JOURNAL OF COMPUTATIONAL PHYSICS 64, 220–229 (1986)

A Generalized Version of the Recursive Residue Generation Method for Vector Computers*

RICHARD A. FRIESNER† AND ROBERT E. WYATT

Department of Chemistry and Institute for Theoretical Chemistry, The University of Texas, Austin, Texas 78712

AND

C. HEMPEL AND B. CRINER

Control Data Corporation, 7719 Woodhollow, Suite 210, Austin, Texas 78731

Received May 7, 1985; revised August 15, 1985

After decomposing a multidimensional potential into a sum of products of simpler terms, an algorithm is presented for computing **HU**, where **H** is a sparse broad-banded Hamiltonian matrix and **U** is an N-vector. Unlike other methods, the new algorithm is *explicitly vectorizable*. When implemented on a vector processor, the algorithm leads to greatly reduced computation times in applications of the recursive residue generation method to quantum dynamics problems. © 1986 Academic Press, Inc.

I. INTRODUCTION

The recursive residue generation method (RRGM) is a new approach to the calculation of time and temperature Green's function matrix elements [1, 2] and quantum statistical averages [3] for multidimensional effective Hamiltonians. Because the computation of eigenvectors is bypassed, extremely large (zero-order) basis sets (up to 10^6 states) can be employed [4]; thus, accurate benchmark calculations can be performed for state-to-state transition probabilities or time correlation functions for systems which have heretofore been studied only via uncontrolled approximations.

The most serious disadvantage of the RRGM to date has been substantial computation time and storage requirements. Within the context of current rapid development and accessibility of supercomputers, a straightforward solution to this problem is implementation of the approach on a vector machine. In favorable cases,

* Supported in part by a grant from the National Science Foundation and the Robert A. Welch Foundation.

† Alfred P. Sloan Foundation Fellow, 1984–1986.

0021-9991/86 $3.00

Copyright © 1986 by Academic Press, Inc.
All rights of reproduction in any form reserved.

reduction of computation time by factors of 20–100 can be achieved compared to mainframe scalar machines, e.g., the CYBER 170/750.

In the RRGM, Green's function matrix elements are computed via the Lanczos algorithm [5], by operating repeatedly with the Hamiltonian matrix **H** on the current ("old") recursion vector to obtain the new recursion vector: $\mathbf{U}_{\text{new}} = \mathbf{H}\mathbf{U}_{\text{old}}$. Performing this step to generate a sequence of recursion vectors consumes about 80% of the total CPU time in our previous versions of the RRGM program. We will focus on this critical part of the calculation in what follows.

Three crucial problems must be addressed in numerical implementation of the RRGM on a vector machine. First, the Hamiltonian should be represented in some way in fast memory; direct storage for large basis sets is impractical, while recomputation of matrix elements (or swapping from secondary storage) is unacceptable for an operation which must be repeated many times. Second, algorithms for the above step ($|\Psi_{\text{new}}\rangle = H|\Psi_{\text{old}}\rangle$) must be devised to take advantage of vector processing capabilities. Third, an appropriate basis set (one which minimizes off-diagonal elements, and therefore reduces the required dimensionality of the basis set, but is still amenable to the first two requirements) must be selected.

We will describe a set of techniques which efficiently addresses all of these concerns for a very general multidimensional Hamiltonian. The method is then applied to a five-mode coupled anharmonic system. Significant reductions in CPU time are obtained with the new algorithm.

We will explicitly consider only a single electronic potential surface (and hence, vibrational/rotational motion); however, the method is easily generalized to treat problems involving more than one electronic level (optical spectroscopy, electron transfer, etc.). Calculations of this type will be reported in subsequent publications.

In applications of the RRGM, we begin with the $N \times N$ Hamiltonian matrix (actually a compacted version, as described later) and a starting recursion vector, $\mathbf{U}_0$. After performing M steps of Lanczos recursion, we have an $M \times M$ tridiagonal (Jacobi) matrix **J**, which is the representation of H in the space spanned by the M recursion vectors, $\mathbf{U}_0$, $\mathbf{U}_1$,..., $\mathbf{U}_{M-1}$. (However, it is only necessary to store $\mathbf{U}_{j-1}$ and $\mathbf{U}_j$ in order to form $\mathbf{U}_{j+1}$.) Of course, each recursion vector is of length N; i.e., it is still an element of the N-dimensional space. It is important to appreciate that the physics developed by the M recursion vectors through **J** is *not* equivalent to carefully selecting M of the *original* basis functions in order to generate an $M \times M$ block of the full Hamiltonian matrix. In order to clarify this, consider an example with $N = 6$. Assume the Hamiltonian matrix

$$H = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 \\ 1 & 0 & 3 & 0 & 1 & 0 \\ 0 & 1 & 0 & 4 & 0 & 1 \\ 0 & 0 & 1 & 0 & 5 & 0 \\ 0 & 0 & 0 & 1 & 0 & 6 \end{bmatrix}$$

and the starting vector

$$\mathbf{U}_0 = [1, 0, 0, 0, 0, 0]^{\mathrm{tr}}.$$

Then the first three Krylov vectors ($\mathbf{K}_n = \mathbf{H}^n \mathbf{U}_0$) are

$$\mathbf{K}_0 = [1, 0, 0, 0, 0, 0]^{\mathrm{tr}},$$

$$\mathbf{K}_1 = [0, 1, 1, 0, 0, 0]^{\mathrm{tr}},$$

$$\mathbf{K}_2 = [2, 2, 3, 1, 1, 1]^{\mathrm{tr}}$$

and the Gram–Schmidt orthonormalized Krylov vectors (the Lanczos vectors) are

$$\mathbf{U}_0 = [1, 0, 0, 0, 0, 0]^{\mathrm{tr}}$$

$$\mathbf{U}_1 = (1/\sqrt{2})[0, 1, 1, 0, 0, 0]^{\mathrm{tr}}$$

$$\mathbf{U}_2 = (\sqrt{2}/\sqrt{7})[0, -\tfrac{1}{2}, \tfrac{1}{2}, 1, 1, 1]^{\mathrm{tr}}.$$

Note that the 2-dimensional space spanned by ($\mathbf{U}_0, \mathbf{U}_1$) contains *three* of the original basis states, while the 3-dimensional space spanned by ($\mathbf{U}_0, \mathbf{U}_1$, and $\mathbf{U}_2$) "feels" the full 6-dimensional space. The Hamiltonian is designing its M-dimensional subspace within the full N-space; since each Lanczos vector is a *linear combination* of the N basis vectors in no way is this equivalent to a subspace spanned by M of the *original* basis vectors.

II. Effective Hamiltonian and Basis Set Selection

We begin with a quantum mechanical Hamiltonian $H(q_1, \ldots, q_M, \partial/\partial q_1, \ldots, \partial/\partial q_M)$ where $q_1, \ldots, q_M$ is some canonical set of M coordinates. The determination of the optimal set of q's for a complicated system is nontrivial, but will not be pursued here.

A zeroth-order Hamiltonian, H_0, is defined as a sum of one-dimensional operators, i.e.,

$$H_0 = \sum_{j=1}^{M} h_j\left(q_j, \frac{\partial}{\partial q_j}\right). \tag{1}$$

In many cases, an adequate choice for H_0 will be obvious; in others, care should be taken to renormalize strong off-diagonal terms in H. Direct product basis functions for H_0

$$|n\rangle = |n_1 \cdots n_M\rangle = |n_1\rangle\, |n_2\rangle \cdots |n_M\rangle \tag{2}$$

are then defined by the relations

$$h_j\, |n_j\rangle = \varepsilon_{n_j}\, |n_j\rangle, \tag{3}$$

i.e., the $|n_j\rangle$ are eigenfunctions of h_j.

Each basis state $|n\rangle$ must be assigned a label (location) in the N element column vector representing the wavefunction $|\Psi\rangle$. We compute the state label L by the algorithm

$$L(n) = 1 + \sum_{j=1}^{M} (n_j - n_j^{(\min)}) \cdot l(j), \tag{4}$$

where $n_j^{(\min)}$ is the lowest quantum number in the basis for mode j, and

$$l(j) = \prod_{k=1}^{j-1} [n_k^{(\max)} - n_k^{(\min)} + 1] \tag{5}$$

with $l(1) \equiv 1$, and $n_j^{(\max)}$ is the largest quantum number in this basis. For example, in a three-mode problem ($M = 3$) with basis states $n_j = 0, 1, 2$ in each mode, the $N = 27$ basis functions are indexed according to the formula ($l(2) = 3$, $l(3) = 9$)

$$L(n_1, n_2, n_3) = 1 + n_1 + 3n_2 + 9n_3.$$

If h_j is a standard operator (e.g., the harmonic oscillator or rigid rotor Hamiltonian), the eigenfunctions will have well known properties, and matrix elements can often be obtained from simple algebraic formulas. However, there is no real barrier to employing an arbitrary one-dimensional basis obtained from numerical diagonalization of a more complicated potential, because each diagonalization need be performed only once as a pre-processing step. This flexibility enables accurate treatment of a wide variety of physical problems (e.g., a dissociative state or highly anharmonic double well-coupled to several vibrational modes).

We note here that it is possible to employ a nondirect product basis, thus allowing elimination of marginally relevant states via, e.g., energy criteria. This (and several other) improvements of the present algorithm will be reported elsewhere [6].

III. Representation of H as a Series of One-Dimensional Transition Vectors

To simplify the notation, we will assume that all derivatives are contained in H_0, so that we can define $V(q) = H - H_0$, where $q = \{q_1, ..., q_M\}$. In addition, we assume that V, a finite polynomial, is a sum of terms of the form

$$V(q) = \sum_{t=1}^{N_t} A_t \prod_{s=1}^{M_t} F_{ts}(q) \tag{6}$$

where N_t is the number of *terms*, M_t is the number of *steps* (multiplicative operators) in term t, the A_t are constants, and F_{ts} is a sum of one-mode terms,

$$F_{ts}(q) = \sum_{l=1}^{N_{ts}} f_{ts}^{(l)}(q_l). \tag{7}$$

This decomposition of V is very general and includes most situations of physical interest. As an example (which is relevant to the numerical results presented later) consider the five-mode, two-term potential, in which the second term has two multiplicative steps ($N_t = 2$, $M_1 = 1$, $M_2 = 2$):

$$V(q_1 \cdots q_5) = Cq_1^4 + Dq_1(q_2 + q_3 + q_4 + q_5), \tag{8}$$

so that

$$A_1 = C, \qquad A_2 = D,$$

$t = 1$

$$F_{11} = q_1^4$$

$t = 2$

$$F_{21} = q_1$$

$$F_{22} = q_2 + q_3 + q_4 + q_5 = f_{22}^{(1)} + f_{22}^{(2)} + f_{22}^{(3)} + f_{22}^{(4)}.$$

Thus, we can rewrite Eq. (8) as

$$V(q) = A_1 F_{11} + A_2 F_{21} F_{22}.$$

The motivation for this decomposition of V will become apparent in what follows.

As noted previously, the time-consuming step in the Lanczos algorithm (on which RRGM is based) is multiplying $\mathbf{H}$ onto an old recursion vector to obtain a new recursion vector,

$$\mathbf{U}_{\text{new}} = \mathbf{H}\mathbf{U}_{\text{old}}. \tag{9}$$

Of course, the action of $\mathbf{H}_0$ on $\mathbf{U}_{\text{old}}$ is trivial because the basis vectors are chosen to be eigenvectors of H_0:

$$H_0 |n\rangle = \varepsilon_n |n\rangle \tag{10}$$

where

$$\varepsilon_n \sum_{j=1}^{M} \varepsilon_{nj},$$

so that

$$H_0 \left\{ \sum_n c_n |n\rangle \right\} = \left\{ \sum_n c_n \varepsilon_n |n\rangle \right\}. \tag{11}$$

Thus, the difficulty arises in efficiently implementing the operation $\mathbf{V}\mathbf{U}_{\text{old}}$.

Consider first the effect of a function of **one** coordinate acting on a basis function $|n\rangle = |n_1\rangle\,|n_2\rangle \cdots |n_M\rangle$,

$$f(q_j)\,|n\rangle = \sum_m \langle m|\,f(q_j)\,|n_j\rangle\,|n_1\rangle \cdots |m\rangle \cdots |n_M\rangle. \tag{12}$$

Now if $\mathbf{U}_{\text{old}} = [C_1, C_2, \ldots, C_N]^{\text{tr}}$ (where N is the total number of basis functions, in which the basis states are indexed according to Eqs. (4)–(5) is the column vector corresponding to

$$|\Psi_{\text{old}}\rangle = \sum_{n=1}^{N} C_n\,|n\rangle,$$

then the following sequence of operations will generate $f(q_j)|\,\Psi_{\text{old}}\rangle$:

(1) Define a set of *transition vectors* $\mathbf{T}_{\Delta n_j}$, *row vectors* of length N whose elements are ($m = 0, 1, \ldots$, for a harmonic basis)

$$(\mathbf{T}_{\Delta n_j})_{m+1} = \langle n_j^{(m)}|\,f(q_j)|\,n_j^{(m)} + \Delta n_j\rangle, \tag{13}$$

where $n_j^{(m)}$ is the quantum number of basis function $(m+1)$ in the jth mode. In most cases, selection rules rigorously limit the number of transition vectors to only a few values of Δn_j, the *shift index*. Consider two examples, for four harmonic basis functions in mode j ($n_j^{(0)} = 0$, $n_j^{(1)} = 1, \ldots, n_j^{(3)} = 3$):

(a) $f(q_j) = q_j$. For this case, two transition vectors are required:

$$\begin{aligned} \Delta n_j = -1: &\qquad \mathbf{T}_{(-1)} = \frac{1}{\sqrt{2}}(0, \sqrt{1}, \sqrt{2}, \sqrt{3}) \\ \Delta n_j = +1: &\qquad \mathbf{T}_{(+1)} = \frac{1}{\sqrt{2}}(\sqrt{1}, \sqrt{2}, \sqrt{3}, 0). \end{aligned} \tag{14}$$

(b) $f(q_j) = 3q_j^2$. For this case, three transition vectors are required:

$$\begin{aligned} \Delta n_j = -2: &\qquad \mathbf{T}_{(-2)} = \tfrac{3}{2}(0, 0, \sqrt{2}, \sqrt{6}) \\ \Delta n_j = 0: &\qquad \mathbf{T}_{(0)} = \tfrac{3}{2}(1, 3, 5, 7) \\ \Delta n_j = +2: &\qquad \mathbf{T}_{(+2)} = \tfrac{3}{2}(\sqrt{2}, \sqrt{6}, 0, 0). \end{aligned} \tag{15}$$

This step, which builds and stores all required transition vectors, is done only once, before initiating the Lanczos recursion, and does not significantly contribute to the CPU time.

(2) The operation $f(q_j)|\,\Psi_{\text{old}}\rangle$ is now done in three steps.

Step (a). For each value of Δn_j, compute the new vector $\mathbf{D}_{\Delta n_j}$, whose mth element is defined by the *Schur product*

$$(\mathbf{D}_{\Delta n_j})_m = (\mathbf{T}_{\Delta n_j})_m \cdot (\mathbf{U}_{\text{old}})_m. \tag{16}$$

An example will clarify this step. For "example (b)" above, let $|\Psi_{\text{old}}\rangle \rightarrow \mathbf{U}_{\text{old}} = [C_1, C_2, C_3, C_4]^{\text{tr}}$. Then

$$\begin{array}{lcccc} & m=1 & m=2 & m=3 & m=4 \\ \hline \mathbf{D}_{(-2)} = (& 0, & 0, & 3\sqrt{2}/2C_3, & 3\sqrt{6}/2C_4 \\ \mathbf{D}_{(0)} = (& 3/2C_1, & 9/2C_2, & 15/2C_3, & 21/2C_4 \quad) \\ \mathbf{D}_{(+2)} = (& 3\sqrt{2}/2C_1, & 3\sqrt{6}/2C_2, & 0, & 0 \quad) \end{array} \tag{17}$$

Step (b). *Shift* the index of each element in $\mathbf{D}_{\Delta n_j}$ by an amount

$$\Delta L = \Delta n_j \cdot l(j), \tag{18}$$

where $l(j)$ is defined in Eq. (5).

Step (c). The *sum* of the new vectors produced in this way is $|\Psi_{\text{new}}\rangle$. The sum is over the different values of Δn_j.

It is best to return to the example that we just used in Step (a). If j means mode 1 in a "pure" one-dimensional example with four basis functions, then $\Delta L = \Delta n_j = 0, \pm 2$. We first obtain from Eq. (17) three *temporary* column vectors:

$$\begin{array}{ccc} \Delta n = -2 & \Delta n = 0 & \Delta n = +2 \\ \begin{bmatrix} 3\sqrt{2}/2C_3 \\ 3\sqrt{6}/2C_4 \\ 0 \\ 0 \end{bmatrix} & \begin{bmatrix} 3/2C_1 \\ 9/2C_2 \\ 15/2C_3 \\ 21/2C_4 \end{bmatrix} & \begin{bmatrix} 0 \\ 0 \\ 3\sqrt{2}/2C_1 \\ 3\sqrt{6}/2C_2 \end{bmatrix} \end{array} \tag{19}$$

As the final step, when these three vectors are added, we obtain the new vector representing $|\Psi_{\text{new}}\rangle$:

$$\mathbf{U}_{\text{new}} = \begin{bmatrix} 3/2C_1 + 3\sqrt{2}/2C_3 \\ 9/2C_2 + 3\sqrt{6}/2C_4 \\ 15/2C_3 + 3\sqrt{2}/2C_1 \\ 21/2C_4 + 3\sqrt{6}/2C_2 \end{bmatrix}. \tag{20}$$

The above vector may be readily verified by noting that the first term in each row arises from the *diagonal* part of the $3q^2$ operator ($a^\dagger$ and a are the usual harmonic oscillator raising and lowering operators)

$$(3q^2)_{\text{diag}} = \tfrac{3}{2}(aa^\dagger + a^\dagger a),$$

while the second term in each row results from either the $\Delta n = +2$ or the $\Delta n = -2$ component of $3q^2$:

$$(3q^2)_{\Delta n = +2} = \tfrac{3}{2}(a^\dagger)^2,$$

$$(3q^2)_{\Delta n = -2} = \tfrac{3}{2}(a)^2.$$

Both parts of Step (2), because they involve multiplying or equivalencing of contiguous arrays, will explicitly vectorize.

If we represent the two operations Schur product followed by index shift with an asterisk, then Steps (1) through (3) above produce the new vector:

$$\mathbf{U}_{\text{new}} = \sum_{\Delta n_j} \mathbf{T}_{n_j} * \mathbf{U}_{\text{old}}. \tag{21}$$

In some cases (e.g., polynomial $f(q)$ in a harmonic oscillator basis) the number of nonzero $\mathbf{T}_{\Delta n_j}$ vectors will be rigorously limited by selection rules. Otherwise, it is possible to make useful approximations by truncating the sum in Eq. (21) at a small value of Δn_j. Symmetry considerations may also reduce the number of required transition operations.

We now consider the action of a product of two one-dimensional functions $f_1(q_1) f_2(q_2)$ on $|\Psi_{\text{old}}\rangle$. Assume that f_1 has N_1 transition vectors associated with it, and f_2 has N_2 vectors. A straightforward matrix representation of this operator in the direct product basis $|n\rangle$ would then contain $N_1 \cdot N_2$ codiagonals, thus requiring storage of $N_1 N_2 \cdot N$ elements and application of $N_1 N_2$ transition vectors (N is the basis size). However, a much more efficient procedure is to act *first* with f_2 on $|\Psi_{\text{old}}\rangle$ and then act with f_1 on the intermediate result. This requires only $(N_1 + N_2) N$ storage elements and $(N_1 + N_2)$ transition operators! As N_1 or N_2 becomes large or as more terms appear in the product in Eq. (21), the saving of computation time and storage space becomes very significant.

Similarly, if we want to apply a term of the form

$$\{[f_1(q_1) + f_2(q_2)] \cdot f_3(q_3)\} |\Psi_{\text{old}}\rangle, \tag{22}$$

the greatest efficiency is obtained by first acting with f_3, and then applying f_2 and f_1 to the resulting vector.

The Schur product and index shift involved in the "* product" in Eq. (21) (which utilizes factorization of a Hamiltonian into sums and products of much simpler terms) is the reason for the efficiency of the present method on vector computers.

IV. Computational Results

We consider a quartic oscillator coupled to four harmonic "bath" modes, with V given by Eq. (8). This generic Hamiltonian is similar to one studied previously

[1, 2], but contains the full quartic potential instead of only those anharmonic terms which are retained in the rotating wave approximation.

For the potential in Eq. (8), there are five transition vectors associated with the first term (Cq_1^4). They are associated with $\Delta n_1 = 0, \pm 2, \pm 4$. In addition, there are ten transition vectors associated with the second term; they are associated with $\Delta n = \pm 1$ for the five coordinates $q_1, ..., q_5$.

For the present study, we employ a harmonic oscillator basis set for all five modes. However, note that for large anharmonicity, it would be useful to incorporate the quartic term in the zeroth-order Hamiltonian for the anharmonic oscillator, thus reducing the required number of basis functions for this mode.

The purpose of the present study is to compare computation times on the CYBER 205 for code written as direct, nonvectorizing matrix multiplication with code utilizing the new algorithms described here. This comparison will illustrate the advantages of semantic vectorization.

For this five-mode problem, with four basis functions in each mode (so that $N = 5^5 = 1024$), 290 different transition amplitudes were computed for 800 time steps. The original code, not employing the algorithm described here, executed in 1050 sec. on the CYBER 170/750. When executed with the automatic vectorizer on the two-vector pipeline CYBER 205, the execution time dropped by a factor of seven to 150 sec. With the new algorithm, the execution time dropped by *another factor of four* to 35 sec. The new code thus executed about 30 times faster on the 205 compared to the original code on the 170/750.

V. Conclusion

The preceding discussion gives considerable insight into the types of problems for which the RRGM algorithm is useful. First, if the Hamiltonian matrix is dense, computation times for the Lanczos procedure will become extremely large; unless time is of no concern, it seems unlikely that a full matrix too large to be diagonalized by conventional methods can be profitably studied by the RRGM. An exception occurs if only a few Green's function matrix elements are required; then, the Lanczos procedure will be preferable independent of the structure of the Hamiltonian matrix.

On the other hand, a sparse matrix with a very small bandwidth (the trivial case is a tridiagonal matrix) is a quite suitable form for attack via standard banded matrix techniques. Again, it seems unlikely that a matrix too large to store in banded form would be amenable to study via RRGM without unacceptable computational expenditures.

The interesting cases are then precisely those discussed in this paper: *matrices which are sparse but possess a large bandwidth.* This situation is a direct consequence of representing a multidimensional potential in a direct product of one-

dimensional basis functions. It is thus the typical situation for realistic anharmonic potentials in molecular systems.

For potentials which can be decomposed into sums of products of simpler terms, Eq. (6), the new explicitly vectorizable algorithm presented here leads to a significant reduction in CPU time on current vector computers. As a result, the RRGM should be applicable to a broader range of chemically interesting problems.

ACKNOWLEDGMENTS

We thank Keith Knapp of the Control Data Corporation for his generous support of this work. In addition, we thank Claude Leforestier for helpful discussions.

REFERENCES

1. A. NAUTS AND R. E. WYATT, *Phys. Rev. Lett.* **51**, 2238 (1983).
2. A. NAUTS AND R. E.WYATT, *Phys. Rev. A* **30**, 872 (1984).
3. R. A. FRIESNER AND R. E. WYATT, *J. Chem. Phys.* **82**, 1973 (1985).
4. K. F. MILFELD AND R. E. WYATT, to be published.
5. C. LANCZOS, *J. Res. Natl. Bur. Stand.* **45**, 255 (1950).
6. C. LEFORESTIER, R. FRIESNER, AND R. E. WYATT, in preparation.