 8$, $tol = 10^{-12}$ and $Iter_{max} = 200$, we use Algorithm 2.3, Algorithm 3.1 and the RGAR method to compute 8 eigenpairs with smallest eigenvalues in modulus of the problem. The numerical results are shown in Table 2 and Fig. 1.

Table 2 shows that IRRPOP(30) converges, its CPU time is much less than $polyeig(8)$, but both ERRPOP(30) and RGAR(30) do not converge even if the number of restarting process reaches the maximum number $Iter_{max}$.

Fig. 1 shows the maximum relative residuals of the eight desired eigenpairs computed by ERRPOP(30), IRRPOP(30) and RGAR(30). It is observed that IRRPOP(30) converges fast, but the maximum relative residuals computed by ERRPOP(30) and RGAR(30) oscillate around 10^{-3} and 10^{-2}, respectively. The numerical results show that IRRPOP(30) is superior to both ERRPOP(30) and RGAR(30).

Example 5.3 (cf. Ref. [2]). Consider the cubic eigenvalue problem (5.1) where

$$A_3 = 5I_n, \qquad A_2 = 3\begin{pmatrix} 3 & -1 & & \\ -1 & 3 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 3 \end{pmatrix} \in \mathbf{R}^{n\times n},$$

and A_1, A_0 come from the Harwell-Boeing test matrix bwm200 [6], $n = 200$, these matrices are 200×200. Using MATLAB function $polyeig(20)$, we find the twenty smallest magnitude eigenvalues of the cubic eigenvalue problem

$$\begin{aligned}
\lambda_{1,2} &= 0.552030959848608 \pm 0.500562603670607i,\\
\lambda_{3,4} &= -0.398318834009417 \pm 0.634872278556881i,\\
\lambda_{5,6} &= -0.754292739026879 \pm 0.134305722792111i,\\
\lambda_{7,8} &= -0.771609287378186 \pm 0.166442127572732i,\\
\lambda_{9,10} &= -0.499000179706779 \pm 0.638535457063888i,\\
\lambda_{11,12} &= 0.668287604009531 \pm 0.472106303419513i,\\
\lambda_{13,14} &= -0.806374320768845 \pm 0.204918559638503i,\\
\lambda_{15,16} &= -0.857090290217150 \pm 0.236798749787440i,\\
\lambda_{17,18} &= -0.629735431980245 \pm 0.644911876069556i,\\
\lambda_{19,20} &= 0.830887675899521 \pm 0.439968307943236i.
\end{aligned}$$

Setting $m = 30$, $k = 20$, $tol = 10^{-12}$ and $Iter_{max} = 100$, we use Algorithm 2.3, Algorithm 3.1, Algorithm 4.1 and the RGAR method to compute 20 eigenpairs with smallest eigenvalues in modulus of the cubic eigenvalue problem. Fig. 2 shows the maximum relative residuals of the twenty desired eigenpairs computed by ERRPOP(30), IRRPOP(30), DIRRPOP(30) and RGAR(30). From Fig. 2 we find that only DIRRPOP(30) converges, while the maximum relative residuals computed by ERRPOP(30), IRRPOP(30) and RGAR(30) oscillate around 10^{-6}, 10^{-4} and 10^{-4}, respectively. Table 3 shows the deflation history of DIRRPOP(30). The numerical results show that the deflation process is essential and efficient when more eigenpairs are wanted.

The numerical results of the ERRPOP, IRRPOP, DIRRPOP and RGAR methods with different dimensions of the projected subspace for Example 5.3 are listed in Table 4. It is observed that the CPU time spent by DIRRPOP(m) increases as the dimension m of the projected subspace increases.

Our numerical experiments show that the implicitly restarted refined partially orthogonal projection method is superior to both the explicitly restarted refined partially orthogonal projection method and the explicitly restarted refined generalized Arnoldi method for computing a few eigenpairs of the polynomial eigenvalue problem, while the implicitly restarted

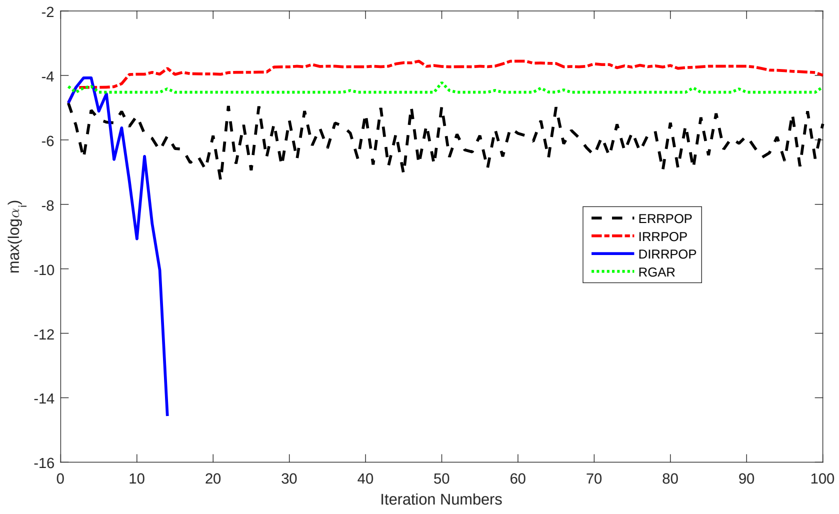

Figure 2: Convergence history of Algorithm 2.3, Algorithm 3.1, Algorithm 4.1 and RGAR for Example 5.3.

Table 3: Deflation history of DIRRPOP(30) for Example 5.3.

computed eigenvalues	α_i	deflation process
$\underline{0.55203095984860}9 \pm \underline{0.5005626036706}28i$	0.6345e-12	the first deflation
$\underline{-0.39831883}3931392 \pm \underline{0.6348722785}43745i$	0.1536e-13	the second deflation
$\underline{-0.7542927390}50764 \pm \underline{0.134305722}407414i$	0.5151e-12	the third deflation
$\underline{-0.49900017970}5161 \pm \underline{0.63853545706}7107i$	0.4632e-12	the fourth deflation
$\underline{0.668287604009}838 \pm \underline{0.47210630341}8599i$	0.5842e-12	
$\underline{-0.77160}8369633790 \pm \underline{0.166442}558624956i$	0.7712e-12	
$\underline{-0.806374}407569414 \pm \underline{0.2049185}60954883i$	0.3477e-13	the fifth deflation
$\underline{0.83088767}0922243 \pm \underline{0.439968}280335488i$	0.4477e-13	
$\underline{-0.62973}8579161092 \pm \underline{0.6449}09212178083i$	0.8892e-13	the sixth deflation
$\underline{-0.8570905}33317283 \pm \underline{0.236798}231042078i$	0.3973e-15	the seventh deflation

refined partially orthogonal projection method with the non-equivalence low-rank deflation may compute more eigenpairs of the polynomial eigenvalue problem.

6. Conclusion

In this paper, based on the implicitly shifted QZ iteration and the refinement strategy, we present an implicitly restarted refined partially orthogonal projection method for computing a few eigenpairs of the polynomial eigenvalue problem. In order to compute more eigenpairs, we develop a novel explicit non-equivalence low-rank deflation technique for

Table 4: Numerical results with different subspaces for Example 5.3.

Methods	m	$Iter$	CPU	ERR	number of converged eigenpairs
ERRPOP	30	100	13.37	0.3146e-05	0
	40	100	24.28	0.6279e-04	0
	50	100	43.26	0.1075e-04	0
IRRPOP	30	100	14.44	0.1010e-03	4
	40	100	28.83	0.8026e-04	10
	50	100	53.99	0.2397e-08	2
DIRRPOP	30	44	2.45	0.6194e-12	20
	40	25	7.93	0.8779e-12	20
	50	25	14.24	0.9824e-12	20
RGAR	30	100	12.97	0.2994e-04	0
	40	100	13.68	0.3007e-04	0
	50	100	18.19	0.2751e-04	4

the polynomial eigenvalue problem, and provide the implicitly restarted refined partially orthogonal projection method with the non-equivalence low-rank deflation. Our numerical results show that the implicitly restarted refined partially orthogonal projection method with the non-equivalence low-rank deflation is efficient and robust.

Acknowledgments

The authors are grateful to the referees for their valuable comments and suggestions which helped to improve the presentation of this paper. The research is supported by the National Natural Science Foundation of China under grant No. 11571171.

References

[1] S. Adhikari and B. Pascual, *Eigenvalues of linear viscoelastic systems*, J. Sound Vib. **325**, 1000-1011 (2009).

[2] L. Bao, Y.-Q. Lin and Y.-M. Wei, *Restarted generalized Krylov subspace methods for solving large-scale polynomial eigenvalue problems*, Numer. Algorithms **50**, 17-32 (2009).

[3] T. Betcke, N.J. Higham, V. Mehrmann, C. Schroder and F. Tisseur, *NLEVP: A collection of non-linear eigenvalue problems*, ACM Trans. Math. Software **39**, 7-28 (2013).

[4] E.K.-W. Chu, *Perturbation of eigenvalues for matrix polynomials via the Bauer-Fike theorems*, SIAM J. Matrix Anal. Appl. **25**, 551-573 (2003).

[5] J. Dedieu and F. Tisseur, *Perturbation theory for homogeneous polynomial eigenvalue problems*, Linear Algebra Appl. **358**, 71-94 (2003).

[6] I.S. Duff, R.G. Grimes and J.G. Lewis, *Sparse matrix test problems,* ACM Trans. Math. Software **15**, 1-14 (1989).

[7] I. Gohberg, P. Lancaster and L. Rodman, *Perturbation theory for divisors of operator polynomials*, SIAM J. Math. Anal. **10**, 1161-1183 (1979).

[8] I. Gohberg, P. Lancaster and L. Rodman, *Matrix Polynomials,* Academic Press, New York (1982).

[9] K.K. Gupta, *On a finite dynamic element method for free vibration analysis of structures*, Comput. Methods Appl. Mech. Eng. **9**, 105-120 (1976).

[10] N.J. Higham, R.-C. Li and F. Tisseur, *Backward error of polynomial eigenproblems solved by linearization*, SIAM J. Matrix Anal. Appl. **29**, 1218-1241 (2007).

[11] N.J. Higham, D.S. Mackey and F. Tisseur, *The conditioning of linearizations of matrix polynomials*, SIAM J. Matrix Anal. Appl. **28**, 1005-1028 (2006).

[12] N.J. Higham and F. Tisseur, *More on pseudospectra for polynomial eigenvalue problems and applications in control theory*, Linear Algebra Appl. **351**, 435-453 (2002).

[13] N.J. Higham and F. Tisseur, *Bounds for eigenvalues of matrix polynomials*, Linear Algebra Appl. **358**, 5-22 (2003).

[14] L. Hoffnung, R.-C. Li and Q. Ye, *Krylov type subspace methods for matrix polynomials*, Linear Algebra Appl. **415**, 52-81 (2006).

[15] W.-Q. Huang, T.-X. Li, Y.-T. Li and W.-W. Lin, *A semiorthogonal generalized Arnoldi method and its variations for quadratic eigenvalue problems*, Numer. Linear Algebra Appl. **20**, 259-280 (2013).

[16] T.-M. Hwang, W.-W. Lin, J.-L. Liu and W. Wang, *Jacobi-Davidson methods for cubic eigenvalue problems*, Numer. Linear Algebra Appl. **12**, 605-624 (2005).

[17] 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).

[18] F.-N. Hwang, Z.-H. Wei, T.-M. Hwang 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).

[19] Z. Jia, *Refined iterative algorithms based on Arnoldi's process for large unsymmetric eigenproblems*, Linear Algebra Appl. **259**, 1-23 (1997).

[20] P.W. Lawrence and R.M. Corless, *Backward error of polynomial eigenvalue problems solved by linearization of Lagrange interpolants*, SIAM J. Matrix Anal. Appl. **36**, 1425-1442 (2015).

[21] D.S. Mackey, N. Mackey, C. Mehl and V. Mehrmann, *Vector spaces of linearizations for matrix polynomials*, SIAM J. Matrix Anal. Appl. **28**, 971-1004 (2006).

[22] D.S. Mackey, N. Mackey, C. Mehl and V. Mehrmann, *Structured polynomial eigenvalue problems: good vibrations from good linearization*, SIAM J. Matrix Anal. Appl. **28**, 1029-1051 (2006).

[23] C.B. Moler and G.W. Stewart, *An algorithm for generalized matrix eigenvalue problems*, SIAM J. Numer. Anal. **10**, 241-256 (1973).

[24] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Halsted Press, New York (1992).

[25] 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).

[26] D.C. Sorensen, *Truncated QZ methods for large scale generalized eigenvalue problems*, Electron. Trans. Numer. Anal. **7**, 141-162 (1998).

[27] F. Tisseur, *Backward error and condition of polynomial eigenvalue problems*, Linear Algebra Appl. **309**, 339-361 (2000).

[28] F. Tisseur and K. Meerbergen, *The quadratic eigenvalue problem*, SIAM Rev. **43**, 235-286 (2001).

[29] W. Wei and H. Dai, *Partially orthogonal projection method and its variations for solving polynomial eigenvalue problem* (in Chinese), Numer. Math. - J. Chinese Universities **38**, 116-133 (2016).