Supercomputer Applications, edited by
R. W. Numrich (Plenum Press, New York, 1985),
pp. 215-254.

LARGE-SCALE QUANTUM MECHANICAL SCATTERING CALCULATIONS ON VECTOR COMPUTERS

David W. Schwenke and Donald G. Truhlar

Department of Chemistry and Supercomputer Institute
University of Minnesota
Minneapolis, MN 55455

I. INTRODUCTION

Energy transfer in molecular collisions is a very fundamental problem in chemical physics, both experimentally and theoretically. Quantitative state-to-state cross sections for energy transfer processes are important for understanding and modelling many kinds of systems, including lasers, shocked gases, planetary atmospheres, and systems containing excited reaction products of any kind. In theory these cross sections can be calculated from first principles using quantum mechanics but in practice this involves severe computational difficulties. The first step is the calculation of molecular interaction potentials, which are a consequence of the electronic structure of the interacting molecules for various geometries. There has been enormous progress in this area in the last few years,[1-3] but we shall not consider it further in this chapter. Instead we shall concentrate on the second step, the dynamical problem that yields the desired inelastic cross sections from the intermolecular potential. Only when both steps have been solved adequately will the ab initio method have reached fruition, but the techniques involved in the two steps are very specialized and to a large extent progress on these steps may occur separately and in parallel. There are two basic approaches to the dynamics problem. First, one can try to develop reliable methods based on simplifying approximations, such as using semiclassical or classical methods or low-order perturbation theory. Second, one can attempt a direct solution of the Schroedinger equation. Since the Schroedinger equation for the systems of interest is not solvable analytically, we are forced to use numerical methods. In this context an "exact" solution of the Schroedinger equation is actually a "converged" solution, converged to within some acceptable but small margin of error (e.g., a few per cent in

the cross section of interest) with respect to the relevant convergence parameters. We shall discuss a general and well studied method for obtaining such a direct solution. This method is variously called the close coupling or coupled-channels (CC) method.[4-6] It is based on a steady-state description of the scattering process in terms of the time-independent Schroedinger partial differential equation, and it converts this equation into a set of coupled ordinary differential equations by expansion of the D-dimensional solution in (D-1)-dimensional basis functions. The resulting CC equations may be cast in many forms. One may directly solve for the wave function, or one may use invariant imbedding.[7,8] The particular invariant imbedding algorithm we use is called R matrix propagation,[5,6,9-12] where the R matrix is Wigner's derivative matrix[13] (not to be confused with another R matrix, the reactance matrix,[4] that also plays a role in the problem). This method involves the calculation of large numbers of multi-dimensional integrals and the diagonalization, inversion, and multiplication of large matrices, as well as linear-equation solving; thus it is well suited to attack on vector computers. Other methods of solving the CC equations involve somewhat different mixes of the various matrix operations, but do not differ in any fundamental way. We shall not consider other methods of solving the Schroedinger equation for dynamics problems in detail. Some of these, like full multi-dimensional finite-difference[14] or finite-element[15] solutions, have been applied successfully to problems of artificially reduced dimensionality and are also well suited to modern vector computers, but at present they seem less efficient for solving real problems in the three-dimensional world.

We concentrate in this chapter on collisions in which both partners start and remain in their ground electronic state and no rearrangements occur. The techniques discussed here are also useful for electronically inelastic[16,17] and reactive collisions,[9,18] as well as for electron scattering.[12,19]

The "state of the art" for converged calculations is the calculation of vibrational-rotational transition probabilities in atom-diatom collisions involving a light diatom[4,20] (heavy molecules have more closely spaced states and hence require larger expansion bases and bigger matrices to obtain converged results for all the states of interest) or the calculation of probabilities for pure rotational quantum number changes in collisions of two light diatoms assumed to be rigid.[21] Already at the level of complexity of rigid diatom-diatom collisions, the number of arithmetic operations scales as the twelfth power of the maximum rotational quantum number involved, and the computational expense rapidly becomes prohibitive.[22] Here we report our

attempts to converge vibrational-rotational state-to-state transitions in diatom-diatom collisions, specifically the collision of two hydrogen fluoride (HF) molecules with enough total energy for both molecules to be vibrationally excited. This allows, even with indistinguishable molecules, one to observe vibration-to-vibration (V-V) energy transfer in which a quantum of vibrational energy originally on one molecule is transferred to the other, e.g., a process in which initially each molecule has one quantum of vibrational excitation and finally one molecule has two and the other has none. Such V-V energy transfer processes are of great interest because they provide the dominant pathway to redistribute vibrational energy in many non-equilibrium chemical mixtures. Prior to the availability of supercomputers, though, accurate quantum mechanical calculations on V-V energy transfer were prohibitively time consuming. Even with supercomputers these calculations are very expensive; thus, for the work we discuss here, we limit ourselves to the case where the total angular momentum is restricted to be zero. Although one cannot calculate observable cross sections or rate constants from this subcase, except at extremely low translational energies, it does serve as a prototype, and exact solutions of the quantal dynamics for this prototype can serve as practical benchmarks for testing approximate, but less computationally expensive, methods for attacking this kind of problem.

Section II gives an introduction to the theory, section III describes our numerical algorithm, section IV discusses the details of the inputs of the calculation, section V discusses vectorizing our program on the Cyber 205 and Cray-1 computers, and section VI compares execution times for scalar and vector versions of the code on one minicomputer and two supercomputers. Section VII gives preliminary results, and section VIII gives our conclusions. Our equations will be given in a form that is valid in any consistent set of units; however we find it convenient to quote numerical parameters in Hartree atomic units. In this system of units, the unit of length is the bohr, abbreviated a_0 ($1\ a_0 = 0.5291771 \times 10^{-10}$ m), the unit of mass is the electron mass, abbreviated m_e ($1\ m_e = 9.109534 \times 10^{-31}$ kg), the unit of energy is the hartree, abbreviated E_h ($1\ E_h = 27.21161$ eV $= 4.359814 \times 10^{-18}$ J), and $\hbar$ (Planck's constant divided by 2π) has the value unity.

II. THEORY

All of the information concerning a collision of molecule A with molecule B is contained in the scattering wave function $\psi_{n_0}(\underset{\sim}{x},\vec{r},E)$, which solves the Schroedinger equation

$$H\psi_{n_0} = E\psi_{n_0}. \tag{1}$$

The symbol n_0 stands for all of the quantum numbers necessary to specify the initial state of the system, $\underset{\sim}{x}$ is the collection of all internal coordinates, with the exception of $\vec{r}$, which is the vector connecting the center of mass of A to that of B, E is the total energy, and H is the system Hamiltonian:

$$H = -\frac{\hbar^2}{2\mu}\nabla^2_{\vec{r}} + H_{int}(\underset{\sim}{x}) + V(\underset{\sim}{x},\vec{r}) \quad , \tag{2}$$

where μ is the reduced mass for relative translational motion of A with respect to B, $\nabla^2_{\vec{r}}$ is the Laplacian with respect to $\vec{r}$, H_{int} is the "internal Hamiltonian", defined as the sum of the Hamiltonians of the isolated A and B subsystems, and V is the interaction potential, which means that V vanishes in the large-r limit. For the next step it is useful to separate $\nabla^2_{\vec{r}}$ into its radial and angular parts and to partition the internal Hamiltonian into a conveniently diagonalized part, $\tilde{H}_{int}$ (which may in some cases simply be H_{int}), and the rest ΔH_{int}. Then we treat the angular part of $-\hbar^2\nabla^2_{\vec{r}}/2\mu$ together with H_{int}; their sum is called the primitive Hamiltonian H^o. This yields

$$H = -\frac{\hbar^2}{2\mu}\left[\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right)\right] + H^o(\underset{\sim}{x},\vec{r}) + V(\underset{\sim}{x},\vec{r}) \quad , \tag{3}$$

where

$$H^o = \frac{\ell^2_{\hat{r}}}{2\mu r^2} + \tilde{H}_{int}(\underset{\sim}{x}) \quad , \tag{4}$$

$$V(\underset{\sim}{x},\vec{r}) = V(\underset{\sim}{x},r) + \Delta H_{int}(\underset{\sim}{x}) \quad , \tag{5}$$

and $\ell^2_{\hat{r}}$ is the quantum mechanical operator for the square of the orbital angular momentum of the relative translational motion of A with respect to B.

To determine ψ_{n_0}, we expand it in terms of simultaneous matrix eigenvectors of $\tilde{H}_{int}$ and $\ell^2_{\hat{r}}$ defined by

$$\int d\hat{r}\int d\underset{\sim}{x}\; X^*_m(\underset{\sim}{x},\hat{r})\tilde{H}_{int}(\underset{\sim}{x})X_n(\underset{\sim}{x},\hat{r}) = \delta_{mn}\tilde{\varepsilon}_n \quad , \tag{6}$$

$$\ell^2_{\hat{r}}X_n(\underset{\sim}{x},\hat{r}) = \hbar^2\ell_n(\ell_n+1)X_n(\underset{\sim}{x},\hat{r}) \quad , \tag{7}$$

and

$$\int d\hat{r}\int d\underset{\sim}{x}\; X^*_m(\underset{\sim}{x},\hat{r})X_n(\underset{\sim}{x},\hat{r}) = \delta_{mn} \quad , \tag{8}$$

where δ_{mn} is the Kronecker delta, and eq. (8) is an orthonormality condition. The expansion is

$$\psi_{n_0}(\underset{\sim}{x},\vec{r},E) = \frac{1}{r}\sum_{n=1}^{N} X_n(\underset{\sim}{x},\hat{r}) f_{nn_0}(r,E) \quad , \tag{9}$$

and the terms in this expansion are called channels. In these equations, $\hat{r}$ is the unit vector which has the direction as $\vec{r}$. Substituting eq. (9) into eq. (1), multiplying by $rX_m^*(\underset{\sim}{x},\hat{r})$, and integrating over $\underset{\sim}{x}$ and $\hat{r}$ yields

$$\left(-\frac{\hbar^2}{2\mu}\frac{d^2}{dr^2} + \frac{\ell_m(\ell_m+1)\hbar^2}{2\mu r^2} + \tilde{\varepsilon}_m\right) f_{mn_0}(r,E) + \sum_n V_{mn}(r) f_{nn_0}(r,E)$$

$$= E f_{mn_0}(r,E) \quad , \quad m = 1,2,\ldots,N, \tag{10}$$

where

$$V_{mn}(r) = \int d\underset{\sim}{x} \int d\hat{r}\; X_m^*(\underset{\sim}{x},\hat{r}) V(\underset{\sim}{x},\hat{r}) X_n(\underset{\sim}{x},\hat{r}) \quad . \tag{11}$$

Equations (10) are the close-coupling equations, and they can be written in the form

$$\frac{d^2}{dr^2}\underset{\sim}{f}(r,E) = \underset{\sim}{D}(r,E)\underset{\sim}{f}(r,E) \quad , \tag{12}$$

where $\sim$ under a symbol denotes a matrix (except for $\underset{\sim}{x}$, for which it denotes a collection of coordinates). The components of $\underset{\sim}{f}$ are just the f_{mn} of eq. (9), and the elements of $\underset{\sim}{D}$ are given by

$$D_{mn}(r,E) = \frac{2\mu}{\hbar^2} V_{mn}(r) + \delta_{mn}\left[\ell_n(\ell_n+1)/r^2 - \tilde{k}_n^2\right] \quad , \tag{13}$$

where

$$\tilde{k}_n^2 = 2\mu(E - \tilde{\varepsilon}_n)/\hbar^2 \quad . \tag{14}$$

$\tilde{k}_n$ is called the primitive wave number. $\underset{\sim}{D}$ is real and symmetric. Notice that the rows of $\underset{\sim}{f}$ correspond to different channels, which are coupled, and the columns to different initial conditions. The channels are ordered so that $\tilde{k}_n^2 \geq \tilde{k}_m^2$ if and only if $n \geq m$.

In order to discuss the large-r boundary conditions on eq. (12) we must consider a transformation that diagonalizes $\underset{\sim}{D}$ at large r. Furthermore, in the R matrix propagation algorithm considered in the next section, we will use transformations to basis functions that diagonalize $\underset{\sim}{D}$ at finite r. At any r we may define functions

$$Z_m(\underset{\sim}{x},\vec{r}) = \sum_{n=1}^{N} U_{nm}(r) X_n(\underset{\sim}{x},\hat{r}) \tag{15}$$

such that

$$\sum_{k,\ell=1}^{N} U_{kn}(r) D_{k\ell}(r,E) U_{\ell m}(r) = \delta_{nm}\left[\lambda_{mm}(r,E)\right]^2 \quad . \tag{16}$$

Notice that, because of the way that E appears in eq. (13), i.e., only on the diagonal, the eigenvalues λ_{mm}^2 depend on E, but the eigenvectors, which are the columns of $\underset{\sim}{U}$, do not. If we use the Z_m as new basis functions, called adiabatic basis functions, the expansion of the wave function becomes

$$\psi_{n_0}(\underset{\sim}{x},\vec{r},E) = \frac{1}{r}\sum_{m=1}^{N} Z_m(\underset{\sim}{x},\vec{r})g_{mn_0}(r,E) \tag{17}$$

where the g_{mn_0} are easily related to the f_{mn_0}. The terms in (17), like those in (9), are called channels. The boundary conditions on the solutions to the close coupling equations may be given in many equivalent forms, but the one that is most convenient for our algorithm is

$$g_{mn} \underset{r\to 0}{\sim} 0 , \qquad 1 \le n, m \le N , \tag{18}$$

$$g_{mn} \underset{r\to\infty}{\sim} \begin{cases} \delta_{mn}\sin[k_m(E)r - \ell_m\pi/2] + a_{mn}(E)\cos[k_m(E)r - \ell_m\pi/2] , \\ \qquad\qquad 1 \le n, m \le P^o , \\ \delta_{mn}\, b\exp[k_m(E)r] + a_{mn}(E)\exp[-k_m(E)r] , \\ \qquad\qquad P^o + 1 \le n, m \le N , \end{cases} \tag{19}$$

where b is arbitrary and k_n is an asymptotic wave number defined by

$$k_n = \lim_{r\to\infty} |\lambda_{nn}(r,E)| . \tag{20}$$

"Open" channels are defined as those with positive k_n^2, and P^o is the number of open channels in the basis. Channels that are not open are called "closed". If H_{int} is the same as $\tilde{H}_{int}$, then k_n equals $|\tilde{k}_n|$ and the boundary conditions (18)-(19) apply to the f_{mn} as well as the g_{mn}.

The final result of the calculation is the $P^o \times P^o$ unitary scattering matrix $\underset{\sim}{S}$ defined by

$$\underset{\sim}{S} = [\underset{\sim}{1} + i\underset{\sim}{R}(E)][\underset{\sim}{1} - i\underset{\sim}{R}(E)]^{-1} , \tag{21}$$

where $\underset{\sim}{1}$ is the unit matrix, $i^2 = -1$, and $\underset{\sim}{R}$ is the real symmetric $P^o \times P^o$ reactance matrix with elements

$$R_{nm} = k_n^{\frac{1}{2}} a_{nm}(E) k_m^{-\frac{1}{2}} , \qquad 1 \le n, m \le P^o . \tag{22}$$

All physical observables may be calculated from the scattering matrix by standard formulas.

The size of N, the number of terms in eq. (9), is determined by including all of the states of interest plus a sufficient number of other states in order for eq. (9) to accurately represent ψ_{n_0}. For the present study, the quantum numbers specified by n or n_0 are[23] v_1 and v_2, the vibrational

quantum numbers of the two molecules, j_1 and j_2, the rotational quantum numbers of the two molecules, j_{12}, the angular momentum quantum number associated with the vector sum of the rotational angular momenta of the two molecules, ℓ (called ℓ_n above when necessary to indicate that ℓ may be different for different n), the orbital angular momentum for relative motion, J and M, the total angular momentum and its projection on a laboratory-fixed Z axis, and η, the symmetry under interchange of the two molecules. The indistinguishability of identical molecules in quantum mechanics makes it impossible to distinguish which molecule has which set of v and j quantum numbers. The parity ζ is $(-1)^{j_1+j_2+\ell}$ and it too is specified by these quantum numbers.

Since J, M, η, and ζ are good quantum numbers, the matrix D is block diagonal in them, and using simultaneous eigenfunctions of their associated operators as our basis functions uncouples the solutions into noninteracting components and reduces the computational work. In the numerical applications considered below we shall only consider the block specified by $J = M = 0$ and $\eta = \zeta = +1$.

III. NUMERICAL METHODS

We solve eq. (12) using R matrix propagation. In this method one first subdivides the coordinate r into N_s sectors with sector midpoints $r_C^{(i)}$ and widths $h^{(i)}$ such that

$$r_C^{(i+1)} = r_C^{(i)} + [h^{(i+1)} + h^{(i)}]/2 \ . \tag{23}$$

In sector i, it is convenient to expand the wavefunction ψ_n in terms of sector-dependent functions defined by

$$Z_m^{(i)}(\underset{\sim}{x},\hat{r}) = \sum_{n=1}^{N} T_{nm}^{(i)} X_n(\underset{\sim}{x},\hat{r}) \ , \qquad 1 \le m \le P^{(i)} \ . \tag{24}$$

The $N \times P^{(i)}$ rectangular matrix $\underset{\sim}{T}^{(i)}$ is made up of the first $P^{(i)}$ columns of the $N \times N$ matrix $\underset{\sim}{U}^{(i)}$, where $\underset{\sim}{U}^{(i)}$ diagonalizes $\underset{\sim}{D}$:

$$\sum_{k,k'=1}^{N} U_{kn}^{(i)} D_{kk'}(r_C^{(i)},E) U_{k'm}^{(i)} = \delta_{nm}[\lambda_{nn}^{(i)}(E)]^2 \ . \tag{25}$$

In terms of the new functions $Z_m^{(i)}$, the wave function is

$$\psi_n(\underset{\sim}{x},\vec{r},E) = \frac{1}{r} \sum_{m=1}^{P^{(i)}} Z_m^{(i)}(\underset{\sim}{x},\hat{r}) g_{mn}^{(i)}(r,E) \ . \tag{26}$$

The new radial functions $\underset{\sim}{g}^{(i)}$ are related to the functions $\underset{\sim}{f}$ by

$$g_{mn}^{(i)}(r,E) = \sum_{k=1}^{N} T_{km}^{(i)} f_{kn}(r,E) \quad , \qquad 1 \le m, n \le P^{(i)} \; . \tag{27}$$

The $\underset{\sim}{g}^{(i)}$ solve the equation

$$\frac{d^2}{dr^2}\, \underset{\sim}{g}^{(i)}(r,E) = \underset{\sim}{L}^{(i)}(r,E)\underset{\sim}{g}^{(i)}(r,E) \; , \tag{28}$$

where

$$L_{nm}^{(i)}(r,E) = \sum_{k,\ell=1}^{N} T_{kn}^{(i)} D_{k\ell}(r,E) T_{\ell m}^{(i)} \; , \qquad 1 \le n, m \le P^{(i)} \; . \tag{29}$$

According to eq. (28), the channels of eq. (26) are uncoupled at $r = r_C^{(i)}$.

In order to enforce eq. (18), it would be convenient if $r_C^{(1)} - h^{(1)}/2$ were equal to zero. However, in atom-molecule and molecule-molecule collisions one finds

$$V_{nn}(r) >> E \; , \qquad \text{all } n,\ r << \sigma \; , \tag{30}$$

where σ is the distance at which the two subsystems begin to repel strongly. This implies that, as a function of r in the decreasing r direction, all $f_{mn}(r)$ and hence all $g_{mn}(r)$ decrease rapidly and are totally negligible for r less than some finite nonzero value. We chose such a nonzero value of r as $r_C^{(1)}$ and thereby avoid the work of propagating the solution to the equations over the region where it is known to be essentially zero(in the present calculations we use $r_C^{(1)}$ equal to 3.0 a_0).

We begin with $P^{(1)} = N$. At large r, because of eq. (19), f_{mn} for $m > P^o$ rapidly decays to zero. In this region we use a criterion described below to allow $P^{(i)}$ to decrease. We constrain $P^{(i+1)}$ to equal $P^{(i)}$ or $P^{(i)} - 1$ in all cases to simplify the algorithm. We also enforce the constraint $P^{(i)} \ge P^o$ for all (i). The $2P^{(i)} \times 2P^{(i)}$ sector propagator $\underset{\sim}{P}^{(i)}$ is defined by

$$\underset{\sim L}{G}^{(i)}(E) = \underset{\sim}{P}^{(i)}(E)\underset{\sim R}{G}^{(i)}(E) \; , \tag{31}$$

where the $2P^{(i)} \times 2P^{(i)}$ matrix $\underset{\sim}{G}^{(i)}$ is given by

$$\underset{\sim}{G}^{(i)}(E) = \begin{bmatrix} \underset{\sim}{g}^{(i)}(r,E) \\ \underset{\sim}{g}'^{(i)}(r,E) \end{bmatrix} \; , \tag{32}$$

where $\underset{\sim}{g}'^{(i)}(r,E)$ denotes $d\underset{\sim}{g}^{(i)}/dr$ and where L and R denote the left $[r_L^{(i)} = r_C^{(i)} - h^{(i)}/2]$ and right $[r_R^{(i)} = r_C^{(i)} + h^{(i)}/2]$ sides of a sector. We partition $\underset{\sim}{P}^{(i)}(E)$ so that

$$\underset{\sim}{P}^{(i)}(E) = \begin{bmatrix} \underset{\sim}{P}_1^{(i)}(E) & \underset{\sim}{P}_2^{(i)}(E) \\ \underset{\sim}{P}_3^{(i)}(E) & \underset{\sim}{P}_4^{(i)}(E) \end{bmatrix} , \tag{33}$$

and the matrices $\underset{\sim}{P}_j^{(i)}(E)$ are all square.

We use the first-order Magnus method[24] for $\underset{\sim}{P}^{(i)}$:

$$[\underset{\sim}{P}_1^{(i)}(E)]_{nm} = [\underset{\sim}{P}_4^{(i)}(E)]_{nm} = \begin{cases} \delta_{nm} \cosh[-h^{(i)}|\lambda_{nn}^{(i)}(E)|] , & \lambda_{nn}^{(i)2}(E) > 0 , \\ \delta_{nm} \cos[-h^{(i)}|\lambda_{nn}^{(i)}(E)|] , & \lambda_{nn}^{(i)2}(E) < 0 , \end{cases} \tag{34}$$

$$[\underset{\sim}{P}_2^{(i)}(E)]_{nm} = \begin{cases} \delta_{nm}|\lambda_{nn}^{(i)}(E)|^{-1} \sinh[-h^{(i)}|\lambda_{nn}^{(i)}(E)|] , & \lambda_{nn}^{(i)2}(E) > 0 , \\ \delta_{nm}|\lambda_{nn}^{(i)}(E)|^{-1} \sin[-h^{(i)}|\lambda_{nn}^{(i)}(E)|] , & \lambda_{nn}^{(i)2}(E) < 0 , \end{cases} \tag{35}$$

$$[\underset{\sim}{P}_3^{(i)}(E)]_{nm} = \begin{cases} \lambda_{nn}^{(i)2}[\underset{\sim}{P}_2^{(i)}(E)]_{nm} , & \lambda_{nm}^{(i)2}(E) > 0 , \\ -\lambda_{nn}^{(i)2}[\underset{\sim}{P}_2^{(i)}(E)]_{nm} , & \lambda_{nm}^{(i)2}(E) < 0 . \end{cases} \tag{36}$$

The error of this propagator is proportional to $h^{(i)3} d\underset{\sim}{L}^{(i)}/dr$.

We can choose the stepsize so that the error term is small. We estimate the error in sector $(i+1)$ by

$$\text{error} \propto h^{(i+1)3}[\frac{1}{N} \sum_{j=1} (\frac{dD_{jj}^{(i)}}{dr})^2]^{\frac{1}{2}} . \tag{37}$$

This translates into the algorithm

$$h^{(i+1)} = \min \begin{cases} \text{EPS}[\frac{1}{N} \sum_{j=1}^{N} (\frac{D_{jj}^{(i)} - D_{jj}^{(i-1)}}{r_C^{(i)} - r_C^{(i-1)}})^2]^{-1/6} \\ h_{max} \end{cases} \tag{38}$$

where EPS and h_{max} are input parameters, and $\min\{^a_b$ means the minimum of a and b.

Continuity of the functions $\underset{\sim}{g}^{(i)}$ across sector boundaries can be expressed by

$$\underset{\sim}{g}_R^{(i-1)}(E) = \underset{\sim}{T}(i-1,i)\underset{\sim}{g}_L^{(i)}(E) \tag{39}$$

$$\underset{\sim}{g}'_R{}^{(i-1)}(E) = \underset{\sim}{T}(i-1,i)\underset{\sim}{g}'_L{}^{(i)}(E) \tag{40}$$

where the overlap matrix $\underset{\sim}{T}(i-1,i)$ is defined by

$$T_{nm}(i-1,i) = \sum_{k=1}^{N} T_{kn}^{(i-1)} T_{km}^{(i)} , \qquad 1 \le n, m \le P^{(i)} . \tag{41}$$

Thus when $P^{(i-1)} \neq P^{(i)}$ only the upper left $P^{(i)} \times P^{(i)}$ part of $\underset{\sim}{g}_R^{(i-1)}$ is used to calculate $\underset{\sim}{g}_L^{(i)}$. We now define the sector R matrix $\underset{\sim}{r}^{(i)}$ by

$$\begin{bmatrix} \underset{\sim}{g}_R^{(i-1)}(E) \\ \underset{\sim}{g}_R^{(i)}(E) \end{bmatrix} = \begin{bmatrix} \underset{\sim}{r}_1^{(i)}(E) & \underset{\sim}{r}_2^{(i)}(E) \\ \underset{\sim}{r}_3^{(i)}(E) & \underset{\sim}{r}_4^{(i)}(E) \end{bmatrix} \begin{bmatrix} \underset{\sim}{g}'^{(i-1)}_R(E) \\ -\underset{\sim}{g}'^{(i)}_R(E) \end{bmatrix} \tag{42}$$

$$\underset{\sim}{r}^{(i)}(E) = \begin{bmatrix} \underset{\sim}{r}_1^{(i)}(E) & \underset{\sim}{r}_2^{(i)}(E) \\ \underset{\sim}{r}_3^{(i)}(E) & \underset{\sim}{r}_4^{(i)}(E) \end{bmatrix} , \tag{43}$$

where the $\underset{\sim}{r}_j^{(i)}$ matrices are $P^{(i)} \times P^{(i)}$. It is easy to show that

$$\underset{\sim}{r}_1^{(i)}(E) = \underset{\sim}{T}(i-1,i)\underset{\sim}{P}_1^{(i)}(E)[\underset{\sim}{P}_3^{(i)}(E)]^{-1}[\underset{\sim}{T}(i-1,i)]^{-1} , \tag{44}$$

$$\underset{\sim}{r}_2^{(i)}(E) = \underset{\sim}{T}(i-1,i)[\underset{\sim}{P}_3^{(i)}(E)]^{-1} , \tag{45}$$

$$\underset{\sim}{r}_3^{(i)}(E) = [\underset{\sim}{P}_3^{(i)}(E)]^{-1}[\underset{\sim}{T}(i-1,i)]^{-1} , \tag{46}$$

$$\underset{\sim}{r}_4^{(i)}(E) = [\underset{\sim}{P}_3^{(i)}(E)]^{-1}\underset{\sim}{P}_4^{(i)}(E) . \tag{47}$$

Note that the matrix $\underset{\sim}{P}_3$ is diagonal so its inversion is not time consuming.

The global R matrix which spans from the left-hand side of the first sector to the right-hand side of sector (i) is defined by

$$\begin{bmatrix} \underset{\sim}{g}_L^{(1)}(E) \\ \underset{\sim}{g}_R^{(i)}(E) \end{bmatrix} = \begin{bmatrix} \underset{\sim}{R}_1^{(i)}(E) & \underset{\sim}{R}_2^{(i)}(E) \\ \underset{\sim}{R}_3^{(i)}(E) & \underset{\sim}{R}_4^{(i)}(E) \end{bmatrix} \begin{bmatrix} \underset{\sim}{g}'^{(1)}_L(E) \\ -\underset{\sim}{g}'^{(i)}_R(E) \end{bmatrix} , \tag{48}$$

where the $R_j^{(i)}$ matrices are all square. It can be shown that if $[\lambda_{nn}^{(1)}]^2 >> 2\mu E/\hbar^2$ for all n, then $\underset{\sim}{R}_2^{(i)}$ and $\underset{\sim}{R}_3^{(i)}$ are approximately zero and that all scattering information, i.e., the a_{mn}, $1 \le m, n \le P^o$, can be determined only from $\underset{\sim}{R}_4^{(N_s)}$. In the present case, it is true that $[\lambda_{nn}^{(1)}(E)]^2 >> 2\mu E/\hbar^2$ for all n, and we only propagate $\underset{\sim}{R}_4^{(i)}$. This matrix depends only on $\underset{\sim}{R}_4^{(i-1)}$ and the $\underset{\sim}{r}_j^{(i)}$:

$$\underset{\sim}{R}_4^{(i)}(E) = \underset{\sim}{r}_4^{(i)}(E) - \underset{\sim}{r}_3^{(i)}(E)[\underset{\sim}{R}_4^{(i-1)}(E) + \underset{\sim}{r}_1^{(i)}(E)]^{-1}\underset{\sim}{r}_2^{(i)}(E) \tag{49}$$

and

$$\underset{\sim}{R}_4^{(1)}(E) = \underset{\sim}{r}_4^{(1)}(E) \quad . \tag{50}$$

When $P^{(i-1)} \neq P^{(i)}$, only the upper-left $P^{(i)} \times P^{(i)}$ part of $\underset{\sim}{R}_4^{(i-1)}(E)$ is used in the calculation of $\underset{\sim}{R}_4^{(i)}(E)$. The determination of the number of channels propagated in a given sector proceeds as follows. If $r_C^{(i)}$ is less than a pre-determined value (10 a_0 for the calculations presented here) or if $P^{(i)}$ has already been reduced to P^o, we set $P^{(i+1)} = P^{(i)}$. Otherwise we check whether $|(\underset{\sim}{r}_2^{(i)})_{nP^{(i)}}| \leq$ EPSRED and $|(\underset{\sim}{r}_2^{(i)})_{P^{(i)}n}| \leq$ EPSRED, $1 \leq n \leq P^{(i)}$, where EPSRED is a pre-set parameter (set equal to 10^{-3} for the calculations reported here). If both inequalities are satisfied, then $P^{(i+1)} = P^{(i)} - 1$; otherwise $P^{(i+1)} = P^{(i)}$.

Before applying the boundary conditions of eq. (19) to $\underset{\sim}{g}^{(i)}$ it is sometimes convenient to reorder the channels in $\underset{\sim}{g}^{(i)}$ and sometimes it is necessary to make linear combinations of the channels in $\underset{\sim}{g}^{(i)}$. It is convenient to reorder the channels if $V(\underset{\sim}{x},\vec{r})$ falls off faster than r^{-2}. This is because for large r it will be approximately true that

$$D_{nm}(r,E) = \delta_{nm}[\ell_n(\ell_n+1)/r^2 - \tilde{k}_n^2(E)] \ , \tag{51}$$

and, depending on r and r', it may be possible that $D_{nn}(r,E) > D_{mm}(r,E)$ and $D_{nn}(r',E) < D_{mm}(r',E)$. Since the subprogram that calculates $\underset{\sim}{T}^{(i)}$ and $[\underset{\sim}{\lambda}^{(i)}]^2$ orders the eigenvalues from lowest to highest, the relative position of channel n and m may change in $\underset{\sim}{g}$ when going from r to r'.

It is necessary to make new linear combinations of the channels in $\underset{\sim}{g}^{(i)}$ if there exist degenerate channels, i.e., those with $[\lambda_{nn}^{(i)}]^2 = [\lambda_{mm}^{(i)}]^2$ and $n \neq m$. This is because our matrix diagonalization routine will mix these states. Degenerate channels will occur in the current calculations at very large r where the term $\ell(\ell+1)/2\mu r^2$ is negligible since there are channels with the same $\tilde{k}_n$ but different values of ℓ_n. (Additional accidental degeneracies would occur if we used the harmonic oscillator and rigid-rotor approximations to calculate asymptotic energies, but we do not make these approximations.) In order to sort out these effects, we make the transformation to new radial functions $\underset{\sim}{h}^{(i)}$ defined by

$$g_{mn}^{(i)}(r,E) = \sum_{\ell=1}^{N} \sum_{k=1}^{P^{(i)}} T_{\ell m}^{(i)} U_{\ell k}^{o} h_{kn}^{(i)}(r,E) \ , \ 1 \leq m \leq P^{(i)}, \ 1 \leq n \leq P^{(i)} \ , \tag{52}$$

where $\underset{\sim}{U}^o$ diagonalizes H_{int} in the X_n basis, and has the channels in some fixed order that does not mix degenerate channels. This equation can be written in matrix notation as

$$\underset{\sim}{g}^{(i)}(r,E) = [\underset{\sim}{T}^{(i)}]^T \underset{\sim}{U}^o \underset{\sim}{h}^{(i)}(r,E) \ . \tag{53}$$

We then define a new global R matrix satisfying

$$\begin{bmatrix} \underset{\sim}{h}_L^{(1)} \\ \underset{\sim}{h}_R^{(i)} \end{bmatrix} = \begin{bmatrix} \tilde{\underset{\sim}{R}}_1^{(i)} & \tilde{\underset{\sim}{R}}_2^{(i)} \\ \tilde{\underset{\sim}{R}}_3^{(i)} & \tilde{\underset{\sim}{R}}_4^{(i)} \end{bmatrix} \begin{bmatrix} \underset{\sim}{h}'^{(1)}_L \\ -\underset{\sim}{h}'^{(i)}_R \end{bmatrix} \tag{54}$$

where

$$\tilde{\underset{\sim}{R}}_1^{(i)} = \{[\underset{\sim}{T}^{(1)}]^T \underset{\sim}{U}^o\}^{-1} \underset{\sim}{R}_1^{(i)} [\underset{\sim}{T}^{(1)}]^T \underset{\sim}{U}^o \tag{55}$$

$$\tilde{\underset{\sim}{R}}_2^{(i)} = \{[\underset{\sim}{T}^{(1)}]^T \underset{\sim}{U}^o\}^{-1} \underset{\sim}{R}_2^{(i)} [\underset{\sim}{T}^{(i)}]^T \underset{\sim}{U}^o \tag{56}$$

$$\tilde{\underset{\sim}{R}}_3^{(i)} = \{[\underset{\sim}{T}^{(i)}]^T \underset{\sim}{U}^o\}^{-1} \underset{\sim}{R}_3^{(i)} [\underset{\sim}{T}^{(1)}]^T \underset{\sim}{U}^o \tag{57}$$

$$\tilde{\underset{\sim}{R}}_4^{(i)} = \{[\underset{\sim}{T}^{(i)}]^T \underset{\sim}{U}^o\}^{-1} \underset{\sim}{R}_3^{(i)} [\underset{\sim}{T}^{(i)}]^T \underset{\sim}{U}^o \tag{58}$$

The matrix $\underset{\sim}{a}$ of eq. (19) is determined by

$$\underset{\sim}{a}(E) = \lim_{i \to \infty} \underset{\sim}{a}^{(i)}(E) \tag{59}$$

where

$$\underset{\sim}{a}^{(i)}(E) = [-\underset{\sim}{F}^{(i)}(E) + \tilde{\underset{\sim}{R}}_4^{(i)}(E)\underset{\sim}{H}^{(i)}(E)]^{-1}[\underset{\sim}{B}^{(i)}(E) + \tilde{\underset{\sim}{R}}_4^{(i)}\underset{\sim}{G}^{(i)}(E)]\underset{\sim}{\Delta} \tag{60}$$

and

$$F_{nm}^{(i)}(E) = \delta_{nm} \begin{cases} \cos[k_m(E) r_R^{(i)} - \ell_m \pi/2] , & 1 \le m \le P^o , \\ \exp[-k_m(E) r_R^{(i)}] , & P^o < m \le P^{(i)}, \end{cases} \tag{61}$$

$$B_{nm}^{(i)}(E) = \delta_{nm} \begin{cases} \sin[k_m(E) r_R^{(i)} - \ell_m \pi/2] , & 1 \le m \le P^o , \\ \exp[k_m(E) r_R^{(i)}] , & P^o < m \le P^{(i)}, \end{cases} \tag{62}$$

$$H_{nm}^{(i)}(E) = k_m(E) B_{nm}^{(i)}(E) , \qquad 1 \le m \le P^{(i)}, \tag{63}$$

$$G_{nm}^{(i)}(E) = k_m(E) F_{nm}^{(i)}(E) , \qquad 1 \le m \le P^{(i)}, \tag{64}$$

$$\Delta_{nm} = \delta_{nm} \begin{cases} 1 , & 1 \le m \le P^o , \\ b , & P^o < m \le P^{(i)} . \end{cases} \tag{65}$$

If it is true that for a given $m > P^o$, $(\underset{\sim}{R}_4^{(i)})_{nm} = (\underset{\sim}{R}_4^{(i)})_{mn} = 0$ for all $n \le P^o$, channel m is not required in the calculation of $a_{nm}^{(i)}$, $1 \le n, m \le P^o$. Our program determines the smallest $m \ge P^o$, called $P_a^{(i)}$, such that $|(\underset{\sim}{R}_4^{(i)})_{nm}| \le$ EPSDR and $|(\underset{\sim}{R}_4^{(i)})_{mn}| \le$ EPSDR for all $n \le P^o$, where EPSDR is some small number (EPSDR equals 10^{-3} in the present calculations), and then

uses the upper left $P_a^{(i)} \times P_a^{(i)}$ subblock of $\underset{\sim}{R}_4^{(i)}$ to calculate $\underset{\sim}{a}^{(i)}$.

It should be noted that the total energy E appears in eq. (13) only as a multiple of the unit matrix, so that the matrices $\underset{\sim}{T}^{(i)}$ are independent of the total energy, and the eigenvalues $[\lambda_{nn}^{(i)}]^2$ at a new energy can easily be determined by

$$[\lambda_{nn}^{(i)}(E)]^2 = [\lambda_{nn}^{(i)}(E')]^2 + 2\mu(E' - E)/\hbar^2 \quad . \tag{66}$$

We use this fact to save computer time on multiple-energy runs by reusing the $\underset{\sim}{\lambda}^{(i)}$ and $\underset{\sim}{T}(i-1,i)$. This makes the calculation of $V_{nm}(r)$, $\underset{\sim}{T}^{(i)}$, and $\underset{\sim}{T}(i-1,i)$ unnecessary for second energies and results in a great savings in computational time for such energies. The possibility of reusing this information is one of the reasons for preferring the present algorithm. Notice, however, that although it decreases the computation time, it greatly increases the storage requirements. There are two ways we have implemented the second-energy calculations. In the first method, the calculations for a given energy are done completely before the calculation for the next energy begins, and to do this as efficiently as possible requires the storage of the $P^{(i)} \times P^{(i)}$ matrices $\underset{\sim}{T}(i-1,i)$ and $[\underset{\sim}{T}(i-1,i)]^{-1}$. Since in the current calculation we will require on the order of 300 sectors, this option requires a great deal of storage space when $P^{(i)}$ is large. The second method we use is to propagate all energies together, that is, the global R matrix for sector (i) is calculated for all of the energies before the global R matrix for sector (i + 1) is calculated for any of the energies. In this case it is only necessary to store the $P^{(i)} \times P^{(i)}$ matrix $\underset{\sim}{R}_4^{(i)}$ for each energy. If there are fewer energies than sectors, which is ordinarily the case, this decreases the storage requirements. For the large-scale calculations described here, we performed calculations for 2-7 energies in a given run, and we used the second method for performing second-energy runs.

IV. THE HF-HF SYSTEM

Our calculations are for the collision of two identical hydrogen fluoride molecules. We take the hydrogen mass to be 1837.15 m_e and the fluorine mass to be 34631.94 m_e. We partition the Hamiltonian into the true diatomic and three-body parts, so that $\tilde{H}_{int}$ equals H_{int}, which is given by

$$H_{int} = H_{int}^{(1)} + H_{int}^{(2)} \quad , \tag{67}$$

$$H_{int}^{(i)} = -\frac{\hbar^2}{2\mu_{HF}}\left[\frac{1}{R_i^2}\frac{\partial}{\partial R_i}\left(R_i^2 \frac{\partial}{\partial R_i}\right)\right] + \hat{j}_{R_i}^2/2\mu R_i^2 + V_{vib}(R_i) \quad , \tag{68}$$

where μ_{HF} is the reduced mass for an HF molecule, R_i is the bond length of molecule i, $j^2_{\hat{R}_i}$ is the quantum mechanical operator for the square of the rotational angular momentum of molecule i, and V_{vib} is the vibrational potential. For V_{vib} we use the function proposed by Murrell and Sorbie,[25] which is a fit to an RKR potential curve determined from experiment. The primitive basis functions $X_n(\underset{\sim}{x},\hat{r})$ are given by[23]

$$X_n(\underset{\sim}{x},\hat{r}) = [2(1 + \delta_{v_1 v_2}\delta_{j_1 j_2})]^{-\frac{1}{2}}[\Phi_\alpha(\underset{\sim}{x},\hat{r}) + \eta(-1)^{j_1+j_2+j_{12}+\ell}\Phi_{\bar{\alpha}}(\underset{\sim}{x},\hat{r})] , \tag{69}$$

$$\Phi_\alpha(\underset{\sim}{x},\hat{r}) = (R_1R_2)^{-1}\chi_{v_1 j_1}(R_1)\chi_{v_2 j_2}(R_2)\Theta^{JM}_{j_1 j_2 j_{12}\ell}(\hat{R}_1,\hat{R}_2,\hat{r}) , \tag{70}$$

$$\Theta^{JM}_{j_1 j_2 j_{12}\ell}(\hat{R}_1,\hat{R}_2,\hat{r}) = \sum_{\substack{m_1 m_2 \\ m_{12} m_\ell}} (j_1 m_1 j_2 m_2 | j_1 j_2 j_{12} m_{12})(j_{12} m_{12} \ell m_\ell | j_{12} \ell JM) \times Y_{j_1 m_1}(\hat{R}_1)Y_{j_2 m_2}(\hat{R}_2)Y_{\ell m_\ell}(\hat{r}) , \tag{71}$$

where χ_{vj} is a vibrational wave function that solves

$$[-\frac{\hbar^2}{2\mu_{HF}}\frac{d^2}{dR^2} + j(j+1)\hbar^2/2\mu_{HF}R^2 + V_{vib}(R)]\chi_{vj}(R) = \varepsilon_{vj}\chi_{vj}(R) , \tag{72}$$

$(\ldots\ \ldots\ \ldots\ \ldots|\ldots\ \ldots\ \ldots\ \ldots)$ is a Clebsch-Gordan coefficient,[26] and $Y_{\ell m}$ is a spherical harmonic. The unit vector $\hat{R}_i$ describes the orientation of molecule i in the laboratory-fixed frame of reference. As discussed at the end of section II, n stands for the quantum numbers v_1, j_1, v_2, j_2, j_{12}, ℓ, J, M, and η, where it is understood that it is not possible to distinguish which molecule has which set of v and j quantum numbers; in contrast, α stands for the quantum numbers v_1, j_1, v_2, j_2, j_{12}, ℓ, J, and M, where formally the two molecules are distinguished so that molecule i is known to have vibrational and rotational quantum numbers v_i and j_i, and $\bar{\alpha}$ means that the quantum numbers for the two molecules are exchanged. For our calculations we obtained the χ_{vj} by solving eq. (72) by the linear variational method in a basis of harmonic oscillator functions. The eigenenergies, ε_{vj}, were obtained, however, from the experimental spectroscopic parameters of Ref. 27. [These are in good agreement with the variational ones from eq. (72).] Notice that

$$\varepsilon_n = \tilde{\varepsilon}_n = \varepsilon_{v_1 j_1} + \varepsilon_{v_2 j_2} . \tag{73}$$

In the final calculations the potential $V(\underset{\sim}{x},\vec{r})$ is the recent fit of Redmon to the extensive ab initio calculations of Binkley.[28] (Some of our early calculations, including the 101-channel timing runs discussed in section VI, used the function of Poulsen et al.[29] These early runs also involve somewhat different values of several of the numerical parameters of the calculations. Except when stated otherwise, the numerical parameters given in the rest of the chapter are the values used in the production runs with the Redmon-Binkley potential.) To evaluate the V matrix elements we first express $V_{nn'}$ in terms of matrix elements over the Φ_α:

$$V_{nn'}(r) = [4(1 + \delta_{v_1v_2}\delta_{j_1j_2})(1 + \delta_{v_1'v_2'}\delta_{j_1'j_2'})]^{-\frac{1}{2}}[V_{\alpha\alpha'}(r) + \eta(-1)^{j_1+j_2+j_{12}+\ell}V_{\bar{\alpha}\alpha'}(r) + \eta'(-1)^{j_1'+j_2'+j_{12}'+\ell'}V_{\alpha\bar{\alpha}'}(r) + \eta\eta'(-1)^{j_{12}+j_{12}'}V_{\bar{\alpha}\bar{\alpha}'}(r)] , \quad (74)$$

where

$$V_{\alpha\alpha'}(r) = \int d\underset{\sim}{x}\int d\hat{r}\ \Phi_\alpha^*(\underset{\sim}{x},\hat{r})V(\underset{\sim}{x},r)\Phi_{\alpha'}(\underset{\sim}{x},\hat{r}) , \quad (75)$$

and we have used the fact that $V_{nn'}$ is diagonal in $(-1)^{j_1+j_2+\ell}$. Since $V_{nn'}$ is diagonal in η, eq. (74) implies that

$$V_{\alpha\alpha'}(r) + \eta(-1)^{j_1+j_2+j_{12}+\ell}V_{\bar{\alpha}\alpha'}(r) = \eta(-1)^{j_1'+j_2'+j_{12}'+\ell'}V_{\alpha\bar{\alpha}'}(r) + (-1)^{j_{12}+j_{12}'}V_{\bar{\alpha}\bar{\alpha}'}(r) . \quad (76)$$

Thus

$$V_{nn'}(r) = \delta_{\eta\eta'}[(1 + \delta_{v_1v_2}\delta_{j_1j_2})(1 + \delta_{v_1'v_2'}\delta_{j_1'j_2'})]^{-\frac{1}{2}}[V_{\alpha\alpha'}(r) + \eta(-1)^{j_1+j_2+j_{12}+\ell}V_{\bar{\alpha}\alpha'}(r)] . \quad (77)$$

One implication of eq. (77) is that only states with $\eta = +1$ will couple with the states with $j_1 + j_2 = 0$

The determination of $V_{\alpha\alpha'}$ requires the evaluation of eq. (75), which is an 8-dimensional integral. We proceed by making the expansion[30]

$$V(\underset{\sim}{x},r) = \sum_{q_1q_2\mu} v_{q_1q_2\mu}(R_1,R_2,r)y_{q_1q_2\mu}(\hat{r}_1,\hat{r}_2) , \quad (78)$$

$$y_{q_1q_2\mu}(\hat{r}_1,\hat{r}_2) = \frac{4\pi}{[2(1 + \delta_{\mu 0})]^{\frac{1}{2}}}[Y_{q_1\mu}(\hat{r}_1)Y_{q_2-\mu}(\hat{r}_2) + Y_{q_1-\mu}(\hat{r}_1)Y_{q_2\mu}(\hat{r}_2)] , \quad (79)$$

where $\hat{r}_i$ is in the same direction as $\hat{R}_i$ but is expressed in the body-fixed frame of reference where the z axis is in the direction of $\hat{r}$. Equation (75) then becomes

$$V_{\alpha\alpha'}(r) = \sum_{q_1 q_2 \mu} B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'} C^{q_1 q_2 \mu}_{\gamma\gamma'}(r) \quad , \tag{80}$$

where

$$B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'} = \int d\hat{R}_1 d\hat{R}_2 \int d\hat{r}\ \Theta^{JM^*}_{j_1 j_2 j_{12} \ell}(\hat{R}_1, \hat{R}_2, \hat{r}) \mathcal{V}_{q_1 q_2 \mu}(\hat{r}_1, \hat{r}_2) \Theta^{J'M'}_{j_1' j_2' j_{12}' \ell'}(\hat{R}_1, \hat{R}_2, \hat{r}), \tag{81}$$

$$C^{q_1 q_2 \mu}_{\gamma\gamma'}(r) = \int dR_1 \int dR_2\ \chi^*_{v_1 j_1}(R_1) \chi^*_{v_2 j_2}(R_2) \nu_{q_1 q_2 \mu}(R_1, R_2, r) \chi_{v_1' j_1'}(R_1) \chi_{v_2' j_2'}(R_2), \tag{82}$$

β stands for the quantum numbers j_1, j_2, j_{12}, J, and M, and γ stands for the quantum numbers v_1, j_1, v_2, and j_2. Equation (81) is a 6-dimensional integral which is independent of r and can be evaluated analytically. Equation (82) is a two-dimensional integral which must be performed at every sector for every vibrational-rotational quantum number pair $\gamma\gamma'$ and every set of $q_1 q_2 \mu$ for which $B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'}$ is nonzero. However, before eq. (82) can be evaluated, it is necessary to determine the $\nu_{q_1 q_2 \mu}$ which are given by

$$\nu_{q_1 q_2 \mu}(R_1, R_2, r) = \frac{1}{4\pi} \int_{-1}^{1} [1 - \cos^2(\phi_1 - \phi_2)]^{-\frac{1}{2}}\, d[\cos(\phi_1 - \phi_2)] \int_{-1}^{1} d(\cos\theta_1) \times \int_{-1}^{1} d(\cos\theta_2) \mathcal{V}^*_{q_1 q_2 \mu}(\hat{r}_1, \hat{r}_2) V(\underset{\sim}{x}, r) \ , \tag{83}$$

where θ_i and ϕ_i are the inclination and azimuthal angles of $\hat{r}_i$. The three-dimensional integral in eq. (83) is independent of the channel indices β, γ, and ℓ.

The $B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'}$ are independent of sector and are evaluated as follows. Let $\tilde{\Theta}^{JM}_{j_1 j_2 j_{12} \Omega}$ be the simultaneous eigenfunctions of the operators with eigenvalues $j_1(j_1+1)$, $j_2(j_2+1)$, $j_{12}(j_{12}+1)$, $J(J+1)$, M, and Ω, where Ω is the projection of J on the body-fixed z axis. $\tilde{\Theta}^{JM}_{j_1 j_2 j_{12} \Omega}$ is given by[31]

$$\tilde{\Theta}^{JM}_{j_1 j_2 j_{12} \Omega}(\hat{r}_1, \hat{r}_2, \hat{r}) = \sum_{m_1 m_2} (j_1 m_1 j_2 m_2 | j_1 j_2 j_{12} \Omega) Y_{j_1 m_1}(\hat{r}_1) \times Y_{j_2 m_2}(\hat{r}_2) [(2J+1)/4\pi]^{\frac{1}{2}} \mathcal{D}^{(J)}_{M\Omega}(\phi, \theta, 0) \ , \tag{84}$$

where $\mathcal{D}^{(J)}_{M\Omega}$ is a rotation matrix[26] and ϕ and θ are the azimuthal and inclination angles of $\hat{r}$. The integral of an expansion function of the potential

between these body-frame basis functions is

$$\tilde{B}^{q_1 q_2 \mu}_{\beta\Omega\beta'\Omega'} = \int d\hat{r}_1 \int d\hat{r}_2 \; \tilde{\theta}^{JM*}_{j_1 j_2 j_{12}\Omega}(\hat{r}_1,\hat{r}_2,\hat{r}) Y_{q_1 q_2 \mu}(\hat{r}_1,\hat{r}_2) \tilde{\theta}^{J'M'}_{j_1' j_2' j_{12}'\Omega'}(\hat{r}_1,\hat{r}_2,\hat{r}), \tag{85}$$

and is given by[31]

$$\begin{aligned}\tilde{B}^{q_1 q_2 \mu}_{\beta\Omega\beta'\Omega'} = {} & \delta_{JJ'}\delta_{MM'}\delta_{\Omega\Omega'}[2(1+\delta_{\mu 0})]^{-\frac{1}{2}}[(2j_1+1)(2q_1+1)(2j_1'+1)(2j_2+1) \\ & \times (2q_2+1)(2j_2'+1)(2j_{12}+1)(2j_{12}'+1)]^{\frac{1}{2}}(-1)^{j_{12}'+j_1'+j_2'+q_1+q_2+\Omega} \\ & \times \begin{pmatrix} j_1 & q_1 & j_1' \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} j_2 & q_2 & j_2' \\ 0 & 0 & 0 \end{pmatrix} \sum_{q_{12}} (2q_{12}+1) \begin{pmatrix} j_{12}' & q_{12} & j_{12} \\ -\Omega & 0 & \Omega \end{pmatrix} \\ & \times \begin{pmatrix} q_{12} & q_1 & q_2 \\ 0 & \mu & -\mu \end{pmatrix} \begin{Bmatrix} j_{12}' & q_{12} & j_{12} \\ j_1' & q_1 & j_1 \\ j_2' & q_2 & j_2 \end{Bmatrix} [1+(-1)^{q_1+q_2+q_{12}}] , \end{aligned} \tag{86}$$

where $\begin{Bmatrix} \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \end{Bmatrix}$ is a $9-j$ symbol.[26] The laboratory-frame basis functions used for the wavefunction expansion are related to these body-frame functions by the transformation[31,32]

$$\Theta^{JM}_{j_1 j_2 j_{12}\ell}(\hat{R}_1,\hat{R}_2,\hat{r}) = \sum_{\Omega} (2\ell+1) \begin{pmatrix} \ell & j_{12} & J \\ 0 & \Omega & -\Omega \end{pmatrix} (-1)^{\Omega} \tilde{\theta}^{JM}_{j_1 j_2 j_{12}\Omega}(\hat{r}_1,\hat{r}_2,\hat{r}) , \tag{87}$$

where $\begin{pmatrix} \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \end{pmatrix}$ is a $3-j$ symbol[26] (which is simply a re-phased and re-normalized Clebsch-Gordan coefficient), and the $B^{q_1 q_2 \mu}_{\beta\ell\beta'\ell'}$ which we need to evaluate are related to the body-frame integrals by

$$B^{q_1 q_2 \mu}_{\beta\ell\beta'\ell'} = (2\ell+1)(2\ell'+1) \sum_{\Omega\Omega'} \begin{pmatrix} \ell & j_{12} & J \\ 0 & \Omega & -\Omega \end{pmatrix} \begin{pmatrix} \ell' & j_{12}' & J' \\ 0 & \Omega' & -\Omega' \end{pmatrix} (-1)^{\Omega+\Omega'} B^{q_1 q_2 \mu}_{\Omega\ell\Omega'\ell'}. \tag{88}$$

Substituting (86) into (88) yields

$$B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'} = \delta_{JJ'} \delta_{MM'} (-1)^{j_{12}+j_1'+j_2'+q_1+q_2+J} [2(1+\delta_{\mu 0})]^{-\frac{1}{2}} [(2j_1+1)(2q_1+1)$$

$$\times (2j_1'+1)(2j_2+1)(2q_2+1)(2j_2'+1)(2j_{12}+1)(2j_{12}'+1)$$

$$\times (2\ell+1)(2\ell'+1)]^{\frac{1}{2}} \begin{pmatrix} j_1 & q_1 & j_1' \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} j_2 & q_2 & j_2' \\ 0 & 0 & 0 \end{pmatrix} \sum_{q_{12}} (2q_{12}+1)$$

$$\times \begin{pmatrix} \ell & \ell' & q_{12} \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} q_{12} & q_1 & q_2 \\ 0 & \mu & -\mu \end{pmatrix} \begin{Bmatrix} \ell & \ell' & q_{12} \\ j_{12}' & j_{12} & J \end{Bmatrix} \begin{Bmatrix} j_{12}' & q_{12} & j_{12} \\ j' & q_1 & j_1 \\ j_2' & q_2 & j_2 \end{Bmatrix}$$

$$\times [1 + (-1)^{q_1+q_2+q_{12}}] \quad , \tag{89}$$

where $\begin{Bmatrix} \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \end{Bmatrix}$ is a 6 - j symbol.[26] For the special case of $J=0$, eq. (89) can be simplified to

$$B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'} = \delta_{j_{12}\ell} \delta_{j_{12}'\ell'} (1+\delta_{\mu 0})^{-\frac{1}{2}} [2(2j_1+1)(2q_1+1)(2j_1'+1)(2j_2+1)$$

$$\times (2q_2+1)(2j_2'+1)(2\ell+1)(2\ell'+1)]^{\frac{1}{2}} (-1)^{j_{12}+j_1'+j_2'+q_1+q_2}$$

$$\times \begin{pmatrix} j_1 & q_1 & j_1' \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} j_2 & q_2 & j_2' \\ 0 & 0 & 0 \end{pmatrix} \sum_m \begin{pmatrix} \ell & j_1 & j_2 \\ 0 & m & -m \end{pmatrix} \begin{pmatrix} \ell' & j_1' & j_2' \\ 0 & -\mu-m & \mu+m \end{pmatrix}$$

$$\times \begin{pmatrix} j_1' & q_1 & j_1 \\ -\mu-m & \mu & m \end{pmatrix} \begin{pmatrix} j_2' & q_2 & j_2 \\ \mu+m & -\mu & -m \end{pmatrix} \quad . \tag{90}$$

In terms of Clebsch-Gordan coefficients eq. (90) is

$$B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'} = \delta_{j_{12}\ell} \delta_{j_{12}'\ell'} (-1)^{j_1'+j_2+\ell'+q_1+q_2+\mu} [(2j_2'+1)(2j_2+1)(1+\delta_{\mu 0})]^{-\frac{1}{2}}$$

$$\times [2(2\ell+1)(2\ell'+1)(2q_1+1)(2q_2+1)]^{\frac{1}{2}} (j_1 0 q_1 0 | j_1 q_1 j_1' 0)$$

$$\times (j_2 0 q_2 0 | j_2 q_2 j_2' 0) \sum_m (\ell 0 j_1 m | \ell j_1 j_2 m)(\ell' 0 j_1' -\mu-m | \ell' j_1' j_2' -\mu-m)$$

$$\times (j_1' -\mu-m q_1 \mu | j_1' q_1 j_1 -m)(j_2' \mu+m q_2 -\mu | j_2' q_2 j_2 m) \quad . \tag{91}$$

The $C^{q_1 q_2 \mu}_{\gamma\gamma'}$ are calculated using a recently proposed optimized quadrature scheme using Gauss-ground-state[33] nodes. Seven points per vibrational coordinate were found sufficient to converge the final S matrix elements for the initial state $v_1 = v_2 = 1$, $j_1 = j_2 = 0$ to better than 1%. These convergence tests were performed with a basis with $v_1 + v_2 = 2$ and $j_1 + j_2 \leq 5$ at a

relative translational energy of 76 meV. N was 54 for these calculations. Thus the integrand of eq. (82) is required at 49 points for each q_1, q_2, μ, γ, γ', and r, and this was evaluated as discussed next.

The three-dimensional integral of eq. (83) must be evaluated numerically in each sector for each of the 49 values of R_1 and R_2 required. The θ_1 and θ_2 integrals were evaluated using N_q-point Gauss-Legendre quadrature and the $\phi_1 - \phi_2$ integral was evaluated using N_q-point Gauss-Chebyshev quadrature, and we took N_q to be $q_{max} + 1$ where q_{max} is the maximum value of q_1 and q_2 used in eq. (78). Note that $0 \leq \mu \leq \min(q_1,q_2)$ where $\min(q_1,q_2)$ is the minimum of q_1 and q_2. N_q ranged from 11 at small r to 4 for r greater than 24.5 a_0. Thus the potential function must be evaluated $49N_q^3$ times per sector. This is an important number because the evaluation of the potential at a given geometry is very time consuming. As coded by us, the Poulsen-Billing-Steinfeld[29] potential requires the evaluation of 3 exponentials, 23 additions or subtractions, 36 multiplications, 1 division, 1 square root, and 6 raisings to a power per potential evaluation. The Redmon-Binkley[28] potential is even more complicated; our version requires 34 exponential evaluations, 303 additions or subtractions, 405 multiplications, 19 divisions, 5 square roots, and 2 raisings to a power per potential evaluation. Note, however, that the number of potential evaluations is independent of the number of channels in the close-coupling expansion.

To complete the specification of our calculations it is necessary to assign values to the number of channels N, i.e., eq. (9), and to the number of terms in the potential expansion, eq. (78). We choose the number of terms in eq. (78) by including all q_1, q_2, and μ with $q_1 + q_2 \leq q_{sum}$ and increasing q_{sum} until eq. (78) accurately represents $V(\underset{\sim}{x},r)$ at a given value of r. In our case accurately means that the expansion differ from $V(\underset{\sim}{x},r)$ on the order of 1% for $r > 4\ a_0$ and slightly more for $r < 4\ a_0$. This yields $q_{sum} = 10$ at small r and $q_{sum} = 3$ for r larger than 24.5 a_0. Equation (78) then contains from 161 to 13 terms. It should be noted that a large number of the $B^{q_1 q_2 \mu}_{\beta\ell\beta'\ell'}$ are zero and to avoid storing up to $322N^2$ numbers, many of which are zero, our program only stores the nonzero values of $B^{q_1 q_2 \mu}_{\beta\ell\beta'\ell'}$, alternating with one word which indicates the values of q_1, q_2, μ, β, ℓ, β', and ℓ'. This requires approximately $30N^2$ words of storage for $q_{sum} = 10$, with the exact coefficient of N^2 depending on N.

The fact that many $B^{q_1 q_2 \mu}_{\beta\ell\beta'\ell'}$ are zero does not necessarily translate into zero values of $V_{nn'}$. In fact at small r, $V_{nn'}$ has no zero elements, but as r increases and we decrease the number of terms in eq. (78), we do obtain zeroes in the V matrix. Table 1 summarizes the percentage of zero elements as a function of r.

Table 1. Potential and sector information

r (a_0)	N_q	Number of terms in eq. (78)	% zero elements in V matrix[a]	Number of sectors in this range[a]
3 to 6.06	11	161	0	52
6.06 to 6.54	10	125	0	8
6.54 to 7.02	9	95	1.1	8
7.02 to 7.56	8	70	3.8	9
7.56 to 11.0	7	50	9.8	51
11.0 to 12.6	6	34	20	12
12.6 to 24.7	5	22	34	51
24.7 to 150	4	13	54	100

[a]based on N = 530

There are many ways to choose N. In the present chapter, the only block of the Schroedinger equation that we attempt to solve is the one with $J = M = 0$, and the only initial state considered explicitly is the one with $v_1 = v_2 = 1$ and $j_1 = j_2 = 0$. As a consequence we need only consider the sub-block with $\eta = \zeta = +1$. Consider a total energy with the $v_1 = v_2 = 1$, $j_1 = j_2 = 0$ channel open, and the $v_1 = v_2 = 1$, $j_1 = 0$, $j_2 = 1$ channel closed. One possible choice for N is to include all open channels plus selected closed channels. However, this would lead to an extremely large value for N since the rotational energy spacing is so much smaller than the vibrational spacing. For the energy under consideration there are 1548 open channels with $J = M = 0$ and $\eta = \zeta = +1$, and this number increases rapidly as either J or the energy increases. We have found previously,[11b] for model diatom-diatom collisions with central potentials, so that only vibrational energy and not rotational energy is transferred, that a good way to choose channels is to include all channels with $v_1 + v_2 \leq v_{sum}$, where v_{sum} is large enough to include at least one closed vibrational level of each molecule. We assume (based on experience[10] with rotational energy transfer in atom-molecule collisions) that highly closed rotational channels may also be excluded, and we also assume that channels with very high j_i, even when open, do not have significant dynamical coupling to an initial state with $j_1 = j_2 = 0$. Thus we finally arrive at a scheme in which we include all channels with $v_1 + v_2 \leq v_{sum}$, $j_1 + j_2 \leq j_{sum}$, where j_{sum} is allowed to depend on v_1 and v_2. In the final analysis we must converge the calculations with respect to v_{sum} and all the j_{sum}. If convergence is obtained the assumptions behind our channel-selection scheme do not affect the accuracy of the final results, but they do affect the rate of convergence with increasing N. Because the HF molecules are indistinguishable and we properly include the interchange

symmetry, it is not a further restriction to generate the channels from a list in which $v_1 \leq v_2$ and $j_1 \leq j_2$ when $v_1 = v_2$. For the present chapter we consider four sizes of the basis set, with N = 55, 101, 400, and 530. The first two are small basis sets used only for debugging and timing analyses; the third and fourth are an attempt to test convergence. The 400-channel basis is obtained using $v_{sum} = 3$, $j_{sum} = 8$ for $v_1 + v_2 \leq 2$, and $j_{sum} = 6$ for $v_1 + v_2 = 3$, and the 530-channel basis is obtained using $v_{sum} = 3$, $j_{sum} = 9$ for $v_1 + v_2 \leq 2$, and $j_{sum} = 7$ for $v_1 + v_2 = 3$. (Another difference is that the 55, 400, and 530-channel runs are for the Redmon-Binkley potential with proper inclusion of interchange symmetry, whereas the 101-channel calculations are simplified as discussed in section VI.) None of these basis sets includes all open channels with $v_1 + v_2 \leq 1$; channels with high $j_1 + j_2$ are excluded. These open channels should not be too important since channels with intermediate $j_1 + j_2$ will have poor translational overlap with the initial state because of the large energy defect and coupling to channels with higher $j_1 + j_2$ require terms in eq. (78) with high q_1 and q_2, and these become relatively less important as q_1 and q_2 increase.

We integrate eq. (12) using a fixed stepsize of 0.06 a_0 for r less than 10 a_0 and then allow the stepsize to increase according to eq. (38) with EPS equal to 0.1 and h_{max} large enough to not limit $h^{(i+1)}$. By the time the calculation is stopped at r = 150 a_0, the stepsize has increased to about 3 a_0, requiring approximately 290 sectors.

V. VECTORIZATION OF THE COMPUTER CODE

On the basis of the theory discussed in section II and the algorithm reviewed in section III, the solution of the close coupling equations may be broken into six segments, as described in Table 2.

A general name for the work of segment V is asymptotic analysis. In general it need only be performed in the last sector, but as a check that the integration has been carried out to large enough r it is usually performed more than once. As mentioned above, runs at second and subsequent energies require less work if appropriate matrices are saved from the calculation at the first energy. For a first energy run with N_s sectors for which the asymptotic analysis is performed N_a times, segments I, II, and IV are performed N_s times, segment III is performed $N_s - 1$ times, and segment V is performed N_a times. Segment VI contains some tasks performed only once, as well as some tasks performed N_s times or N_a times, but the tasks performed more than once all involve very little computational work compared to those singled out as segments I-V. For the various runs discussed here $N_s = 288$ or 291 and $N_a = 4$. Thus most of the computational work is in

Table 2. Segments of the computation

Segment	Principal subprogram	Task
I	POT	Calculation of $V_{nm}(r_C^{(i)})$
II	DCALC	Assembly and diagonalization of $\underset{\sim}{D}^{(i)}$ to obtain $\lambda_{nn}^{(i)}$ and $\underset{\sim}{T}^{(i)}$
III	TAUMTS	Calculation of $\underset{\sim}{T}(i-1,i)$ and/or its inverse
IV	RCALC	Calculation of $\underset{\sim}{R}_4^{(i)}$ and $P^{(i+1)}$
V	GENSCAT	Calculation of $\underset{\sim}{S}$ from $\underset{\sim}{R}_4^{(i)}$
VI	...	everything else

segments I-IV. For convenience in discussing the segments in the rest of this chapter, segments I-V will be named after their principal subprograms and segment VI will be called overhead.

Segment I involves the computation of a very large number, approximately $\frac{1}{2}N^2$, of multi-dimensional integrals, and segments II-V include a number of matrix operations for which the number of arithmetic steps becomes asymptotically proportional to N^3 or $[P^{(i)}]^3$ as the matrix orders are increased. Such matrix operations are called "N^3 steps". Most of the computer time goes into the potential calculation and these N^3 steps, and these are thus the operations for which vectorization offers the most potential benefits. We will next concentrate our attention on the N^3 steps. Table 3 lists the number and type of such steps for each segment. It is useful to comment briefly on the appearance of an inversion step in the table. In general whenever possible, it is more efficient to solve linear equations $\underset{\sim\sim}{A}\underset{\sim}{x} = \underset{\sim}{B}$ directly than to perform the separate steps of inverting $\underset{\sim}{A}$ followed by the matrix multiplication of $\underset{\sim}{A}^{-1}$ times $\underset{\sim}{B}$ to give the solution $\underset{\sim}{x} = \underset{\sim}{A}^{-1}\underset{\sim}{B}$. However, eqs. (44) and (46) are of the form $\underset{\sim}{x} = \underset{\sim}{B}\underset{\sim\sim}{A}^{-1}$. Although this is equivalent to $\underset{\sim}{x}^T = (\underset{\sim}{A}^T)^{-1}\underset{\sim}{B}^T$, we instead solve it by first inverting $\underset{\sim}{T}(i-1,i)$, which plays the role of $\underset{\sim}{A}$, and then form the matrix products $\underset{\sim 1}{B}\underset{\sim}{A}^{-1}$ for eq. (44) and $\underset{\sim 2}{B}\underset{\sim}{A}^{-1}$ for eq. (46). Note that $\underset{\sim 2}{B}$ is diagonal. Multiplying a matrix by a diagonal matrix is, like a matrix times a vector or a matrix plus a matrix, only an "N^2 step", and such operations are not included in Table 3. Figure 1 gives a flow chart of our program, showing the flow of the calculation between the various program segments discussed above. On the first energy, POT, DCALC, TAUMTS, and RCALC are called once per sector. For calculations at second energies, TAUMTS is called once in each sector

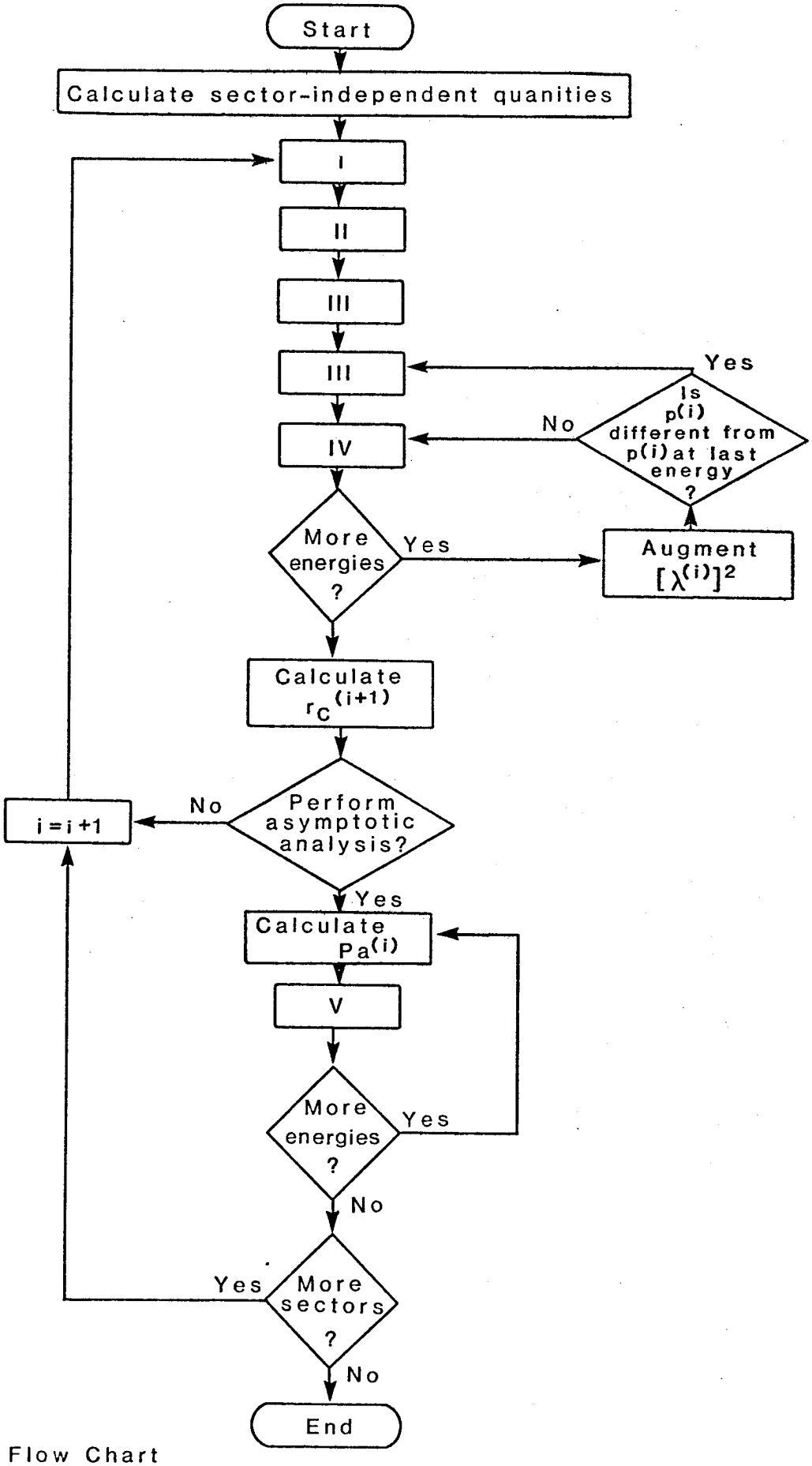

Fig. 1 Flow Chart

Table 3. Number and type of "N^3 steps" per program section[a]

Segment	operation				
	diagonalize	invert matrix	solve linear equations	multiply matrix by matrix	Equations
I. POT	0	0	0	0	...
II. DCALC	(1,0)[b]	0	0	0	(16)
III. TAUMTS	0	[1,0][c]	0	(1,0)	(41),(46)
IV. RCALC	0	0	1	2	(44),(49)
V. GENSCAT	0	0	4	4	(21),(58),(60)

[a] significant operations are ones which scale for large matrix orders as the cube of the matrix order

[b] (1,0) means 1 for the first energy and 0 for subsequent energies

[c] [1,0] means 1 for all segments at the first energy and for sectors in which the number of channels is decreased at any energy, but 0 for other sectors at second and subsequent energies

for which $P^{(i)}$ at this energy is different from $P^{(i)}$ at the last energy, and RCALC is called once in every sector. For the production runs $P^{(i)}$ differs from one sector to the next in about half of the sectors. At selected distances near the end of each calculation, for every energy, GENSCAT is called to perform an asymptotic analysis.

We now discuss the practical implementation of these calculations on three computers. The first is the University of Minnesota Department-of-Chemistry Digital Equipment Corporation VAX 11/780. This is a "minicomputer" equipped with a scalar floating point accelerator and 4 Megabytes of physical memory (equivalent to a half million 64-bit words). The second is the Colorado State University (CSU) Control Data Corporation Cyber 205 equipped with a scalar processor and two vector pipelines and two million 64-bit words of physical memory. (We also performed some of our Cyber 205 calculations on similarly configured machines located at the Control Data Corporation offices in Arden Hills, Minnesota; timing analyses will be given only for the CSU machine.) The third is the University of Minnesota University-Computer-Center Cray-1, equpped with scalar and vector processors and one million 64-bit words of physical memory, of which 0.91 Megawords are available to an individual user. These second and third machines are "class VI supercomputers". These three machines will henceforth be called simply the VAX, the Cyber 205, and the Cray-1, respectively. The VAX and the Cyber 205 are virtual-memory machines, while the Cray-1 is not. In our

discussion of implementing the codes we will pay special attention to utilization of the vector modes available on the two supercomputers.

There are two aspects of vectorizing a program. The first concerns memory usage. Since our calculations require the manipulation of matrices which are quite large, the finite amount of memory and mass storage available to us limits the size and number of matrices which can be used in a given calculation. Thus a considerable amount of effort was spent to modify our code to use the smallest amount of memory possible. This is especially important on the Cray-1 which is not a virtual memory machine. Our vectorized code on the Cray-1 holds only two $N \times N$ matrices in memory, with other $N \times N$ matrices and the B matrix elements stored on disk. This limits us to N less than about 640. It is possible to hold up to four $N \times N$ matrices in memory for N less than about 470, but to avoid presenting results with more than one Cray-1 vector version of our code, we present results here only for the version holding only 2 $N \times N$ matrices in memory. The second aspect of vectorization is to modify the program to use the supercomputers' vector capabilities. This can be done in several ways, as discussed next.

A portion of code that can be vectorized, that is, that which can be replaced with a vector instruction, is a DO-loop that performs exactly the same arithmetic operation on each element of a vector. Examples of DO-loops which cannot be vectorized include those involving recursion, subroutine calls, input or output statements, branch instructions, and most conditional statements. Different machines have different definitions of a vector. On the Cyber 205, a vector is a series of contiguous memory locations. The Cray-1 has 8 vector registers, each capable of containing 64 words (one word is 64 bits), and it can fill these registers with values from a series of memory locations, each incremented by a constant amount.

To understand how a vector machine achieves high-speed arithemtical throughput, it is necessary to first discuss how a scalar machine works. A typical operation, such as the multiplication of two numbers, can be broken down into a series of suboperations. The amount of time required by a suboperation is at least one minor cycle. On a scalar machine, each suboperation must wait until the preceding one has completed before it can start. Thus the multiplication of two numbers will take many minor cycles; the exact number of minor cycles will depend on whether or not the operands were initially in registers or in memory. In a vector machine, the units which perform the suboperations on vectors are more independent. The set of units that perform these suboperations on vectors are called pipelines and the units can be operating simultaneously on different operands, the different vector elements. It is then necessary to wait several minor

cycles until the first element of a vector has received all of the sub-operations, and after that, the results of the operation on the following vector elements will come out at the rate of one per minor cycle. There are several ways to increase the speed even further. For example, one can add more pipelines, as on the Cyber 205 machine we used, which has two. Then, after the startup time, the time necessary for the first result to emerge, a result will be obtained from each pipeline every minor cycle. Another way to speed up the arithmetic is to perform more than one operation at a time; an example of this is multiplying a scalar times a vector and adding it to a different vector (a scalar is a vector of length 1). This is called linking (or chaining), and using it produces the results of two operations per pipe per minor cycle. On the Cyber 205 linking is restricted to two vector operands and each of the four units (add/subtract, multiply/divide, logical, and shift) in the CPU can be used only once. On the Cray-1, all 8 of the vector registers can be used, but again each unit in the CPU can be used only once. On the Cyber 205 it is also possible to use half precision (32 bits) rather than the normal 64-bit precision. In this case one obtains twice as many 32-bit results as 64-bit results. All of these techniques increase the startup time, so it is necessary to have vectors as long as possible to get the maximum speed. The maximum vector length on the Cyber 205 is 65,355. The Cray-1 uses only 64 elements of a vector at a given time, but it has no maximum vector length. In our calculations, we use linking whenever possible and we always use 64-bit precision. For the processes which take up most of the time in segments II-V, the vector length is always less than or equal to $P^{(i)}$. For some operations on the Cyber 205, such as setting a matrix equal to another matrix, we exceed the maximum vector length and have to split up the operation into parts, each of which does not exceed the maximum vector length. In our vectorized version of POT we use vector lengths of up to 1331, however, most of the time required is spent on operations on vectors of length 49.

As stated already above, most of the work in our calculations is spent on matrix manipulations. To take advantage of vectorization one can write utility routines which are vectorizable or use pre-existing library routines supplied by the system. On the Cyber 205 we have used the Math-Geophysical Vector library (MAGEV), which contains routines to diagonalize matrices (these are modifications of EISPACK routines) and solve linear equations and to perform other useful tasks. On the Cray-1, we have used the SCILIB library, which contains similar routines as the MAGEV library.

To help make the vectorization process more clear, we will now consider in detail how one would write a utility routine to multiply two

matrices. We first consider doing this on the Cyber 205, which has the restriction that the components of the vectors be contiguous memory locations. A typical FORTRAN code to multiply an N × M matrix $\underset{\sim}{A}$ times an M × P matrix $\underset{\sim}{B}$ to produce an N × P matrix $\underset{\sim}{C}$ using the inner product method is

```
      DO 1 I=1,N
      DO 1 J=1,P
      C(I,J)=0.0
      DO 1 K=1,M
1     C(I,J)=C(I,J)+A(I,K)*B(K,J).
```

In FORTRAN the elements of arrays are stored in memory in the order produced by varying the first index most rapidly. Thus in the above example only the elements of B are accessed sequentially from memory. Since the elements of A are not accessed sequentially from memory, the inner-most loop cannot be vectorized (this loop could be vectorized on the Cray-1). One possible solution is to store the transpose of the matrix A, but it is much more efficient to simply re-order the loops as follows (this is sometimes known as the outer product method):

```
      DO 1 J=1,P
      DO 1 I=1,N
1     C(I,J)=0.0
      DO 2 J=1,P
      DO 2 K=1,M
      DO 2 I=1,N
2     C(I,J)=C(I,J)+A(I,K)*B(K,J).
```

Here the inner-most loop can be replaced with the linked vector instruction: add element I of the vector starting at C(1,J) to the scalar B(K,J) times element I of the vector starting at A(1,K) and store the result in element I of the vector starting at C(1,J). Since this uses the linking capabilities of the Cyber 205, the matrix multiplication can be performed extremely fast. (The method is not the only efficient way to vectorize a matrix multiply on the Cyber 205, but is the one we use.)

In Cyber 205 FORTRAN there are two ways to indicate vector instructions. The first is to simply turn on the compiler option (after of course modifying the program to allow vector instructions). There are two options, V (for Vectorization) and U (for Unsafe vectorization). The U option causes some classes of DO loops to be replaced with vector instructions which the V option does not vectorize. We found that these options were not useful to us because the V option will not vectorize DO loops with variable limits, and the U option sometimes caused incorrect results. The second way to indicate vector instructions uses an explicit notation. An example of

this notation is A(I,J,K;N) which indicates a vector containing N elements starting with A(I,J,K). Using this notation the matrix multiply example becomes

```
      DO 1 J=1,P
1     C(1,J;N)=0.0
      DO 2 J=1,P
      DO 2 K=1,M
2     C(1,J;N)=C(1,J;N)+A(1,K;N)*B(K,J).
```

On the Cray-1, vectorization is accomplished differently. Since the Cray-1 can fill its vector registers from memory locations which are not contiguous, the first as well as the second multiplication example can be vectorized. We have found that, of the methods we tried, the fastest way to multiply two matrices is to use the DO loops ordered as in the first example but replacing the inner-most DO loop with a call to the Cray-1 dot product function:

```
      DO 1 I=1,N
      DO 1 J=1,P
1     C(I,J)=SDOT(M,A(I,1),N,B(1,J),1)
```

where it has been assumed that the array A has N as its first dimension. Cray-1 FORTRAN does not contain a special explicit notation for vectors; thus to make the compiler use vector instructions it is necessary to use the compiler option V. However, unlike Cyber 205 FORTRAN, Cray-1 FORTRAN allows for what are called compiler directives. These allow one to control whether or not specific DO loops are vectorized.

In the final version of our scattering code on the Cyber 205 we use in-line FORTRAN statements to perform matrix multiplications, and we call routines from the MAGEV library to solve linear equations, invert matrices, and diagonalize matrices. These routines use Gaussian elimination to solve linear equations and modifications of EISPACK routines for matrix diagonalization. In comparison to our own vectorized FORTRAN utility routines to perform these operations, the MAGEV linear equations solver (GEL) is slightly faster and the MAGEV matrix diagonalization routines are significantly faster. The matrix diagonalization routines from the MAGEV library are faster than our FORTRAN routines because matrix diagonalization (first a Householder transformation to tridiagonal form followed by an implicit QL diagonalization of the tridagonal matrix) involves many steps which cannot be vectorized, and the MAGEV routines use many special calls (to what are called STACKLIB routines) which cause the compiler to generate in-line code which efficiently performs these non-vectorizable steps. We have not included any STACKLIB routines in our own coded sections.

On the Cray-1, the final version of our scattering code uses in-line FORTRAN statements and the dot product function SDOT to perform matrix multiplications. To diagonalize the matrix $\underset{\sim}{D}$ we use the SCILIB version of the EISPACK routine RS, and for matrix inversion and linear-equation solving, we use our own vectorized version of a routine made up of LINPACK routines. Our routine consists of the LINPACK routines SGEFA, SGEDI, and SGESL combined together and modified to avoid all unnecessary subroutine calls. The original LINPACK routines use BLAS (Basic Linear Algebra Subroutines) routines to perform the innermost DO loop operations. In our routine, we found it faster to substitute FORTRAN statements to perform the operations done by the BLAS routines rather than to call the BLAS routines contained in the SCILIB library. In contrast to the Cyber 205, the SCILIB EISPACK routines are only slightly faster than using our own vectorized FORTRAN versions. The SCILIB inversion and linear-equation-solver subprogram MINV is faster than our own vectorized FORTRAN routine (based on modified LINPACK routines as discussed above); however we do not use it because the SCILIB version requires matrix input in an augmented matrix form that is inconvenient and also since the ratio of the time required by MINV to the time required by our vectorized FORTRAN routine was approaching one as the matrix size is increased.

VI. EXECUTION TIME ANALYSIS

In this section we present actual execution times for various runs and tasks within runs and for various computers and versions of our code. First we consider a special set of early runs with $N = 101$ for which we made the most detailed timing comparisons. These calculations used the simpler Poulsen-Billing-Steinfeld[29] potential with a smaller number of terms in eq. (78); in particular, we restricted q_1 and μ to 0 and q_2 to be less than or equal to 2. These calculations also differ from the production runs in several other respects; the most important of which are: we used 27-point quadrature (3 points per dimension) to evaluate eq. (83), the vibrational dependence was expanded in a Taylor series, keeping up to quadratic terms in each oscillator coordinate for a total of nine terms; we did not enforce interchange symmetry, we restricted the basis to functions with $j_1 = 0$ (so that the potential is effectively spherically averaged over the first rotor's orientation), and we used 4 sectors and multiplied the times obtained by the appropriate factors to mimic a calculation using $N_s = 291$. In the second part of this section we consider calculations with the Redmon-Binkley[28] surface with $N = 55$, 400, and 530 and with the production values of the numerical parameters, e.g., $N_q = 4$ to 11, depending on r.

Table 4. CPU times (in sec) for various versions of the six program segments on various machines. The times are based on N (number of channels) = 101, N_s (number of sectors) = 291, and N_a (number of asymptotic analyses) = 4.

		Tasks done only at 1st E		Tasks done every E				Totals	
Machine[a]	Version	POT	DCALC	TAUMTS (1st E)[b]	TAUMTS (2nd E)[c]	RCALC[d]	GENSCAT[d]	1st E[b]	2nd E[c]
VA	scalar	3170	18800	14600	4740	29200	894	66800	34800
CY	scalar	69.0(46)[e]	654(29)	427(34)	145(33)	889(33)	27.8(32)	2070(32)	1060(33)
CY	scalar with compiler vectorization (options V and U)	69.0(46)	634(29)	425(34)	141(34)	889(33)	27.4(33)	2060(32)	1060(33)
CY	explicitly vectorized	69.0(46)	81(232)	29(490)	8.8(538)	43(684)	4.45(201)	226(294)	56(628)
CR	scalar	95(33)	551(34)	536(27)	86(55)	1250(23)	35.2(25)	2470(27)	1370(25)
CR	scalar with compiler vectorization	98(32)	269(70)	386(38)	12(393)	1110(26)	28.2(32)	1890(35)	1150(30)
CR	explicitly vectorized	130(24)	117(160)	63(233)	13(375)	116(252)	5.58(160)	432(154)	134(261)

[a] VA: VAX 11/780; CY: Cyber 205; CR: Cray-1

[b] time for first energy

[c] time for second energy or any subsequent energy

[d] time for these tasks is same for 1st or any other energy

[e] numbers in parentheses are the ratio of the VAX time to the time of that particular entry

Table 4 presents the execution times for each of the segments I-V for the 101-channel runs. The times for segment VI (overhead) are not given in this table because they comprise less than 1% of the total time. As stated above, it is assumed that the integration range may be divided into 291 sectors, but an asymptotic analysis to calculate the scattering matrix need be performed 4 times (the fourth to actually obtain the scattering matrix and the first three as convergence checks of the maximum r value r_{max}). For the 101-channel test runs, the segment POT was not vectorized; it was only modified as required to interface with the other program modifications. Table 4 shows that running in scalar mode on the Cyber 205 results in a speedup factor of about 32 as compared to our VAX for both first and second energies. On the Cray-1 this speedup factor is a factor of 26. Vectorizing our program by just turning on the compiler vectorization options (options V and U on the Cyber 205 and option V on the Cray-1) produces very little difference on the Cyber 205, while on the Cray-1 some program sections are significantly speeded up, resulting in an overall speedup factor compared to our VAX of 35 times for first eneriges and 30 times for second energies. Our utility routines to perform diagonalization, determine inverses, and solve linear equations are speeded up significantly on the Cray-1, while our routines to multiply matrices were not speeded up at all, since they were written in a manner which did not allow the compiler to use vector instructions. Explicitly vectorizing our code as described above results in substantial speed increases for both the Cyber 205 and Cray-1. The speedup factor for the Cyber 205 over the VAX increases to 294 times for first energies and 628 times for second energies. The large speedup for second energies is mainly due to the fact that the processes which vectorize relatively less well on the Cyber 205 (the potential evaluation and matrix diagonalization) are not performed for second energies. The situation is similar on the Cray-1 where the speedup factor over the VAX increases to 154 times for first energies and 261 times for second energies. The reason why TAUMTS at second energies is slightly faster in the compiler vectorized version of our code than the explicitly vectorized version is that a different routine to invert matrices is used. We use the slightly slower inverter because it is the routine we have chosen to use to solve linear equations. Some program segments are speeded up even more, e.g., we achieve a factor of 684 for the RCALC segment on the Cyber 205; this indicates that other problems or other algorithms for this problem might be speeded up even more than the cases reported here. Thus it is possible to achieve very large speedup factors if the effort is made to make efficient use of the vector capabilities of a machine.

As discussed in section IV, the Poulsen-Billing-Steinfeld potential is simpler than the Redmon-Binkley one, and as mentioned in an earlier part of this section, we restricted q_{1max}, μ, and q_{2max} to 0, 0, and 2 for test runs with this potential, and the angular integration of eq. (83) was carried out by 27-point rather than N_q^3-point quadrature for the 101-channel timing runs. Nevertheless, the POT segment still consumes about 30% of the total time when the rest of the code is vectorized. In our production runs we used the Redmon-Binkley potential, larger values of q_{max}, and N_q^3-point quadratures with N_q = 4-11 (see Table 1); these improvements, which are necessary for good accuracy, resulted in POT requiring much more time in production runs. Thus we spent extra effort to vectorize our final potential routine for these runs, as described next.

The vectorization of the POT segment proceeded as follows. The first part of the potential matrix evaluation, which is independent of sector, is the calculation of the $B^{q_1 q_2 \mu}_{\beta\ell\beta'\ell'}$ via eq. (91). This is the most time consuming part of the calculations with the versions of the code used for the timing analyses presented here, and it cannot easily be vectorized because most of the $B^{q_1 q_2 \mu}_{\beta\ell\beta'\ell'}$ require different numbers of terms in the sum of eq. (91) and our routines to calculate 3j symbols (or Clebsch-Gordan coefficients) are not easily vectorizable. The sector-dependent parts of POT are also time consuming, but are more amenable to vectorization, as discussed next.

We calculate the $\nu_{q_1 q_2 \mu}$ of eq. (78) by performing the three-dimensional quadrature of eq. (83). In carrying out this integration we evaluate the potential function at all N_q^3 different orientations with fixed R_1, R_2, and r using vector arithmetic prior to doing any of the quadratures. This requires considerable storage, but it enables us to use fairly large vector lengths, equal to N_q^3, which varies from 1331 at small r, and decreasing to 64 at large r. Note that these vector lengths are independent of the number of channels. The sum in the three-dimensional integral to get the $\nu_{q_1 q_2 \mu}$ is done using the vector dot product functions. To do this we write

$$\nu_{q_1 q_2 \mu}(R_1,R_2,r) = \sum_{n_1=1}^{N_q} \sum_{n_2=1}^{N_q} \sum_{n_3=1}^{N_q} \omega_{n_1} \omega_{n_2} \omega'_{n_3} Y^*_{q_1 q_2 \mu}(\theta_{n_1},\theta_{n_2},\phi_{n_3}) \times V(\theta_{n_1},\theta_{n_2},\phi_{n_3},R_1,R_2,r) \quad (92)$$

$$= \sum_{i=1}^{N_q^3} W_i^{q_1 q_2 \mu} V(\underset{\sim}{x}_i,r) \quad (93)$$

where ω_i, ω'_i, θ_i and ϕ_i are the appropriate quadrature weights and nodes,

and $W_i^{q_1 q_2 \mu}$ is a product of three weights times the expansion function, and we use the vector dot product function (SDOT on the Cray-1, Q8SDOT on the Cyber 205) to evaluate eq. (93). The $W_i^{q_1 q_2 \mu}$ depend on the value of N_q and thus on the value of r for which eq. (93) is evaluated; they are calculated once and stored for each r range given in Table 1. As for the evaluation of the potential function, the three-dimensional integral in eq. (83) is independent of the total number of channels. The vector length used in this part of the calculation is again N_q^3, which varies from 1331 to 64.

Our routine then loops through the channel pairs and calculates the $C_{\gamma\gamma'}^{q_1 q_2 \mu}$, the $V_{\alpha\alpha'}$, via eq. (80), and finally the $V_{nn'}$, using eq. (77). The calculation of the $C_{\gamma\gamma'}^{q_1 q_2 \mu}$ requires a 2-dimensional integral and this too is done using the vector dot product functions; that is, we write

$$C_{\gamma\gamma'}^{q_1 q_2 \mu}(r) = \sum_{i=1}^{7} \sum_{j=1}^{7} \tilde{\omega}_i^{v_1 j_1 v_1' j_1'} \tilde{\omega}_j^{v_2 j_2 v_2' j_2'} \nu_{q_1 q_2 \mu}(x_i, x_j, r) \qquad (94)$$

$$= \sum_{k=1}^{49} \tilde{W}_k^{\gamma\gamma'} \nu_{q_1 q_2 \mu}(x_{1k}, x_{2k}, r) \qquad (95)$$

and evaluate eq. (95) using the Cray-1 or Cyber 205 dot product function. The $\tilde{\omega}_i^{vjv'j'}$ are independent of r and are computed once before the start of the R matrix propagation. The $\tilde{W}_k^{\gamma\gamma'}$ are calculated from the $\tilde{\omega}_i^{vjv'j'}$ at each sector for each $\gamma\gamma'$ pair. We compute the $\tilde{W}_k^{\gamma\gamma'}$ on the Cyber 205 by first using the vector Q8VGATHR function to form two vectors of length 49 containing the appropriate $\tilde{\omega}_i^{vjv'j'}$, then by multiplying these two vectors. The $\tilde{W}_k^{\gamma\gamma'}$ are recomputed rather than stored because storing them would involve, for the 530-channel case, an array of length 11.6 million words, whereas the amount of storage required for the $\tilde{\omega}_i^{vjv'j'}$ is only 3.4 thousand words. The integrals in eq. (95) must be performed for each $q_1 q_2 \mu$ with nonzero $B_{\beta\ell\beta'\ell'}^{q_1 q_2 \mu}$ for each of the $\beta\ell\beta'\ell'$ pairs. The number of these integrations is thus proportional to N^2, and depends on the value of r, because the number of terms in eq. (78) depends on the value of r (see Table 1). On the Cyber 205, in the large channel limit where the evaluation of eq. (92) is negligible, these procedures result in a total speedup factor of about 3 for the vector version of segment POT as compared to the scalar version. On the Cray-1, the speedup factor is about 2.

We now consider some computer times for cases with N = 400 and 530. Table 5 gives the times for calculations of 7 energies. (For N = 530 on the Cray-1 these times are estimated from a two-energy run.) The Cyber 205 uses 70% of the time required by the Cray-1 for N = 400 and 50% of the time for N = 530. Also included in Table 5 is the cost of the calculations in

Table 5. Execution times for N = 400 and N = 530 for 7 energies

Machine	CPU hours	billing units (equivalent hours)
	$N = 400$[a]	
Cray-1	11.7	18.1
Cyber 205	7.6	9.2
	$N = 530$[b]	
Cray-1	21.0	31.8
Cyber 205	9.4	15.5

[a] $N_s = 288$

[b] $N_s = 291$

Table 6. Breakdown of execution times (in hours) into program sections for N = 400 and N = 530 on the Cray-1

Program section	N = 400[a]	N = 530[b]
first energy		
Angular integrals[c] ($B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'}$)	1.2	2.4
Potential function evaluation[d]	0.2	0.2
Radial integrals[e]	1.4	2.2
Subtotal for POT[f]	2.8	4.8
DCALC	1.1	1.8
TAUMTS	0.6	1.2
RCALC	1.0	1.6
Total[g]	5.5	9.4
second or subsequent energy		
TAUMTS	0.1	0.3
RCALC	1.0	1.6
Total[g]	1.1	1.9

[a] $N_s = 288$

[b] $N_s = 291$

[c] eq. (91)

[d] Time spent in subprogram FUN, which evaluates the interaction potential

[e] eq. (95)

[f] The evaluation of eq. (92) for all $q_1 q_2 \mu$ and all sectors is included in this total but requires only 28 seconds.

[g] The time for GENSCAT is included but is less than 2 minutes. Overhead is also included but is less than 1%.

equivalent hours. The difference between the CPU time and the equivalent time is due to charges for memory usage, input/output requests, permanent file usage, and, on the Cyber 205, paging to virtual memory. On the Cyber 205, to minimize page fault charges and turnaround time, we mapped all of our large arrays onto large (65,356 word) pages. We let the Cyber 205 operating system schedule all the page requests.

Table 6 shows a breakdown of the total times into those for individual program segments and subsegments for first and second energies on the Cray-1. For first energies for both N = 400 and 530, POT requires about half the total time, while in POT, about half of its total time is spent evaluating the sector-independent coefficients $B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'}$. It should be noted that after these calculations were performed, we were able to implement a new algorithm for which the calculation of the $B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'}$ requires about a factor of 4 less time. Our original routine for calculating the $B^{q_1 q_2 \mu}_{\beta \ell \beta' \ell'}$ used eq. (91) and evaluated the Clebsch-Gordan coefficients using recursion over one of the j indices.[34] The faster version evaluates the Clebsch-Gordan coefficients using recursion over one of the m indices.[34] This saves time since most of the Clebsch-Gordan coefficients required are in a sum over one of the m indices, thus the recursion needs to be performed only a maximum of 6 times as compared to $2 + 4(m_{max} - m_{min})$ times for the original version, where m_{max} and m_{min} are the range of m in eq. (91). For consistency the CPU times in Tables 5 and 6 are based on the slower version.

The bulk of the remaining time in POT is taken up with the 2-dimensional integrals of eq. (95). This is largely as a consequence of the facts that (i) we have not assumed the separability of rotation and vibration and so our vibrational eigenfunctions depend on the rotational state, and (ii) we have retained a very large number of terms in eq. (78) to accurately represent a realistic diatom-diatom potential. The seven points per radial coordinate is essentially optimal[33] and use of any less optimal numerical quadrature in this step would greatly slow the calculations. The program sections DCALC, TAUMTS, and RCALC all require about the same amount of time, about one hour for N = 400 and 1.5 hours for N = 530. GENSCAT and overhead require a very small fraction of the total time. For second energies, the majority of computer time is spent on RCALC.

Table 7 shows times per sector on the Cray-1 for N = 55, 101, 400, and 530. We see that, on this computer, the time required scales approximately as $N^{\frac{1}{2}}$ for N going from 55 to 101, N^3 for N going from 101 to 400, and N^2 for N going from 400 to 530. On the Cyber 205 the approximate scaling for N going from 400 to 530 is N^1.

Table 7. Execution times (in seconds) per sector for N = 55, 101, 400, and 530 on the Cray-1

	Program section			
N	DCALC	TAUMTS (1st E)	RCALC	GENSCAT
55[a]	0.074	0.043	0.060	0.41
101[b]	0.33	0.15	0.27	1.1
400[a]	14.	7.6	12.	8.8
530[a]	22.	14.	19.	23.

[a]Redmon-Binkley potential

[b]Poulsen-Billing-Steinfeld potential

VII. RESULTS

We have computed probabilities for the transitions

$$2HF(v=1,j=0) \rightarrow HF(v_1'=0,j_1') + HF(v_2'=2,j_2') \tag{96}$$

for several energies for N = 500, 530, and 694. However, the results are not converged with respect to increasing N. More complete discussion of the physics will be possible when convergence is achieved.

VIII. CONCLUSIONS

We have successfully used supercomputers to set up and solve quantum mechanical scattering problems representing diatom-diatom collisions with up to 694 coupled equations. These problems are already considerably larger than the biggest calculations performed so far without supercomputers, and thus we may say that the use of supercomputers has definitely extended our capability. Converged solutions of the vibration-vibration energy transfer problem studied here will require even more coupled equations than considered here. We anticipate though that convergence, at least for the prototype case of total angular momentum zero, may be achievable by a combination of (i) further code optimizations, (ii) possible use of algorithms that make more efficient use of vector processors, (iii) use of even more powerful supercomputers, and/or (iv) longer runs.

The speed enhancements we have achieved with respect to the most popular minicomputer, the VAX 11/780 with floating point acceloerator, are very significant. For the total problem, based on 101-coupled-equations test runs and for a case in which seven energies are calculated simultaneously, the speed enhancement corresponds to a factor of 490 on the Cyber 205 and a

factor of 223 on the Cray-1. Some individual program segments are speeded up even more, for example a factor of 684 on the Cyber 205 for a segment involving linear-equation solving and matrix multiplication. We did not run more than 101 coupled equations on the minicomputer but the supercomputers become relatively even more efficient as the equation order and hence vector length increases.

These early calculations were often frustrating and slow because of the difficulties of using new systems, remote usage, and access. At present though many of the problems have been or are being ironed out, and the future looks rosy for wide progress, by other groups as well as our own, for solving very-large-scale quantum mechanical scattering problems on these machines.

IX. ACKNOWLEDGMENTS

We are happy to acknowledge several individuals and grants for contributing to and supporting this research. Nancy Mullaney Harvey and Devarajan Thirumalai collaborated on earlier applications of R matrix propagation to easier problems. Robert W. Numrich of Control Data Corporation provided the crucial encouragement (translate: early funding) that started us on the road to the present vector computations. Computer time was provided by Control Data Corporation, first by grant no. 82-CSUD3 for use of the Cyber 205 at Colorado State University and later by making additional computer time available at Arden Hills, and by the Minnesota Supercomputer Institute, by a grant of time on the University of Minnesota Cray-1. The work at Control Data Corporation was facilitated by both Robert W. Numrich and David Antongiovanni and the work at the University of Minnesota was facilitated by Michael Skow, acting director of the University Computer Center, and his staff. Our departmental minicomputer was purchased with funds obtained in part from the National Science Foundation, and our work on molecular scattering calculations has been supported for many years by the National Science Foundation, most recently by grant no. CHE83-17944. The continued and augmented support for the present project by the chemical physics program officers of the National Science Foundation is greatly appreciated.

1. H. F. Schaefer III, Interaction potentials I: Atom-molecule potentials, in: "Atom-Molecule Collision Theory", R. B. Bernstein, ed., Plenum Press, New York (1979), chap. 2.
2. G. F. Adams, G. D. Bent, R. J. Barlett, and G. D. Purvis, Calculation of potential energy surfaces for HCO and HNO using many-body methods, in: "Potential Energy Surfaces and Dynamics Calculations", D. G. Truhlar, ed., Plenum Press, New York (1981), chap. 5; M. S. Gordon, Potential energy surfaces in excited states of saturated molecules, ibid., chap. 7; K. Morokuma and S. Kato, Potential energy characteristics for chemical reactions, ibid., chap. 10; T. H. Dunning, Jr.,

S. P. Walsh, and A. F. Wagner, Theoretical studies of selected reactions in the hydrogen-oxygen system, ibid., chap. 14.

3. K. Morokuma, S. Kato, K. Kitaura, S. Obara, K. Ohta, and M. Hanamura, Potential energy surfaces of chemical reactions, in: "New Horizons of Quantum Chemistry", P.-O. Löwdin and B. Pullman, eds., D. Reidel, Dordrecht, Holland (1983), chap. 16; W. Kolos, Ab initio methods in calculations of intermolecular interaction energies, ibid., chap. 17.
4. W. A. Lester, Jr., Calculation of cross sections for rotational excitation of diatomic molecules by heavy particle impact: Solution of the close-coupling equations, Meth. Comp. Phys. 10:211 (1973); W. A. Lester, Jr., Coupled-channel studies of rotational and vibrational energy transfer by collision, Adv. Quantum Chem. 9:199 (1975); W. A. Lester, Jr., The N coupled-channel problem, in: "Dynamics of Molecular Collisions, Part A", W. H. Miller, ed., Plenum Press, New York (1976), chap. 1.
5. J. C. Light, Inelastic scattering cross sections I: Theory, in: "Atom-Molecule Collision Theory", R. B. Bernstein, ed., Plenum Press, New York (1979), chap. 6.
6. D. Secrest, Rotational excitation I: The quantal treatment, in: "Atom-Molecule Collision Theory", R. B. Bernstein, ed., Plenum Press, New York (1979), chap. 8.
7. R. Bellman, R. Kalaba, and G. M. Wing, Invariant imbedding and the reduction of two-point boundary value problems to initial value problems, Proc. Natl. Acad. Sci. 46:1646 (1960); R. Bellman, R. Kalaba, and M. Prestrud, "Invariant Imbedding and Radiative Transfer in Slabs of Finite Thickness", American Elsevier, New York (1963).
8. A. Degasperis, Generalization of the phase method to multi-channel potential scattering, Nuovo Cimento 34:1667 (1964); M. E. Riley and A. Kuppermann, Vibrational energy transfer in collisions between diatomic molecules, Chem. Phys. Lett. 1:537 (1968); B. R. Johnson and D. Secrest, Quantum-mechanical calculations of the inelastic cross sections for rotational excitation of para and ortho H_2 upon collision with He, J. Chem. Phys. 48:4682 (1968).
9. J. C. Light and R. B. Walker, An R matrix approach to the solution of coupled equations for atom-molecule reactive scattering, J. Chem. Phys. 65:4272 (1976); E. B. Stechel, R. B. Walker, and J. C. Light, R-matrix solution of coupled equations for inelastic scattering, J. Chem. Phys. 69:3518 (1978).
10. N. A. Mullaney and D. G. Truhlar, The use of rotationally and orbitally adiabatic basis functions to calculate rotational excitation cross sections for atom-molecule collisions, Chem. Phys. 39:91 (1979).
11. (a) N. Mullaney Harvey and D. G. Truhlar, The use of vibrationally adiabatic basis functions for inelastic atom-molecule scattering, Chem. Phys. Lett. 74:252 (1980); (b) D. Thirumalai and D. G. Truhlar, Rapid convergence of V-V energy transfer calculated using adiabatic basis functions and an accurate two-state model for low-energy resonant V-V energy transfer, J. Chem. Phys. 76:5287 (1982).
12. N. A. Mullaney and D. G. Truhlar, Rotationally and orbitally adiabatic basis sets for electron-molecule scattering, Chem. Phys. Lett. 58: 512 (1979); D. G. Truhlar, N. M. Harvey, K. Onda, and M. A. Brandt, Applications of close coupling algorithms to electron-atom, electron-molecule, and atom-molecule scattering, in: "Algorithms and Computer Codes for Atomic and Molecular Quantum Scattering Theory", Vol. I, L. Thomas, ed., National Resource for Computation in chemistry, Lawrence Berkeley Laboratory, Berkeley, CA (1979), chap. 14.
13. E. P. Wigner, Resonance reactions, Phys. Rev. 70:606 (1946); E. P. Wigner and L. Eisenbud, Higher angular momenta and long range interaction in resonance reactions, Phys. Rev. 72:29 (1947); P. G. Burke and W. D. Robb, The R-matrix theory of atomic processes, Adv. At.

Mol. Phys. 11:143 (1975); R. W. Numrich and R. G. Kay, Dissociation dynamics of collinear triatomic systems by the R-matrix method, J. Chem. Phys. 70:4343 (1979); J. Gerratt and I. D. L. Wilson, L^2 R-matrix studies of molecular collision processes; Energy dependence of $\sigma_{vj \to v'j'}$ for $^4He + H_2$, Proc. Roy. Soc. Lond. Ser. A 372:219 (1980).

14. D. J. Diestler and V. McKoy, Quantum mechanical treatment of inelastic collisions. II. Exchange reactions, J. Chem. Phys. 48:2951 (1968); D. G. Truhlar and A. Kuppermann, Exact and approximate quantum mechanical reaction probabilities and rate constants for the collinear $H + H_2$ reaction, J. Chem. Phys. 56:2232 (1972).
15. A. Askar, A. Cakmak, and H. Rabitz, Finite element methods for reactive scattering, Chem. Phys. 33:267 (1978).
16. I. H. Zimmerman, M. Baer, and T. F. George, $F + H_2$ collisions on two electronic potential energy surfaces: Quantum mechanical study of the collinear reaction, J. Chem. Phys. 71:4132 (1979); M. Baer, Quantum mechanical treatment of electronic transitions in atom-diatom exchange collisions, Ber. Bunsenges. Physik. Chem. 86:448 (1982); M. Baer, Quantum mechanical treatment of electronic transitions in atom-molecule collisions, Top. Curr. Phys. 33:117 (1983).
17. T. G. Schmalz, E. B. Stechel, and J. C. Light, Time independent quantum theory of electron transfer collisions using a nonorthogonal basis and R-matrix propagation, J. Chem. Phys. 70:5640 (1979); B. C. Garrett, M. J. Redmon, D. G. Truhlar, and C. F. Melius, Ab initio treatment of electronically inelastic K + H collisions using a direct integration method for the solution of the coupled-channel scattering equations in electronically adiabatic representations, J. Chem. Phys. 74:412 (1981); B. C. Garrett and D. G. Truhlar, The coupling of electronically adiabatic states in atomic and molecular collisions, Theor. Chem. Adv. Perspectives 6A:215 (1981); J. Gerratt, R-matrix theory of charge transfer, Phys. Rev. A 30:1643 (1984).
18. G. C. Schatz and A. Kuppermann, Quantum mechanical reactive scattering for three-dimensional atom plus diatom systems. I. Theory, J. Chem. Phys. 65:4642 (1977); R. E. Wyatt, Direct-mode chemical reactions I: Methodology for accurate quantal calculations, in: "Atom-Molecule Collision Theory", R. B. Bernstein, ed., Plenum Press, New York (1979), chap. 17.
19. D. L. Moores, Applications of the close-coupling method to electron molecule scattering, in: "Electron-Molecule and Photon-Molecule Collisions", T. Rescigno, V. McKoy, and B. Schneider, eds., Plenum Press, New York (1979), chap. 1; M. A. Morrison, The coupled-channels integral-equations method in the theory of low-energy electron-molecule scattering, ibid., chap. 2.
20. D. Secrest, Theory of rotational and vibrational energy transfer in molecules, Annu. Rev. Phys. Chem. 24:379 (1973); M. Faubel and J. P. Toennies, Scattering studies of rotational and vibrational excitation of molecules, Adv. At. Mol. Phys. 13:229 (1977).
21. M. H. Alexander, Close-coupling studies of rotationally inelastic HF-HF collisions at hyperthermal energies, J. Chem. Phys. 73:5735 (1980).
22. H. Rabitz, The dimensionality and choice of effective hamiltonians for molecular collisions, J. Chem. Phys. 63:5208 (1975); D. G. Truhlar, Recent progress in atomic and molecular collisions and the interface with electronic structure theory, Int. J. Quantum chem. Symp. 17:77 (1983).
23. M. H. Alexander and A. E. DePristo, Symmetry considerations in the quantum treatment of collisions between two diatomic molecules, J. Chem. Phys. 66:2166 (1977).
24. P. Pechukas and J. C. Light, On the exponential form of time displacement operators in quantum mechanics, J. Chem. Phys. 44:3897 (1966);

P. Chang and J. C. Light, Exponential solution of the Schrödinger equation: Potential scattering, J. Chem. Phys. 50:2517 (1969); J. C. Light, Quantum calculations in chemically reactive systems, Meth. Comp. Phys. 10:111 (1971).

25. J. N. Murrell and K. S. Sorbie, New analytic form for the potential energy curves of stable diatomic states, J. Chem. Soc. Faraday Trans. II 70:1552 (1974).
26. A. R. Edmonds, "Angular Momentum in Quantum Mechanics", Princeton University Press, Princeton, NJ (1960).
27. D. U. Webb and K. N. Rao, Vibration rotation bands of heated hydrogen halides, J. Mol. Spectry. 28:121 (1968).
28. M. J. Redmon and S. B. Binkley, to be published.
29. L. L. Poulsen, G. D. Billing, and J. I. Steinfeld, Temperature dependence of HF rotational relaxation, J. Chem. Phys. 68:5121 (1978).
30. G. Gioumousis and C. F. Curtiss, Molecular collisions. II. Diatomic molecules, J. Math. Phys. 2:96 (1961).
31. J. M. Launay, Molecular collision processes I. Body-fixed theory of collisions between two systems with arbitrary angular momenta, J. Phys. B 10:3665 (1977).
32. A. E. DePristo and M. H. Alexander, Relationships among the coupled-states, P-helicity decoupling and effective potential methods, Chem. Phys. 19:181 (1977).
33. D. W. Schwenke and D. G. Truhlar, An optimized quadrature scheme for matrix elements over the eigenfunctions of general anharmonic potentials, Comp. Phys. Commun., in press.
34. K. Schulten and R. G. Gordon, Exact recursive evaluation of 3j- and 6j-coefficients for quantum-mechanical coupling of angular momenta, J. Math. Phys. 16:1961 (1975).

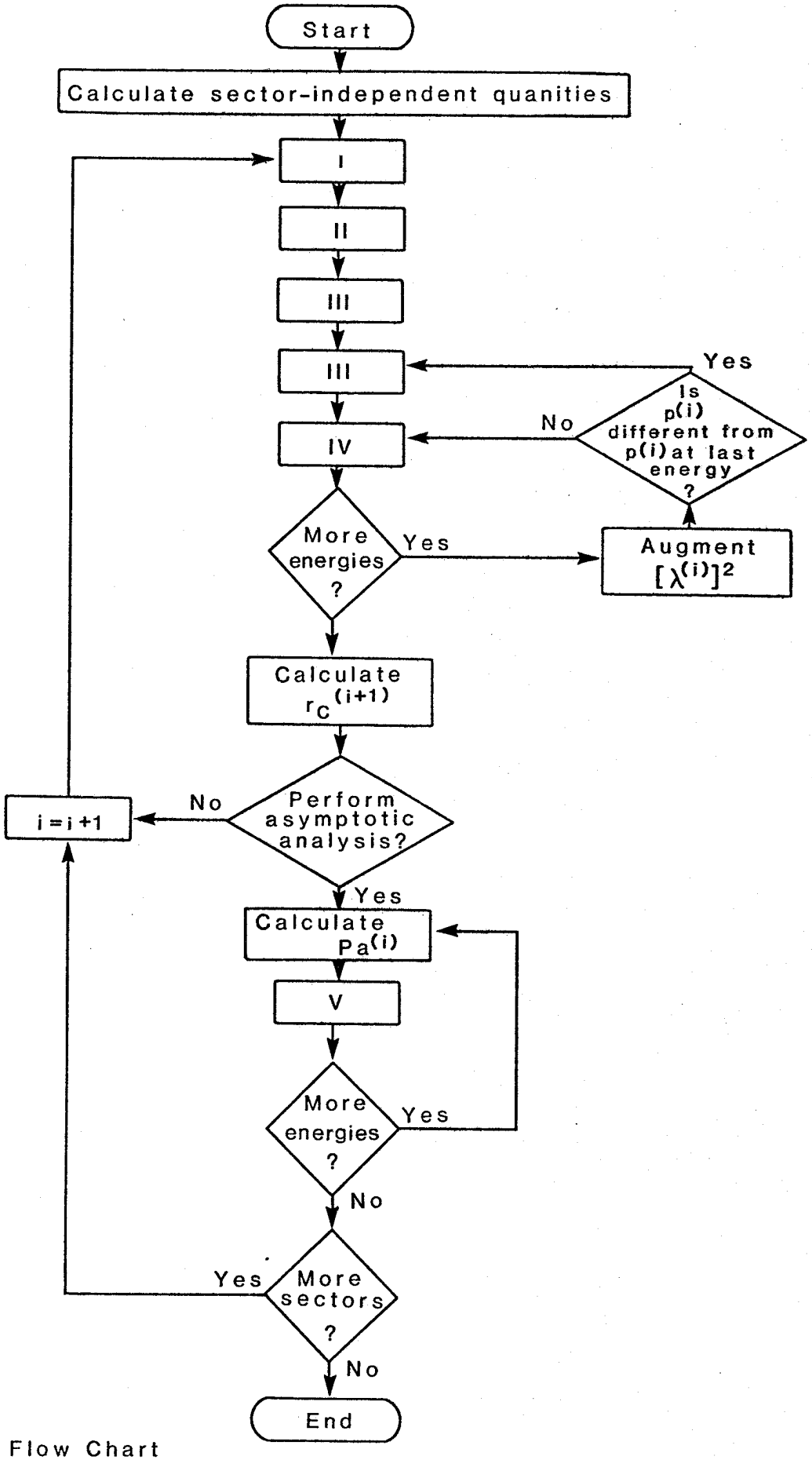

Fig. 1 Flow Chart