A major purpose of the Technical Information Center is to provide the broadest dissemination possible of information contained in DOE's Research and Development Reports to business, industry, the academic community, and federal, state and local governments.

Although a small portion of this report is not reproducible, it is being made available to expedite the availability of information on the research discussed herein.

LA-UR -88-2482

1988

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

LA-UR--88-2482

DE88 014401

TITLE: SOME STRATEGIES FOR ENHANCING THE PERFORMANCE OF THE BLOCK LANCZOS METHOD

AUTHOR(S): J. D. Kress
S. B. Woodruff
G. A. Parker
R. T Pack

SUBMITTED TO: Computer Physics Communications

DISCLAIMER

Los Alamos Los Alamos National Laboratory
Los Alamos, New Mexico 87545

FORM NO 836 R4

Some Strategies for Enhancing the Performance of the Block Lanczos Method

*J. D. Kress, S. B. Woodruff *, G. A. Parker **, and R. T Pack*
Group T-12, MS-J569
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

ABSTRACT

The block Lanczos method is used to calculate the eigenfunctions for a generalized eigenvalue problem constructed for a finite element solution to a 2-dimensional Schrödinger equation on the surface of a hypersphere. This equation results from a treatment of the 3-dimensional reactive scattering problem using Adiabatically adjusting, Principal axes Hyperspherical (APH) coordinates. Three stategies are considered with respect to increasing the CPU performance of the block Lanczos (with selective orthogonalization) method: (1) the effect of varying the Lanczos block size; (2) the effect of solving the block tridiagonal ordinary eigenvalue problem upon *every other* Lanczos iteration; and, (3) the effect of dividing a single problem of finding p eigenvalues into a set of p_i problems, where each subproblem consists of finding p/p_i eigenvalues.

---

* *Group X-7, MS-B257*

** *Present Address: Department of Physics and Astronomy, University of Oklahoma, Norman, Oklahoma 73069*

An accurate quantum theory for the treatment of 3-dimensional reactive (atom-diatom) scattering has been formulated recently using Adiabatically adjusting Principal axes Hyperspherical (APH) coordinates.[1] Expansion of the scattering wavefunction in a sector-adiabatic basis and projection of this basis onto the full Hamiltonian yields a 2-dimensional surface Hamiltonian which depends parametrically on the sector hyperradius ρ_ξ. Expansion of the surface (eigen)functions in a finite element[2] (FE) basis set and projection onto the surface Hamiltonian yields a generalized eigenvalue problem,

$$\mathbf{H\Phi = S\Phi E}, \tag{1}$$

which must be solved for each value of ρ_ξ. Although a non-uniform mesh of elements is (usually) used which places the majority of the nodes in the classically allowed regions of the potential, the resulting Hamiltonian **H** and overlap **S** matrices are typically large (of order $n \sim 3000$ and of average half bandwidth of $m \sim 180$). Furthermore, for most scattering systems of interest, the $p \sim 100$ lowest eigenvalues E_i and eigenfunctions Φ_i (for $i = 1,\ldots,p$) are required at *each* ρ_ξ for $\xi = 1,\ldots,\sim 100$. Solving for the surface functions is the most expensive step of our scattering calculations. Therefore, identifying an efficient method to solve Eq. (1) is of utmost importance.

Application of both the Subspace Iteration[2] (SI) and block Lanczos (BL) methods to the calculation of surface functions has been reported in this issue by some of these authors[3] (hereafter referred to as paper I). The present paper describes some additional strategies we applied to the BL method in order to further improve the computational speed.

In practice, we transform[4] Eq. (1) to obtain an ordinary eigenvalue problem which is then solved using the BL method. Specifically, we form

$$\mathbf{H' = D^{-1/2}L^{-1}SL^{-T}D^{-1/2}} \tag{2}$$

(where **H** is factored as $\mathbf{LDL^T}$) which yields the transformed problem

$$\mathbf{H'\Phi' = \Phi'E'} \tag{3}$$

The block Lanczos procedure[5] reduces $\mathbf{H'}$ to a block tridiagonal form

$$\mathbf{T_k} = \begin{pmatrix} \mathbf{A_1} & \mathbf{B_1^T} & \mathbf{0} & \mathbf{0} \\ \mathbf{B_1} & \mathbf{A_2} & \ddots & \mathbf{0} \\ \mathbf{0} & \ddots & \ddots & \mathbf{B_{k-1}^T} \\ \mathbf{0} & \mathbf{0} & \mathbf{B_{k-1}} & \mathbf{A_k} \end{pmatrix} \tag{4}$$

where $\mathbf{T_k = Q_k^T H' Q_k}$. $\mathbf{Q_k}$ is constructed as $(\hat{\mathbf{Q}}_1, \hat{\mathbf{Q}}_2, \ldots \hat{\mathbf{Q}}_k)$ where each $\hat{\mathbf{Q}}_k$ is a block of n_b column vectors (n_b is defined as the Lanczos block size). The submatrices in Eq. (4), $\mathbf{A_k}$ and $\mathbf{B_k}$, are of dimension $n_b \times n_b$. The $\hat{\mathbf{Q}}_k$ are constructed iteratively once a starting residual matrix $\mathbf{R_0}$ ($\|\mathbf{R_0}\| \neq 0$) is specified. Then the algorithm proceeds for $k = 1,2,\ldots$ (the subscripts denote the k^{th} iteration):

1. Orthonormalize $\mathbf{R_{k-1}}$ (QR factorization): $\mathbf{R_{k-1} = \hat{Q}_k B_{k-1}}$

2. $\mathbf{R}_k \leftarrow \mathbf{H}'\hat{\mathbf{Q}}_k - \hat{\mathbf{Q}}_{k-1}\mathbf{B}_{k-1}^T$ $(\hat{\mathbf{Q}}_0 = 0)$.

3. $\mathbf{A}_k \leftarrow \hat{\mathbf{Q}}_k^T\mathbf{R}_k$.

4. $\mathbf{R}_k \leftarrow \mathbf{R}_k - \hat{\mathbf{Q}}_k\mathbf{A}_k$.

4'. Purge $\mathbf{R}_k$ of any converged eigenvectors of $\mathbf{T}_{k-1}$.

5. Solve using Rayleigh Quotient Iteration (RQI): $\mathbf{T}_k\Theta_k = \Theta_k\mathbf{E}_k$.

6. Check $\mathbf{E}_k$ for convergance of the p lowest eigenvalues. Iterate back to step 1 if necessary.

The version of BL implemented in our code is derived from the SNLASO code of Scott.[6] Along with the BL reduction scheme, SNLASO incorporates the selective orthogonaliztion[7] (SO) technique (step 4'). By purging the converged eigenvectors of $\mathbf{T}_{k-1}$ from $\mathbf{R}_k$, the loss of orthogonality (deviation of $\mathbf{Q}_k^T\mathbf{Q}_k$ from $\mathbf{I}$), which is fatal to the BL reduction, can be delayed to later iterations (larger values of k).

The formation of $\mathbf{H}'\hat{\mathbf{Q}}_k$ in step 2, consisting of a $\mathbf{LDL}^T$ solve and a matrix multiply [the matrix inversions in Eq. (2) are effected as linear equation solves], costs $2n \cdot (2m+1) \cdot n_b$ multiplicative operations (OPs) per iteration to execute. As discussed in paper I, step 2 is the rate-limiting step of the algorithm for small values of p (< 20) since $n \cdot m$ is typically large (200,000 - 500,000). The cost of step 5, that of finding p eigenvalues of $\mathbf{T}_k$ [which is of order $= (k \cdot n_b)$ and of half bandwidth $= (3/2 \cdot n_b)$] using the RQI method, is proportional to $p \cdot k \cdot (n_b)^3$ OPs to the leading term in n_b. On the average, we find that $k_{max} = 3p/n_b$ Lanczos iterations are required to converge p eigenvalues, therefore the total work (summed over k) to perform step 5 is proportional to $3p^3n_b + p^2(n_b)^2$ to the leading terms in p and n_b. The total work (summed over k) to perform step 2 is $6p \cdot n \cdot (2m+1)$. Therefore, for large values of p, the cost of step 5, which is scales as p^3, exceeds the cost of step 2, which only scales linearly in p.

The first strategy investigated involves studying the effect of varying n_b on the performance of the BL method. $\mathbf{H}$ is constructed from the surface Hamiltonian evaluated at $\rho_\xi = 5.0a_0$ using the potential energy surface (PES) for the $LiH + F \rightleftharpoons Li + HF$ system constructed by Laganà and coworkers[8a] from the *ab initio* energies of Chen and Schaefer.[8b] A uniform FE mesh was used which yielded a $\mathbf{H}$ and $\mathbf{S}$ of order $n = 1729$ and half bandwidth $m = 109$. The amount of CPU time required to converge the $p = 60$ lowest eigenvalues is presented in Table I as a function of n_b. (These and subsequent calculations were performed on a CRAY-XMP.) Also provided is the number of iterations (k_{max}) needed for convergence and the order $n_{max} = (k_{max} \cdot n_b)$ of $\mathbf{T}_k$ at the k_{max}^{th} iteration. For $n_b = 2$, the procedure fails due to a loss of orthogonality in the columns of $\mathbf{Q}_k$. For $n_b \geq 4$, the value of k_{max} decreases as n_b is increased, with the net result of n_{max} increasing slightly as a function of n_b. It appears that a minimum number of columns of $\mathbf{Q}_k$, $n_{max} \sim 200$, are necessary to converge the 60 lowest eigenvalues, independent of the value of n_b. The lowest CPU times are obtained for the minimum values of n_b (= 4 and 6). For larger values of n_b, even though k_{max} decreases (fewer iterations performed), the cost to diagonalize $\mathbf{T}_k$, which is proportional to $(n_b)^2$ [see above], is increasingly more expensive. The net result is an overall slower method with respect to increasing n_b.

To minimize the cost of performing step 5, two modifications to the algorithm were tested: 1) decreasing the number of $\mathbf{T}_k$ diagonalizations performed by skipping step 5 for a given interval of iterations; and, 2) dividing the one problem of calculating p eigenvalues into a set of p_i problems, where each subproblem

consists of finding $p' = (p/p_i)$ eigenvalues. The first modification was implemented by performing step 5 (and thus step 4') every n_i^{th} iteration (for $k = n_i, 2n_i, 3n_i, \ldots$). Since steps 1-4 do not depend explicitly on step 5, in principle $\mathbf{T_k}$ and $\mathbf{Q_k}$ can be constructed without the eigenvectors of $\mathbf{T_{k-1}}$. But the SO (step 4') is dependent on step 5, and by postponing the SO to every n_i^{th} iteration, non-orthogonality in $\mathbf{Q_k}$ can potentially appear after fewer total iterations. This behavior places a practical maximum on n_i.

The second modification effectively divides the p eigenvalues into p_i intervals. This approach is operationally possible since the SNLASO code has the capability of using a given number of converged eigenvectors at the beginning of the algorithm. These initial eigenvectors are purged from $\mathbf{R_0}$ and are treated as converged eigenvectors in the SO step. This prevents the duplication of the initial eigenvectors on subsequent iterations. The interval approach starts by finding the first set of p' eigenvalues. The first set of eigenvectors are then purged from the next choice of $\mathbf{R_0}$ used to generate the second set of eigenvalues. After the second set of eigenvalues are found, the eigenvectors from both the first and second set are purged from $\mathbf{R_0}$ constructed to determine the third set of eigenvalues. The above procedure is repeated until all p_i sets of eigenvalues are found. This strategy will be sucessful if the cost to solve the p_i smaller problems is less than the cost to solve the one large problem.

To test these two modifications, FE matrices were calculated at $\rho_\xi = 5.1a_0$ for the $LiH + F \rightleftharpoons Li + HF$ system using the same mesh as before. Table II lists the CPU times required to converge the $p = 100$ lowest eigenvalues using the BL method with a block size of $n_b = 8$ for various combinations of n_i and p_i. Also provided for comparison is the CPU time required for the same problem using the SI method with a subspace of size $q = 150$. The $(p_i = 1; n_i = 1)$ case corresponds to the "standard" method examined in paper I. This case yielded the largest CPU time of all of the BL runs in Table II. Also provided in Table II is the number of iterations (k_{max}) required to converge 100 eigenvalues. By delaying the diagonalization of $\mathbf{T_k}$ to every other iteration $(p_i = 1; n_i = 2)$, we obtain a decrease in CPU time of $\sim 33\%$ with respect to the standard method. By incrementing n_i by one again, $(p_i = 1; n_i = 3)$, we cause orthogonality problems and the method fails. In practice, for the various systems we have investigated, we have found that setting $n_i \geq 3$ is always fatal. The converged eigenvectors must be purged, at the minimum, upon every other iteration $(n_i = 2)$, if not upon every iteration $(n_i = 1)$.

The effect of dividing the $p = 100$ eigenvalue problem into two $p' = 50$ eigenvalue problems $(p_i = 2; n_i = 1)$ also yields a decrease in CPU time with respect to the standard method of $\sim 17\%$. For the $p_i \neq 1$ entries in Table II, the value of k_{max} for each subproblem is listed. Even though the sum total of $k_{max} = 53$ for $(p_i = 2; n_i = 1)$ is greater than the value of $k_{max} = 38$ for $(p_i = 1; n_i = 1)$, less CPU time is required for the former case, since the cost to perform step 5 scales as 2 times $(p')^3$ versus p^3 for the latter case. Eventually, increasing p_i to a larger value increases the overhead involved in performing steps 1-4, such that the $(p_i = 5; n_i = 1)$ case requires more CPU time than the $(p_i = 2; n_i = 1)$ case. The lowest CPU time achieved for this problem resulted from using *both* modifiations, $(p_i = 2; n_i = 2)$, yielding a decrease in CPU time of $\sim 39\%$ with respect to the standard method. Any further attempts to use both modifications in combination with $p_i \geq 2$ and $n_i \geq 2$ terminated the algorithm due to a loss of orthogonality.

At this point in the analysis a *caveat* must be put forth. Varying n_i and p_i from their standard values of 1 may introduce unwanted non-orthogonality which cannot be predicted in any systematic manner. We have found from experience that a given set of (p_i, n_i) will work correctly for a given **H** and **S**, but will fail for a different set of matrices (i.e., those evaluated at a different value of ρ_ξ). Since we require the whole set of surface functions calculated sequentially in ρ_ξ, typically for $\xi = 1, \ldots, 100$, we must have a robust procedure which will *not* fail, for example, when $\xi = 99$.

To provide a measure of the conditions we encounter when we generate surface functions necessary for nearly converged scattering results, we present some CPU requirements encountered for the $F + H_2 \rightleftharpoons HF + H$ system. Using the PES of ref. (9), FE matrices are constructed on a non-uniform mesh of order $n = 3291$ and of half bandwidth $m = 174$. To complete the sector-adiabatic basis when calculating scattering probabilities, 50 unconverged as well as the p=100 converged surface functions are required. Using the BL method, the extra 50 functions are obtained by retaining the lowest 150 eigenvalues from $\mathbf{T}_{k'}$ for $k' = k_{max}$. Using the subspace iteration (SI) method, the 150 functions are obtained by implementing a subspace of size $q = 150$. The BL code used for this example is slightly different than the version used above and in paper I. Machine language subroutines for performing matrix-matrix multiplies and for factoring a banded matrix are now implemented which increase the efficiency of step 5. This modified code was then applied using $n_b = 8$, $n_i = 2$, and $p_i = 1$. 156 sec of CPU time and $k_{max} = 50$ iterations were required to converge the problem. In comparision, the SI method required 15 subspace iterations and 363 sec of CPU time. The decrease in CPU time by a factor of ~ 2.3 for BL *vs.* SI is the best performance ratio we have observed for this class and size of problem.

In conclusion, the effect of three different strategies on the computational efficiency of the block Lanczos (with selective orthogonalization) method to solve large generalized eigenvalues problems was investigated. We found that it was advantageous to use the smallest Lanczos block size which does not introduce the loss of orthogonality in the Lanczos blocks. The second strategy, that of diagonalizing the reduced Lanczos matrix upon *every other* iteration, provided an increased efficiency of $\sim 33\%$ with respect to diagonalizing upon every iteration. The fastest solution approach for the test problem we studied combined the second strategy with the third, where the latter entailed dividing the single problem of finding 100 eigenvalues into two subproblems of finding 50 eigenvalues each. In general, we have found that all three strategies must be used judiciously, as they can all introduce unwanted (and fatal) non-orthogonality between the Lanczos blocks early in the iterative process.

REFERENCES

[1] a) R. T Pack, *Chem. Phys. Lett.* **108**, 333 (1984); b) R. T Pack and G. A. Parker, *J. Chem. Phys.* **87**, 3888 (1987).

[2] K. Bathe and E. L. Wilson, *Numerical Methods in Finite Element Analysis* (Prentice-Hall, New York, 1976).

[3] J. D. Kress, G. A. Parker, R. T Pack, and B. J. Archer, *Comp. Phys. Comm.*, this issue.

[4] T. Ericsson and A. Ruhe, *Math. Comp.* **35**, 1251 (1980).

[5] B. N. Parlett, *The Symmetric Eigenvalue Problem* (Prentice-Hall, Englewood Cliffs, NJ, 1980).

[6] D. S. Scott, *Oak Ridge National Laboratory, report CSD-48,UC-32.* This reference contains the documentation for the basic SNLASO package. The code we used, which was obtained from Argonne National Laboratory, is a modified version of the Oak Ridge code modified by D. S. Scott while at the University of Texas.

[7] B. N. Parlett and D. S. Scott, *Math. Comp.* **33**, 217 (1979).

[8] a) A. Laganà, E. Garcia, and O. Gervasi, *Faraday Discuss. Chem. Soc.* **84** (1987); b) M. M. L. Chen and H. F. Schaefer, *J. Chem. Phys.* **72**, 4376 (1980).

[9] R. Steckler, D. G. Truhlar, and B. C. Garrett, *J. Chem. Phys.* **82**, 5499 (1985); F. B. Brown, R. Steckler, D. W. Schwenke, D. G. Truhlar, and B. C. Garrett, *ibid.*, 188 (1985).

Table I. Block Lanczos (BL) method. $LiH + F$ test problem.
60 converged eigenvalues.
$\rho_\xi = 5.0a_0$, $n = 1729$, and $m = 109$.

n_b	k_{max}	$n_{max} = (k_{max} \cdot n_b)$	CPU time[a] (sec)
2		(Loss of orthogonality in $\mathbf{Q_k}$)	
4	45	180	88.5
6	32	192	88.4
8	27	216	102.0
10	22	220	100.7
12	20	240	114.7
14	18	252	122.4

[a]CPU time on a CRAY-XMP.

Table II. $LiH + F$ test problem.
100 converged eigenvalues.
$\rho_\xi = 5.1a_0$, $n = 1729$, and $m = 109$.

Block Lanczos (BL) $n_b = 8$			
p_i	n_i	k_{max}	CPU time[a] (sec)
1	1	38	127.6
1	2	38	88.5
1	3	(Loss of orthogonality in $\mathbf{Q_k}$)	
2	1	22,31	106.2
5	2	12,17,20, 21,24	110.0
2	2	22,32	78.0
Subspace Iteration (SI) $q = 150$[b]			
			136.3

[a]CPU time on a CRAY-XMP.
[b]Number of vectors in the subspace.