

# Experiences with a Lanczos Eigensolver in High-Precision Arithmetic

Alexander Alperovich<sup>1</sup>, Alex Druinsky<sup>2</sup>, and Sivan Toledo<sup>2</sup>

<sup>1</sup> Microsoft, Haifa, Israel

<sup>2</sup> Tel Aviv University, Tel Aviv, Israel

**Abstract.** We investigate the behavior of the Lanczos process when it is used to find all the eigenvalues of large sparse symmetric matrices. We study the convergence of classical Lanczos (i.e., without re-orthogonalization) to the point where there is a cluster of Ritz values around each eigenvalue of the input matrix  $A$ . At that point, convergence to all the eigenvalues can be ascertained if  $A$  has no multiple eigenvalues. To eliminate multiple eigenvalues, we disperse them by adding to  $A$  a random matrix with a small norm; using high-precision arithmetic, we can perturb the eigenvalues and still produce accurate double-precision results. Our experiments indicate that the speed with which Ritz clusters form depends on the local density of eigenvalues *and on the unit roundoff*, which implies that we can accelerate convergence by using high-precision arithmetic in computations involving the Lanczos iterates.

**Keywords:** Lanczos, mixed-precision arithmetic, Ritz clusters

## 1 Introduction

Existing software libraries offer us several choices when we wish to compute the eigenvalues of a sparse matrix. One option is to use a dense eigensolver, such as one of those implemented in LAPACK [1], which compute all of the eigenvalues at a cost of  $\Theta(n^3)$  arithmetic operations. This is a high cost for a sparse matrix. Krylov-subspace eigensolvers, such as ARPACK [16], take advantage of sparsity, but they only offer to compute small, user-selected subsets of the spectrum. Other algorithms compute the whole spectrum, but they require the matrix to have a special sparsity structure, such as the MRRR method for symmetric tridiagonal matrices [7]. In this paper we explore the possibility of accomplishing both goals simultaneously: computing all of the eigenvalues and also taking advantage of sparsity, even when there is no specific sparsity structure.

The Lanczos process is a long-established and well-known eigensolver [15] (see also [10,17,23,24,29]). It takes as input an  $n$ -by- $n$  Hermitian matrix  $A$  and produces a sequence of matrices  $T^{(m)}$  and  $Q^{(m)}$  such that

$$AQ^{(m)} = Q^{(m)}T^{(m)} + r^{(m)}e_m^*,$$

where  $Q^{(m)}$  is an  $n$ -by- $m$  orthonormal matrix,  $T^{(m)}$  is an  $m$ -by- $m$  tridiagonal matrix,  $e_m$  is the last unit vector of dimension  $m$ , and  $r^{(m)}$  is some  $n$ -vector. The

sequences  $Q^{(m)}$  and  $T^{(m)}$  are nested: each iteration of the Lanczos process adds one column to  $Q$  and a row and a column to  $T$ . The process is a short-recurrence Krylov-subspace iteration; in each iteration, the algorithm multiplies one vector by  $A$  and performs a small number of vector operations on vectors of size  $n$ .

In exact arithmetic, the residual vector  $r^{(m)}$  vanishes after at most  $k$  iterations, where  $k$  is the number of distinct eigenvalues of  $A$ . When  $r^{(m)}$  vanishes,  $T^{(m)}$  is an orthonormal projection of  $A$  onto the column space of  $Q$ , and therefore every eigenvalue of  $T^{(m)}$  is an eigenvalue of  $A$ . For all the starting vectors except for a set of measure 0,  $r^{(m)}$  vanishes after exactly  $k$  iterations and all the eigenvalues of  $A$  appear in the spectrum of  $T^{(k)}$ .

Practitioners quickly discovered that the behavior of Lanczos in floating-point arithmetic differs significantly from that predicted by the theoretical results. In particular, the columns of  $Q$  quickly lose orthogonality, and  $r$  never vanishes in practice. Researchers mostly explored two families of techniques for addressing this difficulty. One set of techniques attempts to prevent the loss of orthogonality in  $Q$ . This can be done using a full orthogonalization process or using selective orthogonalization and related techniques [11,20,25,26]. The other set of techniques [2,31] attempts to extract useful spectral information from the process after a relatively small number of iterations; this rarely results in the identification of all the eigenvalues, but it can result in useful approximations to a subset of the eigenvalues that are important in a given application (e.g., the smallest ones). These families of techniques are not mutually exclusive; many Lanczos codes use both.

However, around 30 years ago a group of researchers explored the use of Lanczos without sophisticated orthogonalization for finding all the eigenvalues of  $A$  [3,9,19]; we refer to such methods as *classical Lanczos* methods. This line of research was based on a deep numerical analysis of the Lanczos process that ultimately showed that in floating point, *the eigenvalues of  $T$  eventually approximate all the eigenvalues of  $A$*  [8]. (This fact was recognized years before it was actually proved; see, for example, [3]). These researchers produced two Lanczos codes, both in the 1980s. To the best of our knowledge none of the Lanczos codes that have been published since 1985 have been *classical* Lanczos. Even though development of new codes has slowed down, there has been intense ongoing theoretical interest in classical Lanczos, resulting in a large body of results (see [17] and the numerous references therein). Experimental studies have also been published [13].

Our goal in this paper is to present the challenges that are involved in developing a classical Lanczos eigensolver, to describe our ideas for addressing these challenges, and to explore the feasibility of these ideas. A major tool in our toolkit is high-precision arithmetic, which our experiments show can be used to tackle hard matrices with tight clusters of eigenvalues and also to devise a reliable termination criterion for the algorithm. Our computational cost of using high-precision arithmetic is negligible because we need it only for forming the matrix  $T$  and not for computing its eigenvalues.

## 2 Background and Methodology

A key issue in Lanczos solvers, including ours, is deciding which Ritz value (eigenvalue of  $T^{(m)}$ ) is an approximate eigenvalue of  $A$ . A growing body of work suggests that non-trivial clusters of Ritz values are only found very close to eigenvalues of  $A$  (see Wülling [32,33], Knizhnerman [14, Theorem 2], Strakoš and Greenbaum [30], and Greenbaum [12]). That is, if we find two or more eigenvalues of  $T^{(m)}$  that are very close to each other, they normally indicate the location of an eigenvalue of  $A$ ; we call such Ritz values *doubly-converged*. This phenomenon was already known to Cullum and Willoughby [3] and to Parlett and Reid [19], but back then there were no provable bounds on the location of eigenvalues relative to non-trivial Ritz clusters. We write that Ritz clusters *normally* indicate eigenvalues because all the results in the literature are conditioned on properties of the spectrum of  $A$  and/or  $T^{(m)}$ , which might not hold. However, exceptions seem very rare and some conditions are easily tested (in particular, conditions that only involve Ritz values).

If all the eigenvalues of  $A$  are simple, we can stop Lanczos once we have  $n$  distinct Ritz clusters (doubly-converged eigenvalues). If there are multiple eigenvalues, we need another strategy. The codes of Cullum and Willoughby [3] and Parlett and Reid [19] used heuristics to decide when to stop. These heuristics sometimes cause the algorithms to fail to find all the eigenvalues; these failures are sometimes silent and sometimes explicit (reported to the user).

In order to address this problem, we use a conceptually simple solution that we call *dispersion*. Instead of running Lanczos on  $A$  itself, we will run it on  $A + P$ , where  $P$  is a random symmetric matrix (from some appropriate distribution) with a small norm  $\|P\|_2 \leq \delta$ . We choose  $P$  so that it is cheap to apply to vectors; this results in Lanczos iterations that are about as cheap as those performed on  $A$  alone. The perturbation  $P$  perturbs the eigenvalues, but only by  $\delta$  or less. Hopefully,  $A + P$  has no multiple eigenvalues; multiple eigenvalues of  $A$  are transformed into clusters of close but distinct eigenvalues of  $A + P$ . The choice of  $P$  determines how close the eigenvalues of  $A + P$  are; we do not have a complete theory that guarantees good separation with high probability, but experiments have shown that dispersion works well. We omit these experiments from this paper, and focus instead on the convergence for a given operator (which the reader can take to be  $A + P$ ).

The size of the perturbation  $\delta$  and machine precision  $\epsilon_{\text{machine}}$  must be tailored according to the accuracy  $\epsilon$  required by the user, using high-precision arithmetic to reduce  $\epsilon_{\text{machine}}$  if necessary. The relation  $\epsilon > \delta > \epsilon_{\text{machine}}$  must hold with sufficient safety margins so that the perturbation can simultaneously separate multiple eigenvalues and preserve the required accuracy.

An alternative approach to obtaining the required accuracy is to use a first-order correction. Here, we consider  $A$  as a matrix that we obtain from  $A + P$  by adding the perturbation  $-P$ . To first order, the eigenvalues of  $A$  are equal to  $\mu_i - v_i^T P v_i$ , where  $\mu_i$  and  $v_i$  are eigenpairs of  $A + P$  for  $i = 1, 2, \dots, n$ ; for details, see [29, pp. 45–48]. Computing the correction  $v_i^T P v_i$  requires that we compute the eigenvectors  $v_i$ , which is accomplished by multiplying the  $n$ -by- $m$  matrix of

iterates  $Q$  by the eigenvectors of  $T$ . This costs  $\Theta(mn)$  arithmetic operations for each  $v_i$  and  $\Theta(mn^2)$  overall. Because  $m > n$ , this is at least as expensive as the  $\Theta(n^3)$  cost of computing the eigenvalues of  $A$  directly using a dense eigensolver. This cost is too high for sparse matrices and therefore we do not use first-order corrections in this paper.

From a broader perspective, matrices with multiple eigenvalues are a sort of singularity, a set of measure zero that causes algorithmic difficulty. The idea of perturbing problems in order to avoid such singularities is well-established in existing research [4,6,28].

To compute the eigenvalues of  $T$  in our experiments we used the LAPACK subroutine DSTEMR and the MPACK subroutine RSTEQR. Both of these subroutines are symmetric tridiagonal eigensolvers; DSTEMR implements the MRRR algorithm and RSTEQR the implicit  $QL$  or  $QR$  methods. The MPACK library, to which RSTEQR belongs, is a collection of multiple-precision versions of BLAS and LAPACK subroutines [18]. Although we only needed the eigenvalues of  $T$  to double-precision accuracy, in some of our experiments we used higher precision, which is why we used MPACK.

### 3 Convergence on Real-World Matrices

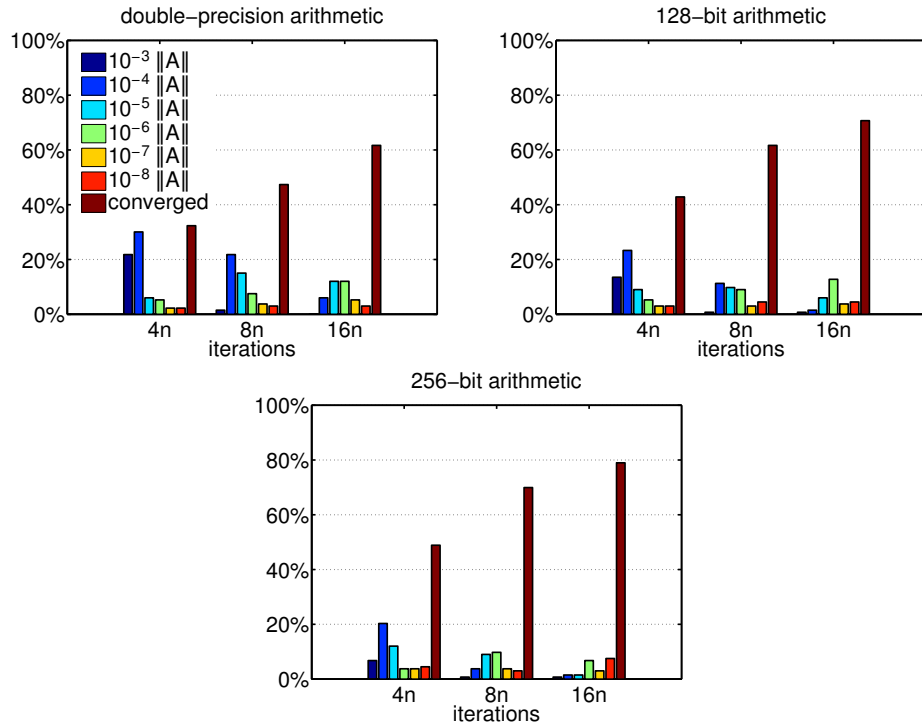
Figure 1 explores the behavior of high-precision Lanczos on a large set of real-world matrices. We ran our code on all 133 symmetric matrices of dimension 2500 or less from the University of Florida Sparse Matrix Collection [5] (we only used matrices with numerical values; we omitted sparsity-pattern-only matrices). We did not attempt to disperse multiple eigenvalues; instead, we compared the eigenvalues computed by our code to those computed by LAPACK and counted how many agree to within  $10^{-9}\|A\|$  or better.<sup>3</sup> This tells us when our code fails to find isolated eigenvalues or entire clusters, but not whether it converges to all the eigenvalues in a tight cluster.

The results show that Lanczos can compute all the eigenvalues of most of the matrices after  $16n$  iterations: of over 60% of the matrices with 64-bit arithmetic, and of more than 70% with 128- and 256-bit arithmetics. After only  $4n$  or  $8n$  iterations, Lanczos can still compute all the eigenvalues of many matrices. As we perform more iterations, Lanczos tends to resolve more eigenvalues *in sparse areas of the spectrum*. The remaining non-converged eigenvalues tend to be shadowed by their neighbors.

### 4 The Effects of Clusters of Eigenvalues

As we saw in Section 3, the key to fast convergence of the Lanczos iteration is dealing with clusters of eigenvalues. One of the problems caused by clustering has been discovered by Parlett et al. [21], who showed that a Ritz value can become

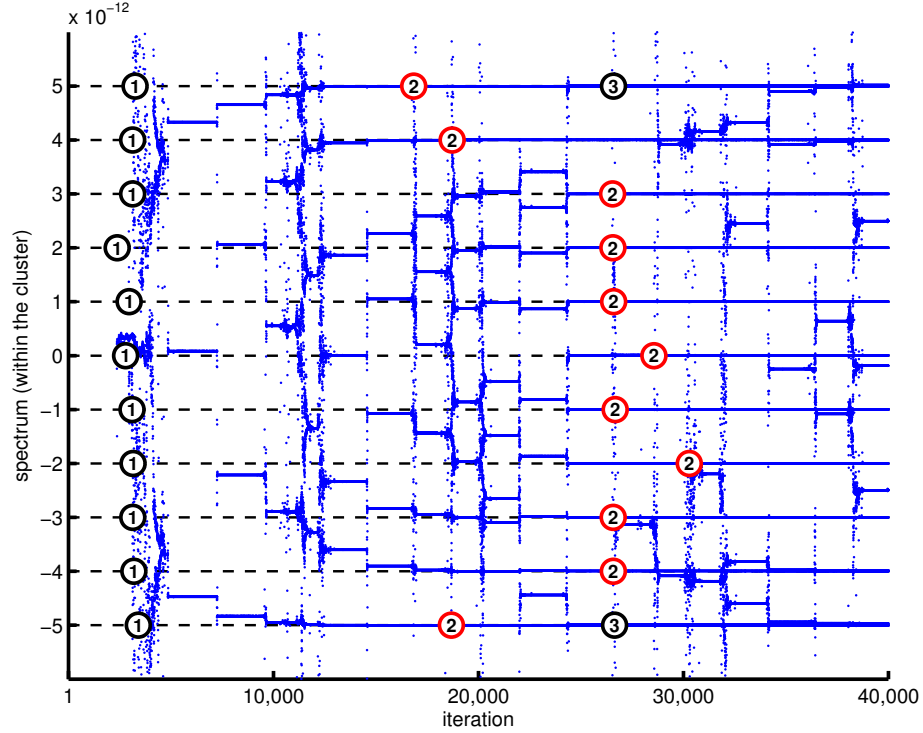
<sup>3</sup> We used the LAPACK unsymmetric eigensolver DGEEV to compute the eigenvalues. Although the symmetric subroutine DSYEV is more efficient, we did not use it here.



**Fig. 1.** Convergence behavior on a set of 133 real-world matrices. The graphs show the percentage of matrices that have converged to all the eigenvalues after  $4n$ ,  $8n$ , and  $16n$  iterations (in brown). The code did not attempt to find the multiplicity of each eigenvalue. The graphs also show, for matrices that have not converged, the greatest distance from a non-converged eigenvalue to its nearest neighbor. The graphs show results for 64-, 128- and 256-bit arithmetic, clockwise from top left. The number of bits refers to the accuracy of the arithmetic with which  $T$  was computed; its eigenvalues were always computed in 64-bit arithmetic.

fixed between two nearby eigenvalues and hold there for a number of iterations before finally migrating towards one of the eigenvalues (see also [22,27]). In this section we study this phenomenon, called *misconvergence*, and we also show that the problems caused by clustering go beyond misconvergence.

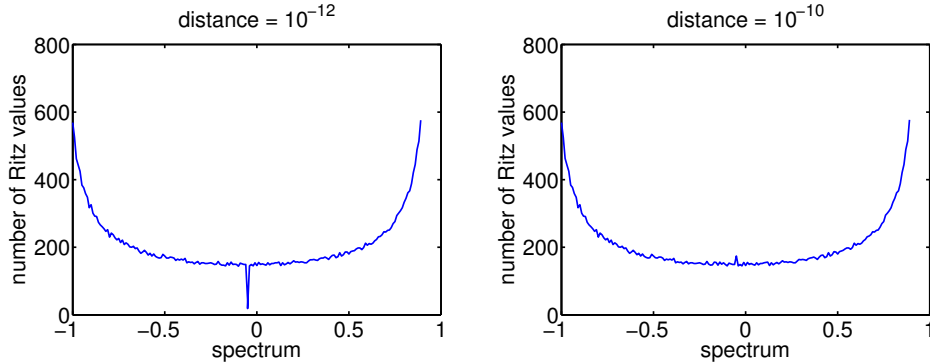
In our experiments we found that small clusters do not substantially affect convergence outside the cluster. When we ran Lanczos on a synthetic matrix whose eigenvalues were evenly spaced in the interval  $[-1, 1]$ , and then added a small cluster of  $0.05n$  eigenvalues, we found that adding the cluster caused no visible artifacts on the plot of the Ritz values produced by the iteration. However, we found severe misconvergence within the cluster. Figure 2 shows that a Ritz value that shows up in a cluster tends to wander around near and between eigenvalues and then typically settles for a long time *in-between* eigenvalues. As



**Fig. 2.** Evolution of Ritz values near a cluster. The cluster consists of 11 eigenvalues  $10^{-12}$  apart, represented by black horizontal dashed lines. Blue dots represent Ritz values. A circled numeral  $k$  shows the first time that there are  $k$  Ritz values near an eigenvalue. Red circles around the numeral 2 show where double convergence first occurs. Blue lines are formed by converged or misconverged Ritz values.

more Ritz values show up, a misconverged eigenvalue tends to shift closer to an eigenvalue, until it actually converges. If we inspect the eigenvalue at  $5 \times 10^{-12}$ , for example (the topmost one), we see a misconverged Ritz value that shifts between 3 or 4 stable locations before converging.

Additional Ritz values appear periodically near an eigenvalue. The most important effect of clusters is on this periodicity. In a cluster, the periodicity is longer; a Ritz value appears near a specific eigenvalue less often than near non-clustered eigenvalues. This is shown in the left plot of Figure 3. This phenomenon causes Lanczos to converge more slowly in the presence of clusters. If we examine the raw density of Ritz values, ignoring the distribution of eigenvalues, we see that the cluster attracts more Ritz values than intervals of the same size elsewhere in the spectrum. This is shown in the right plot of Figure 3. This increased attraction is not sufficient, however, to compensate for the larger number



**Fig. 3.** The number of Ritz values within a distance of  $10^{-12}$  (left) and  $10^{-10}$  (right) from each eigenvalue after  $200n$  iterations. The matrix has order  $n = 200$  and has a cluster of 10 eigenvalues positioned in the center of the spectrum at regular distances of  $10^{-12}$ . There are fewer Ritz values near each eigenvalue within the cluster than elsewhere, yet there are more Ritz values in the cluster area than if there was a single eigenvalue there.

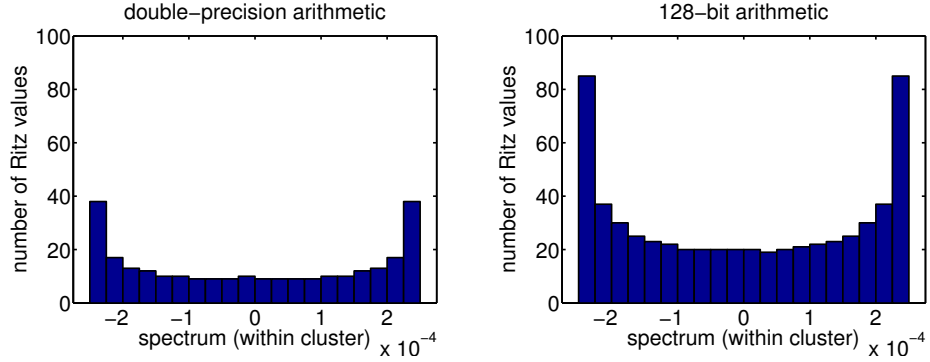
of eigenvalues in the interval, so convergence to all eigenvalues is still adversely affected by the cluster.

## 5 The Effects of High-Precision Arithmetic on the Lanczos Process

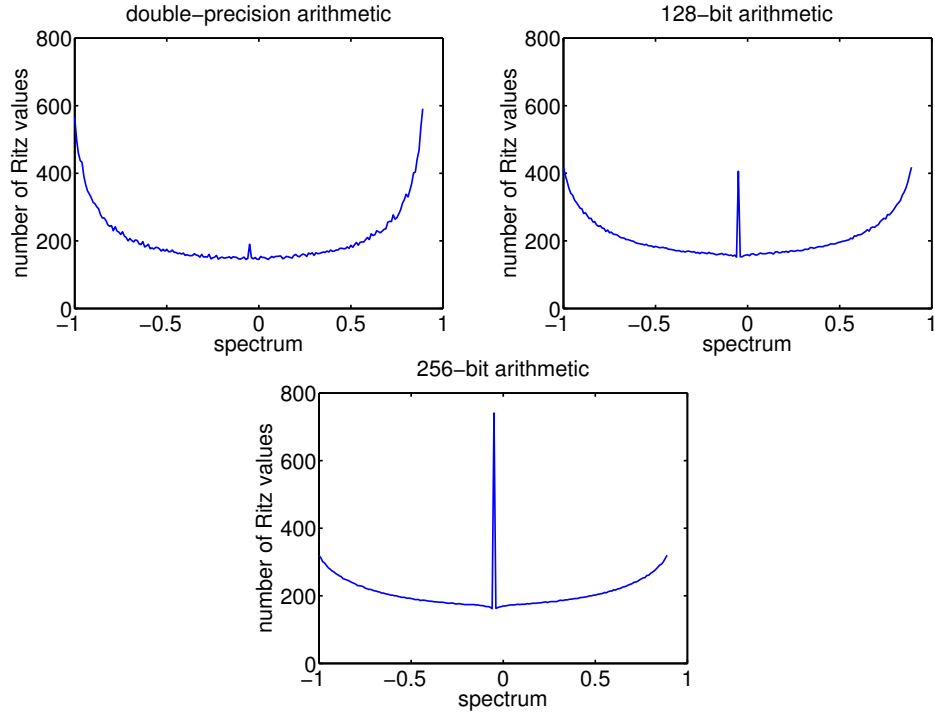
The distribution of Ritz values within a cluster is typically *not* uniform, just like within the spectrum as a whole. When eigenvalues in the cluster are evenly distributed, more Ritz values appear at the edges of the cluster than near its center, as shown in Figure 4. Even when there are *on average* 2 or 3 Ritz values per eigenvalue in the cluster, we may be very far from convergence, because there are not enough Ritz values near eigenvalues in the center of the cluster.

Increasing the precision of the floating-point arithmetic also increases the attractive power of clusters upon Ritz values, as shown in Figure 5. As we increase the precision, the number of Ritz values in a cluster increases, speeding up the convergence. This phenomenon was already observed by Edwards et al. [9], but it does not appear that Lanczos codes have used this insight.

As the size of an eigenvalue cluster grows, its effect on convergence becomes devastating, even in high precision. Figure 6 shows that as the size of a cluster grows, the number of Ritz values in it increases, but not nearly fast enough to obtain convergence on all eigenvalues. When the cluster is small, say containing 10 eigenvalues, there are more than 3 Ritz values per eigenvalue after  $20n$  iterations, even in 64-bit arithmetic (and more Ritz values in higher precision). When the cluster contains 100 eigenvalues, there is not even a single Ritz value per eigenvalue after  $20n$  iterations; we cannot expect convergence in that many iterations. Things get much worse as the cluster size continues to grow.

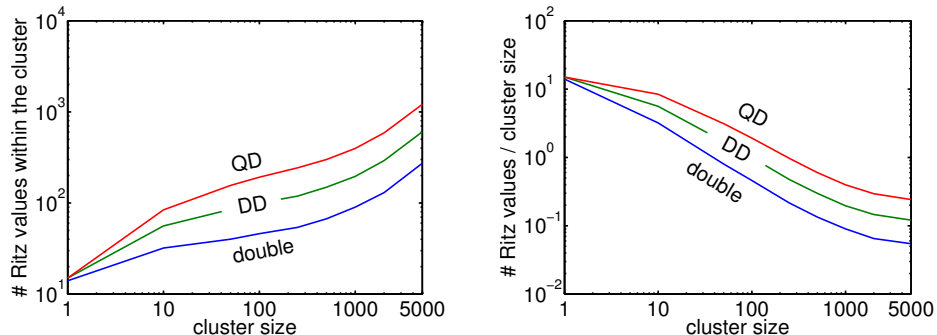


**Fig. 4.** A histogram of the Ritz values within a cluster of 5000 eigenvalues that are spaced  $10^{-7}$  apart after  $20n = 20 \cdot 7000$  Lanczos iterations. Apart from the cluster, the spectrum contains 2000 eigenvalues spaced evenly between  $-1$  and  $1$ . The histogram on the left shows the results in 64-bit arithmetic and the results on the right in 128-bit.



**Fig. 5.** The number of Ritz values within a distance of  $10^{-9}$  of each eigenvalue after  $200n$  iterations. The matrix has order  $n = 200$  and has a cluster of 10 eigenvalues positioned in the center of the spectrum at regular distances of  $10^{-11}$ . The iteration is carried out using different levels of floating-point precision: IEEE-754 double precision (64 bits; top left), 128-bit floating point (top right) and 256-bit floating point (bottom).





**Fig. 6.** The number of Ritz values within a cluster of eigenvalues that are spaced  $10^{-7}$  apart after  $20n$  Lanczos iterations. Apart from the cluster, the spectrum contains 2000 eigenvalues spaced evenly between  $-1$  and  $1$ . In both graphs, the  $X$  axis shows the number of eigenvalues in the cluster, ranging from 1 to 5000. (The dimension of the matrices therefore ranged from 2000 to 7000). On the left, the  $Y$  axis shows the number of Ritz values in the cluster. On the right, the  $Y$  axis shows the same number, but divided by the size of the cluster. Both graphs show the results of computations in 64-bit arithmetic (double), 128-bit (DD), and 256-bit (QD).

## 6 Conclusions

Our experiments suggest several conclusions. The results indicate that Lanczos can find *all* the eigenvalues of many real-world matrices. When it fails, convergence is impaired by the existence of dense areas in the spectrum, which is manifested by misconvergence, and more importantly by relatively low density of Ritz values in such dense areas. High-precision arithmetic helps to neutralize the effect of clustering, but the level of precision must be commensurate with the severity of clustering. How can we find the correct level of precision? One strategy is to start iterating in double precision and then repeatedly increase the accuracy after completing each sequence of  $n$  iterative steps. At the limit we have infinite accuracy and  $n$  additional steps are enough, although we expect a moderate level of accuracy to be sufficient for most matrices. We do not know how effective this strategy is in practice; this question is left for future work.

The slowdown in convergence due to clustering may make Lanczos impractical for problem matrices unless measures are taken to address this issue. Initial experimentation on small matrices suggests that randomized dispersion is effective when the spectrum contains clusters but they are not too large, but ineffective when clusters are very large (say an eigenvalue of multiplicity 3000 in a matrix of dimension 10000).

These technical conclusions lead us to two higher-level observations. First, classical Lanczos may be the only practical way of finding all the eigenvalues for some matrices. If the  $\Theta(n^2)$  space required for dense methods is not available, and if shift-invert operations are too expensive (e.g., matrices for which there is no sparse factorization), and if the spectrum contains only mild clustering,

then classical Lanczos may be the method of choice. This motivates further development of Lanczos codes and techniques.

We are used to resolving details at the scale of  $\epsilon$  using floating-point arithmetic with unit roundoff near  $\epsilon$  ( $\approx 10^{-16}$  for 64-bit arithmetic). For symmetric eigensolvers, resolving eigenvalues at this scale does not mean finding 16 significant digits per eigenvalue; it merely means finding 16 digits relative to the scale of the largest one. For ill-conditioned matrices, resolving eigenvalues to that scale may not be excessive at all. Our second high-level observation is that in Lanczos, resolving eigenvalues at the scale of  $\epsilon$  may require arithmetic with significantly smaller unit roundoff, perhaps  $10^{-32}$  or  $10^{-64}$ , or even less. More efficient implementations of high-precision floating-point arithmetic will enable computational scientists to resolve details that are currently beyond reach, like the eigenvalues of matrices with highly-clustered spectra.

**Acknowledgments** We thank the referees for their valuable comments. The idea of using first-order corrections that we discuss in Section 2 was proposed by one of the referees.

This research was supported in part by grant 1045/09 from the Israel Science Foundation (founded by the Israel Academy of Sciences and Humanities), and by grant 2010231 from the US–Israel Binational Science Foundation.

The first author was at Tel Aviv University while conducting this research.

## References

1. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide. SIAM (1999)
2. Calvetti, D., Reichel, L., Sorensen, D.C.: An implicitly restarted Lanczos method for large symmetric eigenvalue problems. *Electron. Trans. Numer. Anal.* 2, 1–21 (1994)
3. Cullum, J.K., Willoughby, R.A.: *Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Vol. 1 Theory*. Birkhäuser (1985)
4. Davies, E.B.: Approximate diagonalization. *SIAM J. Matrix Anal. Appl.* 29, 1051–1064 (2007)
5. Davis, T.A., Hu, Y.: The University of Florida sparse matrix collection. *ACM Trans. Math. Software* 38, 1:1–1:25 (2011)
6. Dhillon, I.S., Parlett, B.N., Vömel, C.: Glued matrices and the MRRR algorithm. *SIAM J. Sci. Comput.* 27, 496–510 (2005)
7. Dhillon, I.S., Parlett, B.N., Vömel, C.: The design and implementation of the MRRR algorithm. *ACM Trans. Math. Software* 32, 533–560 (2006)
8. Druskin, V.L., Knizhnerman, L.A.: Error bounds in the simple Lanczos procedure for computing functions of symmetric matrices and eigenvalues. *USSR Comput. Math. Math. Phys.* 31(7), 20–30 (1991)
9. Edwards, J.T., Licciardello, D.C., Thouless, D.J.: Use of the Lanczos method for finding complete sets of eigenvalues of large sparse symmetric matrices. *J. Inst. Math. Appl.* 23, 277–283 (1979)

10. Golub, G.H., Van Loan, C.F.: Matrix Computations. Johns Hopkins University Press, 3rd edn. (1996)
11. Grcar, J.F.: Analyses of the Lanczos Algorithm and of the Approximation Problem in Richardson's Method. Ph.D. thesis, University of Illinois at Urbana-Champaign (December 1981)
12. Greenbaum, A.: Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences. *Linear Algebra Appl.* 113, 7–63 (1989)
13. Kalkreuter, T.: Study of Cullum's and Willoughby's Lanczos method for Wilson fermions. *Comput. Phys. Commun.* 95, 1–16 (1996)
14. Knizhnerman, L.A.: The quality of approximations to a well-isolated eigenvalue, and the arrangement of "Ritz numbers" in a simple Lanczos process. *Comput. Math. Math. Phys.* 35(10), 1175–1187 (1995)
15. Lanczos, C.: An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Nat. Bur. Standards* 45(4), 255–282 (October 1950)
16. Lehoucq, R.B., Sorensen, D.C., Yang, C.: ARPACK Users' Guide. SIAM (1997)
17. Meurant, G.: The Lanczos and Conjugate Gradient Algorithms: From Theory to Finite Precision Computations. SIAM (2006)
18. Nakata, M.: The MPACK: Multiple precision arithmetic BLAS and LAPACK. <http://mplapack.sourceforge.net/> (2010)
19. Parlett, B.N., Reid, J.K.: Tracking the progress of the Lanczos algorithm for large symmetric eigenproblems. *IMA J. Numer. Anal.* 1, 135–155 (1981)
20. Parlett, B.N., Scott, D.S.: The Lanczos algorithm with selective orthogonalization. *Math. Comp.* 33, 217–238 (1979)
21. Parlett, B.N., Simon, H., Stringer, L.M.: On estimating the largest eigenvalue with the Lanczos algorithm. *Math. Comp.* 38, 153–165 (1982)
22. Parlett, B.: Misconvergence in the Lanczos algorithm. In: Cox, M., Hammarling, S. (eds.) *Reliable Numerical Computation*, pp. 7–24. Clarendon Press, Oxford (1990)
23. Parlett, B.N.: *The Symmetric Eigenvalue Problem*. Prentice-Hall (1980)
24. Saad, Y.: *Numerical Methods for Large Eigenvalue Problems*. SIAM, 2nd edn. (2011)
25. Simon, H.D.: Analysis of the symmetric Lanczos algorithm with reorthogonalization methods. *Linear Algebra Appl.* 61, 101–131 (1984)
26. Simon, H.D.: The Lanczos algorithm with partial reorthogonalization. *Math. Comp.* 42, 115–142 (1984)
27. van der Sluis, A., van der Vorst, H.A.: The convergence behavior of Ritz values in the presence of close eigenvalues. *Linear Algebra Appl.* 88–89, 651–694 (1987)
28. Spielman, D.A., Teng, S.H.: Smoothed analysis: An attempt to explain the behavior of algorithms in practice. *Commun. ACM* 52, 76–84 (2009)
29. Stewart, G.W.: *Matrix Algorithms, Volume 2: Eigensystems*. SIAM (2001)
30. Strakoš, Z., Greenbaum, A.: Open questions in the convergence analysis of the Lanczos process for the real symmetric eigenvalue problem. IMA Preprint 934, University of Minnesota (April 1992)
31. Wu, K., Simon, H.: Thick-restart Lanczos method for large symmetric eigenvalue problems. *SIAM J. Matrix Anal. Appl.* 22, 602–616 (2000)
32. Wülling, W.: The stabilization of weights in the Lanczos and Conjugate Gradient method. *BIT Numer. Math.* 45, 395–414 (2005)
33. Wülling, W.: On stabilization and convergence of clustered Ritz values in the Lanczos method. *SIAM J. Matrix Anal. Appl.* 27, 891–908 (2006)