SIAM J. SCI. COMPUT.
Vol. 40, No. 3, pp. A1809–A1835

A BLOCK PRECONDITIONED HARMONIC PROJECTION METHOD FOR LARGE-SCALE NONLINEAR EIGENVALUE PROBLEMS*

FEI XUE†

Abstract. We propose a block preconditioned harmonic projection (BPHP) method for solving large-scale nonlinear eigenproblems of the form $T(\lambda)v = 0$. Similar to classical preconditioned eigensolvers such as the locally optimal block preconditioned conjugate gradient method and preconditioned Lanczos, BPHP aims at computing a few eigenvalues of the nonlinear problem close to a specified shift, using preconditioners that enhance the local spectrum, without the need for exact solution of large shifted linear systems. We explore the development of search subspaces, stabilized preconditioning, nonlinear harmonic Rayleigh–Ritz projections, thick restart, and soft deflation capable of resolving linearly dependent eigenvectors. Numerical experiments show that BPHP with a good preconditioner is storage efficient, and it exhibits robust convergence. A moving-window-style partial deflation enables BPHP to reliably compute a large number of eigenvalues.

Key words. nonlinear eigenproblem, preconditioning, harmonic projection, local convergence, soft deflation, thick restart

AMS subject classifications. 65F15, 65F50, 15A18, 15A22

DOI. 10.1137/17M112141X

1. Introduction.

Nonlinear algebraic eigenproblems of the form $T(\lambda)v = 0$ arise in a great variety of scientific and engineering applications [5, 27, 55]. Their theoretical properties and numerical solutions have been a major area of study in numerical linear algebra in recent years; see [16] for a recent survey. Roughly speaking, existing algorithms can be classified as Newton-type methods [7, 12, 23, 32, 37, 50, 52, 53, 57], contour integral methods [8, 9], methods based on polynomial or rational approximations to $T(\lambda)$ [3, 13] and corresponding linearizations [25, 26, 48], and the infinite Arnoldi/Lanczos method [2, 15, 17, 18, 19]. To compute eigenvalues near a specified shift σ to high accuracy, these algorithms typically require direct (exact) solution of the linear systems of the form $T(\sigma)x = b$, with the exception of the Jacobi–Davidson (JD) methods [7, 12, 50, 57]. They in general perform well for small and medium size problems, as the direct linear solves can be done efficiently.

This paper concerns numerical solution of the nonlinear eigenproblems that are large in scale, and $T(\sigma)$ may have certain challenging sparsity patterns and values of elements, for which factorization-based direct linear solvers are prohibitive or infeasible. In this case, it is mandatory to use an iterative linear solver, most well-known examples of which include the preconditioned Krylov subspace methods and the multigrid methods. For difficult large linear systems, it could take quite some computational cost to solve the linear systems to a low or modest accuracy (e.g., to a relative tolerance of 10^{-2}–10^{-6}). The application of iterative linear solvers with a relatively low solution accuracy for eigensolvers such as the infinite Arnoldi method or the compact

*Submitted to the journal's Methods and Algorithms for Scientific Computing section March 17, 2017; accepted for publication (in revised form) February 1, 2018; published electronically June 19, 2018.

http://www.siam.org/journals/sisc/40-3/M112141.html

Funding: This work was supported by the National Science Foundation under grant DMS-1719461.

†Department of Mathematical Sciences, Clemson University, Clemson, SC 29634 (fxue@clemson.edu).

rational Krylov (CORK) method may lead to considerable loss of accuracy in the computed eigenpairs of interest.

To tackle this challenge, we study a block preconditioned harmonic projection (BPHP) method for computing a few eigenvalues of large eigenproblems $T(\lambda)v = 0$ near a specified shift σ, using a proper preconditioner $M \approx T(\sigma)$. The new method is similar in spirit to classical preconditioned eigensolvers such as the locally optimal block preconditioned conjugate gradient (LOBPCG) method [21] and preconditioned Lanczos [39] for computing eigenvalues of linear symmetric problems $Av = \lambda Bv$ and is inspired by recent extensions of these methods to linear nonsymmetric problems [56] and nonlinear Hermitian problems with a variational principle [49, 54]. In each BPHP iteration, a preconditioned Krylov search subspace is developed and used for a nonlinear harmonic Rayleigh–Ritz (NLHRR) projection, where a weighted eigenresidual-based criterion is applied to extract new eigenpair approximations. The rate and robustness of convergence are enhanced by the stabilization of preconditioners with JD-style projectors in the search subspace development and a thick restart that augments the subspace with additional eigenpairs near the desired ones. In short, most existing techniques well known for classical preconditioned eigensolvers can be extended to the new nonlinear setting.

The BPHP method has close connections to JD, which does not require exact solution of linear systems involving $T(\sigma)$. In the single-vector version of BPHP, the search subspace developed is identical to the current eigenvector approximation plus the space of candidate solutions constructed by a Krylov subspace method for solving a corresponding JD correction equation. The use of projectors in BPHP for stabilizing preconditioners is also motivated by JD, which effectively avoids stagnation of convergence toward desired eigenvalues, especially if the desired eigenvalues are clustered around the shift. Moreover, the close relation leads to a local convergence analysis of BPHP based on that of JD. The main difference, however, is that JD computes eigenpairs sequentially, whereas BPHP updates eigenpair approximations simultaneously as other block preconditioned eigensolvers. Consequently, we expect that BPHP is more efficient than JD for computing several eigenvalues around a given shift.

BPHP computes eigenpair approximations to a *minimal* invariant pair (V, L) with $V = [v_1, \ldots, v_q] \in \mathbb{C}^{n\times q}$ and $L = \mathrm{diag}(\lambda_1, \ldots, \lambda_q) \in \mathbb{C}^{q\times q}$ in a decoupled form, instead of a *simple* invariant pair [6, 23, 51]. The decoupled form $\{(\lambda_i, v_i)\}_{i=1}^q$ is easier for the algorithm implementation than an invariant pair $(\widehat{V}, \widehat{L})$ with a triangular or general $\widehat{L}$, since the latter requires solutions of nonlinear matrix equations and computation of functions of matrices. Based on the decoupled form, we propose a simple yet effective soft deflation (locking) [22] for computing multiple eigenpairs. This approach includes converged eigenvectors into the search subspace for the NLHRR projection, and it can easily differentiate updated unconverged eigenvector approximations from the converged ones. Moreover, it can deflate and compute *linearly dependent* eigenvectors associated with different eigenvalues.

The BPHP method can be used with a moving-window-style partial deflation strategy for computing a large number of eigenvalues at a *fixed* storage cost. We may choose a sequence of shifts $\{\sigma_i\}_{i=1}^m$ distributed one after another along a line on the complex plane and apply BPHP to compute eigenvalues near these shifts sequentially. The converged eigenpairs near σ_{i-1} can be locked by our soft deflation when computing the eigenpairs near σ_i; in addition, converged eigenpairs around the shifts far from σ_i do not need to (and should not) be locked, because the preconditioner $M \approx T(\sigma_i)$ only enhances convergence of eigenvalues near σ_i. This partial deflation

has been explored recently in [49], and it is essential to achieve the approximate linear arithmetic complexity for the computation of many eigenvalues.

The rest of the paper is organized as follows. In section 2, we briefly review the preliminaries regarding the nonlinear eigenvalue problem (NLEVP). We propose and study the new method for computing one eigenpair of $T(\lambda)v = 0$ near a shift σ in section 3. The connections to single-vector JD are studied in section 4, where we give a local convergence analysis of the algorithm based on the established analysis of JD. In section 5, the algorithm is extended to block form (hence called BPHP), with discussions about soft deflation, thick restart, and the processing of linearly dependent eigenvectors. Numerical experiments are given in section 6, showing the efficiency and robustness of the full-featured BPHP method, compared with the infinite Arnoldi method and the CORK method; the computation of hundreds of eigenvalues using the moving-window partial deflation is also demonstrated. Finally, we summarize the paper in section 7.

2. Preliminaries. Consider the nonlinear algebraic eigenvalue problem

$$T(\lambda)v = \big(\textstyle\sum_{i=1}^{m} A_i f_i(\lambda)\big)v = 0, \tag{2.1}$$

where $A_i \in \mathbb{C}^{n\times n}$ and $f_i(\cdot) : \mathbb{C} \to \mathbb{C}$ are analytic functions in a domain $\Omega \subset \mathbb{C}$ where the desired eigenvalues are located. Note that this problem is nonlinear in eigenvalues but linear in eigenvectors. We are interested in computing q eigenvalues $\{\lambda_i\}_{i=1}^{q}$ near a specified *finite* scalar $\sigma \in \Omega$, and the corresponding eigenvectors $\{v_i\}_{i=1}^{q}$, such that $T(\lambda_i)v_i = 0$ $(1 \le i \le q)$. This goal is further explained in the following definition and subsequent discussion.

DEFINITION 2.1. *A pair* $(V, L) \in \mathbb{C}^{n\times q} \times \mathbb{C}^{q\times q}$ *is* invariant *for eigenproblem* (2.1) *if*

$$\mathbb{T}(V, L) := \sum_{i=1}^{m} A_i V \mathbf{f}_i(L) = 0_{n\times q},$$

where $\mathbf{f}_i : \mathbb{C}^{q\times q} \to \mathbb{C}^{q\times q}$ *is the matrix function corresponding to* f_i*. An invariant pair is* minimal *if there is an integer* ℓ *such that* $[V^*, (VL)^*, \ldots, (VL^{\ell})^*]^*$ *has full column rank. The smallest such* ℓ *is the* minimality index *of* (V, L)*. A minimal invariant pair* (V, L) *is* simple[1] *if the algebraic multiplicities of the eigenvalues of* L *are identical to the algebraic multiplicities of the corresponding eigenvalues of* (2.1)*. If* (V, L) *is invariant* (*and minimal, simple*)*, then* (VZ^{-1}, ZLZ^{-1}) *is also invariant* (*and minimal, simple*) *with any nonsingular* $Z \in \mathbb{C}^{q\times q}$*.*

Simple invariant pairs are of particular importance because the Fréchet derivative of the nonlinear equations $\mathbb{T}(V, L) = 0$ combined with a normalization condition of (V, L) is nonsingular at a simple invariant pair [6, 23], and this leads to several important algebraic and analytic properties of simple invariant pairs derived in [51].

In this paper, we compute a minimal (not necessarily simple) invariant pair in decoupled form. That is, we approximate (V, Λ), where $V = [v_1, \ldots, v_q]$ and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_q)$, where the multiplicity of each λ_i in Λ is no greater than its corresponding multiplicity of (2.1), and all eigenvectors in V associated with a distinct λ_i are linearly independent (the cardinality of this set of eigenvectors is no greater than the geometric multiplicity of λ_i of (2.1)). In other words, we look for distinct eigenvalues around σ but will obtain a subset (proper or not) of the basis vectors

[1] It is referred to as a "complete" invariant pair in [16].

spanning the eigenspace of each distinct eigenvalue. Using the decoupled form simplifies the algorithm implementation, because computing an invariant pair (V, L) with a triangular or general L by our algorithm needs solution of nonlinear matrix equations and invocation to functions of matrices, and the deflation of converged eigenvalues would also be much more complicated. For computing ill-conditioned eigenvectors associated with tightly clustered eigenvalues (or perturbed Jordan blocks), however, it may be necessary to compute such a (V, L) for numerical stability.

For linear eigenvalue problems $Av = \lambda Bv$, an eigenvalue approximation ρ can be derived from an eigenvector approximation x; namely, $\rho(x) = \frac{x^*Ax}{x^*Bx}$ is the Rayleigh quotient associated with x. Similarly, in our nonlinear setting, we can extend the definition as follows.

DEFINITION 2.2. *Let (λ, v) be an eigenpair of $T(\cdot)$, and assume that $T(\cdot) : \mathbb{C} \to \mathbb{C}^{n\times n}$ is holomorphic in a neighborhood $S_{\delta,\lambda} = \{\mu \in \mathbb{C} : |\mu - \lambda| \le \delta\}$. Define $G_{\omega,v} = \{u \in \mathbb{C}^n \setminus \{0\} : \angle(u, v) \le \omega\}$ with $0 < \omega < \frac{\pi}{2}$. The nonlinear Rayleigh functional is $\rho(\cdot) : G_{\omega,v} \to S_{\delta,\lambda}$, satisfying $\rho(cx) = \rho(x)$ for all $c \neq 0$, $x^*T(\rho(x))x = 0$, and $x^*T'(\rho(x))x \neq 0$.*

LEMMA 2.3 ([38, Theorem 21]). *Under the assumptions of Definition 2.2, assume further that $v^*T'(\lambda)v \neq 0$. Then there exist small ω_0 and δ_0, such that for all $x \in G_{\omega_0,v}$, there is a unique $\rho(x) \in S_{\delta_0,\lambda}$ with $x^*T(\rho(x))x = 0$ and $|\rho(x) - \lambda| \le \frac{10}{3}\frac{\|T(\lambda)\|\|v\|^2}{|v^*T'(\lambda)v|}\tan\angle(x, v)$.*

In particular, we have $\rho(v) = \lambda$. However, since the scalar nonlinear equation $x^*T(\rho)x = 0$ may have multiple solutions, we need to choose the one closest to the specified shift σ to approximate the interested eigenvalue. Definition 2.2 can also be extended to the block case.

DEFINITION 2.4. *Let (V, L) be a simple invariant pair such that $\sum_{i=1}^m A_iV\mathbf{f}_i(L) = 0_{n\times q}$. Let $(X, F) \in \mathbb{C}^{n\times q} \times \mathbb{C}^{q\times q}$ approximate (V, L), and $Y \in \mathbb{C}^{n\times q}$ such that $\sum_{i=1}^m Y^*A_iX\mathbf{f}_i(F) = 0_{q\times q}$. Then F is called a block Rayleigh functional associated with X and Y.*

The existence and uniqueness of the block Rayleigh functional, however, are technical [51, Theorem 4.10]. In this paper, it is most relevant to understand how to compute it, under the assumption of the existence. Let $X = [x_1, \ldots, x_q]$ approximate the desired eigenvectors $V = [v_1, \ldots, v_q]$, and assume that X is sufficiently close to V columnwise in direction. Then there exist q Ritz pairs $\{(\mu_j, y_j)\}_{j=1}^q$ of the projected eigenproblem $\sum_{i=1}^m X^*A_iXyf_i(\mu) = 0$, such that $F = \mathrm{diag}(\mu_1, \ldots, \mu_q)$ and $Y = [y_1, \ldots, y_q]$ satisfy $\sum_{i=1}^m X^*A_iXY\mathbf{f}_i(F) = 0_{q\times q}$. We perform the update $X \leftarrow XY$, so that $X^*\mathbb{T}(X, F) = \sum_{i=1}^m X^*A_iX\mathbf{f}_i(F) = 0_{q\times q}$; that is, $x_i^TT(\mu_j)x_j = 0$ for all $1 \le i, j \le q$. Such a diagonal block Rayleigh functional is most convenient for our computation of minimal invariant pairs in the decoupled form. By Definition 2.2, under relevant assumptions, $|\mu_j - \lambda_j| \le \mathcal{O}(\tan\angle(x_j, v_j))$.

3. Preconditioned harmonic projection (PHP) method. Preconditioned eigensolvers have been developed and used with significant success for solving linear symmetric eigenproblems $Av = \lambda Bv$; see, e.g., [20, 21, 29, 30, 33, 34, 35, 43, 44, 45, 46] and references therein. The primary message we hope to establish in this study is that essentially all the major ideas and techniques developed for classical preconditioned eigensolvers can be extended to tackle our nonlinear eigenproblems, as they have been done for linear nonsymmetric eigenproblems [56] and nonlinear Hermitian eigenproblems with a standard variational characterization of eigenvalues [49, 54].

Preconditioned eigensolvers are of importance whenever state-of-the-art factorization-based direct methods are not feasible for solving the relevant linear systems (equivalently, applying exact shift-invert matrix-vector products) to enhance the desired spectrum. In this section, we present the framework of the single-vector *preconditioned harmonic projection* (PHP(ℓ)) method for computing one eigenpair near σ.

3.1. Preliminary search subspace and harmonic projection. Our choice of the search subspace is motivated by an early development in preconditioned eigensolvers, namely, the *preconditioned projection method* for linear symmetric eigenproblems $Av = \lambda Bv$, suggested by Knyazev; see [1, section 11.3.5] by Knyazev and references therein. This algorithm develops a block $Q_k \in \mathbb{C}^{n\times(\ell+1)}$ containing a basis of the preconditioned Krylov subspace

(3.1)
$$\mathcal{K}_{\ell+1}\left(M^{-1}(A-\rho_k B), x_k\right) = \operatorname{span}\left\{x_k, M^{-1}(A-\rho_k B)x_k, \ldots, [M^{-1}(A-\rho_k B)]^{\ell}x_k\right\}.$$

Then it solves the projected problem $Q_k^*AQ_k y = \mu Q_k^*BQ_k y = \mu y$ for the lowest Ritz pairs (μ, y) and updates the eigenvector approximation $x_{k+1} = Q_k y$.

The above framework can be used directly to solve our nonlinear eigenproblems. Let x_k be the current approximation to the eigenvector v associated with λ near σ, $\rho_k = \rho(x_k)$ the corresponding Rayleigh functional value such that $x_k^*T(\rho_k)x_k = 0$, and $r_k = T(\rho_k)x_k$ the eigenresidual vector. With a preconditioner $M \approx T(\sigma)$, a search space of PHP(ℓ) with a structure similar to (3.1) can be formed as

$$\begin{aligned}\mathcal{K}_{\ell+1}\left(M^{-1}T(\rho_k), x_k\right) &= \operatorname{span}\left\{x_k, M^{-1}T(\rho_k)x_k, \ldots, [M^{-1}T(\rho_k)]^{\ell}x_k\right\} \\ &= \operatorname{span}\{x_k\} + M^{-1}\mathcal{K}_{\ell}\left(T(\rho_k)M^{-1}, r_k\right),\end{aligned} \tag{3.2}$$

where $M^{-1}\mathcal{K}_{\ell}\left(T(\rho_k)M^{-1}, r_k\right)$ is the *correction space* from which an update (correction) to x_k will be extracted. Let $Q_k \in \mathbb{C}^{n\times(\ell+1)}$ contain basis vectors of (3.2), and $Z_k \in \mathbb{C}^{n\times(\ell+1)}$ contain basis vectors of range$\{T(\sigma)Q_k\}$. We perform a nonlinear harmonic Rayleigh–Ritz (NLHRR) projection by imposing a Petrov–Galerkin condition, and solve the projected eigenproblem

$$Z_k^*T(\mu)Q_k y = 0 \tag{3.3}$$

for a proper harmonic Ritz pair (μ, y). The new iterate is set as $x_{k+1} = Q_k y$.

The NLHRR projection is motivated by the harmonic Rayleigh–Ritz projection for linear eigenproblems $Av = \lambda Bv$, which is an instance of the nonlinear problem $T(\lambda)v = 0$ with $T(\lambda) = \lambda B - A$. Given a search subspace $\mathcal{Q} = \operatorname{range}(Q)$, interior eigenvalues of the matrix pencil (A, B) near the shift σ can be found by using the harmonic Rayleigh–Ritz projection with the test space $(A-\sigma B)\mathcal{Q}$, i.e., by imposing the Petrov–Galerkin condition $(A-\mu B)Qy \perp (A-\sigma B)\mathcal{Q}$. This leads to the projected problem $Z^*AQy = \mu Z^*BQy$, where the columns of Z span $(A-\sigma B)\mathcal{Q}$; see, e.g., [1, Chapter 8.4] by Sleijpen and van der Vorst. Note that the regular (symmetric) nonlinear Rayleigh–Ritz (NLRR) projection has been proposed in [24, 49] for nonlinear eigenproblems of our interest. The procedure of both projections is straightforward, but their approximation properties have not been explored yet, to the best of our knowledge. Nevertheless, we will show by experiments that NLHRR does exhibit faster and more robust convergence than NLRR for computing interior eigenvalues near the shift σ.

3.2. Stabilized preconditioners. Preconditioned eigensolvers may encounter difficulties in convergence if M^{-1}, the action of preconditioning, cancels that of the shifted matrix $A-\rho_k B$ or $T(\rho_k)$. This may cause stagnation of convergence, a primary issue of the Davidson method [10]. The JD method [14, 40, 41] fixed this problem by applying appropriate projectors in the solution of correction equations. We will also adopt this idea for stabilizing preconditioners for the PHP(ℓ) method.

Specifically, single-vector JD (without subspace projection and extraction) for the nonlinear eigenproblem can be formed as

(3.4)
$$\begin{cases} \left(I - \dfrac{T'(\rho_k)x_k p_k^*}{p_k^* T'(\rho_k)x_k}\right) T(\rho_k) \left(I - \dfrac{x_k u_k^*}{u_k^* x_k}\right) \Delta x_k = -T(\rho_k)x_k \quad \text{with } \Delta x_k \perp u_k, \\ x_{k+1} = x_k + \Delta x_k, \end{cases}$$

which is mathematically equivalent to the Rayleigh functional iteration (RFI) [37, Chapter 4]. If the correction equation in (3.4) is solved exactly, single-vector JD exhibits local quadratic or cubic convergence toward a simple or semisimple eigenvalue, depending on whether the local symmetry of $T(\lambda)$ exists and whether a corresponding two-sided Rayleigh functional ρ_k is used [37, 38]. The same order of convergence can be maintained, if the correction equation is solved approximately with an appropriate decreasing sequence of tolerances [50].

The projectors we use for PHP(ℓ) are those adopted by JD to solve (3.4). To keep them consistent with those applied in our preconditioned locally minimal residual (PLMR(ℓ)) method for nonlinear Hermitian eigenproblems [49], we let $p_k = x_k$ and $u_k = T'(\rho_k)x_k$ in (3.4), such that the projectors are

$$\Pi = I - \frac{T'(\rho_k)x_k x_k^*}{x_k^* T'(\rho_k)x_k} \text{ and } \Pi^* = I - \frac{x_k x_k^* T'(\rho_k)^*}{x_k^* T'(\rho_k)^* x_k}, \tag{3.5}$$

satisfying $\text{null}(\Pi) = \text{span}\{T'(\rho_k)x_k\}$, $\text{range}(\Pi) = \text{span}\{x_k\}^\perp$, $\text{null}(\Pi^*) = \text{span}\{x_k\}$, and $\text{range}(\Pi^*) = \text{span}\{T'(\rho_k)x_k\}^\perp$. Following the usual development of preconditioned Krylov subspace method for (3.4), we define the *stabilized preconditioner* as

$$\mathbb{M}_\Pi = \Pi M \Pi^*, \tag{3.6}$$

whose action $\mathbb{M}_\Pi^{-1}$ on a vector u can be analyzed as follows. Consider $u = \mathbb{M}_\Pi y = \Pi M \Pi^* y$, where $u \in \text{range}(\Pi)$, and $y \in \text{range}(\Pi^*)$ such that $\Pi^* y = y$; note that choosing such a y or an alternative $\tilde{y} = y + \alpha x_k$ with any scalar α generates the same u. Let $z = M\Pi^* y = My$, which leads to $u = \Pi z = z - \frac{x_k^* z}{x_k^* T'(\rho_k)x_k} T'(\rho_k)x_k = z - \beta_k T'(\rho_k)x_k$. It follows that $z = u + \beta_k T'(\rho_k)x_k$, and hence $y = M^{-1}z = M^{-1}u + \beta_k M^{-1}T'(\rho_k)x_k$. By the assumption that $y \in \text{range}(\Pi^*)$, i.e., $y \perp T'(\rho_k)x_k$, we get $\beta_k = -\frac{x_k^* T'(\rho_k)^* M^{-1} u}{x_k^* T'(\rho_k)^* M^{-1} T'(\rho_k)x_k}$. Therefore,

(3.7)
$$y = \mathbb{M}_\Pi^\dagger u = M^{-1}u + \beta_k M^{-1}T'(\rho_k)x_k = \left(I - \frac{\left(M^{-1}T'(\rho_k)x_k\right)\left(x_k^* T'(\rho_k)^*\right)}{\left(x_k^* T'(\rho_k)^*\right)\left(M^{-1}T'(\rho_k)x_k\right)}\right) M^{-1}u,$$

which is obtained by applying a projector to $M^{-1}u$. The projector in (3.7) involves an extra preconditioned vector $M^{-1}T'(\rho_k)x_k$, computed only once and used for *any* $u \in \text{range}(\Pi)$. Evaluating $\mathbb{M}_\Pi^\dagger u$ incurs marginally higher cost than computing $M^{-1}u$.

With the projected preconditioner, the search subspace for PHP(ℓ) is

$$\mathcal{K}_{\ell+1}\left(\mathbb{M}_\Pi^\dagger T(\rho_k), x_k\right) = \text{span}\{x_k\} + \mathbb{M}_\Pi^\dagger \mathcal{K}_\ell\left(T(\rho_k)\mathbb{M}_\Pi^\dagger, r_k\right). \tag{3.8}$$

The virtue of the projected preconditioner can be intuitively explained as follows. Consider an extreme case when the preconditioner $M = T(\rho_k)$. Then the search subspace (3.2) is simply span$\{x_k\}$, because $M^{-1}T(\rho_k) = I$, and hence any algorithm using this space will suffer from complete stagnation. The new search subspace (3.8) developed with the projected preconditioner, however, for any $\ell \geq 2$, contains

$$\begin{aligned}\mathcal{K}_2\left(\mathbb{M}_\Pi^\dagger T(\rho_k), x_k\right) &= \text{span}\{x_k\} + \text{span}\{\mathbb{M}_\Pi r_k\} \\ &= \text{span}\{x_k\} + \text{span}\left\{\left(I - \frac{\left(T(\rho_k)^{-1}T'(\rho_k)x_k\right)\left(x_k^* T'(\rho_k)^*\right)}{\left(x_k^* T'(\rho_k)^*\right)\left(T(\rho_k)^{-1}T'(\rho_k)x_k\right)}\right)x_k\right\} \\ &= \text{span}\{x_k\} + \text{span}\{T(\rho_k)^{-1}T'(\rho_k)x_k\}.\end{aligned}$$

In particular, the new RFI iterate $T(\rho_k)^{-1}T'(\rho_k)x_k$ lies in this new search subspace. Thanks to the local quadratic or cubic convergence of RFI, the search subspace (3.8) contains significant improvement of the current approximation x_k. In practice, M is rarely identical to $T(\rho_k)$, but a delay in convergence is still possible if M is very close to $T(\mu)$ with $\mu \approx \rho_k$. This effect will be demonstrated by experiments in section 6.3.

As usual, $Q_k \in \mathbb{C}^{n\times(\ell+1)}$ containing an orthonormal (with respect to certain inner product) basis for space (3.8) is recommended. Then NLHRR projection (3.3) is performed, and a new eigenvector approximation is extracted, as explained in the following section.

3.3. Extraction of new eigenvector approximations. The computation of interior eigenvalues by iterative projection methods has to tackle the challenge of choosing the "correct" Ritz pair as new eigenpair approximations. This is mainly because of the existence of spurious Ritz values that are close to the shift σ but have no corresponding accurate eigenvector approximations. Choosing a spurious Ritz pair as the new approximation would significantly delay or disrupt convergence. This difficulty is particularly pronounced if the regular (symmetric) Rayleigh–Ritz projection is used for interior eigenvalue computations. With harmonic projection, the problem tends to be less serious, but may still persist if new eigenpair approximations are chosen solely based on the distance between the Ritz values and σ.

In order to target the harmonic Ritz pair that truly approximates the desired eigenpair closest to σ, we adopt a strategy proposed in [49] that has been shown effective for filtering spurious (harmonic) Ritz values. We first choose a few harmonic Ritz values $\{\mu_i\}_{i=1}^r$ closest to σ as candidates, then compute the relative eigenresidual norm associated with these Ritz pairs $\{\mu_i, Q_k y_i\}_{i=1}^r$, namely, $res_i := \{\frac{\|T(\mu_i)Q_k y_i\|_2}{\|T(\sigma)\|_F \|y_i\|_2}\}_{i=1}^r$. The new eigenpair approximation is set as $x_{k+1} = Q_k y_i$, where the index i satisfies

$$i = \text{argmin}\{|\mu_i - \sigma| \times \text{res}_i^p\}_{i=1}^r, \tag{3.9}$$

with a small positive integer p. This criterion selects the new eigenpair approximation based on *weighted eigenresidual norms* of the candidate harmonic Ritz pairs together with their distance to σ. The user can choose different values for p to assign an appropriate weight to the eigenresidual norms for this criterion to choose new eigenvector approximations. We have no recommendation for a particular value, but our experiments in section 6 show that $p = 1$ leads to considerably more robust convergence than $p = 0$ (choosing the Ritz pair closest to σ as the new eigenpair approximation).

4. Local convergence of PHP. In this section, we show a close connection between PHP and the single-vector JD method, which leads to a local convergence result of the former. In particular, with the same current eigenvector approximation, the new iterate of single-vector JD lies in the search subspace developed by PHP.

Let (ρ_k, x_k) be the current eigenpair approximation of both algorithms. Recall from the basic JD correction equation $\Pi T(\rho_k)\Pi^* \Delta x_k = -T(\rho_k)x_k$ (3.4), and assume that a right-preconditioned Krylov subspace method with the projected preconditioner (3.6) is used to solve this equation. In iteration $\ell - 1$ ($\ell \geq 2$), the Krylov subspace developed for the preconditioned linear system is $\mathcal{K}_\ell(\Pi T(\rho_k)\Pi^* \mathbb{M}_\Pi^\dagger, T(\rho_k)x_k)$, and therefore the approximate solution Δx_k of the original equation (3.4) lies in $\mathbb{M}_p^\dagger \mathcal{K}_\ell\left(\Pi T(\rho_k)\Pi^* \mathbb{M}_p^\dagger, T(\rho_k)x_k\right)$. To understand the effect of the projectors, we recall the explicit expression of $\mathbb{M}_\Pi^\dagger$ from (3.7) and see that

$$\mathbb{M}_\Pi^\dagger \Pi = \Pi^* \mathbb{M}_\Pi^\dagger = \mathbb{M}_\Pi^\dagger.$$

It follows that the new JD iterate $x_{k+1} = x_k + \Delta x_k$ lies in

$$\begin{aligned}
(4.1)\qquad & \operatorname{span}\{x_k\} + \mathbb{M}_\Pi^\dagger \mathcal{K}_\ell\left(\Pi T(\rho_k)\Pi^* \mathbb{M}_\Pi^\dagger, T(\rho_k)x_k\right) \\
&= \operatorname{span}\left\{x_k, \mathbb{M}_\Pi^\dagger T(\rho_k)x_k, \mathbb{M}_\Pi^\dagger \Pi T(\rho_k)\Pi^* \mathbb{M}_\Pi^\dagger T(\rho_k)x_k, \ldots,\right. \\
&\qquad \left. \mathbb{M}_\Pi^\dagger \left(\Pi T(\rho_k)\Pi^* \mathbb{M}_\Pi^\dagger\right)^{\ell-1} T(\rho_k)x_k\right\} \\
&= \operatorname{span}\left\{x_k, \mathbb{M}_\Pi^\dagger T(\rho_k)x_k, \left(\mathbb{M}_\Pi^\dagger T(\rho_k)\right)^2 x_k \ldots, \left(\mathbb{M}_\Pi^\dagger T(\rho_k)\right)^\ell x_k\right\} \\
&= \mathcal{K}_{\ell+1}\left(\mathbb{M}_\Pi^\dagger T(\rho_k), x_k\right),
\end{aligned}$$

which is exactly the search subspace (3.8) developed by PHP(ℓ).

The above observation is summarized in the following lemma.

LEMMA 4.1. *Given the same current eigenvector approximation x_k and the stabilized preconditioner* (3.6)*, single-vector* JD *with* (3.4) *solved approximately by an $(\ell-1)$-step right-preconditioned Krylov subspace method delivers a new eigenvector approximation x_{k+1}^{JD} that lies in the search subspace developed by* PHP(ℓ)*, from which it extracts the new iterate x_{k+1}^{PHP}.*

As a result, if x_{k+1}^{PHP} is of the same quality as x_{k+1}^{JD} for approximating v, the local convergence of PHP(ℓ) can be established as a corollary of the local convergence of single-vector JD, which has been shown for our problem [50, Theorems 7, 11]. Whether the new iterates of the two methods are comparable in quality depends on the approximation properties of the harmonic Rayleigh–Ritz projection. These properties have been established for standard linear eigenproblems $Av = \lambda Bv$; see, e.g., [47, Chapter 4.4] and references therein. Specifically, let v be the desired eigenvector and Q contain basis vectors for the search subspace. Under certain assumptions (typically not stringent), $\angle(v, Qy)$, the angle between v and the corresponding harmonic Ritz vector Qy, is proportional to $\angle(v, \mathcal{Q})$, where $\mathcal{Q} = \operatorname{range}(Q)$.

For nonlinear eigenproblems $T(\lambda)v = 0$, similar approximation properties of the NLHRR projection have not been studied, and a thorough investigation is beyond the scope of this paper. Nevertheless, to our extensive numerical experience with multiple problems, $\angle(v, Qy)$ is also proportional to $\angle(v, \mathcal{Q})$, as long as the eigenvalue

λ associated with v is sufficiently close to the shift σ. We would make the assumption as follows.

ASSUMPTION 4.2. *Let (λ, v) be a simple eigenpair of $T(\lambda)v = 0$, where λ is sufficiently close to σ, and $\mathcal{Q}_k$ be the search subspace* (3.8) *developed in the kth iteration of* PHP(ℓ). *Assume that the* NLHRR *projection extracts a new iterate x_{k+1}, such that $\sin\angle(v, x_{k+1}) \le C \sin\angle(v, \mathcal{Q}_k)$ with a constant C independent of k.*

To study the local convergence of PHP, we first need the following assumption to establish the local convergence of single-vector JD.

ASSUMPTION 4.3. *Let (λ, v) be a simple eigenpair of $T(\lambda)v = 0$, where v is normalized such that $v^*T'(\lambda)v = 1$. Assume that there exists a $\delta > 0$ and a corresponding $\xi_\delta > 0$, such that for any (μ, x) sufficiently close to (λ, v) with $\|[x;\, \mu] - [v;\, \lambda]\| \le \delta$, we have $\|T'(\mu)x\| \le \xi_\delta$.*

With the above assumptions, we provide a main local convergence result of PHP.

THEOREM 4.4. *Let (λ, v) be a simple eigenpair of $T(\lambda)v = 0$ satisfying Assumption* 4.3*, and x_k be the eigenvector approximation after the kth iteration of* PHP(ℓ_k). *Assume that $\angle(x_0, v)$ is sufficiently small, such that $\|[x_0;\, \rho(x_0)] - [v;\, \lambda]\| \le \delta$, and assume that Assumption* 4.2 *holds for each x_k. Suppose that the JD correction equation* (3.4) *is solved by right-preconditioned* GMRES(ℓ_k) *with preconditioner* (3.6)*, where ℓ_k is either a properly large constant or an increasing sequence, such that the single-vector JD converges toward (λ, v) linearly or quadratically with k. Then asymptotically,* PHP(ℓ_k) *with the same preconditioner* (3.6) *converges towards (λ, v) at least linearly or quadratically.*

Proof. Note from Lemma 4.1 that single-vector JD generates the new approximation $x_{k+1}^{\mathrm{JD}} \in \mathcal{Q}_k^{\mathrm{PHP}}$, and hence $\sin\angle(v, x_{k+1}^{\mathrm{JD}}) \ge \sin\angle(v, \mathcal{Q}_k^{\mathrm{PHP}})$. By Assumption 4.2, the NLHRR projection delivers the new eigenvector approximation x_{k+1}^{PHP} comparable to x_{k+1}^{JD}, satisfying $\sin\angle(v, x_{k+1}^{\mathrm{PHP}}) \le C \sin\angle(v, \mathcal{Q}_k^{\mathrm{PHP}})$ for some constant $C > 0$. It follows that $\sin\angle(v, x_{k+1}^{\mathrm{PHP}}) \le C \sin\angle(v, x_{k+1}^{\mathrm{JD}})$, which establishes the locally linear or quadratic convergence of PHP(ℓ_k). ∎

In section 6, we will provide numerical evidence to support Theorem 4.4.

5. BPHP method. We now generalize single-vector PHP to the block version for the computation of several eigenvalues near σ simultaneously. The algorithm is referred to as the BPHP method. Most of the framework developed for PHP can be extended directly to the block case. In addition, we will discuss thick restart, soft deflation (locking), and the computation of linearly dependent eigenvectors.

5.1. Search subspace and stabilized preconditioners. Suppose that we want to compute q eigenvalues $\{\lambda_i\}_{i=1}^q$ near σ, counting multiplicities, together with their eigenvectors $\{v_i\}_{i=1}^q$. For now, we assume for the the sake of simplicity that $\{v_i\}_{i=1}^q$ are linearly independent and so are $\{T'(\lambda_i)v_i\}_{i=1}^q$, and we will discuss the case of linearly dependent eigenvectors later. In iteration k of BPHP, let $X_k = [x_k^{(1)}, \ldots, x_k^{(q)}]$ be the approximation to $V = [v_1, \ldots, v_q]$, and $\Phi_k = \mathrm{diag}(\rho_k^{(1)}, \ldots, \rho_k^{(q)})$ be the associated block Rayleigh functional value, such that

$$X_k^* \mathbb{T}(X_k, \Phi_k) = \sum_{i=1}^{m} X_k^* A_i X_k \mathbf{f}_i(\Phi_k) = 0_{q\times q}.$$

As we explained, this is done by solving the projected eigenproblem

$$\sum_{i=1}^{m} (X_k^* A_i X_k)\, Y_k \mathbf{f}_i(\Phi_k) = 0_{q\times q}$$

for $Y_k \in \mathbb{C}^{q\times q}$ and diagonal Φ_k, and then updating $X_k \leftarrow X_k Y_k$.

Similar to the single-vector algorithm, with a regular preconditioner $M \approx T(\sigma)$, the preliminary search subspace of BPHP(ℓ) can be defined as

$$\mathcal{Q}_k = \mathcal{K}_{\ell+1}\left(M^{-1}\mathbb{T}(\,\cdot\,,\Phi_k), X_k\right), \tag{5.1}$$

where the generating operator is

$$M^{-1}\mathbb{T}(\,\cdot\,,\Phi_k) : X \to M^{-1}\mathbb{T}(X,\Phi_k) = M^{-1}\sum_{i=1}^{m} A_i X \mathbf{f}_i(\Phi_k).$$

The stabilized preconditioner, analogous to that defined in (3.6), can be constructed as

$$\mathbb{M}_{\mathbf{\Pi}} = \mathbf{\Pi} M \mathbf{\Pi}^*, \tag{5.2}$$

where the projector is

$$\mathbf{\Pi} = I - Z_k(X_k^* Z_k)^{-1} X_k^* \ \text{ with } \ Z_k \equiv \mathbb{T}'(X_k,\Phi_k) = \left[T'(\rho_k^{(1)})x_k^{(1)}, \ldots, T'(\rho_k^{(q)})x_k^{(q)}\right], \tag{5.3}$$

a direct extension of the projector Π (3.5), satisfying $\text{null}(\mathbf{\Pi}) = \text{range}(Z_k)$ and $\text{range}(\mathbf{\Pi}) = \text{range}(X_k)^{\perp}$. Note that $X_k^* Z_k$ is nonsingular by our current assumption of linear independence, as $X_k \to [v_1,\ldots,v_q]$ and $\Phi_k \to \text{diag}(\lambda_1,\ldots,\lambda_q)$. Following the derivation of (3.7), the action of $\mathbb{M}_{\mathbf{\Pi}}^{\dagger}$ on a block of vectors $U \in \text{range}(\mathbf{\Pi})$ can be written as

$$\mathbb{M}_{\mathbf{\Pi}}^{\dagger} U = \left(I - M^{-1}Z_k(Z_k^* M^{-1} Z_k)^{-1} Z_k^*\right) M^{-1} U, \tag{5.4}$$

which is obtained at a one-time cost of computing $M^{-1}Z_k$ for any $U \in \text{range}(\mathbf{\Pi})$. With (5.2), the search subspace of BPHP(ℓ) is formulated as

$$\mathcal{Q}_k = \mathcal{K}_{\ell+1}\left(\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathbb{T}(\,\cdot\,,\Phi_k), X_k\right) = \text{span}\{X_k\} + \mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathcal{K}_{\ell}\big(\mathbb{T}(\cdot,\Phi_k)\mathbb{M}_{\mathbf{\Pi}}^{\dagger}, \mathbb{T}(X_k,\Phi_k)\big). \tag{5.5}$$

Here, $\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathcal{K}_{\ell}\big(\mathbb{T}(\cdot,\Phi_k)\mathbb{M}_{\mathbf{\Pi}}^{\dagger}, \mathbb{T}(X_k,\Phi_k)\big)$ is the *correction space*, for which we maintain an orthonormal basis. We keep X_k nonorthogonal since we will use rank-revealing QR to detect potential linear dependence among desired eigenvector approximations; see section 5.4.

5.2. Thick restart and subspace extraction. The basic version of BPHP(ℓ) uses the new eigenvector approximations X_{k+1} to restart the $(k+1)$st iteration. To enhance the convergence, we may augment the search space with additional vectors. One possible option is adopted in LOBPCG [21], which essentially keeps X_k in the search space in iteration $k+1$. This strategy is highly effective for solving linear and

nonlinear symmetric or Hermitian eigenproblems with min-max principle of eigenvalues, but becomes much less attractive in the nonsymmetric case [56]; experimentally, it sometimes does not improve convergence at all or could even be counterproductive. In this study, we take an alternative option and incorporate thick restart (widely used for solving symmetric and nonsymmetric linear eigenproblems) into our new problem setting. Thick restart extracts a few additional (harmonic) Ritz vectors associated with Ritz values near σ and augments the search subspace (5.5) with these Ritz vectors. The augmented search subspace hence contains an enhanced component of desired eigenvectors, which leads to faster and more robust convergence.

Assume that in iteration k, BPHP(ℓ) develops a search subspace $\mathcal{Q}_k$ with proper basis vectors contained in Q_k. The NLHRR projection constructs and solves the projected problem $Z_k^* T(\mu) Q_k y = 0$ (range$(Z_k) = T(\sigma)\mathcal{Q}_k$) for $r \geq q + s$ harmonic Ritz pairs (μ_i, y_i) with μ_i closest to σ. We compute $res_i = \frac{\|T(\mu_i)Q_k y_i\|_2}{\|T(\mu_i)\|_F \|y_i\|_2}$ $(1 \leq i \leq r)$ and the weighted eigenresidual norm $\{|\mu_i - \sigma| \times res_i^p\}$ for each candidate pair and reorder them in the ascending order in the weighted norm; see section 3.3 for the motivation of this criterion. Then we update $X_{k+1} \in \mathbb{C}^{n\times q}$ and $X_{k+1}^{tr} \in \mathbb{C}^{n\times s}$, respectively, to be the first q and the next s reordered harmonic Ritz vectors. In iteration $k + 1$, the augmented search subspace is

$$\mathcal{Q}_{k+1} = \mathcal{K}_{\ell+1}\left(\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathbb{T}(\,\cdot\,, \Phi_{k+1}), X_{k+1}\right) + \text{range}(X_{k+1}^{tr}). \tag{5.6}$$

Thick restart incurs minor extra cost to the original BPHP(ℓ) method. The search space is enlarged from dimension $(\ell+1)q$ to $(\ell+1)q+s$ (linear independence assumed), and X_{k+1}^{tr} is not involved in the construction of the block Krylov subspace. As we will see in section 6, thick restart does improve the rate and robustness of convergence.

5.3. Soft deflation. Deflation (locking) is crucial for computing multiple eigenvalues. It excludes the converged eigenpairs from being further processed, so that the algorithm need only work on the unconverged eigenpair approximations, and it makes sure that no repeated convergence occurs. For linear eigenvalue problems $Av = \lambda Bv$, the so-called "hard deflation" is done by orthogonalizing the search space against the converged eigenspace with respect to certain inner product (B-orthogonalization for the symmetric case and standard orthogonalization based on parital QZ decompositions for the nonsymmetric case). Deflation for nonlinear eigenproblems $T(\lambda)v = 0$ has been done by expanding minimal invariant pairs with the JD method [12] or by applying standard deflation techniques with explicit or implicit restart for Krylov-subspace-type methods, e.g., infinite Arnoldi [2, 18] and compact rational Kyrlov [4]. The infinite Arnoldi methods compute the eigenvalues of $T(\cdot)$ around σ by transforming $T(\lambda)v = 0$ into a linear eigenproblem of an integral operator on a function space, whereas rational Krylov methods tackle the problem by applying polynomial or rational approximations to $T(\cdot)$ and then using certain general (non-structure-preserving) linearizations with a Kronecker structure to solve the approximate problem.

We propose a simple effective "soft deflation" strategy, originally discussed in [22]. The converged eigenvectors are included in the search space $\mathcal{Q}_k$ for NLHRR, and subspace extraction is performed only for active (unconverged) eigenpairs. Specifically, we assume without loss of generality that in iteration k, eigenvector approximations $\{x_k^{(i)}\}_{i=1}^d$ have converged. Let $X_k^{cvg} = [x_k^{(1)}, \ldots, x_k^{(d)}]$ and $X_k^{act} = [x_k^{(d+1)}, \ldots, x_k^{(q)}]$ denote the converged and unconverged approximations. The search subspace of BPHP is

$$\mathcal{Q}_k = \text{range}(X_k^{cvg}) + \mathcal{K}_{\ell+1}\left(\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathbb{T}(\cdot, \Phi_k^{act}), X_k^{act}\right) + \text{range}\left(X_k^{tr}\right), \tag{5.7}$$

where $\Phi_k^{act} = \mathrm{diag}\big(\rho_k^{(d+1)}, \ldots, \rho_k^{(q)}\big)$ is the corresponding block Rayleigh functional value such that $(X_k^{act})^* \mathbb{T}(X_k^{act}, \Phi_k^{act}) = \sum_{i=1}^m \big[\big(X_k^{act}\big)^* A_i X_k^{act}\big] \mathbf{f}_i(\Phi_k^{act}) = 0_{(q-d)\times(q-d)}$. Here, the projector (5.3) used for stabilizing preconditioners remains the same, but due to deflation, the first d columns of $Z_k = [T'(\rho_k^{(1)})x_k^{(1)}, \ldots, T'(\rho_k^{(q)})x_k^{(q)}]$ in (5.3) are locked; therefore we only need to update the active columns of $M^{-1}Z_k$ in subsequent iterations to implement the action of $\mathbb{M}_{\mathbf{\Pi}}^{\dagger}$. The search subspace with deflation decreases in dimension as more eigenvectors converge. From (5.7), it is clear that $\dim(\mathcal{Q}_k) \le d + (q-d)(\ell+1) + s$.

In our algorithm, all columns of X_k (including X_k^{cvg}, X_k^{act}, and X_k^{tr}) are normalized, and the block Krylov subspace contained in (5.7) for the NLHRR projection,

(5.8)
$$\mathcal{K}_{\ell+1}\big(\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathbb{T}(\cdot, \Phi_k^{act}), X_k^{act}\big) = \mathrm{span}\{X_k^{act}\} + \mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathcal{K}_{\ell}\left(\mathbb{T}(\cdot, \Phi_k^{act})\mathbb{M}_{\mathbf{\Pi}}^{\dagger}, \mathbb{T}(X_k^{act}, \Phi_k^{act})\right),$$

is formed by X_k^{act} representing the desired eigenvector approximations and an orthonormal basis for the correction space $\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathcal{K}_{\ell}\big(\mathbb{T}(\cdot, \Phi_k^{act})\mathbb{M}_{\mathbf{\Pi}}^{\dagger}, \mathbb{T}(X_k^{act}, \Phi_k^{act})\big)$. The original desired eigenvector approximations are kept so that rank-revealing QR may detect possible linear dependence among the eigenvectors, whereas an orthonormal basis for the correction space improves numerical stability. We construct the block Q_k consisting of the basis of range(X_k^{cvg}), range(X_k^{act}), the correction space, and range(X_k^{tr}) to be the search space for projection.

Soft deflation is done by performing the NLHRR projection and solving the projected problem $Z_k^* T(\mu) Q_k y = 0$ for unconverged harmonic Ritz pairs. Let $p = d + (q-d)(\ell+1) + s$ be the number of columns in Q_k. We order the normalized primitive harmonic Ritz vectors $\{y_i\}$ such that $\|y_i(1:d)\| \ge \|y_j(1:d)\|$ (i.e., $\|y_i(d+1:p)\| \le \|y_j(d+1:p)\|$) for all $i < j$ and discard $\{y_i\}_{i=1}^d$ because $\{Q_k y_i\}_{i=1}^d$ correspond to the converged eigenvectors. To see this, we note that $X_k^{cvg} = Q_k(:, 1:d)$, and the remaining $p-d = (q-d)(\ell+1)+s$ columns of Q_k are normalized but generally not orthogonal to $\mathcal{X}_k^{cvg} := \mathrm{range}(X_k^{cvg})$. By geometry,

$$\begin{aligned}\sin\varphi_i :&= \sin\angle(Q_k y_i, \mathcal{X}_k^{cvg}) \\ &\le \frac{\|Q_k(:, d+1:p) y_i(d+1:p)\|}{\|Q_k(:, 1:d) y_i(1:d)\|} \le \frac{\|Q_k(:, d+1:p)\|}{\sigma_{\min}(X_k^{cvg})} \frac{\|y_i(d+1:p)\|}{\|y_i(1:d)\|},\end{aligned}$$

where $\frac{\|y_i(d+1:p)\|}{\|y_i(1:d)\|}$ suggests it is most likely that $\sin\varphi_i \le \sin\varphi_j$ for $i < j$ ($\sigma_{\min}(X_k^{cvg}) > 0$, since we have assumed for now that all desired eigenvectors are linearly independent). Therefore, the harmonic Ritz vectors $\{Q_k y_i\}_{i=1}^d$ should not be computed, because they are closer to the deflated eigenspace $\mathcal{X}_k^{cvg}$ than any other harmonic Ritz vectors (in fact, they essentially retrieve $\mathcal{X}_k^{cvg}$). Then we compute other harmonic Ritz vectors and apply the criterion discussed in section 5.2 to choose the most promising $q-d$ and additional s harmonic Ritz vectors to update X_{k+1}^{act} and X_{k+1}^{tr}, respectively. Finally, we compute the block Rayleigh functional value $\Phi_{k+1}^{act} = \mathrm{diag}(\rho_{k+1}^{(d+1)}, \ldots, \rho_{k+1}^{(q)})$ such that $\big(X_{k+1}^{act}\big)^* \mathbb{T}(X_{k+1}^{act}, \Phi_{k+1}^{act}) = 0_{(p-d)\times(p-d)}$ (see the beginning of section 5.1) and restart the next iteration.

5.4. Eigenpairs involving linearly dependent eigenvectors. Nonlinear eigenproblems $T(\lambda)v = 0$ may have linearly dependent eigenvectors associated with distinct eigenvalues. Though this is not often observed for large-scale problems if a small number of eigenpairs are sought, reliable treatment still deserves our attention. Existing techniques include the transformation of the eigenpair (λ, v) of $T(\cdot)$

into an eigenfunction $e^{\lambda t}v$ of a linear integral operator by infinite Arnoldi [19] and the Newton-like/JD methods based on expansion of minimal invariant pairs [12]. The two approaches share a similar spirit: eigenvectors need to be associated with eigenvalues to form a whole entity to differentiate eigenpairs.

Our strategy for computing linearly dependent eigenvectors is as follows. First, note that a *linearly independent* set of vectors in Q_k should be used to represent the search subspace. A linearly dependent set could lead to a nonunique representation of a harmonic Ritz vector and hence possible overestimate of the multiplicity of an eigenvalue. Following the exposition in section 5.3, assume that in iteration k, we have developed the block $Q_k \in \mathbb{C}^{n\times p}$ containing the basis vectors of the search subspace (5.7). Specifically, we let

$$Q_k(:,1:d) \leftarrow X_k^{cvg},\ Q_k(:,p-s+1:p) \leftarrow X_k^{tr},$$
$$\text{range}(Q_k(:,d+1:p-s)) = \mathcal{K}_{\ell+1}\left(\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathbb{T}(\cdot,\Phi_k^{act}), X_k^{act}\right).$$

Then we perform a rank-revealing QR factorization on Q_k (achieved by the `qr` function in MATLAB with three output parameters and a proper tolerance), choose the $\hat{p}$ columns from Q_k that are closest to linearly independent, and form the compressed block $\hat{Q}_k \in \mathbb{C}^{n\times\hat{p}}$. Suppose that the columns of $\hat{Q}_k$ consist of $\hat{d}$ columns from $Q_k(:,1:d)$, $\hat{s}$ columns from $Q_k(:,p-s+1:p)$, and $\hat{p}-\hat{s}-\hat{d}$ columns from $Q_k(:,d+1:p-s)$. We then compute $\hat{Z}_k = T(\sigma)\hat{Q}_k$, perform the NLHRR projection, and solve the projected problem $\hat{Z}_k T(\hat{\mu})\hat{Q}_k\hat{y} = 0$ for unconverged harmonic Ritz vectors. To this end, we order all primitive harmonic Ritz vectors such that $\|\hat{y}_i(1:\hat{d})\| \geq \|\hat{y}_j(1:\hat{d})\|$ for all $i < j$ (see the end of section 5.3). We discard $\{\hat{y}_i\}_{i=1}^{d}$ because the harmonic Ritz vectors $\{\hat{Q}_k\hat{y}_i\}_{i=1}^{d}$ retrieve the deflated eigenspace.

With a linearly independent set of vectors in $\hat{Q}_k$, harmonic Ritz pairs $\{\hat{Q}_k\hat{y}_i\}_{i=d+1}^{q+s}$ approximating potentially linearly dependent eigenvectors can be computed easily, as long as the eigensolver for the projected problem can find all desired and correspondingly linearly dependent primitive harmonic Ritz vectors $\{\hat{y}_i\}_{i=d+1}^{q+s}$. However, to update Φ_{k+1}^{act}, X_{k+1}^{act}, and X_{k+1}^{tr}, care must be taken to prevent active eigenvector approximations linearly dependent to the converged ones from retrieving converged eigenvalues. To achieve this, we use a temporary block $\hat{X}_{k+1} = [\hat{Q}_k\hat{y}_{d+1},\ldots,\hat{Q}_k\hat{y}_{q+s}] \in \mathbb{C}^{n\times\hat{q}}$ ($\hat{q} = q-d+s$) to hold all active harmonic Ritz vectors and compute the associated block Rayleigh functional value Φ_{k+1}^{act} by solving the small eigenproblem $\hat{X}_{k+1}^{*}\mathbb{T}(\hat{X}_{k+1},\Phi_{k+1}^{act}) = \sum_{i=1}^{m}\left(\hat{X}_{k+1}^{*}A_i\hat{X}_{k+1}\right)Y_{k+1}\mathbf{f}_i(\Phi_{k+1}^{act}) = 0_{\hat{q}\times\hat{q}}$. To avoid repeated convergence, we compute sufficiently many Ritz pairs of this small problem, then choose $\hat{q}$ Ritz pairs with Ritz values $\{\rho_{k+1}^{(d+i)}\}_{i=1}^{\hat{q}}$ closest to σ but *not numerically equal to any converged eigenvalues*, and let $\Phi_{k+1}^{act} \leftarrow \text{diag}\big(\rho_{k+1}^{(d+1)},\ldots,\rho_{k+1}^{(d+\hat{q})}\big)$. Then we set $\hat{X}_{k+1} \leftarrow \hat{X}_{k+1}Y_{k+1}$ and reorder the updated pairs $\big\{(\rho_{k+1}^{(d+i)},\hat{X}_{k+1}(:,i))\big\}_{i=1}^{\hat{q}}$ in the ascending order in their weighted eigenresidual norms. We update X_{k+1}^{act} and X_{k+1}^{tr}, respectively, to be the first $q-d$ and the next s reordered harmonic Ritz vectors. This can avoid repeated convergence caused by linearly dependent eigenvectors, but it may get a proper subspace of the eigenspace of an eigenvalue of geometric multiplicity greater than 1, due to the choice of Φ_{k+1}^{act} different from converged eigenvalues. We expect BPHP to find minimal but not necessarily simple invariant pairs.

The BPHP(ℓ) method with soft deflation, thick restart, and the treatment of potential linearly dependent eigenvectors is summarized in Algorithm 1.

Algorithm 1. BPHP(ℓ) method for computing q eigenvalues of $T(\lambda)v=0$ around σ.

Input: Initial eigenvector approximations $X_0 \in \mathbb{C}^{n\times q}$, a shift $\sigma \in \mathbb{C}$, a preconditioner $M \approx T(\sigma)$, integers $\ell, k_{max} > 0$, and a tolerance $\delta > 0$

Output: Eigenpairs $\{(\lambda_i, v_i)\}_{i=1}^{q}$, where $\{\lambda_i\}$ are the eigenvalues of $T(\cdot)$ near σ, such that (V, L) with $V = [v_1, \ldots, v_q]$ and $L = \mathrm{diag}(\lambda_1, \ldots, \lambda_q)$ is a minimal (not necessarily simple) invariant pair of $T(\cdot)$

$X_0^{cvg} \leftarrow [\,]$, $X_0^{tr} \leftarrow [\,]$, $X_0^{act} \leftarrow X_0$, $d \leftarrow 0$, $s \leftarrow 0$, $k \leftarrow 0$; find $\Phi_0^{act} = \mathrm{diag}(\rho_0^{(1)}, \ldots, \rho_0^{(q)})$ and $Y_0 \in \mathbb{C}^{q\times q}$ such that (s.t.) $\sum_{i=1}^{m} \left[\left(X_0^{act}\right)^* A_i X_0^{act}\right] Y_0 \mathbf{f}_i(\Phi_0^{act}) = 0_{q\times q}$, and set $X_0^{act} \leftarrow X_0^{act} Y_0$

while $d < q$ and $k \le k_{max}$ **do**

  $p \leftarrow d + (q-d)(\ell+1) + s$; form $Q_k \in \mathbb{C}^{n\times p}$ by $Q_k(:, 1\!:\!d) \leftarrow X_k^{cvg}$, $Q_k(:, p-s+1\!:\!p) \leftarrow X_k^{tr}$, $Q_k(:, d+1\!:\!q) \leftarrow X_k^{act}(:, 1\!:\!q-d)$, and $Q_k(:, q+1\!:\!p-s) \leftarrow$ an orthonormal basis of $\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathcal{K}_{\ell}\big(T(\cdot, \Phi_k^{act})\mathbb{M}_{\mathbf{\Pi}}^{\dagger}, X_k^{act}(:, 1\!:\!q-d)\big)$, with $\mathbb{M}_{\mathbf{\Pi}}^{\dagger}$ defined in (5.3) and (5.4)

  Form $\hat{Q}_k \in \mathbb{C}^{n\times\hat{p}}$ by choosing the $\hat{p}$ columns of Q_k closest to linearly independent, by rank-revealing QR (assume that these include $\hat{d}$ columns of $Q(:, 1\!:\!d)$, $\hat{p}-\hat{d}-\hat{s}$ columns of $Q(:, d+1\!:\!p-s)$, and $\hat{s}$ columns of $Q(:, p-s+1\!:\!p)$)

  $s \leftarrow$ desired number of thick restart vectors ($s \leftarrow q$ by default)

  Form $\hat{Z}_k = T(\sigma)\hat{Q}_k$, normalize each column of $\hat{Z}_k$, and perform the NLHRR projection

  Solve the projected problem $\hat{Z}_k^* T(\hat{\mu})\hat{Q}_k\hat{y} = 0$ for $q+s$ harmonic Ritz pairs $\{(\hat{\mu}_i, \hat{y}_i)\}$ with $\hat{\mu}_i$ closest to σ; reorder them s.t. $\|\hat{y}_i(1\!:\!\hat{d})\| \ge \|\hat{y}_j(1\!:\!\hat{d})\|$ for any $1 \le i < j \le q+s$

  $\hat{X}_{k+1} \leftarrow \hat{Q}_k[\hat{y}_{d+1}, \ldots, \hat{y}_{d+\hat{q}}]$; compute $\Phi_{k+1}^{act} = \mathrm{diag}(\rho_{k+1}^{(d+1)}, \ldots, \rho_{k+1}^{(d+\hat{q})})$ and $Y_{k+1} \in \mathbb{C}^{\hat{q}\times\hat{q}}$, $\hat{q} = q-d+s$, s.t. $\sum_{i=1}^{m}\left(\hat{X}_{k+1}^* A_i \hat{X}_{k+1}\right) Y_{k+1}\mathbf{f}_i(\Phi_{k+1}^{act}) = 0_{\hat{q}\times\hat{q}}$, where Φ_{k+1}^{act} contains the $\hat{q}$ Ritz values closest to σ but not numerically equal to the converged eigenvalues

  $\hat{X}_{k+1} \leftarrow \hat{X}_{k+1}Y_{k+1}$; reorder the updated harmonic Ritz pairs $\left\{\left(\rho_{k+1}^{(d+i)}, \hat{X}_{k+1}(:, i)\right)\right\}_{i=1}^{\hat{q}}$ in the ascending order in the weighted eigenresidual norms (3.9); let X_{k+1}^{act} and X_{k+1}^{tr} be the first $q-d$ and the rest s reordered harmonic Ritz vectors

  Find the largest integer Δd such that (s.t.) the first Δd pairs of $\left\{\left(\rho_{k+1}^{(d+i)}, X_{k+1}^{act}(:, i)\right)\right\}_{i=1}^{q-d}$ satisfy $\frac{\left\|T\left(\rho_{k+1}^{(d+i)}\right)X_{k+1}^{act}(:,i)\right\|_2}{\left\|T(\sigma)\right\|_F\left\|X_{k+1}^{act}(:,i)\right\|_2} \le \delta$; let $X_{k+1}^{cvg} \leftarrow \left[X_{k+1}^{cvg}\ X_{k+1}^{act}(:, 1\!:\!\Delta d)\right]$, $X_{k+1}^{act} \leftarrow X_{k+1}^{act}(:, \Delta d+1\!:\!q-d)$, $d \leftarrow d+\Delta d$; break the loop if $d \ge q$

  $k \leftarrow k+1$

end while

$\lambda_i \leftarrow \rho_k^{(i)}$, $v_i \leftarrow X_k^{cvg}(:, i)$ $(1 \le i \le d)$ or $X_k^{act}(:, i-d)$ $(d+1 \le i \le q)$; return $\{(\lambda_i, v_i)\}_{i=1}^{q}$

5.5. Computing eigenvalues around multiple shifts. The BPHP method can be used to compute eigenvalues around multiple shifts $\{\sigma_j\}_{j=1}^{r}$ in a straightforward manner, thanks to the soft deflation and treatment of potentially linearly dependent eigenvectors. Assume without loss of generality that BPHP already found q_1 eigenpairs $\{(\lambda_i, v_i)\}_{i=1}^{q_1}$ around σ_1 and now aims to find q_2 eigenpairs around σ_2. To this end, a preconditioner $M \approx T(\sigma_2)$ should be constructed, and BPHP simply includes the converged eigenvectors $\{v_i\}_{i=1}^{q_1}$ into the search subspace and keeps a record of $\{\lambda_i\}_{i=1}^{q_1}$ as well. Let $W = [v_1, \ldots, v_{q_1}]$, and we develop the BPHP(ℓ) search subspace as

$$\mathcal{Q}_k = \mathrm{range}(W) + \mathrm{range}(X_k^{cvg}) + \mathcal{K}_{\ell+1}\left(\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathbb{T}(\cdot, \Phi_k^{act}), X_k^{act}\right) + \mathrm{range}(X_k^{tr}). \tag{5.9}$$

BPHP handles the previously deflated eigenvectors W the same way as it processes the newly converged eigenvectors X_k^{cvg} associated with eigenvalues around σ_2.

Specifically, a rank-revealing QR is used to find a numerically linearly independent set of $\hat{p}$ vectors spanning $\mathcal{Q}_k$, which consists of $\hat{q}_1$ vectors from W, $\hat{d}$ columns from X_k^{cvg}, $\hat{p}-\hat{q}_1-\hat{d}-\hat{s}$ columns from the original vectors spanning $\mathcal{K}_{\ell+1}\big(\mathbb{M}_{\mathbf{\Pi}}^{\dagger}\mathbb{T}(\cdot,\Phi_k^{act}),X_k^{act}\big)$, and $\hat{s}$ columns from X_k^{tr}. These $\hat{p}$ vectors are normalized and assembled into $\hat{Q}_k$, and $\hat{Z}_k = T(\sigma_2)\hat{Q}_k$ is formed and columnwise normalized. Then we solve the NLHRR projected problem $\hat{Z}_k T(\hat{\mu})\hat{Q}_k\hat{y} = 0$ for q_1+q_2+s primitive harmonic Ritz pairs $\{(\hat{\mu}_i,\hat{y}_i)\}$, q_1 of which are closest to σ_1, and q_2+s of which are closest to σ_2 (the latter set needs to be chosen carefully so that it does not overlap with the former). These pairs are ordered such that $\|\hat{y}_i(1\!:\!q_1+d)\| \geq \|\hat{y}_j(1\!:\!q_1+d)\|$ for all valid $i<j$. Here, $\{Q_k y_i\}_{i=1}^{q_1+d}$ should not be computed as they retrieve the converged eigenvectors. Let $\hat{X}_{k+1} = Q_k[y_{q_1+d+1},\ldots,y_{q_1+q_2+s}]$. An associated Y_{k+1} and block Rayleigh functional value Φ_{k+1}^{act} are computed such that $\sum_{i=1}^{m}(\hat{X}_{k+1}^{*}A_i\hat{X}_{k+1})Y_{k+1}\mathbf{f}_i(\Phi_{k+1}^{act}) = 0_{\hat{q}\times\hat{q}}$, with $\hat{q} = q_2-d+s$. Note that Φ_{k+1}^{act} is constructed by choosing the eigenvalues of this small problem that are numerically not equal to the q_1 converged eigenvalues near σ_1 and the d converged ones near σ_2. We set $\hat{X}_{k+1}\leftarrow\hat{X}_{k+1}Y_{k+1}$ and choose the q_2-d updated harmonic Ritz pairs with minimal weighted eigenresiduals to update X_{k+1}^{act}, and the next s pairs to update X_{k+1}^{tr}.

Soft deflation of converged eigenpairs near multiple shifts can be done similarly, but the cost for deflating a large number of eigenpairs is quite expensive. Fortunately, this goal can be fulfilled much more efficiently if the current shift is close to only a small number of shifts already processed and is relatively far from other shifts. In this case, it is sufficient to deflate the converged eigenpairs around the shifts close to the current one. This is because BPHP converges toward eigenvalues around the shift σ_i associated with the preconditioner $M\approx T(\sigma_i)$ but not toward those around the shifts far from σ_i. This locality property has been used to enable a moving-window-like *partial deflation* technique for solving many successive eigenvalues of nonlinear Hermitian eigenproblems that satisfy a standard min-max principle [49]. We will illustrate the effectiveness of this approach for computing many eigenvalues of the general problem $T(\lambda)v=0$ in the next section.

6. Numerical experiments. We provide numerical results demonstrating the local convergence of PHP and the behavior of BPHP with different features enabled. Compared to the infinite Arnoldi method [2, 18, 19] and the CORK method [4], BPHP is storage-efficient, and it exhibits robust convergence with preconditioners of different type and quality, without the need for exact solution of large linear systems. Such advantages are favorable to tackle large-scale problems for which storage efficiency is crucial and the use of approximate linear solvers is mandatory.

We choose six test problems from the collection of NLEVPs toolbox [5], summarized in Table 6.1. For each problem, we show the problem type, size, the shift σ, and the eigenvalue closest to σ. For example, $foundation$ is a quadratic eigenvalue problem of order 3627; we are computing eigenvalues near $\sigma=-2500-100i$, and the eigenvalue closest to σ is $\lambda_1=-2042.2-73.038i$. *gen_hyper*2 is constructed by specifying all its eigenvalues and associated eigenvectors and is used only to demonstrate BPHP computing linearly dependent eigenvectors. The last three problems are constructed by specifying the problem size parameter $n_0 = 32768, 131072$, and 524288, respectively, and using default values for other inputs.

The NLHRR projected problem $\hat{Z}_k T(\hat{\mu})\hat{Q}_k\hat{y}=0$ and the block Rayleigh functional are both solved by `polyeig` in MATLAB for problems *gen_hyper*2, $foundation$, $butterfly$ and *pdde_stability* and by the unrestarted infinite Arnoldi method [19] for *gun* and *loaded_string*. Infinite Arnoldi takes $\min(10+6r,128)$ steps to compute

TABLE 6.1
Description of test problems.

Problem	Type	Order	σ	λ_1
gen_hyper2	quadratic	1024	-2.109	-2.1066
foundation	quadratic	3627	$-2500-100i$	$-2042.2-73.038i$
gun	nonlinear	9956	52000	$54550+459.52i$
butterfly	quartic	32761	$0.8+0.8i$	$0.80327+0.80022i$
pdde_stability	quadratic	131044	-0.1	$-0.10255-6.2741\times10^{-5}i$
loaded_string	rational	524288	1400	1307.3

r (harmonic) Ritz pairs. To speed up the convergence of infinite Arnoldi, we let $\lambda = \kappa\mu + \sigma$ and transform the original problem to $\tilde{T}(\mu)v = T(\kappa\mu+\sigma)v = 0$, so that the eigenvalues μ of $\tilde{T}(\cdot)$ around the origin correspond to those of $T(\cdot)$ near σ; the transformed eigenvalues are more closely clustered than the original ones with a $\kappa > 1$. We let $\kappa = 10000$ and 70 for *gun* and *loaded_string*, respectively.

All experiments are done on an Apple MacBookPro, running OS X 10.11.6 and MATLAB R2016b, with a 2.9 GHz Intel Core i5 CPU and 16 GB 1867 MHz DDR 3 memory.

6.1. Local convergence. First, we show that the (single-vector) PHP(ℓ) method exhibits linear or quadratic local convergence, as summarized in Theorem 4.4. For each of the last five problems in Table 6.1, we seed the MATLAB random number generator by calling `rng('shuffle')`, then initialize the starting vector x_0 by the function `randn`.

To show the linear convergence, we let $\ell = 2$ for PHP(ℓ); for the quadratic convergence, we let $\ell_0 = 1$ and $\ell_{k+1} = 2\ell_k$ in each new iteration. In both cases, PHP is terminated once the eigenresidual norm $res_k = \frac{\|T(\mu_k)x_k\|_2}{\|T(\sigma)\|_F\|x_k\|_2}$ falls below 10^{-12} *and* stops decreasing for two successive iterations. We choose the eigenresidual norms from s successive iterations, after the local convergence behavior has been established and before the residual norms become sufficiently small and begin to decrease twice slower ($\frac{res_k}{res_{k-1}} > 2\frac{res_{k-1}}{res_{k-2}}$) due to the limit of computer arithmetic. Then we perform a least-squares fitting of the form $y = \alpha + \beta x$ for the data points $(\ln res_k, \ln res_{k+1})$ to get an estimated order of convergence β. The average value of β is reported from the experiment repeated 20 times.

To construct the preconditioner $M \approx T(\sigma)$, we perform a sparse LU factorization of $T(\sigma)$ by `lu` in MATLAB with five output parameters, update the upper triangular factor as $U \leftarrow \mathrm{diag}(U)^{-1}U$ (which becomes a unit upper triangular), and drop all entries from the two triangular factors below a threshold. This drop tolerance is 10^{-4} for *pdde_stability* and 10^{-3} for other problems. This approach is not practical for very large problems, yet we use it here for illustration, due to its greater simplicity for tuning than the incomplete factorization function `ilu`. These preconditioners will be used in this section through section 6.5.

Table 6.2 summarizes the average estimated order of local convergence of PHP(ℓ) with $\ell = 2$ and $\ell_{k+1} = 2\ell_k$. For example, for *butterfly*, we sample $s = 16$ successive iterations of PHP(2) and obtain the estimated order of local convergence $\beta = 1.007$; similarly, residual norms from 6 successive iterations of PHP(ℓ_k) with $\ell_{k+1} = 2\ell_k$ lead to an estimated order of convergence $\beta = 1.918$. These estimates support Theorem 4.4 on the linear and quadratic convergence of PHP. For *loaded_string*, PHP(ℓ_k) with $\ell_{k+1} = 2\ell_k$ converges in only 3 iterations, and it is hard to see quadratic convergence clearly.

TABLE 6.2
Average estimated order of local convergence.

Problem	$\ell = 2$		$\ell_{k+1} = 2\ell_k$	
	s	β	s	β
foundation	9	0.997	4	1.851
gun	9	1.027	4	2.050
butterfly	16	1.007	6	1.918
pdde_stability	18	0.958	4	2.051
loaded_string	10	0.836	3	1.337

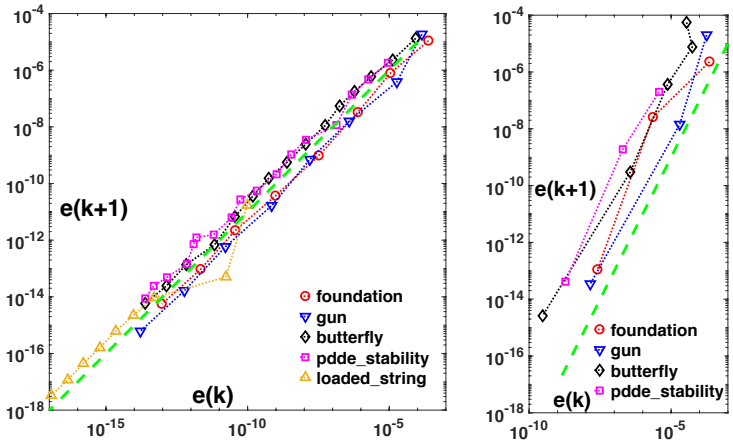

FIG. 6.1. *Typical local convergence history of PHP(ℓ). Left: PHP(2). Right: PHP(ℓ_k) with* $\ell_{k+1} = 2\ell_k$.

In Figure 6.1, we give a typical history of convergence of PHP(2) and PHP(ℓ_k) with $\ell_{k+1} = 2\ell_k$, showing the relation between $\ln res_{k+1}$ and $\ln res_k$. We see that the curves on the left have a slope of 1 and those on the right roughly have a slope of 2 (marked by the green dashed curves), showing linear and quadratic convergence.

6.2. Weighted eigenresidual criterion for subspace extraction. In this and the next few sections, we give numerical results demonstrating several strategies for enhancing the convergence, all important for BPHP to achieve competitive performance.

As explained in section 3.3, when we extract updated eigenpair approximations from the search subspace through NLHRR, choosing harmonic Ritz values closest to the target shift σ and associated harmonic Ritz vectors as the new desired approximations may delay the convergence, due to the presence of potential spurious harmonic Ritz values.

We use BPHP(2) to compute 10 eigenvalues near the shift σ specified in Table 6.1. The tolerance of the relative eigenresidual $\frac{\|T(\mu_k)x_k\|_2}{\|T(\sigma)\|_F\|x_k\|_2}$ is 10^{-15} for *loaded_string* and 10^{-12} for other problems. ILU preconditioners remain the same as in the previous section. In MATLAB, `rng` is set to its default state to generate the initial block X_0. All other features of BPHP discussed before, including stabilized preconditioners, NLHRR projection, thick restart, and soft deflation, are enabled. Such a configuration will be used from now on, unless otherwise specified.

Table 6.3 shows the BPHP iteration counts, with block size 13 (the default block size is $\lceil 1.25q \rceil$, where q is the number of desired eigenpairs) and the weighted eigenresidual norm (3.9) with different values of p used for subspace extraction.

TABLE 6.3
BPHP iteration counts with block size 13 *and subspace extraction* (3.9) *for finding* 10 *eigenvalues.*

Problem	$p = 0$	$p = 1$	$p = 2$	$p = \infty$
foundation	18	16	16	16
gun	100^+ (7)	23	23	23
butterfly	29	31	30	100^+ (6)
pdde_stability	12	12	13	13
loaded_string	7	4	5	4

TABLE 6.4
BPHP(2) *iteration counts with nonstabilized and stabilized preconditioning.*

Problem	Without stabilization	With stabilization
foundation	18	16
gun	26	23
butterfly	100^+ (0)	31
pdde_stability	18	12
loaded_string	9	4

For example, all desired eigenvalues of *loaded_string* are found in at most 5 iterations if $p \geq 1$. Interestingly, the criteria based solely on the distance ($p = 0$) and solely on the eigenresidual norm ($p = \infty$) both lead to failure of convergence for certain problems. By default, we let $p = 1$.

6.3. The effect of stabilized preconditioning. We give numerical results to show the importance of using appropriate projectors for stabilizing the preconditioners to achieve robust convergence. Table 6.4 shows the BPHP(2) iteration counts with nonstabilized and stabilized preconditioning in (5.2) and (5.4). For *foundation* and *gun*, the difference is small, but the stabilization makes a considerable improvement for other problems. In particular, no convergence is achieved at all for *butterfly* without stabilization of preconditioning. Though stabilization requires two additional blocks of storage (Z_k and $M^{-1}Z_k$ in (5.4)), it is clear that this strategy should be enabled by default.

6.4. Regular and harmonic Rayleigh–Ritz. The Rayleigh–Ritz projection and the harmonic variant have been widely used for solving large-scale linear eigenproblems by iterative projection methods. It has been observed that harmonic Rayleigh–Ritz is more favorable for the convergence toward interior eigenvalues than Rayleigh–Ritz [28, 36, 41]. We now show by experiments that such an observation is also true for nonlinear eigenproblems.

Table 6.5 shows the BPHP(2) iteration counts with NLRR and NLHRR projections, respectively. For *foundation*, BPHP(2) with NLRR takes 19 iterations to

TABLE 6.5
BPHP(2) *iteration counts with Rayleigh–Ritz and harmonic Rayleigh–Ritz projection for finding* 10 *eigenvalues near* σ.

Problem	Rayleigh–Ritz	Harmonic Rayleigh–Ritz
foundation	19	16
gun	28	23
butterfly	100^+ (0)	31
pdde_stability	13	12
loaded_string	5	4

TABLE 6.6
BPHP iteration counts without and with thick restart for finding 10 *eigenvalues near* σ.

Problem	No thick restart	Thick restart
foundation	21	16
gun	34	23
butterfly	37	31
pdde_stability	17	12
loaded_string	5	4

converge, but only 16 iterations with NLHRR. Similar observation applies to *gun*, *pdde_stability*, and *loaded_string*. For *butterfly*, BPHP with NLRR did not find any eigenvalues near σ in 100 iterations. Overall, the convergence is clearly more robust and faster with NLHRR projection.

One difference between NLRR and NLHRR is that the former may need less storage, as the later uses different search and test subspaces. In fact, the storage cost is the same if we can perform matrix-vector multiplications involving $T(\sigma)^*$. Note from (2.1) and Algorithm 1 that NLRR and NLHRR construct the projected problems $(\sum_{i=1}^m \hat{Q}_k^* A_i \hat{Q}_k f_i(\hat{\mu}))\hat{y} = 0$ and $(\sum_{i=1}^m \hat{Q}_k^* T(\sigma)^* A_i \hat{Q}_k f_i(\hat{\mu}))\hat{y} = 0$, respectively. Both projections need a temporary storage for $A_i\hat{Q}_k$, but NLHRR overwrites it with $T(\sigma)^* A_i \hat{Q}_k$ (which is done in our MATLAB code). Nevertheless, more storage is needed for NLHRR if matrix-vector multiplications involving $T(\sigma)^*$ cannot be performed, and we have to form $\hat{Z}_k = T(\sigma)\hat{Q}_k$ explicitly.

6.5. Thick restart. We provide numerical evidence to show the effectiveness of thick restart. This approach augments the original search subspace with additional eigenvector approximations selected by our criterion of extraction (3.9).

The iteration counts of BPHP in Table 6.6 illustrate the improvement achieved by thick restart. It leads to a considerable reduction in iteration counts for several problems. Thick restart requires an additional block of storage and a mild increase in arithmetic cost per iteration (solve a larger projected eigenproblem), but the improvement in convergence shows the cost is justified. We recommend this strategy to be enabled by default.

6.6. Preconditioning by alternative techniques. Note that the action of preconditioning for BPHP may be *any* strategy approximating $T(\sigma)^{-1}$, e.g., a generic iterative solver for linear systems of the form $T(\sigma)y = b$ or an ILU preconditioner with different drop tolerances. We implemented the action of M^{-1} on a block of vectors B by applying a right-preconditioned block induced dimension reduction (IDR(s)) method [11] to the block linear systems $T(\sigma)Y = B$. To precondition block IDR(s), we use sparse `lu` in MATLAB with five output parameters, update the upper triangular factor as $U \leftarrow D^{-1}U$, and drop entries below 2×10^{-3} in magnitude from the triangular factors for *pdde_stability* and below 2×10^{-2} for other problems. We choose block IDR(2) and terminate the iterative linear solver once the *relative residual norm* of each linear system falls below 10^{-2}. Such an accuracy for the approximate linear solve is fairly low in general but seems sufficient to help achieve full convergence speed of BPHP in our experiments. We compare this preconditioning action with the exact preconditioning $M = T(\sigma)$ (done by `lu` factorization) and the ILU preconditioner with drop tolerance 10^{-5} for *pdde_stability* and 10^{-4} for others. All the three preconditioners are stronger than the ILU preconditioners used in previous sections, and hence the iteration counts are lower.

Table 6.7 shows the BPHP iteration counts using the exact preconditioning $M = T(\sigma)$, action of ILU with *smaller* drop tolerances, block IDR(2) solve with relative tolerance 10^{-2}, and action of ILU with *larger* drop tolerances. The counts show that our stronger ILU preconditioning and block IDR(2) solves are as effective as exact preconditioning for the test problems. This highlights the potential of BPHP to tackle large-scale problems, where iterative solution of linear systems $T(\sigma)Y = B$ of multiple right-hand side to a low accuracy is typical and much more efficient than exact solutions.

Meanwhile, the results obtained by using ILU with larger drop tolerances show the *robustness* of our BPHP method. With such weak preconditioners, one can improve the convergence by using a larger block size, a larger value ℓ for BPHP(ℓ), and a larger value κ for the infinite Arnoldi method solving the NLHRR projected problems. Also, stronger preconditioners (closer to $T(\sigma)$) are needed if the desired eigenvalues are relatively far from σ.

Details of the eigenresidual history of the 10 eigenvalues of *gun* computed by BPHP with different preconditioners are illustrated in Figure 6.2, showing that ILU

TABLE 6.7
BPHP iteration counts with several preconditioning strategies for finding 10 *eigenvalues.*

Problem	BPHP(2) (block size 10) $M^{-1} = T(\sigma)^{-1}$	BPHP(2) (block size 10) $M^{-1} =$ ILU solve (droptol 10^{-4} or 10^{-5})	BPHP(2) (block size 10) $M^{-1} =$ block IDR(2) solve (rel. tol 10^{-2})	BPHP(4) (block size 13) $M^{-1} =$ ILU solve (droptol 10^{-2} or 10^{-3})
foundation	12	12	13	22
gun	17	17	18	61
butterfly	28	26	26	51
pdde_stability	8	7	7	17

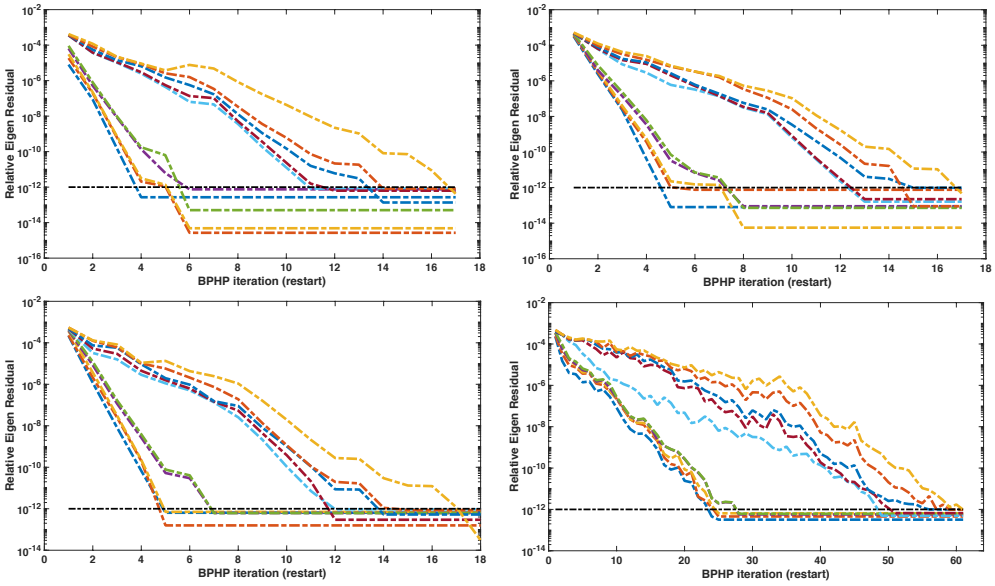

FIG. 6.2. *Eigenresidual history of* 10 *eigenvalues of problem gun computed by BPHP. Top left:* $M^{-1} = T(\sigma)^{-1}$. *Top right:* $M^{-1} =$ *action of* ILU *with drop tolerance* 10^{-4}. *Bottom left:* $M^{-1} =$ block IDR(2) *solve with relative linear tolerance* 10^{-2}. (*All these three are computed by* BPHP(2) *with block size* 10.) *Bottom right:* $M^{-1} =$ *action of* ILU *with drop tolerance* 10^{-2} (*computed by* BPHP(4) *with block size* 13).

preconditioner with drop tolerance 10^{-4} and preconditioning with block IDR(2) solve with relative linear tolerance 10^{-2} are essentially as good as exact preconditioning.

6.7. Computing linearly dependent eigenvectors. We discussed in section 5.4 the computation of eigenpairs involving linearly dependent eigenvectors. This is achieved by removing numerically linearly dependent vectors from the original search subspace, computing updated eigenpairs through NLHRR projection and eigenresidual-based extraction, and finding the Rayleigh functional values of the active eigenvector approximations *different from* the converged eigenvalues. We show the effectiveness of BPHP for solving several artificially constructed problems involving linearly dependent eigenvectors.

Three test problems of order $n = 1024$ are constructed from the quadratic eigenproblem *gen_hyper*2 in NLEVP, as it allows users to specify the eigenvalues and eigenvectors. We choose $2n = 2048$ real eigenvalues by random, ranging from $\lambda_1 = 361.7868$ to $\lambda_{2n} = -317.5928$ in a decreasing order. We consider the set of 10 eigenvalues of interest, namely, $\lambda_{n-4}, \lambda_{n-3}, \ldots, \lambda_{n+5}$, centered around $\sigma = -2.109$. The first problem *gen_hyper*2_*A* has distinct desired eigenvalues; the eigenvectors are $v_{n-j} = e_{n-j}$ for $0 \le j \le 4$ and randomly formed v_{n+j} lying in $\text{span}\{e_{n-4}, e_{n-3}, e_{n-2}, e_{n-1}, e_n\}$ for $1 \le j \le 5$. The second problem *gen_hyper*2_*B* is similar to *gen_hyper*2_*A*, but it has $v_{n+j} = e_{n+1-j}$ for $1 \le j \le 5$; that is, λ_{n-j} and λ_{n+j+1} have the identical eigenvector e_{n-j} for $0 \le j \le 4$. The last problem *gen_hyper*2_*C* has identical eigenvectors as *gen_hyper*2_*B*, but they belong to two semisimple eigenvalues $\lambda_{n-4} = \ldots = \lambda_n$ and $\lambda_{n+1} = \ldots = \lambda_{n+4}$ of multiplicity 5. In terms of the level of linear dependence among eigenvectors, *gen_hyper*2_*A* is the lowest (hence easiest), *gen_hyper*2_*B* is medium, and *gen_hyper*2_*C* is the highest (most difficult).

BPHP(2) with exact preconditioning is used to find the eigenpairs to the relative tolerance 10^{-10}. The initial iterate is formed by calling `rng('shuffle')` followed by invoking `randn`. We count the converged eigenvalues that belong to the sets $\{\lambda_{n-4}, \ldots, \lambda_n\}$ and $\{\lambda_{n+1}, \ldots, \lambda_{n+5}\}$, respectively, and report the average number of eigenvalues found in 100 repeated experiments. For problem *gen_hyper*2_*A*, Table 6.8 shows that BPHP(2) finds all 10 eigenvalues $\{\lambda_{n-4}, \ldots, \lambda_{n+5}\}$ every time. For *gen_hyper*2_*B*, the average numbers decreased slightly, probably due to a stronger linear dependence relation among the eigenvectors of interest. For the last problem, on average, BPHP is able to compute nearly 4.6 (out of 5) linearly independent vectors lying in the eigenspaces for each of the semisimple eigenvalues. In addition, we noticed that the BPHP iteration counts needed for solving the last problem are higher than those needed for solving the first two problems.

As explained in section 1, BPHP aims to compute a minimal invariant pair in decoupled form, which is not necessarily simple. For the first two problems with 10 distinct eigenvalues, BPHP has never converged to any eigenvalue repeatedly; similarly for the last problem, BPHP always found no more than 5 linearly independent eigenvectors associated with each of the semisimple eigenvalues. This method seems reasonably reliable in avoiding repeated convergence (overestimate of the multiplicity) caused by linearly dependent eigenvectors.

6.8. Computing many eigenvalues. It is shown in section 5.5 that a large number of eigenvalues can be computed by a moving-window-style partial deflation technique. The computation of eigenvalues of the current shift only requires the deflation of converged eigenvalues around nearby shifts. Appropriate use of such a "local" effect is essential to achieve an overall arithmetic cost roughly proportional to the number of desired eigenvalues [49].

TABLE 6.8
Average number of linearly dependent eigenvectors found by BPHP(2).

	$\{\lambda_{n-4},\ldots,\lambda_n\}$		$\{\lambda_{n+1},\ldots,\lambda_{n+5}\}$	
Problem	Avg. found	Repeated	Avg. found	Repeated
gen_hyper2_A	5	0	5	0
gen_hyper2_B	4.80	0	4.78	0
gen_hyper2_C	4.56	0	4.56	0

TABLE 6.9
BPHP with moving-window-style partial deflation for computing many eigenvalues.

Problem	σ_1	Direction	# Shifts	ℓ	Tolerance	# EV found	Repeated
foundation	$5500+275i$	$-20-1i$	64	3	10^{-12}	540	0
gun	51000	$1+0i$	64	3	10^{-12}	568	0
butterfly	$0.77+0.6i$	$0.44+1i$	33	5	10^{-10}	246	0
pdde_stability	-600	$1+0i$	134	3	10^{-12}	1224	5

Our experiments showing the effectiveness of this approach are as follows. We choose an initial shift $\sigma_1 \in \mathbb{C}$, and determine a moving direction $\gamma = \alpha + \beta i$ in the complex plane going from σ_1, whose equation is $\alpha\,(y - \mathrm{imag}(\sigma_1)) = \beta\,(x - \mathrm{real}(\sigma_1))$. After q eigenvalues near σ_1 have been found, we project them orthogonally onto this line and let σ_2 be the midpoint between the two projected points on the line that are the second and the third farthest into the moving direction. If BPHP fails to find $f \le \lfloor \frac{q}{2} \rfloor$ eigenvalues near σ_1, we start from those two farthest projected points and walk back from the moving direction by $\lceil \frac{f}{2} \rceil$ projected points; if BPHP fails to find $f > \lfloor \frac{q}{2} \rfloor$ eigenvalues, we walk back by f projected points. Then, σ_2 is set to be the midpoint between the two projected points currently focused.

To achieve more robust convergence, we invoke BPHP(5) for *butterfly* and BPHP(3) for other problems and use the exact preconditioning $M = T(\sigma)$ so that eigenvalues not very close to σ can also be found. For each shift σ_i, we compute $q = 8$ eigenvalues nearby, deflating $3q = 24$ most recently converged eigenpairs. All features of BPHP previously shown effective for improving convergence are enabled.

Table 6.9 summarizes the results. For instance, for problem *foundation*, BPHP(3) starting with the initial shift $\sigma_1 = 5500 + 275i$ and subsequent shifts moving toward direction $-20 - 1i$ found 540 eigenvalues to relative tolerance 10^{-12} around 64 shifts, and no repeated convergence occurred. Note that the total number of eigenvalues found for *foundation* and *gun* are greater than $8 \times 64 = 512$, because around quite a few shifts, more than 8 eigenvalues were found once the 8th eigenvalue converged. For *butterfly*, BPHP(5) found 246 eigenvalues around 33 shifts, about 7.5 per shift on average; also, BPHP(5) terminated at the final shift $0.8890+0.8705i$, because no more eigenvalues are sufficiently close to the shift. In Figure 6.3, we plot the eigenvalues by circles and mark the 5 eigenvalues found repeatedly by diamonds for *pdde_stability*. These repeated eigenvalues arise in the early stage of computation and may be avoided by using a larger window size for deflation.

These results illustrate the potential of BPHP with the moving-window-style partial deflation for computing many eigenvalues at economic arithmetic and reasonable storage cost. The arithmetic cost is roughly proportional to the total number of eigenvalues found, and storage cost is fixed and about $\ell + 4 + 3 \times \mathit{windowsize}$ blocks of vectors ($\ell + 2$ blocks for the search subspace and 2 blocks for the preconditioning stabilization for the active block, plus the window-sized blocks of eigenvectors in soft deflation and twice more for their preconditioning stabilization). For BPHP(3) with

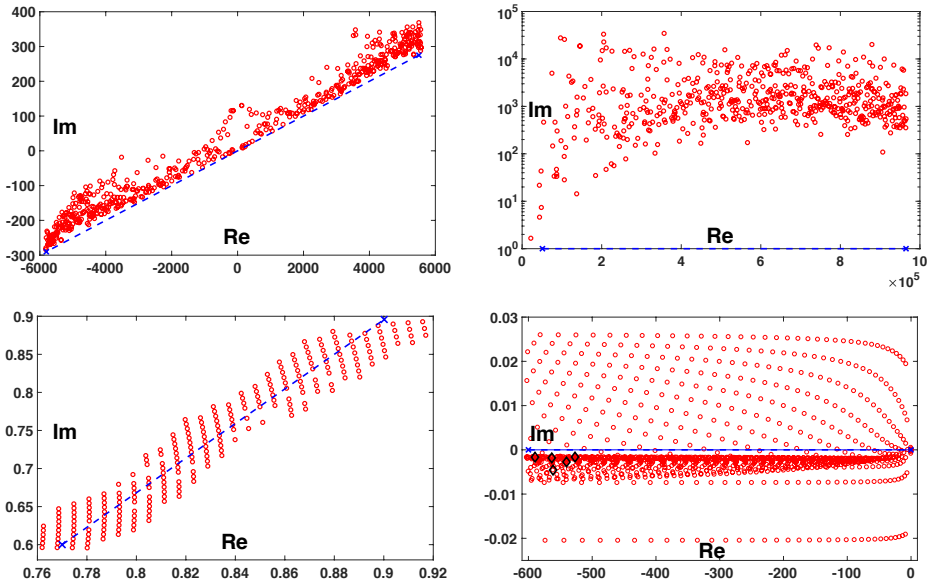

FIG. 6.3. *Eigenvalues around a sequence of shifts found by BPHP with moving-window-style deflation. Top left: foundation. Top right: gun. Bottom left: butterfly. Bottom right: pdde_stability.*

block size 8 and soft deflation of 3 blocks of previously converged eigenvectors, we need about $(3 + 4 + 3 \times 3) \times 8 = 128$ vectors. This estimate does not include the storage for the original preconditioners and solving the projected eigenproblems.

6.9. Comparison with other solvers. We provide numerical results for comparing BPHP with two state-of-the-art nonlinear eigensolvers, namely, the infinite Arnoldi method [2, 18, 19] and the CORK method [4]. We are interested in comparing the storage cost, running time, and robustness of convergence to relatively high accuracy. We use the same test problems as summarized in Table 6.1. For each problem, we compute 12 eigenvalues around the specified shift σ, to the relative tolerance $\frac{\|T(\mu_k)x_k\|_2}{\|T(\sigma)\|_F\|x_k\|_2} \leq 10^{-10}$ for *foundation*, *gun*, *butterfly* and *pdde_stability*, and 10^{-15} for *loaded_string*.

We use BPHP(3) with block size 12, which needs a storage of $(3 + 4) \times 12 = 84$ vectors (1 block for the eigenvectors, 3 blocks for the preconditioned Krylov space, 1 block for thick restart, and 2 blocks for Z_k and $M^{-1}Z_k$ used for preconditioning stabilization (5.4)).

To assess the infinite Arnoldi method, as the code of the implicitly restarted variant [18] is not publicly available, we run the *unrestarted* variant to get the number of Arnoldi steps needed for convergence, then compute a *lower bound* of the time for the implicitly restarted variant. For example, assume that the implicitly restarted variant restarts at Krylov subspace dimension $12 \times 2 + 1 = 25$, takes 10 seconds from dimension 1 through 12 and 20 seconds from dimension 13 through 25, and the unrestarted variant needs 50 steps to converge. Then a lower bound on time for the restarted variant is $10 + 20 \times \frac{50-12}{25-12} = 68.46$ seconds. Such an estimate is conservative, as we assume that the restarted variant needs the same number of Arnoldi steps as the unrestarted variant, and no cost of restart is calculated.

For CORK, we use the first companion form to linearize polynomial eigenproblems, and the rational monomial basis [31] to linearize *loaded_string*; CORK has a

built-in linearization for *gun*. To make the comparison fair, we let CORK use only one pole at σ, and the restart dimension is set as 42 or 84, either half or the same number of vectors used for BPHP(3). This is because the current CORK implementation needs the same amount of storage for the compact Krylov subspace and the desired eigenvectors, and it is not clear if such a storage cost can be reduced. As a result, CORK will use either the same amount of storage or twice as much as BPHP(3). Note that the storage cost is not considered for the preconditioners and iterative linear solvers; also, CORK may need considerable storage for the linearization of fully nonlinear problems such as *gun*. The number of restarted Ritz values is set to 20.

For all problems except *loaded_string*, the preconditioner for BPHP(3) is the block IDR(2), with linear solve relative tolerance 10^{-2}, preconditioned by the ILU preconditioner with drop tolerance 2×10^{-2} for $foundation$, *gun* and $butterfly$, and 2×10^{-3} for *pdde_stability*; the linear solver for infinite Arnoldi and CORK is IDR(4) [42], preconditioned by the same ILU preconditioner, to relative tolerance 10^{-10}. For *loaded_string*, the preconditioner for BPHP(3) and linear solves for infinite Arnoldi and CORK are performed by direct method based on `lu` in MATLAB with five output parameters, since $T(\sigma)$ is symmetric tridiagonal.

The results are summarized in Tables 6.10 and 6.11. For *gun*, e.g., BPHP(3) found all 12 eigenvalues in 11 iterations (restarts) and 66.90 seconds, restarted infinite Arnoldi needs *at least* 63 steps and 58.26 seconds, and CORK is finished in 82 steps with a Kyrlov subspace of dimension 42 (84 vectors used), or 72 steps (144 vectors used) and 70.87 seconds with the space of dimension 84. For the three polynomial problems, CORK fails to find any eigenvalue to relative tolerance 10^{-10}, no matter how large the subspace is and whether direct linear solvers with iterative refinement are used. It is not clear, however, whether CORK would exhibit improved stability with different linearization and system parameters.

These results lead to our general conclusions as follows.

1. Infinite Arnoldi is the most robust for computing eigenvalues to machine precision but is also the most storage-consuming algorithm. Also, primarily due to the orthogonalization cost for such a space of long vectors, the running time for this algorithm is also the longest for relatively large problems such as

TABLE 6.10
Comparison of BPHP, infinite Arnoldi, and CORK for computing 12 *eigenvalues.*

	BPHP(3)			Infinite Arnoldi				CORK		
Problem	Vec.	Iter	Time	Vec.	κ	Step	Time	Vec.	Step	Time
$foundation$	84	8	14.83	625	100	≥ 40	> 11.45	84 or 168	∞	∞
gun	84	11	66.90	625	10^4	≥ 63	> 58.26	84	82	81.83
								144	72	70.87
$butterfly$	84	19	106.16	625	1	≥ 43	> 42.32	84 or 168	∞	∞
pdde_stability	84	6	216.34	625	1	≥ 38	> 261.26	84 or 168	∞	∞
loaded_string	84	5	94.28	625	400	≥ 48	> 272.53	62	31	26.29

TABLE 6.11
Comment on CORK for solving several polynomial eigenvalue problems.

$foundation$	subspace dim 42: 9 eigenvalues converged to tol. 10^{-6} in 78 steps, 34.88 secs
	subspace dim 84: 9 eigenvalues converged to tol. 10^{-6} in 84 steps, 31.39 secs
	any space dim ≥ 20: 2 converged to tol. 10^{-7}; no convergence to 10^{-8}
$butterfly$	any space dim ≥ 20: 2 converged to tol. 10^{-4}; no convergence to 10^{-5}
pdde_stability	subspace dim 42: 12 eigenvalues converged to tol. 10^{-7} in 27 steps, 135.41 secs
	any space dim ≥ 20: 2 converged to tol. 10^{-8}; no convergence to 10^{-9}

pdde_stability and *loaded_string*. Another issue is that the method needs an appropriate scaling factor κ to keep a balance between the rate of convergence and numerical stability.

2. If the linear systems of the form $T(\sigma)x = b$ can be solved efficiently by factorization, such as for *loaded_string*, CORK can be the most efficient, outperforming BPHP and infinite Arnoldi by a big margin. If the linear systems are solved iteratively, BPHP and CORK *might be* comparable (see the result for *gun*), but comprehensive comparison on additional larger problems is needed to draw a conclusion, which most likely depends on the efficiency for solving the linear systems iteratively.
3. The disadvantage of the current CORK code seems to be its lack of robustness of convergence to relatively high accuracy, at least with the first companion linearization of polynomial eigenproblems and the parameters used above. Table 6.11 suggests that the highest relative eigenpair tolerance achievable for $foundation$ and *pdde_stability* is only about 10^{-6} to 10^{-7}, and the problem is most severe for $butterfly$. The code applies reorthogonalization of Krylov subspace vectors and iterative refinement for solving $T(\sigma)x = b$, not cheap for large problems. More experiments are needed to see if the robustness can be improved with structure-preserving linearizations.
4. Considering the overall storage cost, running time, the robust of convergence with preconditioners of different type and quality, and flexibility in the use of approximate linear solves, BPHP would be a competitive method for solving large nonlinear eigenproblems around a specified shift σ, if the linear systems of the form $T(\sigma)x = b$ are difficult to solve by direct methods and necessitate the use of iterative solvers.

7. Conclusion. We proposed a BPHP method for computing several eigenvalues of large-scale nonlinear eigenproblems $T(\lambda)v = 0$ around a specified shift. This algorithm is motivated by recent development of new preconditioned eigensolvers for large nonsymmetric and nonlinear eigenproblems. We discussed the construction of the search subspace, stabilization of preconditioners, local convergence analysis, harmonic projection, weighted eigenresidual criterion of subspace extraction, thick restart, soft deflation, computation of linearly dependent eigenvectors, and the computation of many eigenvalues. Experiments show that BPHP does not need exact solution of relevant linear systems, works effectively with preconditioning techniques, and is storage-efficient; in addition, it exhibits robust convergence and can find linearly dependent eigenvectors.

REFERENCES

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

[2] R. van Beeumen, E. Jarlebring, and W. Michiels, *A rank-exploiting infinite Arnoldi algorithm for nonlinear eigenvalue problems*, Numer. Linear Algebra Appl., 23 (2016), pp. 607–628.

[3] R. van Beeumen, K. Meerbergen, and W. Michiels, *A rational Krylov method based on Hermite interpolation for nonlinear eigenvalue problems*, SIAM J. Sci. Comput., 35 (2013), pp. A327–A350.

[4] R. van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for nonlinear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 820–838.

[5] T. Betcke, N. J. Higham, V. Mehrmann, C. Schröder, and F. Tisseur, *NLEVP: A collection of nonlinear eigenvalue problems*, ACM Trans. Math. Software, 39 (2013), pp. 7:1–7:28.

[6] T. Betcke and D. Kressner, *Perturbation, extraction and refinement of invariant pairs for matrix polynomials*, Linear Algebra Appl., 435 (2011), pp. 514–536.
[7] T. Betcke and H. Voss, *A Jacobi-Davidson-type projection method for nonlinear eigenvalue problems*, Future Generation Comput. Syst., 20 (2004), pp. 363–372.
[8] W.-J. Beyn, *An integral method for solving nonlinear eigenvalue problems*, Linear Algebra Appl., 436 (2012), pp. 3839–3863.
[9] W.-J. Beyn, C. Effenberger, and D. Kressner, *Continuation of eigenvalues and invariant pairs for parameterized nonlinear eigenvalue problems*, Numer. Math., 119 (2011), pp. 489–516.
[10] E. R. Davidson, *The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real symmetric matrices*, J. Comput. Phys., 17 (1975), pp. 87–94.
[11] L. Du, T. Sogabe, B. Yu, Y. Yamamoto, and S.-L. Zhang, *A block IDR(s) method for nonsymmetric linear systems with multiple right-hand sides*, J. Comput. Appl. Math., 235 (2011), pp. 4095–4106.
[12] C. Effenberger, *Robust successive computation of eigenpairs for nonlinear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 1231–1256.
[13] C. Effenberger and D. Kressner, *Chebyshev interpolation for nonlinear eigenvalue problems*, BIT, 52 (2012), pp. 933–951.
[14] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst, *Jacobi-Davidson style QR and QZ algorithms for the reduction of matrix pencils*, SIAM J. Sci. Comput., 20 (1998), pp. 94–125.
[15] S. W. Gaaf and E. Jarlebring, *The infinite Bi-Lanczos method for nonlinear eigenvalue problems*, SIAM J. Sci. Comput., 39 (2017), pp. S898–S919.
[16] S. Güttel and F. Tisseur, *The nonlinear eigenvalue problem*, Acta Numer., 26 (2017), pp. 1–94.
[17] E. Jarlebring, K. Meerbergen, and W. Michiels, *A Krylov method for the delay eigenvalue problem*, SIAM J. Sci. Comput., 32 (2010), pp. 3278–3300.
[18] E. Jarlebring, K. Meerbergen, and W. Michiels, *Computing a partial Schur factorization of nonlinear eigenvalue problems using the infinite Arnoldi method*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 411–436.
[19] E. Jarlebring, W. Michiels, and K. Meerbergen, *A linear eigenvalue algorithm for the nonlinear eigenvalue problem*, Numer. Math., 122 (2012), pp. 169–195.
[20] A. V. Knyazev, *Preconditioned eigensolvers – an oxymoron?*, Electron. Trans. Numer. Anal., 7 (1998), pp. 104–123.
[21] A. V. Knyazev, *Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method*, SIAM J. Sci. Comput., 23 (2001), pp. 517–541.
[22] A. V. Knyazev, *Hard and soft locking in iterative methods for symmetric eigenvalue problems*, presented at the 8th Copper Mountain Conference on Iterative Methods, Copper Mountain, CO, 2004, http://math.ucdenver.edu/~aknyazev/research/conf/cm04_soft_locking/cm04.pdf.
[23] D. Kressner, *A block Newton method for nonlinear eigenvalue problems*, Numer. Math., 114 (2009), pp. 355–372.
[24] B.-S. Liao, Z. Bai, L.-Q. Lee, and K. Ko, *Nonlinear Rayleigh-Ritz iteratve method for solving large scale nonlinear eigenvalue problems*, Taiwanese J. Math., 14 (2010), pp. 869–883.
[25] D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann, *Structured polynomial eigenvalue problems: Good vibrations from good linearizations*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 1029–1051.
[26] D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann, *Vector spaces of linearizations for matrix polynomials*, SIAM. J. Matrix Anal. Appl., 28 (2006), pp. 971–1004.
[27] V. Mehrmann and H. Voss, *Nonlinear eigenvalue problems: A challenge for modern eigenvalue methods*, GAMM-Mitt., 27 (2005), pp. 121–152.
[28] R. B. Morgan, *Computing interior eigenvalues of large matrices*, Linear Algebra Appl., 154–156 (1991), pp. 289–309.
[29] R. B. Morgan and D. S. Scott, *Generalizations of Davidson's method for computing eigenvalues of sparse symmetric matrices*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 817–825.
[30] R. B. Morgan and D. S. Scott, *Preconditioning the Lanczos algorithm for sparse symmetric eigenvalue problems*, SIAM J. Sci. Comput., 14 (1993), pp. 585–593.
[31] Y. Nakatsukasa and F. Tisseur, *Eigenvector error bounds and perturbation for nonlinear eigenvalue problems*, presented at the Manchester Workshop on Nonlinear Eigenvalue Problems (NEP14), University of Manchester, UK, 2014, http://www.cl.eps.manchester.ac.uk/medialand/maths/archived-events/workshops/www.mins.manchster.ac.uk/events/workshops/NEP14/talks/d03s01t03-nakatsukasa.pdf.
[32] A. Neumaier, *Residual inverse iteration for the nonlinear eigenvalue problem*, SIAM J. Numer. Anal., 22 (1985), pp. 914–923.

[33] Y. Notay, *Combination of Jacobi-Davidson and conjugate gradients for the partial symmetric eigenproblem*, Numer. Linear Algebra Appl., 9 (2002), pp. 21–44.
[34] E. Ovtchinnikov, *Convergence estimates for the generalized Davidson method for symmetric eigenvalue problems* I*: The preconditioning aspect*, SIAM J. Numer. Anal., 41 (2003), pp. 258–271.
[35] E. Ovtchinnikov, *Convergence estimates for the generalized Davidson method for symmetric eigenvalue problems* II*: The subspace acceleration*, SIAM J. Numer. Anal., 41 (2003), pp. 272–286.
[36] C. C. Paige, B. N. Parlett, and H. A. van der Vorst, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Numer. Linear Algebra Appl., 2 (1995), pp. 115–133.
[37] K. Schreiber, *Nonlinear Eigenvalue Problems: Newton-type Methods and Nonlinear Rayleigh Functionals*, Ph.D. thesis, Department of Mathematics, Technische Universistät Berlin, 2008.
[38] H. Schwetlick and K. Schreiber, *Nonlinear Rayleigh functionals*, Linear Algebra Appl., 436 (2012), pp. 3991–4016.
[39] D. S. Scott, *Solving sparse symmetric generalised eigenvalue problems without factorisation*, SIAM J. Numer. Anal., 18 (1981), pp. 102–110.
[40] 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 Numer. Math., 36 (1996), pp. 595–633.
[41] G. L. G. Sleijpen and H. A. van der Vorst, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM Rev., 42 (2000), pp. 267–293.
[42] P. Sonneveld and M. B. van Gijzen, *IDR(s): A family of simple and fast algorithms for solving large nonsymmetric linear systems*, SIAM J. Sci. Comput., 31 (2008), pp. 1035–1062.
[43] A. Stathopoulos, *Nearly optimal preconditioned methods for Hermitian eigenproblems under limited memory. Part* I*: Seeking one eigenvalue.*, SIAM J. Sci. Comput., 29 (2007), pp. 481–514.
[44] A. Stathopoulos and C. F. Fischer, *A Davidson program for finding a few selected extreme eigenpairs of a large, sparse, real, symmetric matrix*, Comput. Phys. Comm., 79 (1994), pp. 268–290.
[45] A. Stathopoulos and J. R. McCombs, *Nearly optimal preconditioned methods for Hermitian eigenproblems under limited memory. Part* II*: Seeking many eigenvalues*, SIAM J. Sci. Comput., 29 (2007), pp. 2162–2188.
[46] A. Stathopoulos and J. R. McCombs, *PRIMME: Preconditioned iterative multimethod eigensolver – methods and software description*, ACM Trans. Math. Software, 37 (2010), 21.
[47] G. W. Stewart, *Matrix Algorithms Volume* 2*: Eigensystems*, SIAM, Philadelphia, 2001.
[48] Y. Su and Z. Bai, *Solving rational eigenvalue problems via linearization*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 201–216.
[49] D. B. Szyld, E. Vecharynski, and F. Xue, *Preconditioned eigensolvers for large-scale nonlinear Hermitian eigenproblems with variational characterizations.* II*. Interior eigenvalues*, SIAM J. Sci. Comput., 37 (2016), pp. A2969–A2997.
[50] D. B. Szyld and F. Xue, *Local convergence analysis of several inexact Newton-type algorithms for general nonlinear eigenvalue problems*, Numer. Math., 123 (2012), pp. 333–362.
[51] D. B. Szyld and F. Xue, *Several properties of invariant pairs of nonlinear algebraic eigenvalue problems*, IMA J. Numer. Anal., 34 (2014), pp. 921–954.
[52] D. B. Szyld and F. Xue, *Local convergence of Newton-like methods for degenerate eigenvalues of nonlinear eigenproblems.* I*. Classical algorithms*, Numer. Math., 129 (2015), pp. 353–381.
[53] D. B. Szyld and F. Xue, *Local convergence of Newton-like methods for degenerate eigenvalues of nonlinear eigenproblems.* II*. Accelerated algorithms*, Numer. Math., 129 (2015), pp. 383–403.
[54] D. B. Szyld and F. Xue, *Preconditioned eigensolvers for large-scale nonlinear Hermitian eigenproblems with variational characterizations.* I*. Extreme eigenvalues*, Math. Comp., 85 (2016), pp. 2887–2918.
[55] F. Tisseur and K. Meerbergen, *The quadratic eigenvalue problem*, SIAM Rev., 43 (2001), pp. 235–286.
[56] E. Vecharynski, C. Yang, and F. Xue, *Generalized preconditioned locally harmonic residual method for non-Hermitian eigenproblems*, SIAM J. Sci. Comput., 38 (2016), pp. A500–A527.
[57] H. Voss, *A Jacobi-Davidson method for nonlinear and nonsymmetric eigenproblems*, Comput. Struct., 85 (2007), pp. 1284–1292.