NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2005; **00**:1–6 *Prepared using* nlaauth.cls *[Version: 2002/09/18 v1.02]*

Studies on Jacobi-Davidson, Rayleigh Quotient Iteration, Inverse Iteration Generalized Davidson and Newton Updates

Yunkai Zhou [†]

Dept. of Computer Science & Engineering
University of Minnesota
Minneapolis, MN 55455, USA

SUMMARY

We study Davidson-type subspace eigensolvers. Correction equations of Jacobi-Davidson and several other schemes are reviewed. New correction equations are derived. A general correction equation is constructed, existing correction equations may be considered as special cases of this general equation. The main theme of this study is to identify the essential common ingredient that leads to the efficiency of a diverse form of Davidson-type methods. We emphasize the importance of the approximate Rayleigh-quotient-iteration direction.

KEY WORDS: Eigenproblem; Davidson-type subspace method; Correction equations; Rayleigh quotient; Jacobi-Davidson; IIGD; Newton method.

1. Introduction

We study Davidson-type subspace methods for the standard eigenvalue problem

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

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, 5, 7, 17, 27, 31, 40, 43] contain in-depth discussions on properties and algorithms for eigenvalue problems, together with many sources from science and engineering applications.

[*] Correspondence to: University of Minnesota, 200 Union St. SE., Minneapolis, MN 55455, USA.
[†] Email: zhou@msi.umn.edu

Contract/grant sponsor: This work was partially 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; and partially supported by another U.S. Department of Energy contract; contract/grant number: DE-FG02-03ER25585.

Received Aug. 13, 2003
Revised Oct. 24, 2005
Accepted Nov. 28, 2005

Krylov subspace methods constitute a major group of methods for (1). They include the Arnoldi method [1], the Lanczos method [20], and their important variants: the implicit restart Arnoldi method [36], the rational Krylov method [30, 21] and the Krylov-Schur method [41]. In contrast, Davidson-type subspace methods are not restricted to Krylov structures. They include the Davidson method [9], several generalizations [22, 6, 38], and the Jacobi-Davidson method [33, 34, 15, 18].

A key advantage of a subspace method over a single vector update approach is that the convergence may be more robust and the convergence rate may be faster. (For a symmetric eigenvalue problem the reason is easy to see since $\forall f : \mathbb{R}^n \to \mathbb{R}$ (e.g., f is the Rayleigh-quotient), $\min_{\mathbf{x} \in S_1} f(\mathbf{x}) \leq \min_{\mathbf{x} \in S_2} f(\mathbf{x})$ for $S_2 \subseteq S_1 \subseteq \mathbb{R}^n$. This is often called the *subspace acceleration*.

In this paper we study mainly the Davidson-type subspace methods for (1). Within each step of a Davidson-type method, a standard Rayleigh-Ritz procedure is usually applied. It is known that exterior eigenvalues are approximated more efficiently than interior ones by Davidson-type methods with standard Rayleigh-Ritz procedures. For approximation of interior eigenvalues, the *harmonic Rayleigh-Ritz* procedure [33] may be used. Harmonic Rayleigh-Ritz is related to shift-invert but its clever formulation avoids matrix inversion, hence it can be more favorable for large problems.

In large scale eigenvalue computation, solving correction equations inexactly via (preconditioned) iterative methods is a crucial part for most of the fast eigensolvers. It is known (e.g., [33, 15, 34, 37]) that when suitable correction equations are found, preconditioned formulations can be obtained by modifying the correction equations. Hence the focus of this paper is to study different correction equations. We wish to identify the essential ingredient that a diverse form of correction equations share which leads to a fast Davidson-type eigensolver.

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. And we use μ to denote Ritz values.

2. Davidson-type Subspace Eigensolver

Before getting into details we present in Algorithm 2.1 the framework of a Davidson-type subspace method for (1), where for simplicity we only compute one eigenpair that best satisfies a selection criterion. The criterion may be largest/smallest real part (LR/SR), imaginary part (LI/SI), or magnitude (LM/SM); or that the eigenvalue is closest to a given value.

For large eigenproblems restart is necessary because of memory constraint. Moreover, effective restart can refine the starting vector, which is important for the convergence rate of a subspace eigensolver. The *implicit restart Arnoldi* (IRA) [36] provides an efficient restart strategy for Krylov-type subspace methods. The Krylov-Schur method [41] is mathematically equivalent to IRA but uses Schur vectors for restart. Restart for Davidson-type methods are addressed in, e.g., [15, 39]. All the restart techniques try to retain as much useful information that have been accumulated in the current basis vectors as possible. As for Algorithm 2.1, the restart corresponds to computing the Schur decomposition of the Rayleigh quotient matrix: $\mathbf{H}_{j+1}\mathbf{Y} = \mathbf{YS}$ (at step 5.(e) when $j = m$), where the upper triangular form $\mathbf{S}$ is ordered s.t. the first k diagonal elements best approximate the wanted eigenvalues. Then the algorithm

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

```
1. Start with an initial unit vector $\mathbf{x}$, $\mathbf{V}_1 = [\mathbf{x}]$.
2. Compute $\mathbf{w} = \mathbf{A}\mathbf{x}$, $\mathbf{W}_1 = [\mathbf{w}]$.
3. Compute $\mu = \mathbf{x}^*\mathbf{w}$, $\mathbf{r} = \mathbf{w} - \mu\mathbf{x}$; let $\mathbf{H}_1 = [\mu]$.
4. If $\|\mathbf{r}\| <= \epsilon$, return $(\mu, \mathbf{x})$ as the eigenpair; else, set $k_1 = 1$.
5. Outer Loop: for $j = k_1, k_1 + 1, ..., m$ 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 $\mathbf{w} = \mathbf{A}\mathbf{v}$; $\mathbf{V}_{j+1} = [\mathbf{V}_j \mid \mathbf{v}]$, $\mathbf{W}_{j+1} = [\mathbf{W}_j \mid \mathbf{w}]$.
    (d) Compute $\mathbf{H}_{j+1} = \begin{bmatrix} \mathbf{H}_j & \mathbf{V}_j^*\mathbf{w} \\ \mathbf{v}^*\mathbf{W}_j & \mathbf{v}^*\mathbf{w} \end{bmatrix}$.
    (e) Compute the eigenpair $(\mu, \mathbf{y})$ $(\|\mathbf{y}\| = 1)$ of $\mathbf{H}_{j+1}$ that best satisfies the
        selection criterion. Set $\beta =$ the largest magnitude of eigenvalues of $\mathbf{H}_{j+1}$.
    (f) Compute the Ritz vector $\mathbf{x} = \mathbf{V}_{j+1}\mathbf{y}$
        and the residual vector $\mathbf{r} = \mathbf{A}\mathbf{x} - \mu\mathbf{x} = \mathbf{W}_{j+1}\mathbf{y} - \mu\mathbf{x}$.
    (g) Test for convergence: if $\|\mathbf{r}\| <= \epsilon\beta$, return $(\mu, \mathbf{x})$ as the eigenpair.
6. Restart, set $k_1 = k$, goto step 5.
```

restarts from

$$\mathbf{V}_k = \mathbf{V}_{m+1}\mathbf{Y}(:, 1:k),\ \mathbf{W}_k = \mathbf{W}_{m+1}\mathbf{Y}(:, 1:k),\ \mathbf{H}_k = \mathbf{S}(1:k, 1:k).$$

It is often useful to do deflation when more eigenpairs are sought. Deflation means fixing converged wanted Schur vectors; the active subspace basis is forced to be orthogonal to these Schur vectors. For more detailed discussions on deflation, we refer to standard books, e.g. [17, 31, 40].

One 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. The other problem is that MGS can not be expressed by Level-2 BLAS, hence parallel implementation needs more communication [2, 13]. Another choice for orthogonalization is the DGKS method [8]. DGKS can be implemented more efficiently in parallel because DGKS is actually GS with reorthogonalization when necessary. As a referee pointed out, DGKS is preferred over MGS for eigenproblem also because that for a preconditioned eigensolver where linear systems are solved by GMRES, MGS for GMRES may fail to be stable when the residual is small.

What distinguishes each subspace method is the vector augmentation at step *5(a)*. When iterative methods (e.g., GMRES [32], BICGSTAB [42]) are used to solve for the augmentation vector, step *5(a)* necessarily contains the *inner iteration* of the subspace methods.

The (generalized) **Davidson** method [9, 22, 6] constructs $\mathbf{t}$ as follows:

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

$$(\mathbf{M}_j - \mu_j \mathbf{I})\,\mathbf{t} = -\mathbf{r}. \tag{2}$$

For notational simplicity, we omit the subindex of any Ritz pairs $(\mu_j, \mathbf{x}_j)$ wherever there is no confusion, and use $\mathbf{x}$ to denote the normalized Ritz vector.

In the original paper [9], Davidson derived this formulation by taking the derivative of the Rayleigh quotient

$$Q(\mathbf{z}) = \frac{\mathbf{z}^*\mathbf{A}\mathbf{z}}{\mathbf{z}^*\mathbf{z}}, \qquad \forall \mathbf{z} \neq \mathbf{0}, \tag{3}$$

at $\mathbf{x}$, varying only the i-th component of $\mathbf{x}$ (denoted as $\mathbf{x}_{(i)}$) each time:

$$\left.\frac{\partial Q(\mathbf{x})}{\partial \mathbf{x}_{(i)}}\right|_{\mathbf{x}_{(i)}+\delta\mathbf{x}_{(i)}} = 0, \qquad i = 1, 2, ..., n. \tag{4}$$

From (4) we get $\delta\mathbf{x}_{(i)} = (\mu - a_{ii})^{-1}(\mathbf{A}\mathbf{x} - \mu\mathbf{x})_{(i)}$. Hence $\mathbf{M}_j = diag(\mathbf{A})$ is proposed in [9]. In [10] 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 [43, 22, 38]), then the Davidson method converges very fast. Hence the diagonal matrix $(diag(\mathbf{A}) - \mu\mathbf{I})$ was considered as a straightforward preconditioner to $\mathbf{A} - \mu\mathbf{I}$. 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 [22, 6], leading to the so called *generalized Davidson method.* As has been pointed out (e.g., in [22, 33]), the preconditioner interpretation does not explain the improved convergence rate well since the exact preconditioner $(\mathbf{A} - \mu\mathbf{I})^{-1}$ leads to stagnation ($(\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{r}$ is $\mathbf{x}$, which lies in the original projection subspace).

The **Jacobi-Davidson** method [33] is an important advance for Davidson-type subspace methods. Jacobi-Davidson solves the following projected equation:

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

$$(\mathbf{I} - \mathbf{x}\mathbf{x}^*)(\mathbf{A} - \mu\mathbf{I})(\mathbf{I} - \mathbf{x}\mathbf{x}^*)\mathbf{t} = -\mathbf{r}. \tag{5}$$

Usually (5) is solved inexactly by iterative methods. The matrix $(\mathbf{I} - \mathbf{x}\mathbf{x}^*)(\mathbf{A} - \mu\mathbf{I})(\mathbf{I} - \mathbf{x}\mathbf{x}^*)$ is always singular on $\mathbb{C}^n$, but not on $\mathbf{x}^\perp$, therefore this singularity poses no essential difficulty to iterative methods for (5).

An equivalent formulation to (5) (see [33] §4) is the following *inverse iteration generalized Davidson* (IIGD) method [24]:

5(a) Solve (approximately)

$$(\mathbf{A}-\mu\mathbf{I})\mathbf{t}_1=\mathbf{r} \quad \text{and} \quad (\mathbf{A}-\mu\mathbf{I})\mathbf{t}_2=\mathbf{x};$$
$$\text{then set} \ \ \mathbf{t}=-\mathbf{t}_1+\tau\mathbf{t}_2, \quad \text{where} \quad \tau=\frac{\mathbf{x}^*\mathbf{t}_1}{\mathbf{x}^*\mathbf{t}_2}. \tag{6}$$

The τ is set to ensure $\mathbf{t}\perp\mathbf{x}$. From (6) it is clear that the expansion vector $\mathbf{t}$ contains information in both directions $(\mathbf{A}-\mu\mathbf{I})^{-1}\mathbf{r}$ and $(\mathbf{A}-\mu\mathbf{I})^{-1}\mathbf{x}$. 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 [40]).

A major advantage of Davidson-type subspace methods is that the correction equations allow approximate solutions, hence they can be solved by existing iterative methods; preconditioners may be incorporated. In contrast, shift-invert Arnoldi-type methods generally require rather accurate system solves, which presents difficulty for preconditioned iterative linear solvers. One main reason for the different accuracy requirement is that, in an Arnoldi-type method, the residual vector at each iteration is computed as a byproduct of the Arnoldi decomposition, it needs high accuracy in order to keep the Arnoldi structure; while a Davidson-type method preforms the Rayleigh-Ritz procedure and computes a new residual vector ($\mathbf{r}=\mathbf{A}\mathbf{x}-\mu\mathbf{x}$) at each iteration step. The extra work frees a Davidson-type method from the need to solve its correction equations accurately.

3. Correction Equations Based on Newton Updates

The locally fast convergence property of the (single vector) Newton method is an attractive feature for designing fast algorithms. In [35] and [24] it was argued that Jacobi-Davidson and IIGD are (inexact) Newton-Raphson methods. The generalized Rayleigh quotient $Q(\mathbf{z},\mathbf{y}):=\frac{\mathbf{z}^*\mathbf{A}\mathbf{y}}{\mathbf{z}^*\mathbf{y}}$ was used in [35] to establish the argument. [16] also contains discussions on Jacobi-Davidson method as subspace accelerated Newton method.

In this section we derive a Newton update straightforwardly from the standard Rayleigh quotient (3), discuss its several disadvantages. We then review several more efficient correction equations based on applying the Newton method to transformed formulations of (1). Note that because of the normalization, the Newton updates discussed here are not Newton method in the strict sense. This is pointed out in [28].

Throughout this section we assume $\mathbf{A}$ is real and symmetric. The gradient of the Rayleigh quotient (3) is

$$\nabla Q(\mathbf{z})=\frac{2\mathbf{A}\mathbf{z}}{\mathbf{z}^T\mathbf{z}}-\frac{2\mathbf{z}^T\mathbf{A}\mathbf{z}\mathbf{z}}{(\mathbf{z}^T\mathbf{z})^2}=\frac{2}{\mathbf{z}^T\mathbf{z}}(\mathbf{A}\mathbf{z}-Q(\mathbf{z})\mathbf{z}). \tag{7}$$

The Hessian of (3) is

$$\nabla^2 Q(\mathbf{z}) \quad = \quad \frac{2}{\mathbf{z}^T\mathbf{z}}(\mathbf{A}-Q(\mathbf{z})\mathbf{I})-\frac{4}{(\mathbf{z}^T\mathbf{z})^2}(\mathbf{A}\mathbf{z}\mathbf{z}^T+\mathbf{z}\mathbf{z}^T\mathbf{A}^T-2Q(\mathbf{z})\mathbf{z}\mathbf{z}^T). \tag{8}$$

The $\mathbf{A}^T$ comes from expressing $\nabla Q(\mathbf{z})^T$ straightforwardly. Let $\mathbf{z}$ be the normalized Ritz vector $\mathbf{x}$ and denote $\mu=Q(\mathbf{x})$, then (8) becomes

$$\nabla^2 Q(\mathbf{x})=2(\mathbf{A}-\mu\mathbf{I})-4(\mathbf{A}\mathbf{x}\mathbf{x}^T+\mathbf{x}\mathbf{x}^T\mathbf{A}^T-2\mu\mathbf{x}\mathbf{x}^T). \tag{9}$$

We note that the formula of the Hessian for a more general Rayleigh-quotient $\frac{\mathbf{z}^T\mathbf{A}\mathbf{z}}{\mathbf{z}^T\mathbf{M}\mathbf{z}}$ may be found in [29].

By the assumption $\mathbf{A} = \mathbf{A}^T$, also note that $\mathbf{r} = \mathbf{A}\mathbf{x} - \mu\mathbf{x}$, we may simplify the three correction terms in (9) into two terms:

$$\mathbf{A}\mathbf{x}\mathbf{x}^T + \mathbf{x}\mathbf{x}^T\mathbf{A}^T - 2\mu\mathbf{x}\mathbf{x}^T = \mathbf{r}\mathbf{x}^T + \mathbf{x}\mathbf{r}^T. \tag{10}$$

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

$$(\mathbf{A} - \mu\mathbf{I} - 2(\mathbf{r}\mathbf{x}^T + \mathbf{x}\mathbf{r}^T))\mathbf{t} = -\mathbf{r}. \tag{11}$$

Since $\mathbf{A} = \mathbf{A}^T$, equation (11) is equivalent to:

$$(\mathbf{I} - 2\mathbf{x}\mathbf{x}^T)(\mathbf{A} - \mu\mathbf{I})(\mathbf{I} - 2\mathbf{x}\mathbf{x}^T)\mathbf{t} = -\mathbf{r}. \tag{12}$$

From (12) it is clear that the Hessian is singular when μ is an eigenvalue. Recall that one of the standard conditions for a Newton method applied to $f(\mathbf{x}) = 0$ to have quadratic convergence rate is that ∇f is nonsingular at the solution [†] [12, 19], hence (11) and (12) are not very exciting Newton formulations. A more serious problem is that $-\mathbf{x}$ is the only solution of (11) or (12) when μ is not an eigenvalue of $\mathbf{A}$. Therefore (11) or (12) has the same stagnation problem associated with the Davidson method (2) using $\mathbf{M}_j = \mathbf{A}$. We present (11) and (12) mainly to show that it can be misleading to assume that a formulation derived by a Newton method always has at least quadratic convergence rate. The standard conditions for a quadratic rate should be taken into account when determining the rate of a Newton formulation.

Another shortcoming about the above derivation based on the standard Rayleigh-quotient (3) is the condition $\mathbf{A} = \mathbf{A}^T$. Note also that the derivatives in (7) and (8) are w.r.t. $\mathbf{z}$, but there are $n+1$ unknowns $(\mathbf{z}, \mu)$; the extra unknown μ is fixed as $Q(\mathbf{z})$.

Note that essentially we want the residual to be zero. A much better approach, as taken in [28, 34, 44], is to apply Newton method to

$$F(\mathbf{x}, \mu) := \begin{bmatrix} (\mathbf{A} - \mu\mathbf{I})\mathbf{x} \\ -\frac{1}{2}\mathbf{x}^T\mathbf{x} + \frac{1}{2} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}. \tag{13}$$

This avoids the $\mathbf{A} = \mathbf{A}^T$ constraint, and the natural normalization condition $\mathbf{x}^T\mathbf{x} = 1$ makes (13) $n+1$ equations for $n+1$ unknowns $(\mathbf{x}, \mu)$. It is straightforward to verify that the Newton equation w.r.t. $(\mathbf{x}, \mu)$ for (13) is

$$\begin{bmatrix} \mathbf{A} - \mu\mathbf{I} & -\mathbf{x} \\ -\mathbf{x}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{t} \\ \eta \end{bmatrix} = \begin{bmatrix} -\mathbf{r} \\ 0 \end{bmatrix}. \tag{14}$$

The Jacobian matrix $\begin{bmatrix} \mathbf{A} - \mu\mathbf{I} & -\mathbf{x} \\ -\mathbf{x}^T & 0 \end{bmatrix}$ is generally nonsingular, even when $(\mu, \mathbf{x})$ is a simple eigen pair of $\mathbf{A}$. The nonsingularity was proved in [28]. Therefore the Newton formulation (14) for (13) has quadratic convergence rate. Actually the design of Jacobi-Davidson [33] exploits this nonsingularity, since (5) is equivalent to (14).

[†] A simple example in the one dimensional case is $f(s) = s^n$ for any integer $n > 1$, Newton method applied to this $f(s) = 0$ is only linearly convergent because $f'(0) = 0$.

In [44] two other interesting updates are presented. The first one is the *constrained Newton recurrence* obtained by applying Newton method to

$$\mathbf{Ax} - \mathbf{xx}^T\mathbf{Ax} = \mathbf{0}. \tag{15}$$

The Newton equation for (15) is:

$$[\mathbf{A} - \mu\mathbf{I} - \mathbf{xx}^T(\mathbf{A} + \mathbf{A}^T)]\mathbf{t} = -\mathbf{r}. \tag{16}$$

The second update is the following,

$$[\mathbf{A} - \mu\mathbf{I} + \alpha\mathbf{xx}^T]\mathbf{t} = -\mathbf{r}. \tag{17}$$

It is based not on Newton method but instead on a heuristic. Nevertheless it is called the *inflated Newton recurrence* in [44], The value $\alpha = 1$ is used in [44] for the numerical tests, while in [14] different values of α are tested and no significant difference is observed.

What seems tricky is when μ is an eigenvalue of $\mathbf{A}$ with geometric multiplicity greater than one. In this case the matrices in (14) and (16) (and also (17)) are singular, hence one of the standard conditions for Newton method to have quadratic convergence rate is not satisfied. Therefore the quadratic convergence rate for (14) and (16) may not be established by simply referring to the Newton method. We will address this in the next section. It turns out that the Rayleigh-quotient-iteration, whose convergence rate is not affected by eigenvalue multiplicity, will resolve this unfavorable case nicely.

4. Unification by Rayleigh-Quotient-Iteration and a Generalization

In this section we unify the Jacobi-Davidson (JD), IIGD and the methods (14), (16) and (17) in Section 3 by the Rayleigh-quotient-iteration (RQI). A general formulation that captures the essence of all these correction equations is presented.

It is well-known that RQI is cubicly convergent for normal matrices and quadratically convergent for nonnormal matrices [26]. The convergence rate is independent of the eigenvalue multiplicity [27, p.78]. Extensive studies on RQI may be found, e.g., in [25, 26, 27, 40, 11, 23] and references therein.

Note that the subspace method with updating equations (5), (6), (11) and (12) are special cases of the following (18),

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

$$(\mathbf{I} - \alpha\,\mathbf{xx}^*)(\mathbf{A} - \mu\mathbf{I})\mathbf{t} = -\mathbf{r}, \tag{18}$$

where α is a suitable nonzero constant.

To discuss the validity of (18), we first establish the following result.

Proposition 4.1. *Let the initial unit vector be the same* $\mathbf{x}$, *assuming that* μ *is not an eigenvalue of* $\mathbf{A}$ *and that* $\mathbf{x}^*(\mathbf{A} - \mu\mathbf{I})\mathbf{t} \neq 0$. *Then the subspace method via (18) produces a subspace that includes the (approximate) RQI direction during the next iteration.*

Proof : It is well known that the RQI direction is

$$\mathbf{x}_R = (\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{x}. \tag{19}$$

(Note that when the inverse is done via approximate solve, we get an approximate direction. For simplicity of notation, we do not use extra symbols for the approximate solves and approximate directions throughout.) Equation (18) can be written as

$$(\mathbf{A} - \mu\mathbf{I})\mathbf{t} = -\mathbf{r} + \eta\,\mathbf{x}. \tag{20}$$

From the assumption $\mathbf{x}^*(\mathbf{A} - \mu\mathbf{I})\mathbf{t} \neq 0$ and $\alpha \neq 0$, we see $\eta \neq 0$. Equation (20) is essentially the IIGD equation (6). From (20)

$$\mathbf{t} = (\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{r} + \eta\,(\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{x} = -\mathbf{x} + \eta\,\mathbf{x}_R. \tag{21}$$

Note that $\mathbf{x}$ is in the original projection subspace, say, $\mathbf{V}_j$. Therefore the RQI direction $\mathbf{x}_R$ will be included in the augmented projection subspace $\mathbf{V}_{j+1}$ when *5(b)* and *5(c)* are performed in Algorithm 2.1. ■

This proposition shows that, under the specified conditions, the subspace method with (18) is able to retain the RQI direction. But under what condition does $\mathbf{x}^*(\mathbf{A} - \mu\mathbf{I})\mathbf{t} \neq 0$ hold true for a solution $\mathbf{t}$ of (18)? The following lemma answers this question.

Lemma 4.1. *In exact arithmetic, any solution* $\mathbf{t}$ *of (18) satisfies* $\mathbf{x}^*(\mathbf{A} - \mu\mathbf{I})\mathbf{t} = 0$ *if* $\alpha \neq 1$.

Proof : Since $\alpha \neq 1$, it is easy to show that $(\mathbf{I} - \alpha\,\mathbf{x}\mathbf{x}^*)$ is invertible and has an eigenvector $\mathbf{x}$ which corresponds to the eigenvalue $1 - \alpha$, the other eigenvalues are all 1's. The eigen-decomposition of $(\mathbf{I} - \alpha\,\mathbf{x}\mathbf{x}^*)$ can be written as

$$(\mathbf{I} - \alpha\,\mathbf{x}\mathbf{x}^*) = \mathbf{U}\begin{bmatrix} 1-\alpha & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}\mathbf{U}^*, \tag{22}$$

where $\mathbf{U} = [\mathbf{x}, \mathbf{u}_2, ..., \mathbf{u}_n]$, $\mathbf{U}^*\mathbf{U} = \mathbf{I}$ (note $\|\mathbf{x}\| = 1$).

Together with the known condition $\mathbf{x}^*\mathbf{r} = 0$, (18) and (22) lead to

$$\mathbf{x}^*(\mathbf{A} - \mu\mathbf{I})\mathbf{t} = \frac{1}{1-\alpha}\mathbf{x}^*\mathbf{r} + \sum_{i=2}^{n}(\mathbf{x}^*\mathbf{u}_i\mathbf{u}_i^*\mathbf{r}) = 0. \quad ■$$

A much simpler proof is to multiply (18) on the left by $\mathbf{x}^*$ to get

$$(1 - \alpha)\mathbf{x}^*(\mathbf{A} - \mu\mathbf{I})\mathbf{t} = 0. \tag{23}$$

Under the condition that μ is not an eigenvalue of $\mathbf{A}$ (as in Proposition 4.1), we see that the matrix $(\mathbf{I} - \alpha\,\mathbf{x}\mathbf{x}^*)(\mathbf{A} - \mu\mathbf{I})$ is nonsingular if $\alpha \neq 1$. In this case (18) has only one solution which is readily verified to be $-\mathbf{x}$. However, this means that (18) with $\alpha \neq 1$ has the same stagnation problem associated with the exact preconditioned Davidson method. Therefore, Lemma 4.1 reveals that $\alpha = 1$ (JD [33]) is the only viable choice for the class of methods in (18) to retain the RQI direction.

The proof of Proposition 4.1 shows that the essence is to make the coefficient of $\mathbf{x}_R$ nonzero. But Lemma 4.1 means that (18) does not generalize much over JD, except that if $\tilde{\mathbf{t}}$ is a solution of (5), then $\mathbf{t} = (\mathbf{I} - \mathbf{x}\mathbf{x}^*)\tilde{\mathbf{t}}$ is a solution of (18) for $\alpha = 1$.

To really generalize the formulation so that α is not restricted only to 1, we first construct the following

$$(\mathbf{I} - \alpha\,\mathbf{x}\mathbf{z}^*)(\mathbf{A} - \mu\mathbf{I})\mathbf{t} = -\mathbf{r}, \tag{24}$$

where $\mathbf{z}$ represents one of certain suitable unit vectors. It is not hard to see a more general formulation, which we express in the following as one step of Algorithm 2.1:

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

$$(\mathbf{A} - \mu\mathbf{I} - \alpha\,\mathbf{x}\mathbf{y}^*)\mathbf{t} = -\mathbf{r}, \tag{25}$$

where $\mathbf{y}$ is a given suitable vector and α is a suitable nonzero constant.

Clearly one can absorb α into $\mathbf{y}$, or keep α but make $\mathbf{y}$ of unit length. We do not make this distinction since it is non-essential. Note that (24) is a special case of (25) when $\mathbf{y}$ is in the direction of $(\mathbf{A}^* - \bar{\mu}\mathbf{I})\mathbf{z}$. So we focus discussion on (25).

The following result establishes what kind of a given $\mathbf{y}$ can be suitable (in the sense that $\mathbf{x}_R$ is retained in the subspace generated by Algorithm 2.1).

Theorem 4.1. *Assuming that μ is not an eigenvalue of $\mathbf{A}$. If $\mathbf{y}$ and a nonzero scalar α satisfy*

$$\alpha\mathbf{y}^*(\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{x} \neq 1 \quad \text{and} \quad \mathbf{y}^*\mathbf{x} \neq 0, \tag{26}$$

then Algorithm 2.1 via (25) produces a subspace that includes the (approximate) RQI direction.

Proof : By assumption $(\mathbf{A} - \mu\mathbf{I})$ is nonsingular. Define

$$\sigma := \mathbf{y}^*(\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{x} - \frac{1}{\alpha}. \tag{27}$$

The first part of (26) leads to $\sigma \neq 0$. Therefore, by the Sherman-Morrison formula [12, p.188], $(\mathbf{A} - \mu\mathbf{I} - \alpha\,\mathbf{x}\mathbf{y}^*)$ is nonsingular and its inverse is

$$\left[\mathbf{I} - \frac{1}{\sigma}(\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{x}\mathbf{y}^*\right](\mathbf{A} - \mu\mathbf{I})^{-1}.$$

Note that $\mathbf{x} = (\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{r}$ and $\mathbf{x}_R = (\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{x}$, we see that the unique solution to (25) is

$$\mathbf{t} = -\mathbf{x} + \frac{1}{\sigma}(\mathbf{y}^*\mathbf{x})\mathbf{x}_R. \tag{28}$$

Since $\frac{\mathbf{y}^*\mathbf{x}}{\sigma} \neq 0$, $\mathbf{x}_R$ is clearly retained in $\mathbf{t}$, hence it is in the subspace generated by Algorithm 2.1 using (25). ■

Because $\mathbf{x}_R$ is unknown, it may be difficult to make sure $\alpha\mathbf{y}^*\mathbf{x}_R \neq 1$. However, (28) reveals that essentially one only needs to guarantee $\mathbf{y}^*\mathbf{x} \neq 0$; while the choice of α can be rather free, since a small σ (i.e., $\alpha\mathbf{y}^*\mathbf{x}_R \approx 1$) is actually beneficial.

The generality of (25) is in the choice of $\mathbf{y}$ and α. E.g., $\mathbf{y} = -\mathbf{x}$ leads to the *inflated Newton* scheme (17); while $\mathbf{y} = (\mathbf{A} + \mathbf{A}^*)\mathbf{x}$ and $\alpha = 1$ lead to the *constrained Newton* scheme (16). And from (20) we see that IIGD is a special case of (25). Note that the simplified JD (18) with $\alpha = 1$ is a special case of (25) with $\mathbf{y} = (\mathbf{A}^* - \bar{\mu}\mathbf{I})\mathbf{x}$ and $\alpha = 1$. Clearly, the corresponding matrix should be singular. This is also easy to see by the Sherman-Morrison Lemma, noticing that with the chosen $(\mathbf{y}, \alpha)$ the σ defined by (27) is zero. Theorem 4.1 does not apply to this

$(\mathbf{y}, \alpha)$ pair because $\mathbf{y}^*\mathbf{x} = 0$. The magic of JD is that although the matrix in (5) is singular, it is nonsingular on $\mathbf{x}^\perp$ unless μ is a multiple eigenvalue of $\mathbf{A}$. By choosing $\alpha = 1$, JD naturally satisfies the condition (23), and there are ample freedom to make $\mathbf{x}^*(\mathbf{A} - \mu\mathbf{I})\mathbf{t} \neq 0$ so that the approximate RQI direction is retained. Another interesting point is that if Theorem 4.1 is literaturely applied to JD (rigorously it does not apply), then JD corresponds to the limit $0/0$ case in (28).

Theorem 4.1 shows that when μ is not an eigenvalue of $\mathbf{A}$, there are many $(\mathbf{y}, \alpha)$ pairs which can make the matrix in (25) nonsingular. One may choose $(\mathbf{y}, \alpha)$ to make (25) more favorable for an iterative linear solver.

In the real case, a nontrivial difference between (25) with $\mathbf{y} = (\mathbf{A}+\mathbf{A}^T)\mathbf{x}$ and the *constrained Newton* scheme (16) is worth mentioning: There exists many values of α for the following (29) to have similar numerical efficiency as (16),

$$[\mathbf{A} - \mu\mathbf{I} - \alpha\mathbf{x}\mathbf{x}^T(\mathbf{A} + \mathbf{A}^T)]\mathbf{t} = -\mathbf{r}, \qquad (\alpha \neq 1). \tag{29}$$

This is very much the same as what is observed in [14] on the choice of α for the *inflated Newton* scheme (17). Both cases seem may be explained by the normalization step (e.g., 5.(b) in Algorithm 2.1).

Note that the Jacobian matrix in (16) is derived by applying Newton method to (15). The above observation suggests that Newton method may not be the most straightforward way to explain the efficiency of the discussed Davidson-type methods. We think that the essence for the efficiency is in properly retaining the (approximate) RQI direction in the subspace. In Section 6 we will present several concrete numerical results to support this view. One interesting numerical experiment is to choose $\mathbf{y}$ in (25) as a random vector (this will generally make $\mathbf{y}^*\mathbf{x} \neq 0$) and compare it with the better known schemes JD (5), the constrained Newton (16) and the inflated Newton (17).

As mentioned earlier, the quadratic convergence rate of the Newton method generally requires that the Jacobian be nonsingular at a solution, this can pose difficulty for eigenvalues with geometric multiplicity greater than 1. In contrast, the convergence property of RQI does not have this constraint (actually there is no concept of Jacobian involved in RQI), and the convergence rate of RQI is not affected by eigenvalue multiplicity. Moreover, RQI for normal matrices has cubic convergence rate; while it requires verifying additional condition (relating to the second derivative at a solution) to claim that a Newton scheme has a cubic convergence rate. Therefore, interpretations via RQI can be more convenient than via the Newton method in understanding the Davidson-type or JD-type methods discussed in this paper. (Note that in [33] RQI is used to discuss the convergence rate of JD.) The subtle difference is possibly caused by the fact that Newton method is mainly for solving equations, where both the direction and the magnitude of the direction matter; while for the $\mathbf{x}$ in $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$, only its direction is essential.

5. Simplification of Jacobi-Davidson and IIGD

We present two schemes that simplified JD and IIGD in some sense.

By Lemma 4.1, the only viable choice for (18) is $\alpha = 1$, as done in JD. This leads to the following simplified JD, *5(a)* Solve approximately for $\mathbf{t}$ from

$$(\mathbf{I}-\mathbf{x}\mathbf{x}^*)(\mathbf{A}-\mu\mathbf{I})\,\mathbf{t} = -\mathbf{r}. \tag{30}$$

Clearly, $-\mathbf{x}$ is a solution of (30), and this solution should be avoided. Fortunately, when we solve (30) by a Krylov method starting for a vector not parallel to $\mathbf{x}$, then the solution generally has a strong component in $\mathbf{x}^{\perp}$ because of the projection $(\mathbf{I}-\mathbf{x}\mathbf{x}^*)$. If the initial vector of the Krylov solver is chosen to be orthogonal to $\mathbf{x}$ (e.g., a zero initial vector), then the approximate solution is actually in $\mathbf{x}^{\perp}$, as would be obtained by JD. One referee pointed out that [15] (§3.2 remark 9) mentioned that the right projection in (5) can be dropped if (5) is solved by a Krylov method with an initial vector orthogonal to $\mathbf{x}$.

Numerical results in the next section will show that (30) performs as efficiently as JD. When (30) and (5) are solved by GMRES with small fixed iteration steps, there is not much difference in residual reduction and convergence steps between the two equations. We note that the right projection by $(\mathbf{I}-\mathbf{x}\mathbf{x}^*)$ in (5) is not performed at each iteration step, and the final $\mathbf{t}\perp\mathbf{x}$ is waived. The more essential term in (5) is the left projector $(\mathbf{I}-\mathbf{x}\mathbf{x}^*)$, which not only retains the (approximate) RQI direction in $\mathbf{t}$ but also improves the conditioning of the correction equation on the $\mathbf{x}^{\perp}$ subspace.

For $\mathbf{A}=\mathbf{A}^*$, generally the original JD (5) should be preferred over (30) since one can apply a symmetric solver to (5). For $\mathbf{A}\neq\mathbf{A}^*$ problems, or for $\mathbf{A}=\mathbf{A}^*$ when an available symmetric solver requires the matrix to be positive definite, then (30) may be preferred over (5). Since (30) saves projections, it can use less total cputime than (5).

One advantage of JD over IIGD is that IIGD requires two equation-solves in the inner iteration, which is more expensive than the one equation-solve JD. Equality (21) suggests the following simplified IIGD, which requires only one equation-solve.

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

$$(\mathbf{A}-\mu\mathbf{I})\,\mathbf{t} = \mathbf{x}; \quad \text{then compute} \;\; \eta = 1/(\mathbf{x}^*\mathbf{t}),\; \mathbf{t}\leftarrow(\eta\mathbf{t}-\mathbf{x}). \tag{31}$$

This modification mainly exploits the fact that in exact solve for (6), $(\mathbf{A}-\mu\mathbf{I})^{-1}\mathbf{r}=\mathbf{x}$. It saves almost half the computational cost at each inner iteration of IIGD. A second look at (31) reveals that essentially it is approximate RQI with an additional scaled Gram-Schmidt step, the same scheme presented in observation *4.1(c)* of [33], except that (31) adds a Gram-Schmidt step. The above observation to (31) shows that, in exact solve case, IIGD (6) is also essentially RQI with a scaled Gram-Schmidt step to make $\mathbf{t}\perp\mathbf{x}$. Related remarks may be found in [35, 33]. One difference between (6) and the JD type method (30) is that the projection $(\mathbf{I}-\mathbf{x}\mathbf{x}^*)$ is applied to the solution of the RQI equation for IIGD, while for (30) it is applied to $(\mathbf{A}-\mu\mathbf{I})$ before the system solve. The latter has advantage because of better conditioning on the $\mathbf{x}^{\perp}$ subspace.

In very crude approximate solve cases, approximate RQI is known not to be better than (5). We present (31) in order to see how well approximate RQI performs when the approximate solve becomes more accurate. Numerical results show that actually (31) performs well with increasingly accurate system solve by GMRES. (One example in [33, 35] shows that the approximate RQI without the scale Gram-Schmidt step performs well when the linear-solves become quite accurate.) But this good performance is not observed if the linear solver is

BICGSTAB. The reason may be that GMRES is less sensitive to ill conditioning. As for JD type equation (5) or (30), there is no as big difference between GMRES and BICGSTAB as (31).

In the large scale setting, the correction equations are often solved by preconditioned iterative methods, so we briefly comment on preconditioned forms of correction equations. As mentioned in the Introduction, preconditioned methods for linear systems can be applied to the correction equations (e.g.[33, 15, 34, 37]). We use (31) as a simple example: If at step j we have a good preconditioner $\mathbf{K}_j$ for $(\mathbf{A} - \mu\mathbf{I})$, then we solve $\mathbf{K}_j\mathbf{t} = \mathbf{x}$ and compute $\eta = \frac{1}{\mathbf{x}^*\mathbf{t}}$. Then the updating vector can be obtained by $(\eta\mathbf{t} - \mathbf{x})$, which is orthogonal to $\mathbf{x}$ — the wanted direction in order to augment the projection basis. The straightforward preconditioned form of (6) would require solving two preconditioned equations to get the orthogonalization constant η ($\eta = \frac{\mathbf{x}^*\mathbf{K}_j^{-1}\mathbf{r}}{\mathbf{x}^*\mathbf{K}_j^{-1}\mathbf{x}}$), unless fixed preconditioners are used. Equation (31) reduces the two system-solves to one system-solve, even for the preconditioned formulation. The problem is that in the approximate solve scenario, $\mathbf{K}_j^{-1}\mathbf{r}$ can contain useful information that enhance the approximate RQI direction $\mathbf{K}_j^{-1}\mathbf{x}$. As pointed out in [33], JD offers the better combination of both directions by solving only one equation at each iteration. This is gained by using a well designed correction equation. Clearly, schemes represented by (25) also offer better combination of both directions by only one equation-solve per iteration; preconditioned forms based on (25) can offer a similar advantage. By working with better conditioned matrices, the important RQI direction may be approximated better via an iterative equation solver.

6. Numerical Results and Discussions

The purpose of this section is to numerically show the importance of the approximate RQI direction on the efficiency of the different schemes discussed. Because of the locally fast convergent nature of these methods, for the numerical tests we first compute the required eigenvector $\mathbf{v}_1$ by Matlab built-in function *eigs*, then perturb $\mathbf{v}_1$ as $\mathbf{v}_1 \leftarrow \mathbf{v}_1 + pert * rand(n, 1)$ and use the same perturbed vector as the initial vector. The value of *pert* and the mode of the sought eigenvalues are printed on each plot. (For approaches that try to improve the global convergence, interested readers are referred to [22, 15, 4, 3, 45].)

The linear solvers used are the Matlab built-in *gmres.m* and a modified BICGSTAB code based on the *bicgstab.m* for [19]. This *bicgstab.m* does two matrix-vector products at each of its iteration step. The Matlab built-in *bicgstab.m* returns a zero vector far too often when the maximum iteration number is set small, hence it is not used for the tests. We note that the Matlab built-in *gmres.m* is more fine tuned than the modified *bicgstab.m* used.

The Matlab code used for the tests is more sophisticate than Algorithm 2.1. We implemented both restart and deflation. For all the numerical tests, the outer iteration tolerance (the ϵ in Algorithm 2.1) is set to 10^{-12}.

All numerical tests are performed using Matlab version 7.0.1 (R14) on an Intel-xeon PC (2G RAM, dual 3G Hz cpu), the OS is Debian Linux with Kernel version 2.6.10.

In approximate solve cases, IIGD is not better than JD, hence in the first two numerical tests, we mainly compare JD (5), JDm (30) and IIGDm (31).

The first two tests are on silicon quantum dot models $Si_{10}H_{16}$ and $Si_{34}H_{36}$ from material science. The matrices correspond to silicon atoms passivated by hydrogen atoms. They are

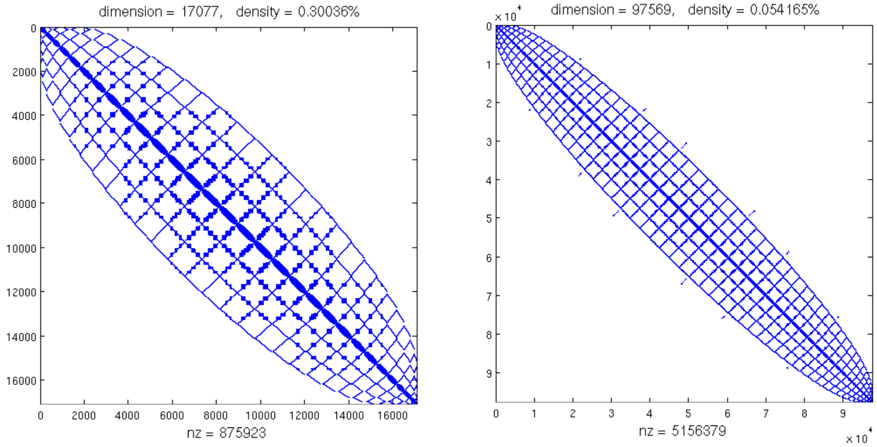

Figure 1. Sparsity structures of $Si_{10}H_{16}$ (left) and $Si_{34}H_{36}$ (right).

obtained from *ab initio* calculations after the self-consistency is reached. Both of them are symmetric real. Figure 1 shows their sparsity structures.

The maximum subspace dimension is set to 20 for these two tests. When the subspace dimension exceeds 20, the outer loop restarts from a subspace of dimension 6.

Figure 2 shows the convergence behavior of one eigenpair for JD (5), JDm (30) and IIGDm (31) solved by GMRES and BICGSTAB for $Si_{10}H_{16}$. The maximum inner iteration number (*itmax*) is set to 5, 8, 11 and 17, with inner iteration tolerance 10^{-3} (this is the *tol* passed to a linear solver). We observe no difference for JD and JDm by GMRES; while there are noticeable difference between JD and JDm by BICGSTAB (we note that the *bicgstab.m* used is not a sophisticated and stable enough implementation of BICGSTAB, we present the results by this *bicgstab.m* only as a reference to the Matlab builtin *gmres.m*); for IIGDm, there is always a big difference between solves by GMRES and solves by BICGSTAB, which means that JD and JDm are more stable than IIGDm. But we note that IIGDm can outperform JD and JDm when the linear solves by GMRES become more accurate.

Figure 3 shows the convergence behaviors of JD (5), JDm (30) and IIGDm (31) solved by GMRES and BICGSTAB for $Si_{34}H_{36}$. The *itmax* is set to 8, 11, 14 and 20, with fixed linear solve tolerance 10^{-4}. Again we see no difference for JD and JDm by GMRES. And again IIGDm by GMRES performs as well as JD and JDm with increasingly accurate linear solves. The good performance of IIGDm by GMRES is one example to show the importance of the approximate RQI direction.

The rest of the tests numerically compare the *inflated Newton*, the *constrained Newton* and their modifications with the JD-type methods (5) and (30). Tests are performed on nonsymmetric matrices as well as on symmetric ones. Two of the matrices are from material science, they correspond to $GaAsH_6$ and Si_5H_{12}. Their sparsity structures are shown in Figure 4. These two matrices are very close to being symmetric. When the matrices are nonsymmetric, we compute a partial Schur decomposition with eigenvalues satisfying a specified mode.

GMRES is used as the linear solver for all the remaining tests. We present results on

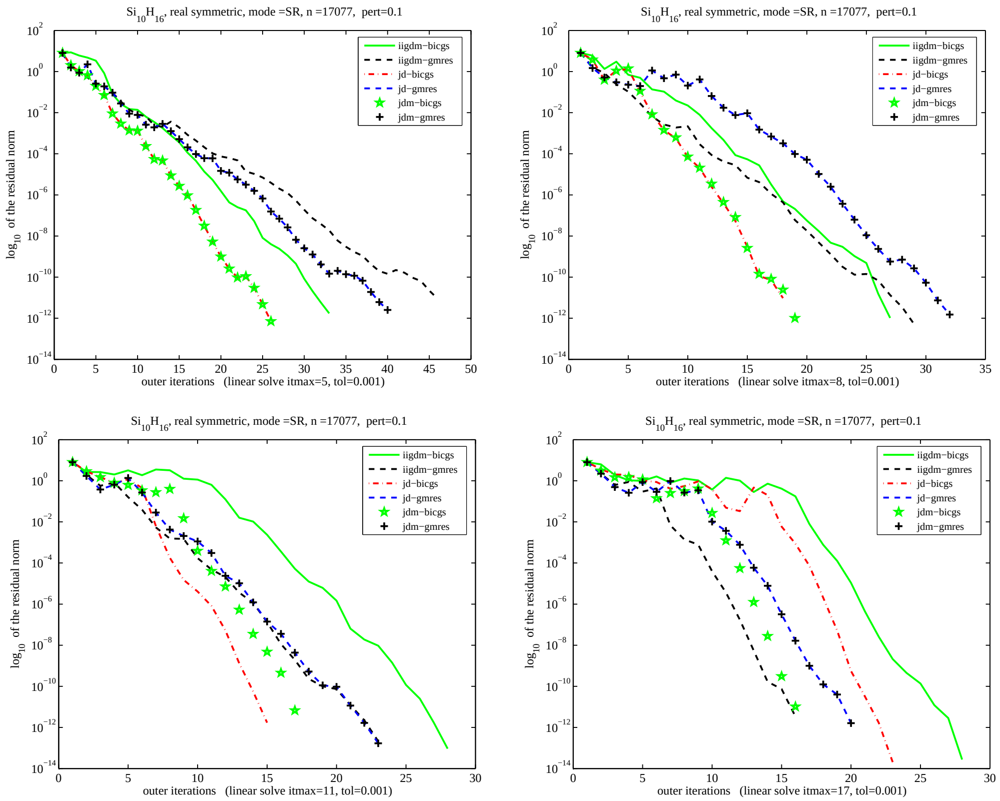

Figure 2. Residual reduction of Model $Si_{10}H_{16}$, $n = 17017$

computing multiple eigen/Schur pairs. Both restart and deflation are applied, so that the results would represent the efficiency of the tested schemes in more realistic situations.

In Figures 5–8, the legends signal the following: the `'inflate'` denotes the *inflated Newton* (17); the `'random'` denotes (25) with $\mathbf{y}$ being set to a new random vector at each matrix-vector product of the inner iteration; the `'cons'` denotes the *constrained Newton* (16); the `'consm'` denotes (29), we actually simplify it into

$$[\mathbf{A} - \mu\mathbf{I} + \alpha\mathbf{x}\mathbf{x}^*\mathbf{A}]\mathbf{t} = -\mathbf{r}, \tag{32}$$

with $\alpha \neq -2$; and the `'jd'` and `'jdm'` refer to JD and JDm respectively as before. We also experiment with the following equation

$$[\mathbf{A} - \mu\mathbf{I} + \mathbf{x}(\mathbf{r} + \beta\mathbf{x})^*]\mathbf{t} = -\mathbf{r}, \tag{33}$$

where $\beta = \mu$ if $|\mu| > 1$, and $\beta = 1$ otherwise. Clearly (33) is just another special case of (25), we use `'rpx'` to denote (33). Setting $\mathbf{y} = \mathbf{A}(\mathbf{r} + \beta\mathbf{x})$ in (25) leads to another equation, with similar behavior to (33), so only (33) is used for the tests. Noticing that $\mathbf{A}\mathbf{x} = \mathbf{r} + \mu\mathbf{x}$, we see that (33) is $[\mathbf{A} - \mu\mathbf{I} + \mathbf{x}\mathbf{x}^*\mathbf{A}^*]\mathbf{t} = -\mathbf{r}$ when $|\mu| > 1$.

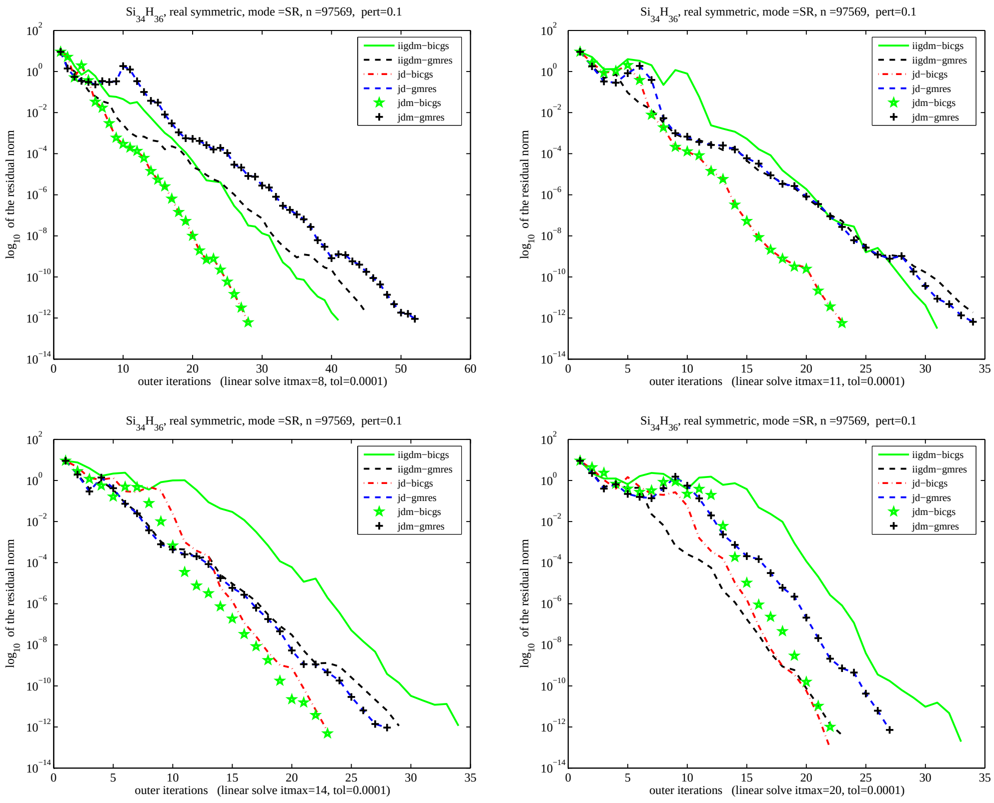

Figure 3. Residual reduction of Model $Si_{34}H_{36}$, $n = 97569$.

Figure 5 shows the residual reduction plot for the $GaAsH_6$ model. We see that the `'random'` scheme is as efficient as other schemes in computing the first two eigenvalues, but it becomes less efficient computing the rest 3 eigenvalues. Figure 6 shows the residual plot for the Si_5H_{12} model. We see that the gap between `'random'` and the other schemes shrinks when the linear solves become more accurate.

Tables I–II contain other two comparisons of the schemes on computing a size-10 eigen/Schur decomposition. The maximum subspace dimension used is 30 in both cases; and the outer loop starts from a subspace of dimension 15 after restart. The results of `'rpx'` is not reported (to save space in the table), but we note that for both of these models `'rpx'` is slightly worse than the other schemes except `'random'`.

A common feature seen in Tables I–II is that under-solving or over-solving the correction equations is not cost-effective. And over-solving usually can be rather expensive in terms of cputime, although it uses less total iteration steps.

We performed many other test runs on different type of matrices with different parameters, and we observed similar phenomena to what are presented here.

Now we spend some discussion on the `'random'` scheme. This scheme is chosen mainly for

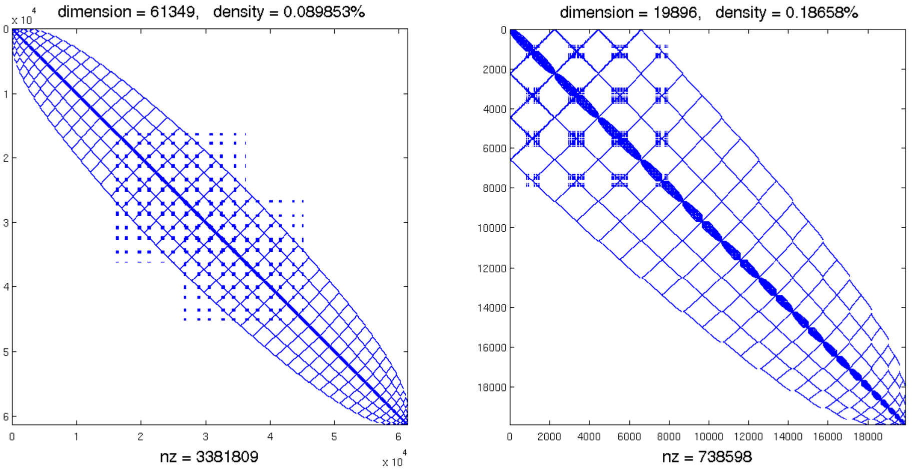

Figure 4. Sparsity structures of $GaAsH_6$ (left) and Si_5H_{12} (right).

itmax	inflate	jd	jdm	cons	consm	random
9	344.8 (250)	356.6 (254)	348.2 (253)	1607.2 (253)	361.9 (263)	775.8 (546)
12	331.3 (185)	351.5 (193)	336.3 (188)	1501.3 (185)	336.8 (186)	586.1 (315)
15	369.3 (162)	388.5 (168)	371.1 (164)	1589.8 (160)	362.7 (160)	570.7 (245)
18	383.1 (137)	377.3 (133)	378.9 (136)	1654.9 (139)	379.1 (136)	555.9 (195)
21	399.3 (119)	395.8 (117)	389.1 (117)	1585.2 (116)	391.4 (117)	510.7 (149)
24	404.3 (102)	409.2 (102)	404.1 (102)	1674.3 (107)	418.5 (105)	499.8 (122)
27	438.3 (94)	438.9 (93)	439.1 (94)	1710.6 (97)	442.8 (95)	538.5 (112)

Table I. CPU seconds of different correction equations solved by *GMRES* for Model $GaAsH_6$. The digits in the parenthesis are total number of outer iterations for the size-10 Schur decomposition with 'SR' mode to converge. The computed eigenvalues (keeping 9 digits) are: -1.14249099e+00, -9.10538900e-01, -6.69170075e-01, -6.69170075e-01, -5.70362441e-01, -5.70362441e-01, -5.22600834e-01, -7.75795983e-02, -4.35314664e-02, -5.99235789e-03. The cputime of 'cons' is larger because of the A^* term, but we note that with more careful programming, the cost of each 'cons' step can be made similar to that of 'consm' (32), even in the nonsymmetric case.

itmax	inflate	jd	jdm	cons	consm	random
9	447.7 (208)	452.4 (207)	441.0 (206)	437.8 (205)	418.9 (196)	791.0 (358)
12	441.2 (157)	443.7 (155)	443.9 (158)	441.8 (157)	449.6 (160)	667.0 (230)
15	457.9 (128)	473.2 (130)	468.8 (131)	465.5 (130)	494.6 (138)	629.8 (171)
18	510.4 (115)	523.5 (116)	518.9 (117)	541.6 (122)	541.6 (122)	632.3 (139)
21	548.4 (102)	547.6 (100)	544.1 (101)	586.4 (109)	636.3 (118)	655.7 (119)
24	641.8 (100)	604.1 (93)	596.2 (93)	622.6 (97)	701.0 (109)	714.9 (109)
27	718.7 (95)	720.4 (94)	711.0 (94)	748.9 (99)	703.7 (93)	804.7 (105)

Table II. CPU seconds of different correction equations solved by *GMRES* for Model $Si_{34}H_{36}$. The digits in the parenthesis are total number of outer iterations for the 10 eigenpairs with 'SA' mode to converge. The computed eigenvalues (keeping 9 digits) are: -1.15862684, -1.12438398, -1.12438283, -1.12438126, -1.07704853, -1.07704726, -1.06245140, -1.06245056, -1.06245016, -1.01335308.

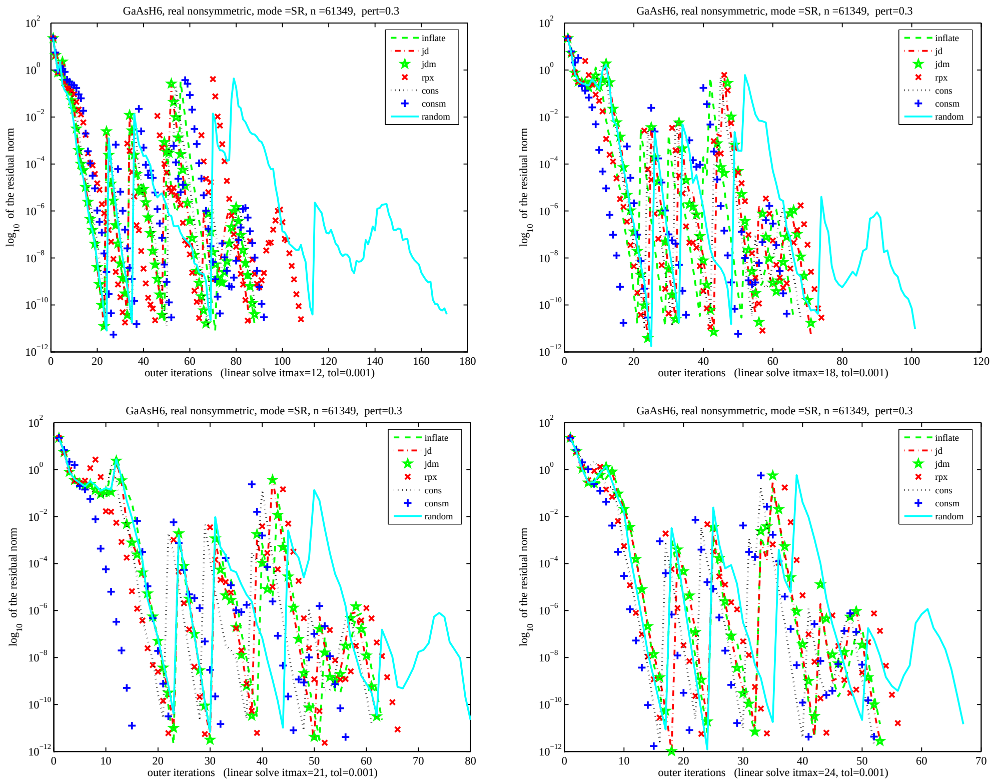

Figure 5. Residual reduction of Model $GaAsH_6$, $n = 61349$. The size-5 partial Schur decomposition with 'SR' eigenvalue mode are computed.

numerical experiments. As seen from Figures I–II, `'random'` works fine for computing the first eigen/Schur pair. Similar phenomenas are observed from many other tests (including those in Tables I – II, although it can not been seen from the total iteration steps). But `'random'` can become much less efficient than other schemes when more eigenpairs are computed, especially when the solves are crude. Clearly one can not expect a scheme that uses a random vector at each step of iteration to be competitive; a subspace method that uses a random vector for augmentation generally will not converge. What is different here is that using a random vector $\mathbf{y}$ in (25) always leads to convergence for all the tests we performed; and the other schemes with known fast convergent rate seldom converge twice faster than `'random'`. We consider this another good example to show the importance of the approximate RQI direction. One reason that `'random'` becomes less efficient in computing more eigenpairs may be related to reusing information, i.e., it may not use the information that have been built into the subspace when converging other eigenpairs as efficiently as other schemes. Another more probable reason may be that when the linear solves are crude, the wanted RQI-direction is not well approximated because of higher noise in the correction term ($\alpha\mathbf{x}\mathbf{y}^*$ to $\mathbf{A} - \lambda\mathbf{I}$); this seems to explain why the difference between `'random'` and the other schemes become smaller when the linear solves

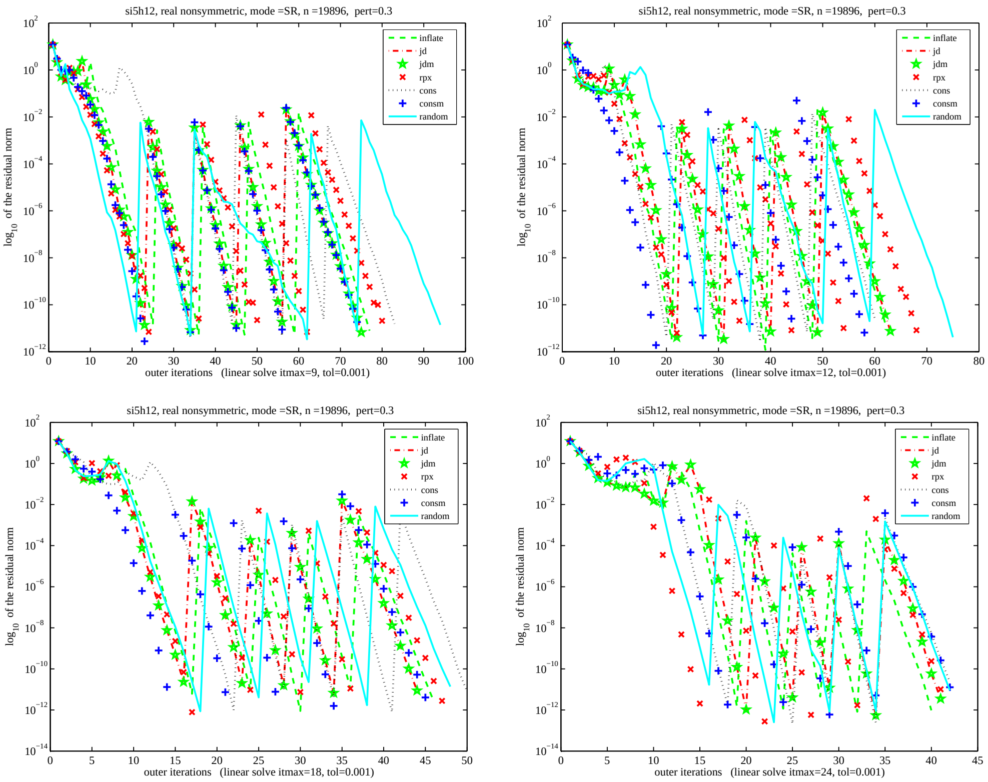

Figure 6. Residual reduction of Model Si_5H_{12}, $n = 19896$. The size-5 partial Schur decomposition with 'SR' eigenvalue mode are computed.

become more accurate.

The analysis in Section 4 shows that, in exact arithmetic, the essence of the discussed schemes to be efficient is to retain the RQI direction in the subspace; and all these schemes may be considered equivalent in this sense. Moreover, the numerical results in this section show that, in rounded arithmetic, with approximate equation solves, and with restart and deflation, these schemes (except the `'random'` one) also have rather similar efficiency. We consider this as a good agreement between theory and numerical schemes, where roundoff errors should subject to well designed numerical schemes based on correct theory.

A rough conclusion is that none of the discussed schemes is a clear winner; one scheme may be more efficient than the others for a certain type of problems. (This agrees with a similar conclusion in [44].) Moreover, the linear solvers used can also affect the efficiency of a scheme. We note that in [14] where the linear solver is MG and the eigenproblems are from structure dynamics, the *inflated Newton* (17) scheme is reported to have better performance than JD. As for the problems tested here, such a consistent advantage is not observed. But we agree that (17) is one of the best schemes represented by (25).

We draw some other conclusions from the numerical experiments: 1. Including the approximate RQI direction in the subspace is very important for the discussed Davidson-type methods to be efficient. 2. JD-type equations (5) and (30) are more stable than IIGD-type (31), in the sense that applying different linear solvers to them does not lead to drastic differences. 3. The simplified JDm (30), i.e., without the right projection for the matrix-vector product at each inner iteration and the final $\mathbf{t} \perp \mathbf{x}$ requirement, can be as stable and efficient as JD (5) that performs these projections. 4. The general formula (25) represents a large class of schemes that can be as efficient as JD-type methods.

Note that preconditioners are not used in the experiments, but the above conclusions should be valid for preconditioned equation solves, because approximate solves can be regarded as solving from preconditioned equations.

Finally, we report an interesting observation related to the *inflated Newton* scheme that is not noted in [44, 14]. For the above presented results and many other results not reported here, the value of α used is within 1–50, other values do not have any significant difference. But we observed quite significant difference between a small value α and a large value α for some matrices from the NIST matrix market. Two typical matrices are `olm1000` and `sherman3`, where eigenvalues with large magnitude are computed. As seem from Figures 7 and 8, a small α value leads to the much slower `'inflate'` and `'random'` which do not include information related to the current eigenvalue in the correction term; while schemes that use this information, (i.e., the $\mathbf{Ax}$ term in `'cons'` and `'consm'`, and the $\beta\mathbf{x}$ term in `'rpx'`) are not affected by the small α value. After increasing α to the magnitude of μ, all the schemes become similar again. This can be explained by Theorem 4.1 as follows: Because in both of these examples, the $\mathbf{x}_R$ is of small norm (in the order of $\frac{1}{|\mu|}$), so the $\mathbf{y}^*\mathbf{x}_R$ is also in the order of $\frac{1}{|\mu|}$ when $\mathbf{y}$ is set to $\mathbf{x}$ or a random vector. Note that the σ as defined in (27) satisfies $\frac{1}{\sigma} = \frac{\alpha}{\alpha\mathbf{y}^*\mathbf{x}_R - 1}$, which is now in the order of α. Therefore, the $\frac{\mathbf{y}^*\mathbf{x}}{\sigma}\mathbf{x}_R$ term in (28) is in the order of $\frac{\alpha}{\mu}$, which is insignificant comparing with the unit length $\mathbf{x}$ when α is small. An α comparable to μ strengthens the RQI direction in (28), hence improves the convergence speed. Once again we see the importance of the approximate RQI direction. And these two examples show that in rounded arithmetic, one needs to include a numerically significant approximate RQI direction for a Davidson-type scheme to be efficient.

7. Concluding remarks

We studied Jacobi-Davidson, IIGD and several schemes based on the Newton method for eigenvalue computations. We analysed that being able to keep the RQI direction in the solution of a correction equation constitutes the essence of efficiency for a diverse form of Davidson-type methods. The left projector $(\mathbf{I} - \mathbf{xx}^*)$ is found to be essential for Jacobi-Davidson to retain the approximate RQI direction. A simplified Jacobi-Davidson is proposed based on this study. A general formulation which has more freedom in keeping the approximate RQI direction is constructed. Numerical results confirmed that the approximate RQI direction at each outer iteration is crucial for the efficiency of the studied schemes.

ACKNOWLEDGEMENTS

The author gratefully acknowledges Gerard L.G. Sleijpen for his many insightful comments and

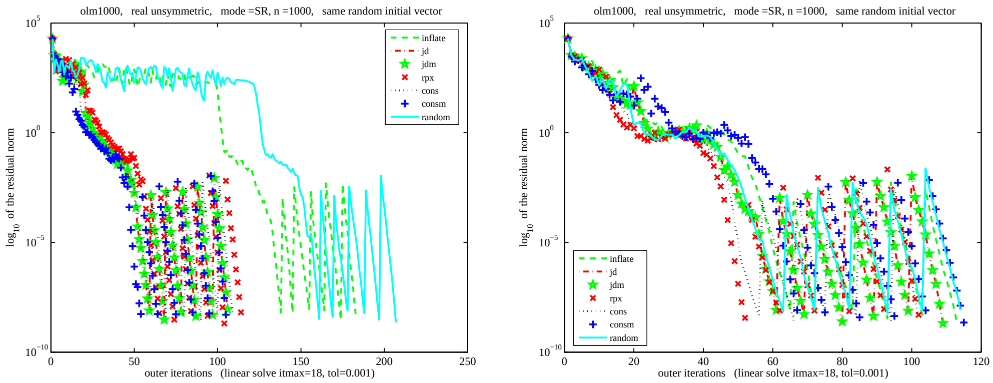

Figure 7. Residual reduction of `olm1000`. Linear solver used is GMRES. The six computed 'SR' eigenvalues are: -1.01633831e+04, -1.01630831e+04, -1.01625831e+04, -1.01618831e+04, -1.01609833e+04, -1.01598835e+04. Left picture, $\alpha = 3$; right picture, $\alpha = \mu$.

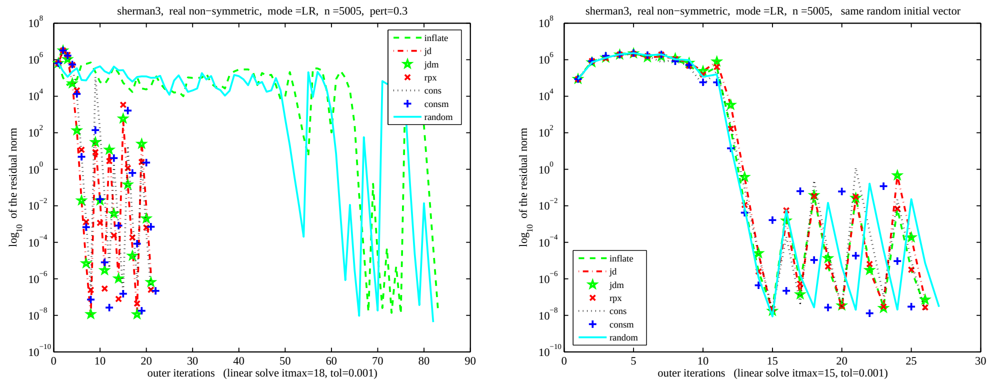

Figure 8. Residual reduction of `sherman3`, where $cond(\mathbf{A}) > 6e+16$. The five computed 'LR' eigenvalues are: 6.75844005e+06, 6.35064445e+06, 5.70779495e+06, 4.88197142e+06, 3.94007715e+06. Left picture, $\alpha = 3$; right picture, $\alpha = \mu$.

substantial contributions that significantly improve the presentation of this paper. The superb reviews from the referees are greatly acknowledged. Sincere thanks also go to Yousef Saad for helpful discussions and for his research support, and to Dan Sorensen for introducing the author to eigenvalue computations. Ron Shepard and Mike Minkoff at Argonne national Laboratory are acknowledged for interesting discussions. The valuable NSF travel support from the Institute of Pure and Applied Mathematics 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.

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. J. H. Brandts, *The Riccati method for eigenvalues and invariant subspaces of matrices with inexpensive action*, Linear Algebra and Its Applications, 358 (2003) pp. 333-363.
4. J. H. Brandts, *Solving eigenproblems: from Arnoldi via Jacobi-Davidson to the Riccati method*, Springer Lecture notes in computer science, 2542 (2003) pp. 167-173.
5. F. Chaitin-Chatelin, *Eigenvalues of Matrices*, John Wiley & Son, 1993.
6. M. Crouzeix, B. Philippe, M. Sadkane, *The Davidson Method,* SIAM Journal on Scientific Computing, 15 (1994) pp. 62-76.
7. J. K. Cullum, R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Vol. I: Theory*, SIAM, Classics in Applied Mathematics 41, 2002.
8. 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.
9. 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.
10. E. R. Davidson, *Monster Matrices: Their Eigenvalues and Eigenvectors*, Computers in Physics, 7 (1993) pp. 519-522.
11. A. Dax, *The Orthogonal Rayleigh Quotient Iteration Method*, Linear Algebra and Its Applications, 358 (2003) pp. 23-43.
12. J. E. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM classics in Applied Mathematics, 16, 1996.
13. J. Dongarra, I. Duff, D. C. Sorensen, H. A. van der Vorst, *Numerical Linear Algebra for High-Performance Computers*, SIAM publications, 1998.
14. Y. T. Feng, *An Integrated Multigrid and Davidson Method for Very Large Scale Symmetric Eigenvalue Problems,* Computational Methods in Applied Mechanics and Engineering, (190) 2001, pp. 3543-3563.
15. 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.
16. D. R. Fokkema, G. L. G. Sleijpen, H. A. van der Vorst, *Accelerated Inexact Newton Schemes for Large Systems of Nonlinear Equations,* SIAM Journal on Scientific Computing, 19 (1998) pp. 657-674.
17. G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 3rd ed., 1996.
18. M. E. Hochstenbach, G. L. G. Sleijpen, *Two-sided and Alternating Jacobi-Davidson,* Linear Algebra and Its Applications, 358 (2003) pp. 145-172.
19. C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Frontiers in Applied Mathematics 16, 1995.
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*, 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, *Computation of eigenvalues and vectors*, in Sparse Matrix Techniques, V.A. Barker, ed., Lecture Notes in Mathematics, 572 (1977) pp. 130-184.
30. A. Ruhe, *Rational Krylov Sequence Methods for Eigenvalue Computations*, Linear Algebra and Its Applications, 58 (1984) pp. 391-405.
31. Y. Saad, *Numerical Methods for Large Eigenvalue Problems,* John Wiley, 1992. `http://www-users.cs.umn.edu/~saad/books.html`.
32. 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.

33. 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.
34. 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.
35. 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.
36. 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.
37. D. C. Sorensen, *Numerical Methods for Large Eigenvalues Problems*, Acta Numerica, (2002) pp. 519-584.
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. 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.
43. J. H. Wilkinson, *The Algebraic Eigenvalue Problem,* Oxford University Press, 1965.
44. 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.
45. Y. Zhou, Y. Saad, *A Chebyshev-Davidson algorithm for large symmetric eigenproblems,* Submitted to SIAM Journal on Matrix Analysis and Applications.