LAPACK WORKING NOTE 163: HOW THE MRRR ALGORITHM CAN FAIL ON TIGHT EIGENVALUE CLUSTERS[§]

BERESFORD N. PARLETT[†] AND CHRISTOF VÖMEL[‡]

Technical Report UCB//CSD-04-1367
Computer Science Division
UC Berkeley

Abstract. In the 90s, Dhillon and Parlett devised a new algorithm (Multiple Relatively Robust Representations, MRRR) for computing numerically orthogonal eigenvectors of a symmetric tridiagonal matrix T with $\mathcal{O}(n^2)$ cost. It has been incorporated into LAPACK version 3.0 as routine STEGR.

We have discovered that the MRRR algorithm can fail in extreme cases. Roundoff error does not always come to rescue and clusters of eigenvalues can agree to working accuracy. In this paper, we describe and analyze these failures and various remedies.

Key words. Multiple relatively robust representations, numerically orthogonal eigenvectors, symmetric tridiagonal matrix, tight clusters of eigenvalues, glued matrices, Wilkinson matrices.

AMS subject classifications. 15A18, 15A23.

1. Introduction. In the 90s, Dhillon and Parlett devised a new algorithm (Multiple Relatively Robust Representations, MRRR) for computing numerically orthogonal eigenvectors of a symmetric tridiagonal matrix T with $\mathcal{O}(n^2)$ cost [10, 12, 13, 14, 15]. This algorithm was tested on a large and challenging set of matrices and has been incorporated into LAPACK version 3.0 as routine STEGR. The algorithm is described, in the detail we need here, in Section 2. For a more detailed description see also [6]. An example of MRRR in action is given in Section 3.

In 2003, one of us (Vömel) came to Berkeley to assist in the modification of STEGR to compute a subset of k eigenpairs with $\mathcal{O}(kn)$ operations. When testing STEGR on more and more challenging matrices, he discovered cases of failure.

Investigation of these cases brought to light assumptions made in STEGR that hold in exact arithmetic and in the majority of cases in finite precision arithmetic but can fail. These assumptions, why they are reasonable, and how they can fail, are the subject of Section 4.

In Section 5, we propose and analyze various remedies for the aforementioned shortcomings and show how to incorporate them into MRRR. We also look at the cost of these modifications. We select as most suitable an approach that is based on small random perturbations and introduces artificial roundoff effects. This approach preserves the complexity of the original algorithm.

2. The algorithm of Multiple Relatively Robust Representations. The MRRR algorithm [10, 12, 13, 14, 15] is a sophisticated variant of inverse iteration [9]. It addressed the challenge of computing numerically orthogonal eigenvectors of a symmetric tridiagonal matrix $T \in R^{n \times n}$ for eigenvalues that are very close to each

[†]Mathematics Department and Computer Science Division, University of California, Berkeley, CA 94720, USA. parlett@math.berkeley.edu

[‡]Computer Science Division, University of California, Berkeley, CA 94720, USA. voemel@eecs.berkeley.edu

[§]This work has been supported by a grant from the National Science Foundation (Cooperative Agreement no. ACI-9619020).

other. The difficulty of this task is that each computed eigenvector tends to be corrupted by components from its neighbors, leading to non-orthogonal vectors.

One way to overcome this obstacle is to achieve highly accurate eigenpairs satisfying

$$\|(T - \lambda_i I)v_i\| = \mathcal{O}(n\epsilon|\lambda_i|); \tag{2.1}$$

the eigenpairs have a *relatively* small residual norm with respect to $|\lambda_i|$. For a small eigenvalue λ, this is a much stronger demand on accuracy then the common condition

$$\|(T - \lambda_i I)v_i\| = \mathcal{O}(n\epsilon\|T\|). \tag{2.2}$$

(Throughout this paper, ϵ denotes the unit roundoff.)

2.1. Relatively Robust Representations. The first contribution of MRRR is the observation that for most shifts σ, the standard triangular factorization $T - \sigma I = LDL^T$, L unit bidiagonal, D diagonal, exists and has the property that small relative changes in the nontrivial entries of L and D cause small relative changes in each small eigenvalue of LDL^T (despite possible element growth in the factorization). This is in sharp contrast to the entries in $T - \sigma I$ which have this property only in special cases, see [1]. We call (L, D) a *Relatively Robust Representation (RRR)* when it determines a subset Γ of the eigenvalues to high relative accuracy Armed with an RRR, we can then compute an approximate eigenvalue to high relative accuracy. This can be done by bisection for a single eigenvalue λ with $\mathcal{O}(n)$ work, yielding $\hat{\lambda}$ satisfying

$$|\lambda - \hat{\lambda}| \le Kn\epsilon|\hat{\lambda}|, \tag{2.3}$$

where K is a modest constant independent of T and λ. Moreover, when the matrix is definite, then dqds may be used to compute all eigenvalues to this accuracy in $\mathcal{O}(n^2)$ operations.

2.2. The FP vector. The second innovative feature of MRRR is based on the fact that, for an accurate approximate eigenvalue $\hat{\lambda}$, it is possible to find the index r of the largest component of the true wanted eigenvector v using a double factorization

$$(LDL^T - \hat{\lambda}I) = L_+D_+L_+^T = U_-D_-U_-^T \tag{2.4}$$

with $\mathcal{O}(n)$ work. This was discovered in the mid 1990's independently by Godunov and by Fernando, see [12] and the references therein. From

$$\sin^2\angle(e_r, v) = 1 - \cos^2\angle(e_r, v) = 1 - v^2(r), \tag{2.5}$$

we see that e_r, the r-th column of the identity, cannot be a bad approximation to the true eigenvector. The algorithm solves

$$(LDL^T - \hat{\lambda}I)z = e_r\gamma_r, \ \ z(r) = 1, \tag{2.6}$$

with a normalization factor γ_r and $\gamma_r^{-1} = [(LDL^T - \hat{\lambda}I)^{-1}]_{rr}$. An eigenvector expansion [12] shows that

$$\frac{\|(LDL^T - \hat{\lambda}I)z\|_2}{\|z\|_2} = \frac{|\gamma_r|}{\|z\|_2} \le \frac{|\lambda - \hat{\lambda}|}{|v(r)|}. \tag{2.7}$$

In particular, if r is chosen such that $|v(r)| \geq 1/\sqrt{n}$, then an upper bound on (2.7) is $\sqrt{n}\,|\lambda - \hat{\lambda}|$. Now (2.7) together (2.3) with yields (2.1), a relatively small residual norm. In [12], the resulting z from (2.6) is called the *FP vector*, FP for Fernando and Parlett. The reward for the relatively small residual norm of the FP vector is revealed by the classical gap theorem, often attributed to Davis and Kahan, see [12]. It shows that

$$|\sin\angle(z,v)| \leq \frac{\|LDL^T z - z\hat{\lambda}\|_2}{\|z\|_2 \operatorname{gap}(\hat{\lambda})} \leq \frac{|\lambda - \hat{\lambda}|}{\|v\|_\infty \operatorname{gap}(\hat{\lambda})}, \tag{2.8}$$

where $\operatorname{gap}(\hat{\lambda}) = \min\left\{|\hat{\lambda} - \mu| : \lambda \neq \mu, \mu \in \operatorname{spectrum}(LDL^T)\right\}$. By (2.3), we thus obtain

$$|\sin\angle(z,v)| \leq \frac{Kn\epsilon|\hat{\lambda}|}{\|v\|_\infty \operatorname{gap}(\hat{\lambda})} = \frac{Kn\epsilon}{\|v\|_\infty \operatorname{relgap}(\hat{\lambda})}, \tag{2.9}$$

defining the relative gap, $\operatorname{relgap}(\hat{\lambda})$. Thus, when $\hat{\lambda}$ and $\operatorname{gap}(\hat{\lambda})$ are of not too different size, the angle between the FP vector and the eigenvector is $\mathcal{O}(n\epsilon)$.

2.3. The representation tree and the differential qd algorithm. The third new idea of the MRRR algorithm is to use *multiple representations* in order to achieve large relative gaps. By large we mean larger than a threshold, usually $1.e-3$. If the relative gaps of a group of eigenvalues are too small, then they can be increased by shifting the origin close to one end of the group. Furthermore, the procedure can be repeated for clusters within clusters of close eigenvalues. The tool for computing the various RRRs is the stationary *differential qd algorithm.* It has the property that tiny relative changes to the input (L, D) and the output (L_+, D_+) with $LDL^T - \sigma I = L_+D_+L_+^T$ give an exact relation. The MRRR procedure for computing eigenpairs is best represented by a rooted tree. Each node of the graph is a pair consisting of a representation (L, D) and a subset Γ of the wanted eigenvalues for which it is an RRR. The root node of this *representation tree* is the initial representation that is an RRR for all the wanted eigenvalues, each leaf corresponds to a singleton, a (shifted) eigenvalue whose relative separation from its neighbor exceeds a given threshold.

2.4. Twisted factorizations. Another ingredient of the MRRR algorithm is the way the FP vector is computed. One uses a *twisted factorization* to solve (2.6) as follows. With the multipliers from (2.4), let

$$N_r = \begin{pmatrix}
1 &&&&&&&&& \\
L_+(1) & 1 &&&&&&&& \\
& \ddots & \ddots &&&&&&& \\
&& L_+(r-2) & 1 &&&&&& \\
&&& L_+(r-1) & 1 & U_-(r) &&&& \\
&&&&& 1 & U_-(r+1) &&& \\
&&&&&& 1 & \ddots && \\
&&&&&&& \ddots & U_-(n-1) & \\
&&&&&&&& 1 &
\end{pmatrix},$$

and, with the pivots from (2.4) and γ_r from (2.6), define

$$\Delta_r = \operatorname{diag}(D_+(1), \ldots, D_+(r-1), \gamma_r, D_-(r+1), \ldots, D_-(n)).$$

Then the twisted factorization at $\hat{\lambda}$ (with twist index r) is given by

$$LDL^T - \hat{\lambda} I = N_r \Delta_r N_r^T. \tag{2.10}$$

Since $N_r e_r = e_r$ and $\Delta_r e_r = \gamma_r e_r$, the solution of (2.6) is equivalent to solving

$$N_r^T z = e_r, \;\; z(r) = 1. \tag{2.11}$$

The great attraction of using this equation is that z can be computed solely with multiplications, no additions or subtractions are necessary. By (2.11), we obtain

$$z(i) = \begin{cases} -L_+(i) z(i+1) & , \; i = r-1, \ldots, 1, \\ -U_-(i-1) z(i-1) & , \; i = r+1, \ldots, n. \end{cases} \tag{2.12}$$

This procedure together with the use of the differential qd algorithms, permits the error analysis that shows that roundoff does not spoil the overall procedure [15].

2.5. Putting it all together: the MRRR algorithm. Combining all the ideas from the previous sections, we obtain the following algorithm:

Algorithm 1 The MRRR algorithm.

```
for the current level of the representation tree do
  for each eigenvalue with a large relative gap do
    Compute an approximation that is good to high relative accuracy.
    Compute its FP vector.
  end for
  for each of the remaining groups of eigenvalues do
    Choose shift σ close to the outside of the group.
    Compute new RRR L+D+L+^T = LDL^T − σI.
    Refine the eigenvalues.
  end for
  Proceed to the next level of the representation tree.
end for
```

3. The MRRR algorithm in practice. In [17], Wilkinson introduced two classes of symmetric tridiagonal matrices W^+_{2m+1} and W^-_{2m+1} for positive integers m. W^+_{2m+1} is more interesting for our purposes. It is defined as

$$W^+_{2m+1} = \text{tridiag}\begin{pmatrix} & 1 & & 1 & & \ldots & & 1 & & 1 & \\ m & & m-1 & & \ldots & 0 & \ldots & & m-1 & & m \\ & 1 & & 1 & & \ldots & & 1 & & 1 & \end{pmatrix}$$

We want to to illustrate the action of MRRR on W^+_{21}. The algorithm first computes a shift σ so that $T - \sigma I$ becomes definite, then all eigenvalues of the root representation $LDL^T = T - \sigma I$ are computed by dqds. In Table 3.1 we show the eigenvalues of W^+_{21} and of the root representation whose shift is $\sigma = 10.746194215443904$, a computed upper bound on the largest eigenvalue of T (hence LDL^T is negative definite). The relative gaps are with respect to the root representation.

MRRR uses a threshold to decide which relative gaps are large enough, currently $1.E-3$. (That is, eigenvalues that agree to fewer than three digits are declared relatively isolated.) The last column of Table 3.1 shows that the relative gaps of the first nine eigenvalues exceed the threshold and thus are singletons for the algorithm. The corresponding FP vectors are computed immediately. The remaining eigenvalues

Index	$\lambda_i(T)$	$\lambda_i(LDL^T)$	$\lvert\lambda_{i+1}-\lambda_i\rvert$	$\lvert\lambda_{i+1}-\lambda_i\rvert/\lvert\lambda_i\rvert$
1	-1.125441522119984	-11.87163573756389	1.379	0.116
2	0.253805817096679	-10.49238839834723	0.694	0.066
3	0.947534367529293	-9.79865984791461	0.842	0.086
4	1.789321352695081	-8.95687286274882	0.341	0.038
5	2.130209219362507	-8.61598499608140	0.831	0.096
6	2.961058884185726	-7.78513533125818	0.082	0.011
7	3.043099292578824	-7.70309492286509	0.953	0.124
8	3.996048201383624	-6.75014601406028	8.306E-3	1.230E-3
9	4.004354023440857	-6.74184019200305	0.995	0.148
10	4.999782477742902	-5.74641173770100	4.619E-4	8.039**E-5**
11	5.000244425001912	-5.74594979044199	1.000	0.174
12	6.000217522257097	-4.74597669318681	1.651E-5	3.479**E-6**
13	6.000234031584167	-4.74596018385974	1.004	0.211
14	7.003951798616375	-3.74224241682753	4.109E-7	1.100**E-7**
15	7.003952209528675	-3.74224200591523	1.035	0.277
16	8.038941115814273	-2.70725309962963	7.015E-9	2.591**E-9**
17	8.038941122829025	-2.70725309261488	1.172	0.433
18	9.210678647304919	-1.53551556813899	5.641E-11	3.673**E-11**
19	9.210678647361332	-1.53551556808257	1.535	0.100
20	10.746194182903322	-3.25405817949E-8	7.160E-14	2.200**E-6**
21	10.746194182903393	-3.25405101953E-8		

TABLE 3.1
Eigenvalues, right gaps, and relative right gaps of the negative definite root representation of W_{21}^+.

come in pairs which are well isolated from each other. For each of these pairs, the algorithm finds a close shift, computes a new RRR, $LDL^T - \tau I = L_+D_+L_+^T$, and refines the eigenvalues of that new RRR. In Table 3.2, we show these refined local eigenvalues with respect to their new RRRs. Note that the relative gaps inside the pairs have been significantly improved over those of the root representation (they are all of order 1) because $|\lambda_{2i+1} - \lambda_{2i}| \approx |\lambda_{2i}|,\ i = 5, \dots, 10$. With respect to the appropriate representation, each eigenvalue is now a singleton. In Figure 3.1, we show the representation tree associated to these computations.

Index	local λ ($L_+D_+L_+^T$)	$\lvert\lambda_{i+1}-\lambda_i\rvert/\lvert\lambda_i\rvert$	τ (shift of new RRR)
10	-4.619472590129691E-4	1.0	-5.745949790441989
11	-1.821136708729214E-15		
12	-1.6509327074115723E-5	1.0	-4.745960183859731
13	-5.239123482043716E-15		
14	-4.109123007352570E-7	1.0	-3.742242005915228
15	-3.071368719442505E-17		
16	-7.014749992198161E-9	1.0	-2.7072530926148803
17	-6.432284272916864E-17		
18	-5.641544663900284E-11	1.0	-1.5355155680825696
19	-1.932676195102966E-15		
20	-7.159957809285440E-14	1.0	-3.25405101953065E-8
21	-9.193866613291496E-23		

TABLE 3.2
Eigenvalues and relative right gaps of the respective children of the root representation of W_{21}^+. The last column shows the (incremental) shift τ of the respective representation with respect to the shift of the root representation.

4. Fallible assumptions. In this section, we show how MRRR can fail. Our illustrations use (large) Wilkinson and glued Wilkinson matrices. By a glued (sym-

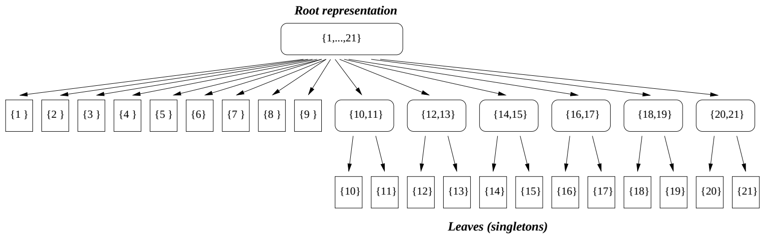

FIG. 3.1. *The representation tree for W_{21}^+. Square boxes correspond to singletons, boxes with round corners correspond to eigenvalue groups for which an individual representation is needed to improve relative gaps.*

metric tridiagonal) matrix, we denote a matrix

$$T(\gamma) = \left[\begin{array}{c|c|c} T_1 & & \\ \hline & \ddots & \\ \hline & & T_p \end{array}\right] + \gamma \sum_{i=1}^{p-1} v_i v_i^T \quad ,$$

where v_i is a vector with only two nonzero entries of size 1 in the positions corresponding to the last row of T_i and the first row of T_{i+1} in T, where $T_1, \ldots, T_p$ are symmetric tridiagonal matrices.

4.1. Finding large relative gaps. ASSUMPTION 4.1. *By suitable shifting, at least one eigenvalue in a tight cluster of eigenvalues of an unreduced symmetric tridiagonal matrix can be given a relative gap exceeding any given threshold.*

In exact arithmetic, the eigenvalues of an unreduced symmetric tridiagonal matrix are simple. Thus, a shift to either end of a cluster of close eigenvalues yields a local eigenvalue that is (nearly) zero and a relative gap of (almost) ∞. So at least one local eigenvalue becomes a singleton.

When working in finite precision with roundoff, we expect that a tight cluster, however close in exact arithmetic, will be computed with differences exceeding $\epsilon\lambda_i$. By shifting close to the end of a cluster, we can expect the relative gaps to become large enough. A typical example was shown in Table 3.2. For example, an incremental shift of -3.25E-8 from the root representation to the RRR for eigenvalues $20, 21$ increased the relative gap of eigenvalue 20 to 1 since $|\lambda_{21} - \lambda_{20}| \approx |\lambda_{20}|$. For matrices with tighter clusters, more steps of shifting and gradual refinement of the eigenvalues might be needed to increase the relative separation.

In the following experiment, we look at the second-smallest cluster of a matrix T obtained from five copies of W_{201}^+ glued together by $\sqrt{\epsilon}$. According to the strategy of finding large relative gaps, we refine the extremal eigenvalues of the cluster to high accuracy. Then we compute LDL^T factorizations at both ends of the cluster and choose the end with the smaller element growth for the representation. If the code performed as usual, by shifting closer and closer to the cluster, we would see for a certain shift the cluster splitting up into subgroups and singletons. Instead, we see a refinement of the eigenvalues of the cluster down to the underflow threshold without being able to find subclusters. In terms of the representation tree, this corresponds to a long chain and had not been observed in former tests. In Table 4.1, we show for some parts of the chain the extremal eigenvalues of the cluster together with the corresponding depth of the representation tree. An explanation of this phenomenon is given in Section 4.3.

Index	Level 1	Level 2	Level 10	Level 21
6	1.0524021997591E-13	-7.6189638190266E-30	3.83337214181111E-149	1.14585459648E-312
7	1.0524021997591E-13	-7.6189638190266E-30	3.83337214181111E-149	1.14585459649E-312
8	1.0524021997591E-13	-7.6189638190266E-30	3.83337214181111E-149	1.14585459651E-312
9	1.0524021997591E-13	-7.6189638190266E-30	3.83337214181111E-149	1.14585459653E-312
10	1.0524021997591E-13	-7.6189638190266E-30	3.83337214181111E-149	1.14585459654E-312

TABLE 4.1

(Local) eigenvalues of the second-smallest cluster of the glued matrix T by representation tree level. (We define the root representation as Level 0 and increase the level number when descending in the tree from a parent to a child.)

4.2. Finding an RRR. ASSUMPTION 4.2. *It is always possible to find an RRR close enough to one (or both) ends of a tight cluster to yield a singleton eigenvalue.* This statement differs from Assumption 4.1 by bringing in the property that the representation should be relatively robust.

In exact arithmetic, the triangular factorization of a matrix $T - \sigma I = LDL^T$ exists for all but a finite set of shifts σ, the so-called Ritz values of T. Each Ritz value is the eigenvalue of a leading principal submatrix. The element growth, that is $\|D\|$, is bounded except in a very small interval around these Ritz values.

It can be shown that the even-indexed eigenvalues of the Wilkinson matrices W_{2m+1}^+ are also Ritz values. In fact, they are the eigenvalues of the leading (and trailing) $m \times m$ submatrix, see [17]. This implies that an LDL^T factorization with such a shift will break down, or in other words: an RRR with such a shift does not exist. Luckily, for the simple Wilkinson matrices, it is usually not necessary to choose such a shift for a representation. For the well isolated eigenvalues at the lower end of the spectrum, one need not shift close to any Ritz value in order to obtain large relative gaps between the eigenvalues. At the upper end of the spectrum, the eigenvalues come in pairs consisting of two eigenvalues, the smaller having an even and the larger and odd index. In this case, one need not shift close to the left (Ritz-) eigenvalue, a shift to the right of the pair is excellent.

In order to find an RRR, the MRRR algorithm first factors at both ends of the cluster and monitors the element growth. If the element growth at one end is small, it is chosen as origin for the RRR. Otherwise, the shift backs off from the ends and tries again. The benefit from backing off is the possibility of smaller element growth, the danger is to obtain an RRR whose eigenvalues have small relative gaps.

In our experiment, we consider again the second-smallest cluster of five copies of W_{201}^+ glued together by $\sqrt{\epsilon}$. As seen in Table 4.1, the glue does not change the finite precision approximation of the true eigenvalues. Thus, in limited precision, we face an eigenvalue/Ritz value of multiplicity five. A representation in a small neighborhood cannot exist. Normally, we would back off enough to ensure modest element growth. However, since the clustered eigenvalues have zero gaps, backing off is pointless. Even if we found an RRR, it would not be suitable for the algorithm. Note that the problem does not occur in the (unglued) matrix but only in the glued case for a reason that will be explained in Section 4.3.

4.3. Why glued matrices are difficult for MRRR. Table 4.1 of Section 4.1 shows that the eigenvalues of the glued matrix are equal to working accuracy despite roundoff error. The following simple model can provide a good intuition of the phenomenon: Suppose that the glue connecting the copies were zero but the algorithm 'was not aware of it', that is, treated it as a non-splitting matrix. Then, *for all shifts*, there would be five equal computed eigenvalues. So, in practice, if there are eigenvalues for which the glue parameter has a negligible effect, then the algorithm will fail. This is exactly what happens in this case. In fact, it can be shown that the magnitude of the effect of the glue on eigenvalues is tied to top and bottom entries of the corresponding eigenvectors. As an intuitive argument, we remind the reader of Theorem 4.3 for the case of two glued matrices that plays an important role in the Divide & Conquer algorithm [2, 3, 7], a complete analysis of glued matrices is out of the scope of this paper, we refer to [16].

THEOREM 4.3. *Let* $T_1 = Q_1\Lambda_1 Q_1^T, T_2 = Q_2\Lambda_2 Q_2^T$ *and define*

$$D = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix}, \; u = \begin{bmatrix} Q_1^T & 0 \\ 0 & Q_2^T \end{bmatrix} v = \begin{bmatrix} \text{last column of } Q_1^T \\ \text{first column of } Q_2^T \end{bmatrix}.$$

Let u *only have nonzero entries, then the eigenvalues of* T *are the solutions of the secular equation*

$$\det(I + \gamma(D-\lambda)^{-1}uu^T) = 1 + \gamma u^T(D-\lambda)^{-1}u = 1 + \gamma\sum_{i=1}^{n}\frac{u_i^2}{d_i - \lambda} = 0. \tag{4.1}$$

It is well known that when u_i is exactly zero, then d_i is an eigenvalue of the glued matrix. Likewise, when u_i is tiny (as in the case of the top and bottom entry of the second eigenvector of W_{201}^+), the effect of the glue on the separation of the eigenvalues is negligible. Thus, for these glued matrices, the special structure can defeat the normal effects of rounding errors.

4.4. Computing the FP vector. ASSUMPTION 4.4. *Let the eigenvalue approximation* $\hat\lambda$ *have high relative accuracy. By starting inverse iteration according to (2.6) with* e_r, r *being the largest component of the true eigenvector, we are guaranteed that the FP vector has a small angle to the true eigenvector according to (2.9). The FP vector solves (2.11) and can be computed by (2.12).*

This is very satisfactory and correct in exact arithmetic. In finite precision arithmetic, we might not be able to accurately solve (2.6) (or (2.11)) by (2.12). Although we obtain a vector with small residual, we need not obtain the true FP vector which is guaranteed to have a small angle to the true eigenvector. The culprit is underflow as we now explain.

The eigenvectors of the largest eigenvalue pair of a Wilkinson matrix W_{2m+1}^+, m sufficiently large, provide examples. Both eigenvectors have their 'essential' nonzero part concentrated at the top and the bottom and decay rapidly towards the middle. This decay commonly causes MRRR (and other methods as well) to return the bisectors instead of the true eigenvectors, see the example of MATLAB in Figures 4.1 and 4.2.

These bisectors are perfectly satisfactory from the point of view of numerically small residual norms and numerical orthogonality. Moreover, the true eigenvectors can be obtained as the sum and the difference of the two bisectors.

We now look at the execution of (2.12) on the two largest eigenpairs of W_{201}^+. The largest entries of the true eigenvectors are at the first and last position. When executing (2.12), underflow must occur for large enough m (and in particular occurs for $m = 201$), which sets the rest of the vector to zero, yielding the bisector instead of the FP vector.

The danger is that MRRR might choose the same twist index for both vectors, yielding the same bisector. In exact arithmetic, there are are two maximal entries

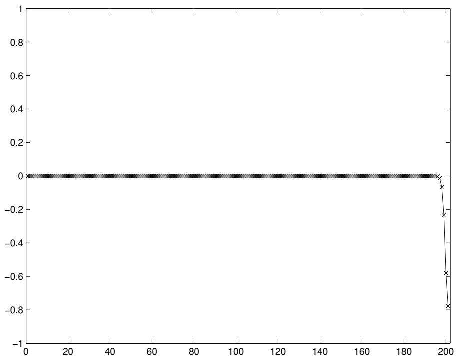

FIG. 4.1. *Eigenvector 200 of W_{201}^{+} as computed by MATLAB.*

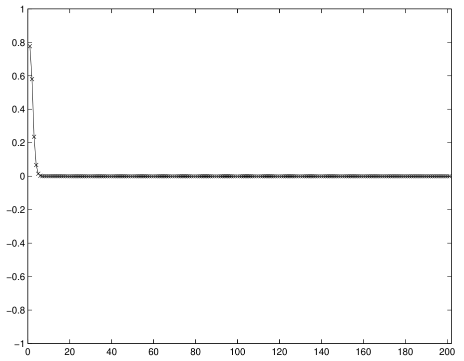

FIG. 4.2. *Eigenvector 201 of W_{201}^{+} as computed by MATLAB.*

in the true eigenvectors, and all would be well if the code chose the twist indices at opposite ends. However, currently there is no mechanism in the MRRR algorithm that guarantees this.

In summary, the FP vector in exact arithmetic is always a good approximation to the true eigenvector. However, in the face of underflow, the computed vector will not satisfy (2.6) and its angle to the true eigenvector will not be small. Furthermore, MRRR is not guaranteed to produce orthogonal vectors.

5. Modifications to MRRR. This section sketches possible modifications to MRRR and discusses their principal features.

Section 5.1 essentially invokes the earlier LAPACK routine STEIN, based on inverse iteration with Modified Gram-Schmidt orthogonalization. We apply it only to very tight clusters on which MRRR has failed (as seen in Section 4).

Section 5.2 covers Parlett's submatrix method which is very efficient but is applicable only for clusters that are well isolated.

Both these methods can be used as fall-back for the MRRR algorithm on a difficult cluster. However, such a cluster might only be relatively isolated from neighboring clusters and not well separated by an absolute gap. Then the orthogonality among the two clusters cannot be guaranteed without additional work. For this purpose, we discuss the Rayleigh-Ritz projection in Section 5.3.

In the worst case, all these modifications will lead to an algorithm that costs $\mathcal{O}(nk^2)$ operations, for a cluster of size k. In Section 5.4, we describe Dhillon's elegant proposal to simply get rid of these difficult cases by small componentwise relative random perturbation of the root of the representation. This approach does not sacrifice the $\mathcal{O}(n^2)$ complexity of the original MRRR algorithm.

5.1. Selective inverse iteration with Modified-Gram-Schmidt orthogonalization. In this section, we take a look at inverse iteration with Modified Gram-Schmidt orthogonalization [5, 8, 9], available in LAPACK's STEIN. Its efficiency problems with matrices from analysis of molecules prompted the research that led to the development of the MRRR algorithm. It is interesting that STEIN can cope with the very tight glued clusters and thus complement STEGR in this respect.

Given a set of eigenvalues ordered by increasing value, the procedure is as follows. First, if an eigenvalue is too close to its left neighbor, it is perturbed by a small relative amount. Then one step of inverse iteration with a random right-hand side is performed. The resulting vector is kept orthogonal to all previously computed eigenvectors that belong to eigenvalues within distance $ORTOL$ of the current eigenvalue. (The choice of the parameter $ORTOL = 10^{-3}||T||_1$ determines the level of numerical orthogonality.) The vector obtained is scaled and used as new right-hand side until the iteration converges. However, it is the orthogonalisation feature that makes STEIN inefficient.

Thus, we could use STEIN as backup for the case when MRRR cannot achieve relative separation for a cluster. However, this modification would destroy the claim that MRRR has $\mathcal{O}(n^2)$ complexity in the worst cases. While inverse iteration itself costs only $\mathcal{O}(nk)$ operations, the total cost of the algorithm is dominated by the orthogonalization of a computed vector to all previously found eigenvectors, yielding an $\mathcal{O}(nk^2)$ algorithm.

5.2. The submatrix method. In this section, we sketch briefly the ideas behind the submatrix method as a means of obtaining an orthogonal basis for the subspace defined by a tight isolated cluster. For more details, we refer the reader to [11].

The envelope vector of an invariant subspace of the (symmetric) matrix T is defined by the property that its j-th entry is the maximal $j-th$ entry over all unit vectors in the subspace. For a tight isolated cluster of k eigenvalues, the envelope has k hills separated by $k-1$ valleys. The method chooses k submatrices of T with two properties:

- Each submatrix has a simple eigenvalue in the cluster interval whose normalized eigenvector has very small first and last entries (except for the first entry of the submatrix at the top and the last entry of the submatrix at the bottom).
- The indices of each submatrix overlap, at worst, the indices of its adjacent neighbors.

The respective k (small) eigenvectors of these k submatrices, when padded appropriately with zeros, form a distinguished basis for the subspace. When the cluster is well isolated, these basis vectors are numerically orthogonal and have a small residual norm (with respect to T). Otherwise, they might be not quite orthogonal (but, by construction, are guaranteed to be linearly independent). Suitable linear combinations are needed to form the eigenvectors of T. We show how to do that in the next section.

The main attraction of the submatrix method is that by construction, each submatrix has exactly one eigenvalue in the interval spanned by the cluster, thus in terms of the submatrix, we have to compute the eigenvector of an isolated eigenvalue which is an easy task.

Typically, the submatrix method is an $\mathcal{O}(nk)$ process. However, its limitation is that the tight cluster needs to be isolated. Some large glued Wilkinson matrices produce small tight clusters that are not well isolated, an example is shown in the next section. We could not find a modification of the submatrix method that did not spoil its complexity in that case.

5.3. Rayleigh-Ritz projections and how to couple MRRR with selective inverse iteration. In the previous sections, we have seen how to obtain numerically orthogonal vectors spanning the approximate invariant subspace of a tight cluster. These basis vectors will be (automatically) orthogonal to other eigenvectors belonging to eigenvectors outside the cluster, provided that the cluster is isolated. However, when the the (absolute) gaps on either side of the tight cluster are not large enough, orthogonality cannot be guaranteed. For this reason, neither inverse iteration nor the submatrix method alone can provide a fall-back solution for tight clusters on which the MRRR algorithm fails.

A possible solution is the following post-processing step whenever the MRRR algorithm fails to resolve a tight non-isolated cluster and resorts to a special method to compute a subspace basis. After all eigenvectors have been computed, we can group together those of the tight cluster and all those belonging to eigenvalues which are close to it so that the enlarged set has gaps greater than $ORTOL$ to the rest of the spectrum. (Here, we choose the parameter $ORTOL$ in the same way as in Section 5.1.) Then, the Rayleigh-Ritz-values and -vector give us the best approximation from that subspace. The Ritz vectors are orthogonal among themselves and the orthogonality to the eigenvectors of the rest of the spectrum is governed in the same way as for STEIN, see Section 5.1.

Specifically, let Z_k be the matrix whose k normalized linearly independent column vectors $z_i, 1 \leq i \leq k$ span the enlarged subspace. Furthermore, let

$$||Tz_i - \hat{\lambda}_i z_i|| = \mathcal{O}(n\epsilon ||T||),\ \forall i \in \{1, \ldots, k\}\,.$$

Define the matrix $A = (a_{ij}) \in \mathcal{R}^{k \times k}$ by

$$a_{ij} = \frac{\lambda_i + \lambda_j}{2}(z_i, z_j),$$

and the matrix $B = (b_{ij}) \in \mathcal{R}^{k \times k}$ by $b_{ij} = (z_i, z_j)$.

If Λ holds the (generalized) eigenvalues and the columns of G hold the corresponding generalized eigenvectors of the pencil (A, B), $AG = BG\Lambda$, then Rayleigh-Ritz theory states that

$$(\Lambda, Z_k G) = \arg \min_{(\hat{\Lambda}, \hat{Z})} ||T\hat{Z} - \hat{Z}\hat{\Lambda}||_F.$$

In practice, the need for the *post-processing* step of computing Ritz approximations from a subspace does not occur very often. This is very fortunate since a cost analysis reveals that for a cluster of size k, it has $\mathcal{O}(nk^2)$ costs.

Next, we give an example where the post-processing is needed. We consider a matrix T obtained from five copies of W_{201}^{+} glued together by $\sqrt{\epsilon}$. Originally, the largest eigenvalue pair of the matrix W_{201}^{+} is very close to 100.74619418290335. By gluing 5 copies of the matrix together, we obtain a cluster of 10 eigenvalues that, at Level 1 of the representation tree (where we shift close to the right side of the cluster), that splits up into two (tight) clusters of four eigenvalues each and a (not quite so tight) cluster of two in the middle. (The relative gaps between eigenvalues 999 and 1000 and eigenvalues 1001 and 1002 are both large enough to pass MRRR's threshold test.) This is shown in Table 5.1. In order to show that the eigenvalues belong to different clusters and thus to different representations, we separate them at the cluster boundaries by a horizontal line. Since the two eigenvalues in the middle cluster have large relative gaps between themselves, the corresponding eigenvectors can be immediately computed by MRRR. However, the algorithm is not able to refine the two outer clusters any more, so it chooses representations closer and closer to the cluster ends without detecting large enough relative gaps in the eigenvalues which keep decreasing towards the roundoff level. Recall the behavior observed in Section 4 and documented in Table 4.1. We can cure this situation in each cluster individually by applying inverse iteration with modified Gram Schmidt orthogonalization on each group of eigenvalues as described in Section 5.1.

Index	Level 1	Level 2	Level 10
996	-3.841845502299982E-17	3.2287628082475236E-32	1.4072142703872178E-152
997	-3.841845502299985E-17	3.2287628082475225E-32	1.4072142703872178E-152
998	-3.841845502299985E-17	3.2287628082475225E-32	1.4072142703872178E-152
999	-3.841845502299985E-17	3.2287628082475225E-32	1.4072142703872178E-152
1000	-3.827326426940154E-17		
1001	-6.336625249605947E-22		
1002	-3.680255591970263E-21	-1.1612364990015818E-36	6.952868336216883E-157
1003	-3.680255591970262E-21	-1.1612364990015825E-36	6.952868336216888E-157
1004	-3.680255591970262E-21	-1.1612364990015825E-36	6.952868336216888E-157
1005	-3.680255591970262E-21	-1.1612364990015825E-36	6.952868336216888E-157

Table 5.1

The largest (local) eigenvalues of the glued matrix T by representation tree level.

However, this procedure will only produce vectors that are orthogonal to the other three vectors within their cluster of four, mutual orthogonality within the cluster of ten eigenvalues of T is not guaranteed and indeed not given. However, the ten vectors are linearly independent and provide a good basis for the invariant subspace. By computing the orthogonal Ritz basis, we obtain orthogonal vectors with small residuals. In Table 5.2, we display the computed eigenvalues of T and the Ritz values from the post-processing process.

Index	Eigenvalues	Ritz-values
996	100.74619418290335	100.74619418290335
997	100.74619418290335	100.74619418290337
998	100.74619418290335	100.74619418290337
999	100.74619418290335	100.74619418290341
1000	100.74619418290335	100.74619418290344
1001	100.74619418290335	100.74619418290345
1002	100.74619420089603	100.74619420089599
1003	100.74619420089603	100.74619420089603
1004	100.74619420089603	100.74619420089604
1005	100.74619420089603	100.74619420089614

Table 5.2

The largest eigenvalues of the glued matrix T and the corresponding Ritz values computed in the post-processing step.

5.4. Small relative perturbations of the root representation. In Section 5.1, we presented inverse iteration with Modified Gram-Schmidt orthogonalization as one remedy for tight clusters arising from gluing. Two key factors for that algorithm's success are

- The perturbation of an eigenvalue by a small (relative) amount to separate it from the rest of the spectrum.
- The choice of a random right-hand side for the inverse iteration.

As mentioned, the important drawback of the approach is its repeated use of reothogonalization that increases its complexity.

Dhillon had the clever idea to perturb the elements of the root representation by a (relatively) small amount in order to achieve eigenvalue separation and break up the tight clusters. After having found an LDL^T factorization of T (with appropriate shift), he suggests using

$$\hat{d}_i = d_i * (1 + \mu_i k\epsilon),\ 1 \le i \le n, \tag{5.1}$$

$$\hat{l}_j = l_j * (1 + \nu_j k\epsilon), \ 1 \le j \le n, \tag{5.2}$$

where k is a small number, say 4, and $\mu_i, \nu_j \in [-1, 1]$ are chosen at random from a uniform distribution. The size of the perturbation is at the same level as the errors in the differential qd algorithms.

We have verified by experiments that this technique works very well and will show an example below. The rationale for our approach is as follows.

- Symmetric tridiagonal matrices do not necessarily define their eigenvalues to high relative accuracy. A result of Demmel and Kahan [4] states that bidiagonal matrices always do. As long as k from (5.1) and (5.2) is small enough, the eigenvalues of the root representation are perturbed only by a relatively small amount. For this reason, we apply the perturbation to the root representation and not the matrix T itself.
- So far, all our tests indicate that no other representations need to be perturbed. Thus, MRRR computes the eigenvalues of the perturbed root representation $\hat{L}\hat{D}\hat{L}^T$ instead of the computed LDL^T factorization of T.
- In order to reproduce numerical results, we advocate the use of a pseudo-random generator with fixed seed.

In brief, we introduce artificial roundoff into the root representation since machine roundoff might not come to rescue.

In the following, we study the effect of random perturbations on the second-smallest cluster of a matrix T obtained from five copies of W_{201}^+ glued together by $\sqrt{\epsilon}$. As seen in Section 4, this matrix is a counterexample to the basic assumptions of MRRR. Table 4.1 showed a refinement of the eigenvalues of the cluster up to the underflow threshold without being able to find subclusters. This behavior will be compared to the use of MRRR on a perturbed root representation with $k = 4$ in (5.1) and (5.2). We remark that the respective children at Level 1 of the unperturbed and the perturbed root representation have (slightly) different shifts. For this reason the local eigenvalues appear very different. However, for MRRR, we note that the child of the perturbed representation has (local) eigenvalues with large relative gaps; see the right-most column of Table 5.3. Hence we can compute the corresponding eigenvectors immediately with guaranteed numerical orthogonality. The relative changes in the eigenvalues of the root representation (Level 0) are of order $\mathcal{O}(n\epsilon)$, n = 1005 in this case.

	LDL^T (unperturbed root)		$\hat{L}\hat{D}\hat{L}^T$ (perturbed root)	
Index	Level 0	Level 1	Level 0	Level 1
6	0.2538058170966395	1.052402199759149E-13	0.2538058170965921	-6.243561316319056E-13
7	0.2538058170966395	1.052402199759149E-13	0.2538058170966135	-6.028454790143962E-13
8	0.2538058170966395	1.052402199759149E-13	0.2538058170966301	-5.862486570542836E-13
9	0.2538058170966395	1.052402199759149E-13	0.2538058170966449	-5.714964747742379E-13
10	0.2538058170966395	1.052402199759149E-13	0.2538058170966671	-5.492735741423265E-13

Table 5.3

Investigation of the second-smallest cluster of eigenvalues of the matrix T obtained from five copies of W_{201}^+ glued together by $\sqrt{\epsilon}$, see also Table 4.1. Shown are the unshifted eigenvalues of the (unperturbed) root representation LDL^T and its perturbed counterpart $\hat{L}\hat{D}\hat{L}^T$. Furthermore, we display the eigenvalues of the representation of the corresponding child clusters that belong to level 1 of the respective representation trees.

In conclusion, random perturbations of the root representation work well in practice. The perturbations are made by default, not just in difficult cases which might be hard to recognize a priori. In all our tests, we have yet to find a case where the perturbed clusters are not broken up quickly.

6. Summary and conclusions. The MRRR algorithm is based on very reasonable assumptions that are theoretically sound and hold in the majority of cases in finite precision arithmetic. In this paper, we have shown very subtle and non-obvious difficulties that can lead to failures. We have illustrated and explained these failures by examples. Furthermore, we have proposed and evaluated possible remedies. These will be adopted for an update of the current LAPACK 3.0 version of STEGR.

REFERENCES

[1] J. Barlow and J. Demmel. Computing accurate eigensystems of scaled diagonally dominant matrices. *SIAM J. Numer. Anal.*, 27(3):762–791, 1990.
[2] J. Bunch, P. Nielsen, and D. Sorensen. Rank-one modification of the symmetric eigenproblem. *Numer. Math.*, 31:31–48, 1978.
[3] J. J. M. Cuppen. A divide and conquer method for the symmetric tridiagonal eigenproblem. *Numer. Math.*, 36:177–195, 1981.
[4] J. W. Demmel and W. Kahan. accurate singular values of bidiagonal matrices. *SIAM J. Sci. Stat. Comput.*, 11(5):873–912, 1990.
[5] I. S. Dhillon. Current inverse iteration software can fail. *BIT*, 38:4:685–704, 1998.
[6] I. S. Dhillon, B. N. Parlett, and C. Vömel. LAPACK working note 162: The design and implementation of the MRRR algorithm. Technical Report UCBCSD-04-1346, University of California, Berkeley, 2004. (also as LAPACK Working Note #162).
[7] M. Gu and S. C. Eisenstat. A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem. *SIAM J. Matrix Anal. Appl.*, 16(1):172–191, 1995.
[8] I. C. F. Ipsen. A history of inverse iteration. In B. Huppert and H. Schneider, editors, *Helmut Wielandt, Mathematische Werke, Mathematical Works*, volume II: Matrix Theory and Analysis. Walter de Gruyter, Berlin, 1996.
[9] I. C. F. Ipsen. Computing an eigenvector with inverse iteration. *SIAM Review*, 39(2):254–291, 1997.
[10] B. N. Parlett. *Acta Numerica*, chapter The new qd algorithms, pages 459–491. Cambridge University Press, 1995.
[11] B. N. Parlett. Invariant subspaces for tightly clustered eigenvalues of tridiagonals. *BIT*, 36(3):542–562, 1996.
[12] B. N. Parlett and I. S. Dhillon. Fernando's solution to Wilkinson's problem: an application of double factorization. *Linear Algebra and Appl.*, 267:247–279, 1997.
[13] B. N. Parlett and I. S. Dhillon. Relatively robust representations of symmetric tridiagonals. *Linear Algebra and Appl.*, 309(1-3):121–151, 2000.
[14] B. N. Parlett and I. S. Dhillon. Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices, 2004.
[15] B. N. Parlett and I. S. Dhillon. Orthogonal eigenvectors and relative gaps. *SIAM J. Matrix Anal. Appl.*, 25(3):858–899, 2004.
[16] B. N. Parlett and C. Vömel. Eigenvalues and eigenvectors of glued matrices. University of California, Berkeley, 2004. In preparation.
[17] J. H. Wilkinson. *The Algebraic Eigenvalue Problem*. Oxford University Press, Oxford, 1965.