Eigenvalue Computation from the Optimization Perspective: On Jacobi-Davidson, IIGD, RQI and Newton Updates[*]

Yunkai Zhou[†]

Abstract

We discuss the close connection between eigenvalue computation and optimization using the Newton method and subspace methods. From the connection we derive a new class of Newton updates. The new update formulation is similar to the well-known Jacobi-Davidson method. This similarity leads to simplified versions of the Jacobi-Davidson method and the inverse iteration generalized Davidson (IIGD) method. We prove that the projection subspace augmented by the updating direction from each of these methods is able to include the Rayleigh quotient iteration (RQI) direction. Hence, the locally quadratic (cubic for normal matrices) convergence rate of the RQI method is retained and strengthened by the subspace methods. The theory is supported by extensive numerical results. Preconditioned formulations are also briefly discussed for large scale eigenvalue problems.

Keywords: Eigenvalue; Rayleigh quotient; Subspace method; Jacobi-Davidson; IIGD; Newton method

1 Introduction

We study subspace methods for the standard eigenvalue problem

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \tag{1}$$

where $\mathbf{A}$ is an n by n matrix, λ is the eigenvalue and $\mathbf{x}$ is the corresponding eigenvector. Eigenvalue problem (1) is central to many scientific applications. Several excellent monographs [2, 3, 5, 13, 27, 30, 40, 42] contain in-depth discussions on properties and algorithms for eigenvalue problems, together with many sources from science and engineering applications.

Subspace methods include the Arnoldi method [1], Lanczos method [20] and the more sophisticated but important variants, the implicit restart Arnoldi method [35] and the rational Krylov method [29]. Other important subspace methods include the Davidson method [7], Jacobi-Davidson method [32, 33, 16], Krylov-Schur method [41], truncated RQ method [37, 45], and preconditioned subspace eigensolvers [4, 12, 14, 18, 19, 22, 38].

[*]This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, SciDAC Program, Office of Science, U.S. Department of Energy, under contract W-31-109-ENG-38.

[†]Argonne National Laboratory, Mathematics and Computer Science Division, Chemistry Division, Argonne, IL 60439, U.S.A. Email: `zhou@mcs.anl.gov`

In this paper we study mainly the Jacobi-Davidson type subspace methods for (1). We highlight the connection between eigenvalue computation and optimization; the crucial link is the Newton method for Rayleigh quotient combined with subspace techniques. This link is well known and is central to the success of several eigensolvers, for example, the Davidson method and Jacobi-Davidson method. But often the connection to the Newton method is emphasized, which is used to account for the (locally) fast convergence property of the eigensolvers. The importance of subspace methods is no doubt well known. But we feel the following discussion may still contribute to the understanding of the Newton method applied within a subspace approach. We try to emphasize the subspace approach.

We begin the discussion from optimization. The traditional optimization algorithms are often one dimensional; that is, at each iteration step, the optimization is done within a one dimensional space. As an example we look at the Newton method for

$$\min_{\mathbf{x}} f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n, \tag{2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be twice continuously differentiable. At step k we obtain an approximate minimizer $\mathbf{x}_k$. Then we solve the Newton correction equation:

$$\left[\nabla^2 f(\mathbf{x}_k)\right] \ \delta\mathbf{x}_k = -\nabla f(\mathbf{x}_k). \tag{3}$$

The approximate minimizer at step $k+1$ is updated from $\mathbf{x}_k$ and $\delta\mathbf{x}_k$ as

$$\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \ \delta\mathbf{x}_k, \tag{4}$$

where α_k is some line search parameter obtained by an approximate minimization; $\alpha_k = 1$ is the standard Newton method. Often the Armijo-Goldstein condition or Wolfe condition [10] is used to ensure global convergence (without line search and other globalization techniques Newton method may converge to local maximal or saddle point). In other words, the minimizer is sought within a one dimensional space $span\{\delta\mathbf{x}_k\}$ with affine translation $\mathbf{x}_k$.

The key advantage of a subspace approach is that the convergence may be more robust and the convergence rate may be faster than that of a single vector update approach. This may be seen clearly from the following: Instead of looking for a minimizer within a one dimensional space $\mathcal{S}_1$, we look for a minimizer within a multidimensional subspace $\mathcal{S}_k$ that contains $\mathcal{S}_1$ as its subspace; hence the minimum in $\mathcal{S}_k$ must be no greater than the minimum in $\mathcal{S}_1$. Again we use the Newton method for (2) as an example. At step $k+1$, instead of looking for a single vector update (4), we augment the subspace by the computed Newton direction (i.e., $span\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k, \delta\mathbf{x}_k\}$). Then we look for a minimizer $\tilde{\mathbf{x}}_{k+1}$ within this augmented subspace. Then it is very clear to see that $f(\tilde{\mathbf{x}}_{k+1}) \leq f(\mathbf{x}_{k+1})$. By augmenting the subspace in this way, we retain the locally fast convergence property of the Newton method. A second advantage of the subspace approach is for clustered eigenvalues. Since even for the symmetric case the *gap theorem* [27] tells us that a single vector approach (without deflation) will not lead to a meaningful eigenvector, we have to compute the invariant subspace related to the clustered eigenvalues. Another advantage of the subspace approach is that one can seek more than one optimizer by working with different subspaces. This feature is particularly important for eigenvalue computation because one often needs a portion of the spectrum and the related eigenvectors instead of a single eigenpair.

Clearly the advantage of subspace technique is not gained without price. At each iteration of a subspace method one has to do more matrix vector products. And since orthonormal basis is numerically more stable than non-orthonormal ones, one needs to orthonormalize the new direction against the former subspace basis. These imply that the subspace dimension can not be large for large scale problems. To gain efficiency, one often needs to do restart and deflation [13, 30, 35, 40]. *Restart* is an indispensable technique which ensures that the starting vector becomes progressively better, so that the convergence becomes faster and the subspace dimension is kept relatively small to n. *Deflation* is also important with subspace methods. It tries to fix converged eigenvectors so that they will not be recomputed; and by using only vectors orthogonal to the converged eigenvectors to augment the projection basis, the projection subspace will be augmented in a more efficient way.

The important link between the eigenvalue problem (1) and optimization is the following nonlinear function called the *Rayleigh quotient*:

$$Q(\mathbf{x}) = \frac{\mathbf{x}^*\mathbf{A}\mathbf{x}}{\mathbf{x}^*\mathbf{x}}, \qquad \forall \mathbf{x} \neq \mathbf{0}. \tag{5}$$

For symmetric or Hermitian $\mathbf{A}$, $\min_{\mathbf{x}} Q(\mathbf{x})$ or $\max_{\mathbf{x}} Q(\mathbf{x})$ gives the extreme eigenvalue of $\mathbf{A}$. For general matrix $\mathbf{A}$, the Rayleigh quotient defines the *field of value* [15, 17]:

$$\mathcal{F}(\mathbf{A}) := \{Q(\mathbf{x}) \mid \mathbf{x} \in \mathbb{C}^n \backslash \{\mathbf{0}\}\}.$$

If $\mathbf{A}$ is normal, then $\mathcal{F}(\mathbf{A})$ is the convex hull of the eigenvalues of $\mathbf{A}$. In general $\mathcal{F}(\mathbf{A})$ contains the spectrum of $\mathbf{A}$. With suitable subspaces one can locate the eigenvalues of $\mathbf{A}$ via Rayleigh quotient; this is the well known *Rayleigh-Ritz* procedure. Certainly when the eigenvalues are located in the complex plane, the optimization interpretation becomes obscure, since one can not compare complex values. However, if we interpret optimizers not only as the end points of an interval in $\mathbb{R}$ but also as the boundary of a region in $\mathbb{C}$, then the optimization interpretation still makes sense. As an example of this interpretation we look at the Lanczos- or Arnoldi-type algorithms. It is well known that these algorithms usually approximate the exterior eigenvalues. Since the subspace at each iteration essentially contains the residual $\mathbf{r}_k = \mathbf{A}\mathbf{x}_k - \lambda_k\mathbf{x}_k$, for symmetric problems $\mathbf{r}_k$ is in the direction of the gradient of (5) at $\mathbf{x}_k$ (i.e., the steepest descent or ascent direction). For nonsymmetric problems $\mathbf{r}_k$ may not be readily considered as gradient since $Q(\mathbf{x})$ may not be differentiable [26], but it is likely that the subspace augmented by $\mathbf{x}_k$ contains the steepest descent or ascent direction (measured by vector norm) of $Q(\mathbf{x})$. This can also be considered as the advantage of the subspace approach: one does not have to compute the exact direction directly, but the span of the vectors contains the best direction. In this vein of thought, there would be no surprise that the algorithms converge to the exterior eigenvalues—the optimizers in the complex plane. As for interior eigenvalues, the common approach is to apply shift and invert, that is, to map the interior eigenvalues of the original matrix into extreme eigenvalues of a transformed matrix, and then apply standard solvers to the transformed matrix.

Some notations: Throughout this paper we use boldface letters for matrices and vectors. The upper script $()^*$ denotes the transpose $()^T$ in the real case, and the conjugate transpose $()^H$ in the complex case.

2 Subspace Eigensolvers: Davidson and Jacobi-Davidson

The Davidson method and Jacobi-Davidson method are two of the typical examples of subspace methods for eigenvalue computation. Before getting into details we present the framework of subspace methods for the eigenvalue problem (1) in Algorithm 2.1, where for simplicity we assume that $\mathbf{A} = \mathbf{A}^*$ and that only the smallest eigenvalue and its corresponding eigenvector are sought. Note that the optimization within a subspace is done at step *5(e)*.

Algorithm 2.1 *Framework of subspace methods for the eigenvalue problem (1):*

```
1. Start with an initial unit vector $\mathbf{x}$, $\mathbf{V}_1 \leftarrow [\mathbf{x}]$.
2. Compute $\mathbf{w} = \mathbf{A}\mathbf{x}$, $\mathbf{W}_1 \leftarrow [\mathbf{w}]$.
3. Compute $\lambda = \mathbf{H}_1 = \mathbf{x}^*\mathbf{w}$, $\mathbf{r} = \mathbf{w} - \lambda\mathbf{x}$.
4. If $\|\mathbf{r}\| <= \epsilon$, return $(\lambda, \mathbf{x})$ as the eigenpair.
5. Outer Loop: for $j = 1, 2, ..., m$ do
    (a) $\boxed{\text{Call specific subspace method to construct an augmentation vector } \mathbf{t}.}$
    (b) Orthonormalize $\mathbf{t}$ against $\mathbf{V}_j$ to get a unit vector $\mathbf{v}$.
    (c) Compute $\mathbf{w} = \mathbf{A}\mathbf{v}$, $\mathbf{W}_{j+1} \leftarrow [\mathbf{W}_j \mid \mathbf{w}]$, $\mathbf{V}_{j+1} \leftarrow [\mathbf{V}_j \mid \mathbf{v}]$.
    (d) Compute $\begin{bmatrix} \mathbf{h} \\ \alpha \end{bmatrix} = \mathbf{V}_{j+1}^*\mathbf{w}$, $\mathbf{H}_{j+1} = \begin{bmatrix} \mathbf{H}_j & \mathbf{h} \\ \mathbf{h}^* & \alpha \end{bmatrix}$.
    (e) Compute the smallest eigenpair $(\lambda, \mathbf{y})$ of $\mathbf{H}_{j+1}$. $(\|\mathbf{y}\| = 1)$.
    (f) Compute the Ritz vector $\mathbf{x} = \mathbf{V}_{j+1}\mathbf{y}$
        and the residual vector $\mathbf{r} = \mathbf{A}\mathbf{x} - \lambda\mathbf{x} = \mathbf{W}_{j+1}\mathbf{y} - \lambda\mathbf{x}$.
    (g) Test for convergence: if $\|\mathbf{r}\| <= \epsilon$, return $(\lambda, \mathbf{x})$ as the eigenpair.
6. Restart: Set $\mathbf{V}_1 = [\mathbf{x}]$, $\mathbf{W}_1 = [\mathbf{W}_{j+1}\mathbf{y}]$, $\mathbf{H}_1 = [\lambda]$, goto step 5.
```

Note also that the stopping criterion $\|\mathbf{r}\| <= \epsilon$ is in the relative sense; i.e., ϵ is related to the computed Ritz values. This is more important to the nonnormal problems. For $\mathbf{A} \neq \mathbf{A}^*$, one can easily adapt the framework by modifying step *5(d)* as

$$\mathbf{H}_{j+1} = \begin{bmatrix} \mathbf{H}_j & \mathbf{V}_j^*\mathbf{w} \\ \mathbf{v}^*\mathbf{W}_j & \mathbf{v}^*\mathbf{w} \end{bmatrix},$$

note only the $\mathbf{h}^*$ term at step *5(d)* needs to be changed into $\mathbf{v}^*\mathbf{W}_j$. If more eigenpairs are required, one can compute eigenpairs of $\mathbf{H}_{j+1}$ according to different selection criteria at step *5(e)*. The selection criteria include the first l $(l \geq 1)$ eigenvalues of the largest or smallest real parts, imaginary parts or magnitudes. Then at step *5(a)* one computes the augmentation vectors from the residual vectors related to the selected eigenpairs of $\mathbf{H}_{j+1}$.

The other important numerical step is the *orthogonalization* at step *5(b)*. A common choice is the modified Gram-Schmidt (MGS) method. MGS is in practice much more stable than Gram-Schmidt (GS), but the orthogonality of the basis constructed by MGS depends on the condition number of the original set of vectors. Another problem is that MGS can not be expressed by Level-2 BLAS, hence parallel implementation needs more communication [2, 11]. Another choice for orthogonalization is the DGKS correction [6]. DGKS can be implemented more efficiently in parallel because DGKS is actually GS with selective reorthogonalization. To make $\mathbf{t}$ orthogonal to $\mathbf{V}_j$ we do the following,

$$\mathbf{t}_{tmp} \leftarrow \mathbf{V}_j^* \mathbf{t}, \quad \mathbf{v}_{j+1} \leftarrow \mathbf{t} - \mathbf{V}_j \mathbf{t}_{tmp}, \quad \mathbf{v}_{j+1} \leftarrow \frac{\mathbf{v}_{j+1}}{\|\mathbf{v}_{j+1}\|}, \tag{6}$$

$$\text{if } (\|\mathbf{t}_{tmp}\| > tol * \|\mathbf{v}_{j+1}\|) \text{ then do the reorthogonalization :}$$

$$\mathbf{t}_{tmp} \leftarrow \mathbf{V}_j^* \mathbf{v}_{j+1}, \quad \mathbf{v}_{j+1} \leftarrow \mathbf{v}_{j+1} - \mathbf{V}_j \mathbf{t}_{tmp}, \quad \mathbf{v}_{j+1} \leftarrow \frac{\mathbf{v}_{j+1}}{\|\mathbf{v}_{j+1}\|}, \tag{7}$$

where the criteria means that if the angle between $\mathbf{t}_{tmp}$ and $\mathbf{v}_{j+1}$ at line (6) is smaller than $actan(tol)$, then reorthogonalization is necessary.

The subspace dimension and the convergence rate are a big concern for large scale problems. For subspace eigensolvers the convergence rate often depends heavily on the quality of the starting vector. Restart can be used to refine the starting vector. We know that before restart, useful information may have been accumulated in the recent expansion vectors. Thus it may not be desirable to keep only the most recent expansion vector and throw away all the rest. In [39, 44] *thick restart* techniques are proposed. The idea is to keep as much useful information as possible by keeping more than one expansion vector at the first step of restart. This idea has proved to be efficient and easy to incorporate into the framework.

What distinguishes each subspace method is the vector augmentation at step *5(a)*. When iterative methods (e.g., GMRES [31], BICGSTAB [46]) are used to solve for the augmentation vector, step *5(a)* necessarily contains the *inner iteration* of the subspace methods.

The **Davidson** method [4, 7, 22, 30] constructs $\mathbf{t}$ as follows:

5(a) Construct preconditioner $\mathbf{M}_j$; solve for $\mathbf{t}$ from:

$$(\mathbf{M}_j - \lambda \mathbf{I})\, \mathbf{t} = -\mathbf{r}.$$

In the original paper by Davidson [7], $\mathbf{M}_j = diag(\mathbf{A})$. Originally Davidson derived this formulation by taking the derivative of $Q(\mathbf{x}_j)$, varying only the i-th component of $\mathbf{x}_j$ (denoted as $\mathbf{x}_{j(i)}$) each time:

$$\left.\frac{\partial Q(\mathbf{x}_j)}{\partial \mathbf{x}_{j(i)}}\right|_{\mathbf{x}_{j(i)}+\delta \mathbf{x}_{j(i)}} = 0, \qquad i = 1, 2, ..., n. \tag{8}$$

From (8) we get $\delta \mathbf{x}_{j(i)} = (\lambda_j - a_{ii})^{-1} (\mathbf{A}\mathbf{x}_j - \lambda_j \mathbf{x}_j)_{(i)}$. In [8] Davidson related this formulation to the Newton method. It was observed that the convergence rate is related to how well the diagonal matrix $diag(\mathbf{A})$ approximates $\mathbf{A}$. For example, if $\mathbf{A}$ is diagonally dominant (in eigenvalue problems, diagonal dominance means that the off-diagonal elements are small compared with the changes in magnitude between diagonal elements [42, 22, 38]), then the Davidson method converges very fast. Hence the diagonal matrix ($diag(\mathbf{A}) - \lambda_j \mathbf{I}$) was considered as a straightforward preconditioner to $\mathbf{A} - \lambda_j \mathbf{I}$. The main advantage of a diagonal preconditioner is that the preconditioned system (which is diagonal) is trivial to solve, but it requires $\mathbf{A}$ to be diagonally dominant. More sophisticated preconditioners than the diagonal matrix have been tried [4, 22], leading to the so called *generalized Davidson method.* As has been pointed out (e.g., in [32]), the preconditioner interpretation does not explain the improved convergence rate well since the exact preconditioner $(\mathbf{A} - \lambda_j \mathbf{I})^{-1}$ leads to stagnation ($(\mathbf{A} - \lambda_j \mathbf{I})^{-1}\mathbf{r}$ is $\mathbf{x}_j$, which lies in the original projection subspace).

The **Jacobi-Davidson** method [32] is an important advance for subspace methods in eigenvalue computation. At each iteration the correction vector $\mathbf{t}$ is required to reside within the orthogonal complement of the existing projection subspace. Jacobi-Davidson solves the following projected equation:

5(a) Solve (approximately) for $\mathbf{t}$ s.t. $\mathbf{t} \perp \mathbf{x}$ and

$$(\mathbf{I} - \mathbf{x}\mathbf{x}^*)(\mathbf{A} - \lambda \mathbf{I})(\mathbf{I} - \mathbf{x}\mathbf{x}^*)\mathbf{t} = -\mathbf{r}. \tag{9}$$

Usually (9) is solved inexactly by iterative methods. The matrix $(\mathbf{I} - \mathbf{x}\mathbf{x}^*)(\mathbf{A} - \lambda \mathbf{I})(\mathbf{I} - \mathbf{x}\mathbf{x}^*)$ is always singular, but this singularity poses no essential difficulty to iterative methods for (9).

Existing preconditioners for linear systems may be applied when solving (9) iteratively. This is regarded as one of the major advantages of (Jacobi-)Davidson-type subspace methods, since preconditioners may not be easily incorporated into the Arnoldi-type methods. Rational Krylov-type or (single vector) RQI-type methods often require rather exact system solves, while (Jacobi-)Davidson-type methods allow approximate system solves.

An equivalent formulation to (9) is the *inverse iteration generalized Davidson* (IIGD) method by Olsen *et al* [24]. IIGD solves

$$(\mathbf{A} - \lambda \mathbf{I})\mathbf{t} = -\mathbf{r} + \mathbf{x}\tau, \tag{10}$$

where τ is set to ensure $\mathbf{t} \perp \mathbf{x}$, i.e., $\tau = \frac{\mathbf{x}^*(\mathbf{A}-\lambda\mathbf{I})^{-1}\mathbf{r}}{\mathbf{x}^*(\mathbf{A}-\lambda\mathbf{I})^{-1}\mathbf{x}}$. From (10) it is clear that the expansion vector $\mathbf{t}$ contains information in both directions $(\mathbf{A} - \lambda \mathbf{I})^{-1}\mathbf{r}$ and $(\mathbf{A} - \lambda \mathbf{I})^{-1}\mathbf{x}$. In the original paper [24] the authors dealt with a huge problem at that time and they could not afford the subspace approach; hence they ended up using single vector updating approach in their program (this is also pointed out in [41]).

In [34] and [24] it was argued that Jacobi-Davidson and IIGD are (inexact) Newton-Raphson methods. The generalized Rayleigh quotient $Q(\mathbf{x}, \mathbf{y}) := \frac{\mathbf{x}^*\mathbf{A}\mathbf{y}}{\mathbf{x}^*\mathbf{y}}$ was used in [34] to establish the argument. In the following section we derived a Newton update straightforwardly from the standard Rayleigh quotient (5).

3 A New Class of Newton Update

The locally fast convergence property of the Newton method in optimization is an attractive feature for designing fast algorithms. Davidson, Jacobi-Davidson and IIGD methods all tried to establish a connection to Newton method to explain their fast convergent behavior. Yet no Newton method combined with subspace projection applied directly to the Rayleigh quotient (5) seems to have been tried. In this section we establish this interesting application.

We assume that $\mathbf{A}^* = \mathbf{A}$. (This is not for notational simplicity reason, since the nonnormal case may be completely different.) The gradient of the Rayleigh quotient (5) is

$$\begin{aligned} \nabla Q(\mathbf{x}) &= \frac{2\mathbf{Ax}}{\mathbf{x}^*\mathbf{x}} - \frac{2\mathbf{x}^*\mathbf{Axx}}{(\mathbf{x}^*\mathbf{x})^2} \\ &= \frac{2}{\mathbf{x}^*\mathbf{x}}(\mathbf{Ax} - Q(\mathbf{x})\mathbf{x}). \qquad (11) \end{aligned}$$

The Hessian of (5) is

$$\begin{aligned} \nabla^2 Q(\mathbf{x}) &= \frac{2\mathbf{Ax}}{\mathbf{x}^*\mathbf{x}} - \frac{4\mathbf{Axx}^*}{(\mathbf{x}^*\mathbf{x})^2} - \frac{2}{\mathbf{xx}^*}\left(\mathbf{x}(\nabla Q(\mathbf{x})^*) + Q(\mathbf{x})\mathbf{I}\right) + \frac{4}{(\mathbf{x}^*\mathbf{x})^2}\left(Q(\mathbf{x})\mathbf{xx}^*\right) \\ &= \frac{2}{\mathbf{x}^*\mathbf{x}}(\mathbf{A} - Q(\mathbf{x})\mathbf{I}) - \frac{2}{\mathbf{x}^*\mathbf{x}}(\mathbf{x}(\nabla Q(\mathbf{x})^*) + (\nabla Q(\mathbf{x}))\mathbf{x}^*) \\ &= \frac{2}{\mathbf{x}^*\mathbf{x}}(\mathbf{A} - Q(\mathbf{x})\mathbf{I}) - \frac{4}{(\mathbf{x}^*\mathbf{x})^2}(\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A}^* - 2Q(\mathbf{x})\mathbf{xx}^*). \qquad (12) \end{aligned}$$

This seemingly complicated formula can be simplified. If we assume $\mathbf{x}$ is already normalized and denote $\lambda = Q(x)$, then

$$\nabla^2 Q(\mathbf{x}) = 2(\mathbf{A} - \lambda\mathbf{I}) - 4(\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A}^* - 2\lambda\mathbf{xx}^*). \qquad (13)$$

So the Newton equation $\nabla^2 Q(\mathbf{x})\mathbf{t} = -\nabla Q(\mathbf{x})$ is

$$[(\mathbf{A} - \lambda\mathbf{I}) - 2(\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A}^* - 2\lambda\mathbf{xx}^*)]\mathbf{t} = -(\mathbf{Ax} - \lambda\mathbf{x}) = -\mathbf{r}. \qquad (14)$$

To see the relation with the Jacobi-Davidson equation (9), we expand the matrix in the left hand side of (9) and get: (note $\mathbf{x}^*\mathbf{x} = 1$)

$$\begin{aligned} & [(\mathbf{I} - \mathbf{xx}^*)(\mathbf{A} - \lambda\mathbf{I})(\mathbf{I} - \mathbf{xx}^*)]\mathbf{t} \\ = \; & [(\mathbf{A} - \lambda\mathbf{I}) - (\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A} - 2\lambda\mathbf{xx}^*)]\mathbf{t} = -\mathbf{r}. \qquad (15) \end{aligned}$$

From (14) and (15) we see the essential difference is that the scaling of the correction term $(\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A}^* - 2\lambda\mathbf{xx}^*)$ to $(\mathbf{A} - \lambda\mathbf{I})$ in (14) is twice as much as the one in (15). The formula we derived here is Newton method, hence it preserves the locally quadratic convergence rate. No additional treatment is needed to establish the connection to the Newton method.

Our new algorithm using Newton direction within the subspace method framework (Algorithm 2.1) may be stated as follows:

5(a) Solve (approximately) the following equation for $\mathbf{t} \perp \mathbf{x}$:

$$(\ \mathbf{A} - \lambda\mathbf{I} - 2(\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A}^* - 2\lambda\mathbf{xx}^*)\ )\mathbf{t} = -\mathbf{r} \qquad (16)$$

In nonlinear optimization, the Hessian matrix is often difficult to obtain, hence low rank updates such as DFP and BFGS are used to obtain approximate Hessian matrix, resulting in the Quasi-Newton methods with a super linear convergence rate. For eigenvalue computation, the situation is better because we can compute the explicit Hessian of the Rayleigh quotient. Actually in [43] the correct Hessian was derived, but the formula in [43] was not simplified and it was more complicated than (12); hence the authors did not try applying Newton method to the Rayleigh Quotient, while they proposed other very neat Newton-type methods based on transformed formulations of the eigenvalue problem.

The other difference between (14) and the Jacobi-Davidson equation (15) is that the $\mathbf{xx}^*\mathbf{A}^*$ term in (16) is $\mathbf{xx}^*\mathbf{A}$ in (15). This certainly makes no difference under the assumption $\mathbf{A} = \mathbf{A}^*$. Actually under this assumption, equation (16) is equivalent to:

$$(\mathbf{I} - 2\mathbf{xx}^*)(\mathbf{A} - \lambda\mathbf{I})(\mathbf{I} - 2\mathbf{xx}^*)\mathbf{t} = -\mathbf{r}. \tag{17}$$

This looks very similar to the Jacobi-Davidson equation (9). We note that (15) was derived from computational insight (where the idea of the Jacobi orthogonal component correction is exploited as the orthogonal projector). We will discuss later that this is the truly valuable insight that makes the Jacobi-Davidson to be so efficient. While (17) is derived rigorously by the Newton method. The major difference is that $(\mathbf{I} - 2\mathbf{xx}^*)$ is perfectly conditioned since $(\mathbf{I} - 2\mathbf{xx}^*)^2 = \mathbf{I}$ ($\mathbf{I} - 2\mathbf{xx}^*$ is also called the *Householder reflector*), while the orthogonal projector $(\mathbf{I} - \mathbf{xx}^*)$ is singular. Hence, away from an eigenvalue, (17) is better conditioned than (15). When near an eigenvalue λ, both (15) and (17) relate to the inverse iteration on $(\mathbf{A} - \lambda\mathbf{I})$. As discussed in [27, 28], the ill-conditioning of $(\mathbf{A} - \lambda\mathbf{I})$ does not prevent computing the direction of the eigenvector for λ correctly.

For large scale problems, computing the Newton direction exactly is usually not practical. Equation (17) need to be replaced by some preconditioned equation. We can incorporate preconditioners in the same way as the generalized Davidson method or Jacobi-Davidson method does. I.e., if we construct a good preconditioner $\mathbf{M}_j$ at step j, then we solve the following:

5(a) Applying preconditioner $\mathbf{M}_j$, solve the following equation for $\mathbf{t} \perp \mathbf{x}$:

$$(\mathbf{I} - 2\mathbf{xx}^*)(\mathbf{M}_j - \lambda\mathbf{I})(\mathbf{I} - 2\mathbf{xx}^*)\mathbf{t} = -\mathbf{r}. \tag{18}$$

If the formulation in (16) is preferred, then we solve for $\mathbf{t} \perp \mathbf{x}$ from

$$(\ \mathbf{M}_j - \lambda\mathbf{I} - 2(\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A}^* - 2\lambda\mathbf{xx}^*)\ )\mathbf{t} = -\mathbf{r}. \tag{19}$$

In both formulations (18) and (19), we may apply the easily available diagonal preconditioner $\mathbf{M}_j = diag(\mathbf{A})$. We note that they now can not be solved as trivially as diagonal systems. We also note that (19) contains more information about the original $\mathbf{A}$ than does (18).

The overhead caused by the correction terms in (18) or (19) is marginal for iterative methods because one mainly computes the approximate solution by matrix-vector products. But we point out that the condition number of the matrix (which need not be formed explicitly) affects the convergence rate of any iterative method. For eigensolvers that utilize preconditioners from linear systems directly, another complication may occur, as pointed out in [36], that a good preconditioner $\mathbf{M}_j$ for $\mathbf{A}$ (such as preconditioners from multigrid method) need not necessary be a good preconditioner for $\mathbf{A} - \lambda\mathbf{I}$ or other modifications to $\mathbf{A}$. We also note that the interpretation of conditioning is different between linear systems and eigenvalue problems: for linear system, the conditioning depends solely on $\sigma_{max}/\sigma_{min}$ where σ denotes the singular value; while for eigenvalue problems, each eigenvalue has its own conditioner number and the separation of eigenvalues is closely related to the convergence rate of eigensolvers.

Note that $\mathbf{r} = \mathbf{Ax} - \lambda\mathbf{x}$ is computed at each step of the Davidson-type methods. If $\lambda \in \mathbb{R}$, the three correction terms in (16) and (19) may be simplified into two terms:

$$\mathbf{Axx}^* + \mathbf{xx}^*\mathbf{A}^* - 2\lambda\mathbf{xx}^* = \mathbf{rx}^* + \mathbf{xr}^*. \tag{20}$$

This simplification makes updating the matrix vector products at each iteration easier, and we use it in our code for the numerical tests.

4 Unification of Jacobi-Davidson, IIGD and Newton Updates via RQI

In large scale eigenvalue computation, solving preconditioned system (inexactly) via iterative methods is a crucial part for most of the fast eigensolvers. We see from Section 3 that when suitable correction equations are found, preconditioners may be readily adapted into the existing subspace methods by modifying the correction equations. Hence in the rest of this paper we concentrate on the derivation of the correction equations, without being distracted by preconditioners.

When we talk about fast convergent algorithms for eigenvalue problems, an important candidate that comes to mind is the Rayleigh quotient iteration (RQI). RQI is cubic convergent for normal matrices and quadratic convergent for nonnormal matrices [26]. Extensive studies of RQ and RQI exist. Properties of RQI and different type of generalizations may be found, for example, in [9, 23, 25, 26, 27, 40] and references therein.

In the following we unify the Jacobi-Davidson, IIGD and the proposed methods in Section 3 by RQI. This unification leads us to propose a simplified version of the Jacobi-Davidson method and a simplified IIGD method. We first establish the following theorem.

Theorem 4.1 *Let the initial unit vector be the same* $\mathbf{x}$*, assuming the Rayleigh quotient* $\lambda = \mathbf{x}^*\mathbf{Ax}$ *is not an eigenvalue of* $\mathbf{A}$ *yet. Then the subspace method with equation (9) or (10) or (17) each produces a subspace that includes the RQI direction during the next iteration.*

Proof: The subspace method with updating equation (9) or (10) or (17) is a special case of the following (with suitable α and β):

5(a) Solve (approximately) the following equation for $\mathbf{t} \perp \mathbf{x}$:

$$(\mathbf{I} - \alpha\,\mathbf{xx}^*)(\mathbf{A} - \lambda\mathbf{I})(\mathbf{I} - \beta\,\mathbf{xx}^*)\mathbf{t} = -\mathbf{r}, \qquad \forall\ \alpha \neq 0. \tag{21}$$

We now prove the following stronger statement: Solving (21) for $\mathbf{t} \perp \mathbf{x}$, then the subspace augmented by $\mathbf{t}$ will include the RQI direction during the next iteration.

It is well known that during the next iteration, RQI produces the direction

$$\mathbf{x}_R = (\mathbf{A} - \lambda\mathbf{I})^{-1}\mathbf{x}.$$

Under the restriction $\mathbf{x}^*\mathbf{t} = 0$, equation (21) becomes

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{t} = -\mathbf{r} + \eta\, \mathbf{x}. \tag{22}$$

This is actually the IIGD equation. From (22)

$$\mathbf{t} = (\mathbf{A} - \lambda\mathbf{I})^{-1}\mathbf{r} + \eta\, (\mathbf{A} - \lambda\mathbf{I})^{-1}\mathbf{x} = -\mathbf{x} + \eta\, \mathbf{x}_R, \tag{23}$$

where $\eta = 1/(\mathbf{x}^*\mathbf{x}_R)$ is chosen to ensure $\mathbf{x}^*\mathbf{t} = 0$. Note that $\mathbf{x}$ is in the original projection subspace, say, $\mathbf{V}_j$. We see that the RQI direction $\mathbf{x}_R$ will be included in the augmented projection subspace $\mathbf{V}_{j+1}$ when we carry out *5(b)* in Algorithm 2.1. ■

This theorem shows that the subspace method with (21) is able to retain the desirable fast convergence property of RQI. As discussed in the introduction section, the subspace approach possibly converges faster than RQI, and it also possesses the other advantages of a subspace method over a single-vector method.

Formula (21) may be linked to the Newton method in other ways. From (21) and $\mathbf{x}^*\mathbf{t} = 0$ we see

$$(\mathbf{I} - \alpha\, \mathbf{x}\mathbf{x}^*)(\mathbf{A} - \lambda\mathbf{I})\mathbf{t} = -\mathbf{r}.$$

This leads to:

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{t} - \eta\, \mathbf{x} = -\mathbf{r}, \qquad \mathbf{x}^*\mathbf{t} = 0. \tag{24}$$

Equation (24) is equivalent to the following bordered equation:

$$\begin{bmatrix} \mathbf{A} - \lambda\mathbf{I} & -\mathbf{x} \\ -\mathbf{x}^* & 0 \end{bmatrix} \begin{bmatrix} \mathbf{t} \\ \eta \end{bmatrix} = \begin{bmatrix} -\mathbf{r} \\ 0 \end{bmatrix}. \tag{25}$$

This bordered equation also appears in [28, 43] where Newton method is applied to the following nonlinear equation:

$$\begin{cases} (\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = \mathbf{0}, \\ -\frac{1}{2}\mathbf{x}^*\mathbf{x} + \frac{1}{2} = 0. \end{cases} \tag{26}$$

The bordered matrix turns out to be the Jacobian matrix of (26). The Newton correction equation to (26) is exactly (25). From this point of view, each of the algorithms — Jacobi-Davidson, IIGD and the generalized form (21) — can be viewed as a Newton method for (26) carried out with subspace acceleration. A hidden reason for the success of the Jacobi-Davidson and IIGD methods may be that the bordered matrix $\begin{bmatrix} \mathbf{A} - \lambda\mathbf{I} & -\mathbf{x} \\ -\mathbf{x}^* & 0 \end{bmatrix}$ is generally nonsingular (even when λ is a simple eigenvalue of $\mathbf{A}$). The nonsingularity was first proved in [28]. In actual computations one does not need to form the bordered matrix. Discussion of the preconditioned form of (25) may be found in [36].

We prefer unifying the methods by RQI instead of by the Newton method, since the convergence rate for RQI is locally cubic for normal matrices and locally quadratic for nonnormal matrices. Whereas the Newton method usually explains only the locally quadratic convergence rate for both cases.

As seen from the proof, the fast convergence rate of these methods depends mainly on including the RQI direction to augment the projection subspace. This fact is well known for the Jacobi-Davidson method. But to cast it in a more general setting as we did here is new. Also, the proof leads to other interesting observations, which we discuss in the next section.

5 Simplification of Jacobi-Davidson and IIGD

A more critical look at the proof of Theorem 4.1 reveals that the constants before the correction terms in (9) and (17) (hence also (16)) surprisingly seem not essential in establishing the link to RQI. As seen from (21), any nonzero α seems to work. The arbitrary choice of the parameters (α, β) in (21) may be convenient for algorithm design, but certainly it is not mathematically pleasant.

The difference may be that from (21) to (22) we require both $\mathbf{x}^*\mathbf{t} = 0$ and implicitly $\eta = \alpha\, \mathbf{x}^*(\mathbf{A} - \lambda\mathbf{I})\mathbf{t} = 1/(\mathbf{x}^*\mathbf{x}_R)$. This implies that different choices of α (and possibly β) may not be equivalent. But this explanation does not go further.

We have determined, through extensive numerical experiments, that $\alpha = 1$ in (21) is the most essential character that leads to the success of the Jacobi-Davidson method. In the Jacobi-Davidson method [32], $\alpha = \beta = 1$ is inspired by the Jacobi orthogonal component correction. This construction is indeed of deep computational insight. While in (17), $\alpha = \beta = 2$ is derived from the Newton method for the Rayleigh quotient of $\mathbf{A}(= \mathbf{A}^*)$. Even though $(\mathbf{I} - 2\mathbf{x}\mathbf{x}^*)$ is a perfectly conditioned matrix, without the additional restriction $\mathbf{x}^*\mathbf{t} = 0$, (16) and (17) generally converge more slowly than do the Jacobi-Davidson and IIGD. At first sight this may seem questionable because from Theorem 4.1 all the methods should be equivalent. The explanation is that multiplying $(\mathbf{A} - \lambda\mathbf{I})\mathbf{t} = -\mathbf{r}$ by perfectly conditioned matrices may not move the solution $\mathbf{t}$ away from the direction of $\mathbf{x}$; hence the stagnation for the perfectly preconditioned Davidson method may also happen to (16) and (17), though not as seriously. Additionally requiring $\mathbf{t} \perp \mathbf{x}$, if done after solving the correction equations (16) or (17) by direct methods, may lead to a poor quality $\mathbf{t}$. In this case, often the DGKS method at step *5(b)* is not enough to ensure orthogonality. One would have to perform three steps of Gram-Schmidt orthogonalization, which often results in a vector that does not augment the projection subspace efficiently. If the $\mathbf{t} \perp \mathbf{x}$ is enforced at each step of any iterative methods for (16) or (17), then from the equality $(\mathbf{I} - \mathbf{x}\mathbf{x}^*)(\mathbf{I} - \gamma\mathbf{x}\mathbf{x}^*) = (\mathbf{I} - \gamma\mathbf{x}\mathbf{x}^*)(\mathbf{I} - \mathbf{x}\mathbf{x}^*) = (\mathbf{I} - \mathbf{x}\mathbf{x}^*)$, for $\forall\gamma$, we see that (16) and (17) are essentially Jacobi-Davidson.

As seen from Algorithm 2.1, after solving the correction equation for $\mathbf{t}$ in step *5(a)*, this $\mathbf{t}$ is orthonormalized against a subspace that contains $\mathbf{x}$. Hence, theoretically speaking, requiring $\mathbf{t} \perp \mathbf{x}$ at step *5(a)* is not necessary; but for numerical reasons, especially when iterative methods are used to solve the correction equation, it may be necessary to orthogonalize the solution $\mathbf{t}$ against $\mathbf{x}$ once in a while or at every step of the inner iteration (as suggested in [32]).

We found out that choosing $\alpha = 1$ in (21) is crucial in moving $\mathbf{t}$ sufficiently away from the direction of $\mathbf{x}$. While the value of β does not appear to be important, it may be set to any value not too large as to lead to overflow. For cost-effective reasons, we may set $\beta = 0$ and simplify the Jacobi-Davidson method as follows:

5(a) Solve (approximately) for $\mathbf{t}$ s.t. $\mathbf{t} \perp \mathbf{x}$ and:

$$(\mathbf{I} - \mathbf{x}\mathbf{x}^*)(\mathbf{A} - \lambda\mathbf{I})\,\mathbf{t} = -\mathbf{r}. \tag{27}$$

Note that (27) implies

$$(\mathbf{I} - \mathbf{x}\mathbf{x}^*)(\mathbf{A} - \lambda\mathbf{I})\,\mathbf{t} = -(\mathbf{I} - \mathbf{x}\mathbf{x}^*)\mathbf{r},$$

so we actually solve the equation from a right hand side vector orthogonal to $\mathbf{x}$.

For $\mathbf{A} \neq \mathbf{A}^*$, the unsymmetric matrix in (27) may not cause objection. For $\mathbf{A} = \mathbf{A}^*$, it may suggest that the symmetric matrix in (9) would be preferred. We show by extensive numerical experiments that for both the symmetric and unsymmetric cases, if direct methods are used to solve (27) and (9), then the cheaper (27) performs as well as (9). We point out that if iterative methods are used to solve (27) and the solution at each step is orthogonalized to $\mathbf{x}$, then (27) is the same as the original Jacobi-Davidson method. The excellent numerical behavior of direct methods for (27) suggests that in iterative methods, the orthogonalization of the solution to $\mathbf{x}$ may be carried out after several iteration steps instead of after each step.

One advantage of Jacobi-Davidson over IIGD is that IIGD requires two system-solves per inner iteration, which is more expensive than the one system-solve Jacobi-Davidson.

Also from (23) in the proof of Theorem 4.1, we see that in order to capture the fast convergence property of RQI and to move $\mathbf{t}$ away from the $\mathbf{x}$ direction, the η should not be too small. And (23) suggests the following simplified IIGD, which requires only one system-solve:

5(a) Solve (approximately) for $\mathbf{t}$ from:

$$(\mathbf{A} - \lambda\mathbf{I})\,\mathbf{t} = \mathbf{x}, \tag{28}$$

Compute the constant $\eta = 1/(\mathbf{x}^*\mathbf{t})$, then assign $\mathbf{t} \leftarrow (\eta\mathbf{t} - \mathbf{x})$.

This modification is simple. Other researchers may have noticed similar equivalent formulas several years ago, but to our knowledge, to explicitly propose solving the IIGD equation by this one-solve approach is new, and no numerical results seem have been published that compare the one-solve approach with the mainly used two-solve approach. Our modification actually saves almost half the computational cost at each inner iteration of IIGD. Even with one less system-solve, it produces a $\mathbf{t}$ that is readily orthogonal to $\mathbf{x}$ (even when $(\mathbf{A}-\lambda\mathbf{I})^{-1}\mathbf{x}$ makes a small angle with $\mathbf{x}$. This is an improvement on the observation 4.1.(c) in [32]). Both the above analysis and the extensive numerical experiments that we performed show that (28) works as well as the IIGD. Because of the reduced (often near-singular) system solves, (28) can be expected to perform better than (10). Most important, it is the least expensive method but equally efficient to Jacobi-Davidson and IIGD.

We note that existing preconditioned methods for linear systems may be easily incorporated into (27) and (28). We use (28) as an example: If at step j we have a good preconditioner $\mathbf{K}_j$ for $(\mathbf{A} - \lambda_j\mathbf{I})$, then we solve $\mathbf{K}_j\mathbf{t} = \mathbf{x}_j$ and compute $\eta = \frac{1}{x_j^* t}$. Then the updating vector can be obtained by $(\eta\mathbf{t} - \mathbf{x}_j)$, which is always orthogonal to $\mathbf{x}_j$ — the direction we are interested in to augment the projection basis. The significance of (28) may be seen from the following: If we stick to the preconditioned form of (10) (i.e., $\mathbf{K}_j\mathbf{t} = -\mathbf{r}_j + \eta\mathbf{x}_j$), then two preconditioned equations would have to be solved for the orthogonalization constant η ($\eta = \frac{\mathbf{x}_j^*\mathbf{K}_j^{-1}\mathbf{r}_j}{\mathbf{x}_j^*\mathbf{K}_j^{-1}\mathbf{x}_j}$) since $\mathbf{K}_j^{-1}\mathbf{r}_j$ may not be equal to $\mathbf{x}_j$. But our modified scheme (28) aptly reduced the two system-solves into one system-solve, even for the preconditioned formulation.

6 Numerical Results and Discussions

The purpose of this paper is not to present actual implementations of solvers for large scale eigenvalue problems. Our purpose is to study the efficiency of different correction equations. Different preconditioning techniques may be applied readily after the correction equations are derived. Currently one of the most efficient and well known eigensolvers is the Jacobi-Davidson method (9). IIGD (10) is equivalent to Jacobi-Davidson and is well known in the computational chemistry community. We propose the generalized form (21). In this section we present numerical behavior of the more specific forms (16), (17), (27) and (28), with comparisons among themselves and to the well known (9) and (10). In the result-plot Figures, N1 and N2 denote (16) and (17), respectively; JD and IIGD denote (9) and (10), respectively; and JDm and IIGDm denote the modified forms (27) and (28), respectively.

The numerical experiments were done by using Matlab v6.5 on Pentium IV PCs with OS Linux Redhat7.3. The models are restricted to relatively small scale problems ($200 \leq n \leq 1000$). We use direct methods (Matlab "\") to solve the correction equations, since for matrices of this scale "\" is often at least ten times faster and more accurate than iterative methods. We note that when $\mathbf{M}$ is singular, $\mathbf{M}\backslash\mathbf{r}$ give the solution $\mathbf{M}^+\mathbf{r}$ where $\mathbf{M}^+$ is the pseudo-inverse of $\mathbf{M}$.

As pointed out in Section 5, in Algorithm 2.1, the solution $\mathbf{t}$ from step *5(a)* is made orthogonal to $\mathbf{x}$ in step *5(b)*. Hence, for the Jacobi-Davidson type equations (9), (16), (17) and (27), we **did not** additionally require $\mathbf{t} \perp \mathbf{x}$ in step *5(a)*. This modification will tell which equation would *naturally* provide a solution that can augment the subspace efficiently.

We tried the methods on models from the Matrix Market.[1] Models from this website are often related to realistic and difficult problems. For all the tests we chose relative tolerance $tol = 10^{-10}$ as the stopping measure for the residual norm. We report results for the models: `bwm200, ck400, pde225, pde900, rdb200, rdb450, rdb968, rw496`. The name of each model is listed in the first term at the title of each subgraph. These models may be retrieved easily by searching for eigenvalue problems in Matlab format at the Matrix Market website. The physical meaning of these models may be found at the same website. The `mode` denotes the eigenvalue selection criteria, where LR, LM, SR and SM stand for largest real part, largest magnitude, smallest real part and smallest magnitude, respectively. The modes were chosen according to the description of each model. Since there are fewer symmetric standard eigenvalue problems in Matlab format at Matrix Market, we also applied the methods to matrices $(\mathbf{A} + \mathbf{A}^*)/2$ when $\mathbf{A} \neq \mathbf{A}^*$. We note that this simple symmetrization still led to problems that are more realistic (and usually more difficult) than randomly constructed models.

The methods discussed in this paper generally converge fast locally, but not globally. For

[1] `http://math.nist.gov/MatrixMarket/`

the problems from Matrix Market, without a good initial vector, each method may take a long time to arrive at the fast convergent region. Hence, we chose the initial vector as follows. We computed the eigenvector corresponding to the required eigenvalue by the Matlab function *eig* and perturbed it as the initial vector. For the nonsymmetric problems, we perturbed the eigenvector by $10^{-2} * rand(n, 1)$; for symmetric problems, the perturbation was set as $5 * 10^{-2} * rand(n, 1)$. By choosing the initial vector this way, the locally fast convergence behavior of each method becomes clear to observe and easy to compare. Certainly in real applications it is not easy to find a good initial guess without a good understanding of the models at hand, and one may have to resort to restart techniques. The approach we took in this section was mainly to verify the analysis we did in this paper, as well as to check the viability of the proposed formulations.

As seen from the numerical results, the Jacobi-Davidson, IIDG and their simplified versions (27) (28) do show locally quadratic convergence for nonsymmetric problems and cubic convergence for symmetric problems. And the convergence appeared faster than the theoretic rate in some cases, which is due to the subspace acceleration. For all the problems tried, JDm and IIGDm behaved almost the same as Jacobi-Davidson and IIGD; as seen from the plots, the four lines are often indistinguishable. This result also agrees with our analysis in Sections 4 and 5.

What needs to be explained more is the behavior of (16) and (17). Although for some constructed models (16) and (17) appeared to be as competitive as JD and IIGD, for most of the more realistic models they performed more slowly. The observed convergence behavior for (16) and (17) is locally quadratic for symmetric problems and linear for nonsymmetric problems. It may be super-quadratic or super-linear for each case because of the subspace acceleration. The reason is that the derivation of the gradient and Hessian in Section 3 holds true for $\mathbf{A} = \mathbf{A}^*$, whereas the gradient for the Rayleigh quotient of $\mathbf{A} \neq \mathbf{A}^*$ is different and may not exist. We point out that even though in the Matlab codes we did not require $\mathbf{t} \perp \mathbf{x}$ at step *5(a)* for (16) and (17), adding this requirement may not help much if the correction equations are solved by direct methods. The reason is discussed in Section 5. If this requirement is enforced at each step of an iterative method for (16) and (17), then (16) and (17) essentially become Jacobi-Davidson and IIGD, as seen from the proof of Theorem 4.1.

The Jacobi-Davidson equation (9) together with equations (16) and (17) led to the generalized formulation (21), which led to the unified theory and the two simplifications (27) and (28). The modifications are cheaper and simpler than the original formulations. Analysis and numerical evidences show that they are as efficient as the original formulations. We expect that preconditioned formulations based on (27) and (28) will be practical in actual applications.

The numerical results we present here evidently show that $\alpha = 1$ in (21) is fundamental for the efficiency of the methods we discussed in this paper. We hope that they also contribute to the understanding and appreciation of the Jacobi-Davidson method and the IIGD method.

One problem related to Jacobi-Davidson and IIGD type methods is the possible slow convergence when a starting vector is poor. Our unreported numerical tests show that with a random initial vector many iterations may be needed to reach the fast convergent region (some of the numerical results in [16] showed similar behavior). This suggests that efficient globalization techniques may need to be developed. A practical approach is to apply a few steps of cheaper Lanczos or Arnoldi iteration to get a better starting vector and then integrate restart during the Davidson type outer iteration. Another approach is to modify shifts in the correction equations.

This may require a better understanding on the difference between preconditioning for linear systems and preconditioning for eigenvalue problems. [38] contains very interesting results in this direction for the symmetric eigenvalue problem.

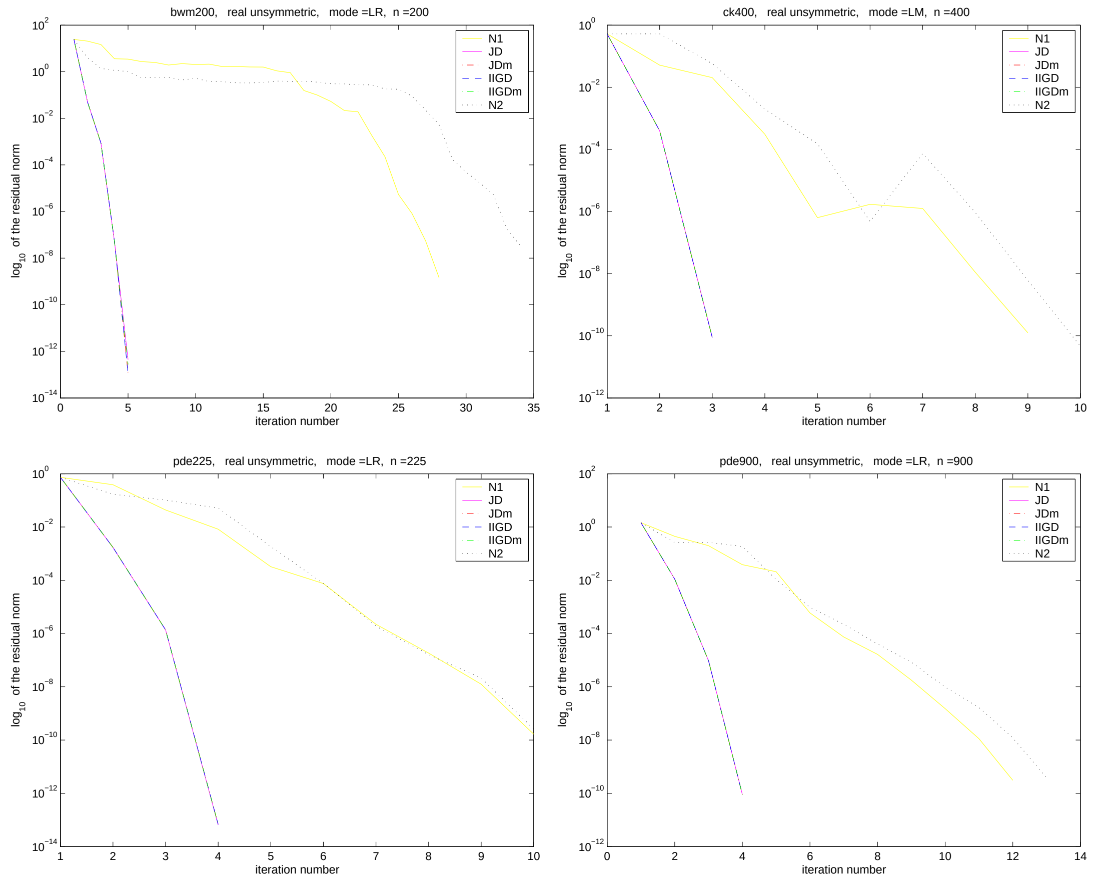

Figure 1: Comparison of performance for nonsymmetric models: bwm200, ck400, pde225, pde900.

7 Concluding remarks

In this paper we highlighted the value of subspace methods in eigenvalue computation. We also emphasized the link between eigenvalue computation and optimization through the Newton method. We derived a class of Newton updates for symmetric eigenvalue problems. From the study of the Newton updates and their similarity to the Jacobi-Davidson method, we discovered the generalized correction equation (21). From this generalized formula we arrived at simplified forms of the Jacobi-Davidson method and the IIGD method. The modification (28) actually halves the computational cost for the inner iteration of IIGD. We showed by extensive numerical results that the simplified versions perform as efficiently as the Jacobi-Davidson and IIGD meth-

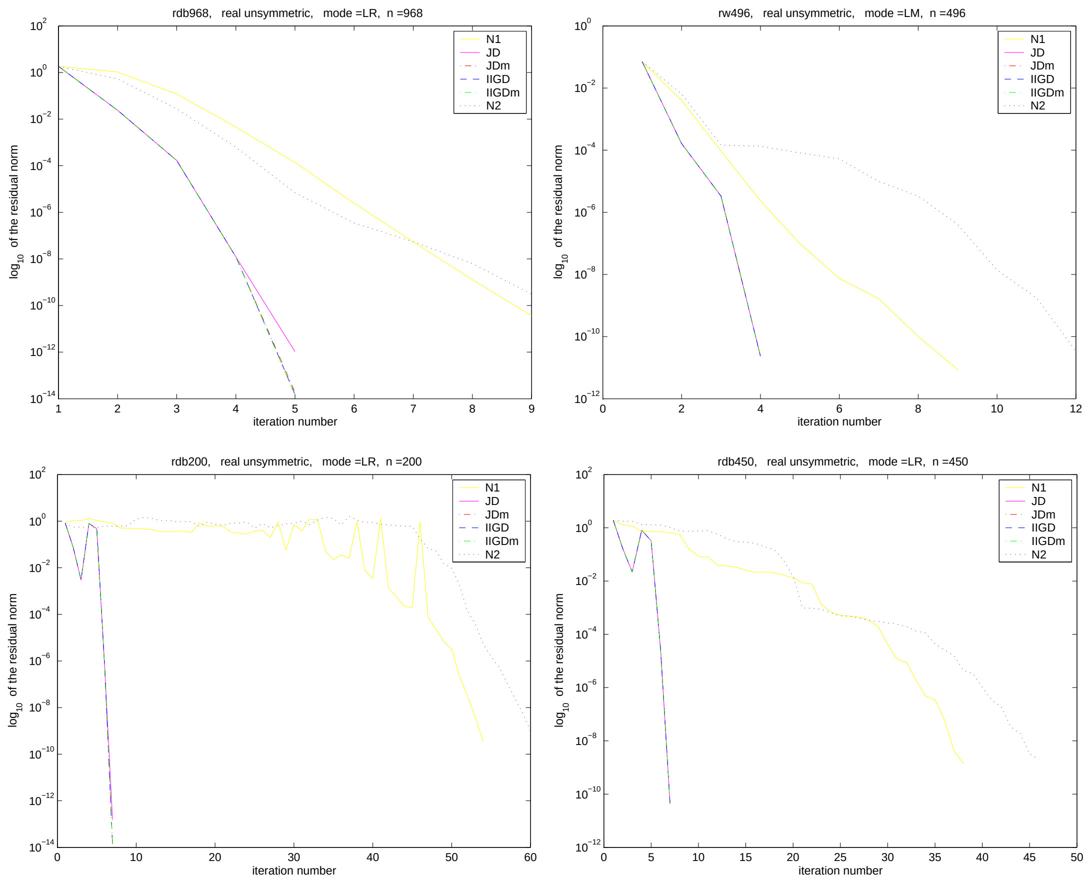

Figure 2: Comparison of performance for nonsymmetric models: rdb968, rw496, rdb200, rdb450.

ods. Preconditioned forms based on (27) and (28) are expected to be efficient and practical for large scale eigenvalue problems from real-world applications; this will be the subject of ongoing research.

Acknowledgments: The author thanks Prof. Dan Sorensen for introducing him to eigenvalue computations. He thanks R. Shepard and M. Minkoff for some interesting discussions. The valuable NSF travel support from the Institute of Pure and Applied Mathematics (IPAM) to attend the Nanoscale science and Engineering workshops at UCLA is also acknowledged.

References

[1] W. E. Arnoldi, *The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem*, Quarterly Journal of Applied Mathematics, 9 (1951) pp. 17-29.

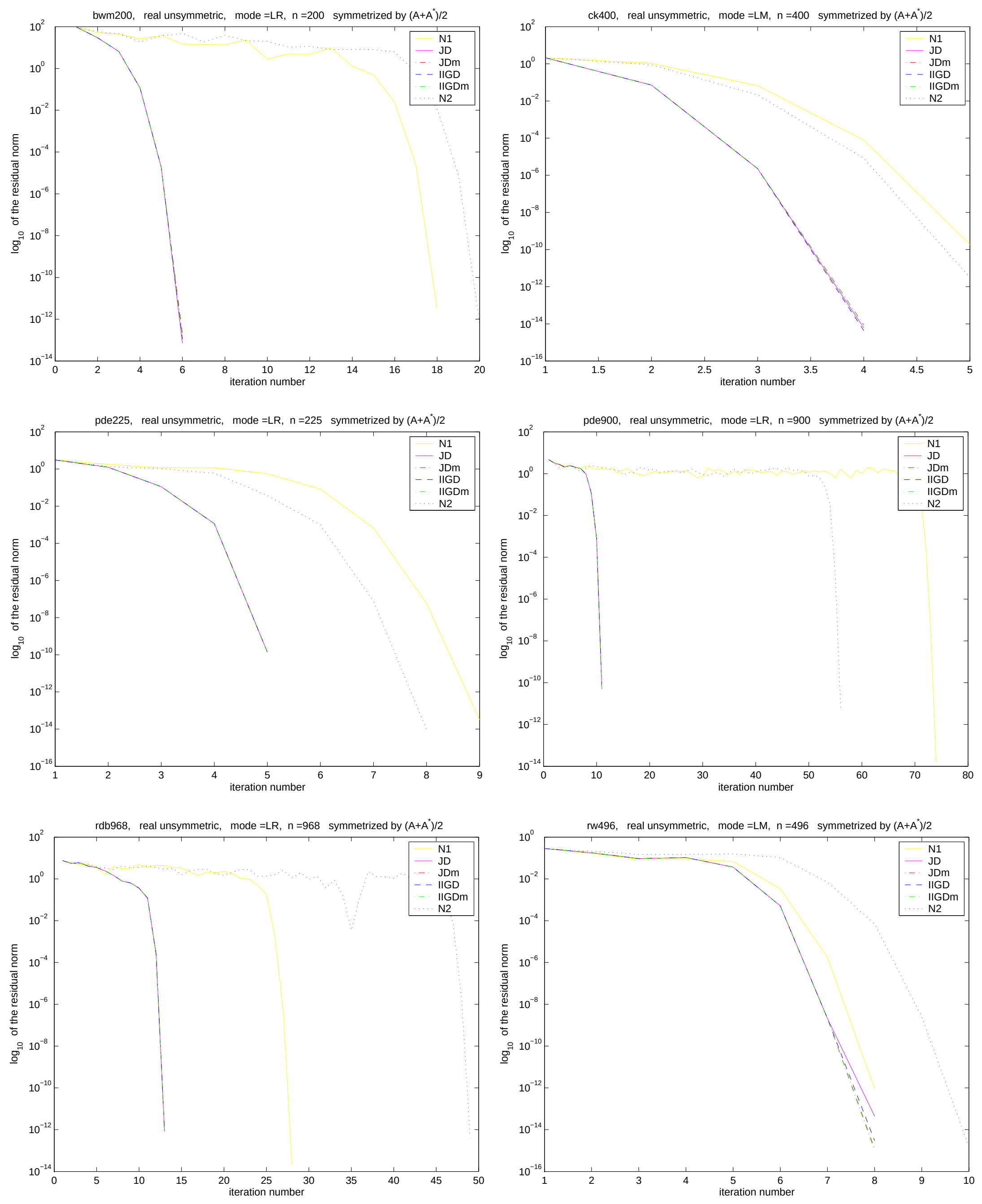

Figure 3: Comparison of performance for symmetric models: bwm200s, ck400s, pde225s, pde900s, rdb968s, rw496s.

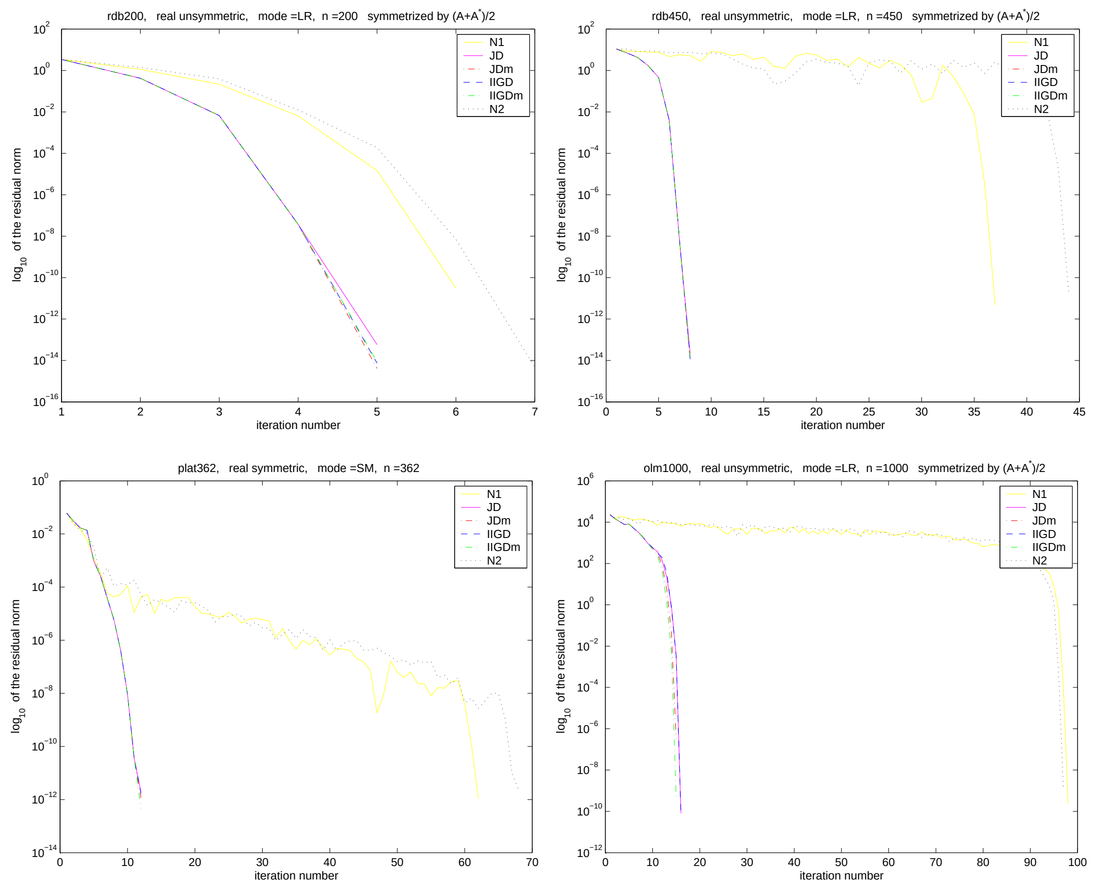

Figure 4: Comparison of performance for symmetric models: rdb200s, rdb450s, plat362, olm1000s.

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

[3] F. Chaitin-Chatelin, *Eigenvalues of Matrices*, John Wiley & Son, 1993.

[4] M. Crouzeix, B. Philippe, M. Sadkane, *The Davidson Method,* SIAM Journal on Scientific Computing, 15 (1994) pp. 62-76.

[5] J. K. Cullum, R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Vol. I: Theory*, SIAM, Classics in Applied Mathematics 41, 2002.

[6] J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart, *Reorthogonalization and Stable Algorithms for Updating the Gram-Schmidt QR Factorization*, Mathematics of Computation, 30 (1976) pp. 772-795.

[7] E. R. Davidson, *The Iterative Calculation of A Few of the Lowest Eigenvalues and Corresponding Eigenvectors of Large Real Symmetric Matrices,* Journal of Computational Physics, 17 (1975) pp. 87-94.

[8] E. R. Davidson, *Monster Matrices: Their Eigenvalues and Eigenvectors*, Computers in Physics, 7 (1993) pp. 519-522.

[9] A. Dax, *The Orthogonal Rayleigh Quotient Iteration Method*, Linear Algebra and Its Applications, 358 (2003) pp. 23-43.

[10] J. E. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, 1983; updated reprint, SIAM classics in Applied Mathematics, 16, 1996.

[11] J. Dongarra, I. Duff, D. C. Sorensen, H. A. van der Vorst, *Numerical Linear Algebra for High-Performance Computers*, SIAM publications, 1998.

[12] D. R. Fokkema, G. L. G. Sleijpen, H. A. van der Vorst, *Jacobi-Davidson Style QR and QZ Algorithms for the Reduction of Matrix Pencils,* SIAM Journal on Scientific Computing, 20 (1998) pp. 94-125.

[13] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 3rd ed., 1996.

[14] G. H. Golub, Q. Ye, *An Inverse Free Preconditioned Krylov Subspace Method for Symmetric Generalized Eigenvalue Problems*, SIAM Journal on Scientific Computing, 24 (2002) pp. 312-334.

[15] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, SIAM, Frontiers in Applied Mathematics, v17, 1997.

[16] M. E. Hochstenbach, G. L. G. Sleijpen, *Two-sided and Alternating Jacobi-Davidson,* Linear Algebra and Its Applications, 358 (2003) pp. 145-172.

[17] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, 1991.

[18] A. V. Knyazev, *Toward the Optimal Preconditioned Eigensolver: Locally Optimal Block Preconditioned Conjugate Gradient Method,* SIAM Journal on Scientific Computing, 23 (2001) pp. 517-541.

[19] A. V. Knyazev, Klaus Neymeyr, *Efficient Solution of Symmetric Eigenvalue Problems Using Multigrid Preconditioners in the Locally Optimal Block Conjugate Gradient Method,* Electronic Transactions on Numerical Analysis, 15 (2003), pp. 38-55.

[20] C. Lanczos, *An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators*, J. Res. Nat. Bur. Standards, 45 (1950) pp. 255-282.

[21] R. B. Lehoucq, K. Meerbergen, *Using Generalized Cayley Transformations within an Inexact Rational Krylov Sequence Method,* SIAM Journal on Matrix Analysis and Applications, 20 (1998) pp. 131-148.

[22] R. B. Morgan, D. S. Scott, *Generalization of Davidson's Method for Computing Eigenvalues of Sparse Symmetric Matrices*, SIAM Journal on Statistical and Scientific Computing, 7 (1986) pp. 817-825.

[23] Y. Notay, *Convergence Analysis of Inexact Rayleigh Quotient Iteration,* SIAM Journal on Matrix Analysis and Applications, 24 (2003) pp. 627-644.

[24] J. Olsen, P. Jørgensen, J. Simons, *Passing the One-billion Limit in Full Configuration-Interaction (FCI) Calculations,* Chemical Physics Letters, 169 (1990) pp. 463-472.

[25] A. M. Ostrowski, *On the Convergence of the Rayleigh Quotient Iteration for the Computation of Characteristic Roots and Vectors. I–VI,* Arch. Rational Meth. Anal., v. 1–4, 1958/59, pp. 233-241,423-428,325-340,341-347,472-481,153-165. (cited in [26]).

[26] B. N. Parlett, *The Rayleigh Quotient Iteration and Some Generalizations for Nonnormal Matrices,* Mathematics of Computation, 28 (1974) pp. 679-693.

[27] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, 1980; updated reprint, SIAM, Classics in Applied Mathematics 20, 1998.

[28] G. Peters, J. H. Wilkinson, *Inverse Iteration, Ill-conditioned Equations and Newton's Method*, SIAM Review, 21 (1979) pp. 339-360.

[29] A. Ruhe, *Rational Krylov Sequence Methods for Eigenvalue Computations*, Linear Algebra and Its Applications, 58 (1984) pp. 391-405.

[30] Y. Saad, *Numerical Methods for Large Eigenvalue Problems,* John Wiley, 1992. `http://www-users.cs.umn.edu/~saad/books.html`.

[31] Y. Saad and M. Schultz, *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*, SIAM Journal on Statistical and Scientific Computing, 7 (1986) pp. 856-869.

[32] G. L. G. Sleijpen and H. A. van der Vorst, *A Jacobi-Davidson Iteration Method for Linear Eigenvalue Problems*, SIAM Journal on Matrix Analysis and Applications, 17 (1996) pp. 401-425. Reprinted in SIAM Review, 42 (2000), pp. 267-293.

[33] G. L. G. Sleijpen, A. G. L. Booten, D. R. Fokkema, H. A. van der Vorst, *Jacobi-Davidson Type Method for Generalized Eigenproblems and Polynomial Eigenproblems,* BIT, 36 (1996) pp. 595-633.

[34] G. L. G. Sleijpen and H. A. van der Vorst, *The Jacobi-Davidson Method for Eigenvalue Problems and Its Relation to Accelerated Inexact Newton Schemes,* Proceeding of the second IMACS International Symposium on Iterative Methods in Linear Algebra, June, 1995.

[35] D. C. Sorensen, *Implicit Application of Polynomial Filters in a k-step Arnoldi Method*, SIAM Journal on Matrix Analysis and Applications, 13 (1992) pp. 357-385.

[36] D. C. Sorensen, *Numerical Methods for Large Eigenvalues Problems*, Acta Numerica, (2002) pp. 519-584.

[37] D. C. Sorensen, C. Yang, *A Truncated RQ Iteration for Large Scale Eigenvalue Calculations,* SIAM Journal on Matrix Analysis and Applications, 19 (1998) pp. 1045-1073.

[38] A. Stathopoulos, Y. Saad, C. F. Fisher, *Robust Preconditioning of Large, Sparse, Symmetric Eigenvalue Problems,* Journal of Computational and Applied Mathematics, 64 (1995) pp. 197-215.

[39] A. Stathopoulos, Y. Saad, K. Wu, *Dynamic Thick Restarting of the Davidson, and the Implicit Restarted Arnoldi Methods,* SIAM Journal on Scientific Computing, 19 (1998) pp. 227-245.

[40] G. W. Stewart, *Matrix Algorithms, Volume II: Eigensystems,* SIAM publications, 2001.

[41] G. W. Stewart, *A Krylov–Schur Algorithm for Large Eigenproblems*, SIAM Journal on Matrix Analysis and Applications, 23 (2001) pp. 601-614.

[42] J. H. Wilkinson, *The Algebraic Eigenvalue Problem,* Oxford University Press, 1965.

[43] K. Wu, Y. Saad, A. Stathopoulos, *Inexact Newton Preconditioning Techniques for Large Symmetric Eigenvalue Problems*, Electronic Transactions on Numerical Analysis, 7 (1998) pp. 202-214.

[44] K. Wu, H. Simon, *Thick-restart Lanczos Method for Large Symmetric Eigenvalue Problems,* SIAM Journal on Matrix Analysis and Applications, 22 (2000) pp. 602-616.

[45] C. Yang, *Convergence Analysis of an Inexact Truncated RQ Iterations,* Electronic Transactions on Numerical Analysis, 7 (1998) pp. 40-55.

[46] H. A. van der Vorst, *Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems,* SIAM Journal on Statistical and Scientific Computing, 13 (1992) pp. 631-644.