CONVERGED CALCULATIONS OF ROTATIONAL EXCITATION AND V-V ENERGY TRANSFER IN THE COLLISION OF TWO MOLECULES

David W. Schwenke and Donald G. Truhlar

Department of Chemistry and Supercomputer Institute
University of Minnesota
Minneapolis, Minnesota 55455, U.S.A.

ABSTRACT

We present the results of large-scale quantum mechanical calculations of state-to-state transition probabilities for the collision of two hydrogen fluoride molecules. We use a potential energy surface obtained by adding a vibrational dependence to the interaction potential of Alexander and DePristo, and we consider zero total angular momentum. We have calculated converged transition probabilities for rotational energy transfer in the rigid rotator approximation and for vibration-to-vibration energy transfer in calculations including full vibration-rotation coupling. The V-V calculations include up to 948 coupled channels. Final production runs were carried out with a highly vectorized code on the University of Minnesota Cyber 205 and Cray-1 computers; earlier test runs were carried out as well on a Cray Research X-MP/48 machine.

1. INTRODUCTION

In molecular quantum mechanics the amount of effort required to treat problems involving systems with two, three, or four atoms increases enormously as each atom is added to the system. The present status in molecular scattering theory is that atom-atom collisions may be treated routinely and atom-diatom collisions may be treated only with great difficulty unless one makes dynamical approximations or artificially restricts or eliminates one or more coordinates.[1] Exact treatments of four-atom systems are generally considered to lie "beyond the state of the art." Encouraged by the great computational enhancements afforded by the class VI computers (i.e., vector pipeline machines like the Cray-1, the Control Data Corporation Cyber 205, and the Cray X-MP series), we have embarked on a quest to obtain converged results for a prototype diatom-collision. Although it is not the simplest such system we chose HF-HF as the prototype for study. The basic reasoning behind this choice of system is as follows. First, the number of internal states involved is much smaller for hydrides than nonhydrides. Second, among collisions of hydrides the HF-HF system has been most widely studied experimentally.[2] In fact the HF-HF system may even be considered the experimental prototype for vibration-to-vibration (V-V) energy transfer, and V-V energy transfer in turn is the dominant energy relaxation mechanism under most conditions where such relaxation is of interest. Furthermore, there have been several studies of the HF-HF potential energy surface,[3] and a knowledge of the potential energy surface is a prerequisite for a dynamics calculation.

We calculate the scattering matrix for the HF-HF collision system by the close coupling method,[1] which for nonreactive scattering essentially involves the solution of N coupled ordinary differential equations, where N is the number of channels. In two earlier reports[4,5] on V-V energy transfer in HF-HF collisions, we discussed calculations including up to 694 channels and here we report 948-channel calculations. The largest number of channels included in previous studies is 510, for which a single column of the scattering matrix was calculated.[6] The method of solution used here and in Refs. 4 and 5 yields all columns. The largest other coupled channel calculation of the whole scattering matrix of which we are aware involves 250 coupled channels.[7] Most close coupling calculations are, however, still restricted to less than 100 channels. Since the number of floating point operations scales as the cube of the number of channels, the present calculations are computationally more intensive than a 100-channel calculation by a factor of $(948/100)^3$ i.e., two and a half orders of magnitude.

We have carried out two kinds of calculations, the first having the diatoms restricted to be rigid, i.e., nonvibrating but free to rotate, and the second with the diatoms also free to vibrate. In both cases we consider only collisions with zero total angular momentum. The major difficulty in these calculations is the representation of the dependence of the wave function on the rotational degrees of freedom. The rigid-rotator calculations, although of considerable interest in their own right,[8] are also designed to help assess the convergence of the rotational basis to provide guidance for the vibrating rotator calculations, whose goal is to converge V-V energy transfer probabilities.

As mentioned above, there have been many studies of the potential energy surface for the HF-HF collisions. In our own previous dynamical calculations of V-V energy transfer in this system[4,5,9] we have used the potentials of Poulsen *et al.*[3c] and Redmon and Binkley.[31] In the present work we build a new potential by starting with the rigid-rotator potential of Alexander and DePristo[3b] and modifying it to have a dependence on vibrational coordinates. This is accomplished by extending a procedure proposed by Gianturco and co-workers.[3e]

2. THEORY

In this section we summarize the methods we use for the quantum mechanical scattering calculations of the collisions of two identical molecules. A more detailed treatment is given in Ref. 4. The quantum mechanical wave function is written as

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

where $\vec{r} \equiv (r, \hat{r})$, r is the distance between the centers of mass of the two molecules, $\hat{r}$ denotes the orientation of a vector from the center of mass of molecule 1 to the center of mass of molecule 2 in a laboratory-frame coordinate system, $\underset{\sim}{x}$ denotes all other internal coordinates of the two-molecule system, E denotes the total energy, X_n is a symmetrized basis function, and f_{nn_0} is a radial translational wave function. The expansion (1) is called the close coupling expansion, X_n is called a channel function, and n and n_0 are channel indices. The second index n_0 on f_{nn_0} denotes which of the radial translational wave functions is given initial-state boundary conditions in the particular solution of (1) under consideration; all other radial translational wave functions have outgoing or exponentially decaying boundary conditions at large r. Those with outgoing boundary conditions are called open channels, and those with exponentially decaying boundary conditions are called closed channels. The equations to be solved for the radial translational functions are obtained by substituting Eq. (1) into the time-independent Schrödinger equation, multiplying in turn by each X_n^*, multiplying by $(-2\mu_{rel}r/\hbar^2)$, where μ_{rel} is the reduced mass equal to one half the molecular mass, and integrating over $\underset{\sim}{x}$ and $\hat{r}$. This yields

$$\left(\frac{d^2}{dr^2} - \frac{l_n(l_n+1)}{r^2} + k_n^2\right) f_{nn_0}(r) = \frac{2\mu_{rel}}{\hbar^2} \sum_m V_{nm}(r)\, f_{mn_0}(r) \quad , \tag{2}$$

l_n is the translational orbital angular momentum quantum number for channel n, k_n is the magnitude of the wave vector for channel n, and V_{mn} is a matrix element of the interaction potential V in the basis X_n. The equations (2) are called the close coupling equations. In practice the summations in (1) and (2) are truncated at N terms, but the results become exact only in the limit as $N \rightarrow \infty$. The coupled equations (2) are integrated numerically using the R-matrix propagation algorithm with the usual scattering boundary conditions.[4,10,11] The results of the calculation is a matrix of complex scattering matrix elements whose absolute values squared are probabilities for transitions from state n_0 to state m. For the vibrating molecule calculations, we calculated k_n^2 using

$$k_n^2 = 2\mu_{rel}\ (E - \varepsilon_n)\ /\ \hbar^2 \quad , \tag{3}$$

where ε_n is the sum of two diatomic vib-rotational energy levels evaluated from the constants of Webb and Rao.[12] For the rigid rotator calculations, we calculated rotational energy levels by the formula given by DePristo and Alexander[8a] to be consistent with earlier work on this potential.

In order to simplify the calculations we take advantage of the conservation of total angular momentum and the fact that the spatial wave function (1) must be symmetric under the interchange of the coordinates of the two identical molecules. Thus we use basis functions that are eigenfunctions of the total angular momentum and the molecule interchange operator; this block diagonalizes V_{nm} in the total angular momentum quantum number J, its component M on a laboratory-fixed axis, and the interchange symmetry quantum number η and therefore allows the close coupling equations to be solved separately for each J and η (the solutions are independent of M). We are only going to consider initial states with identical quantum numbers for the two molecules; these states have even interchange symmetries, and so they are only coupled to other symmetric basis functions, and we restrict our basis to such symmetric functions. The index n then specifies a set of quantum numbers (v_1, v_2), (j_1, j_2), j_{12}, l, J, M, and η, where v_1 and v_2 are vibrational quantum numbers, j_1 and j_2 are rotational quantum numbers, and j_{12} is the quantum number for the vector sum of the rotational angular momenta of the two molecules. The symmetric basis functions are given by

$$X_n = \left[2(1 + \delta_{v_1 v_2}\delta_{j_1 j_2})\right]^{-1/2}\left[\phi_\alpha(\underset{\sim}{x}, \hat{r}) + (-1)^{j_1+j_2+j_{12}+l}\phi_{\bar{\alpha}}(\underset{\sim}{x}, \hat{r})\right] \tag{4}$$

where ϕ_α is a distinguishable-molecule vibrational-rotational-orbital basis function, α is a set of distinguishable-molecule quantum numbers, and $\bar{\alpha}$ denotes the basis function which has v_1 and j_1 interchanged with v_2 and j_2 as compared to the basis function specified by α. Thus n stands for the quantum numbers v_1, j_1, v_2, j_2, j_{12}, l, J, M, and η, where, because it is not possible to distinguish which molecule has which set of quantum numbers, only sets with $v_1 > v_2$ or $v_1 = v_2$ and $j_1 \geq j_2$ are included, 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. Another consequence of the interchange symmetry is that the physically meaningful transition probabilities are to symmetric final states that can be labelled either by $v_1 > v_2$ with any pair of j_1 and j_2 or by $v_1 = v_2$ with $j_1 \geq j_2$.

When Eq. (4) is employed, the matrix elements of V over symmetric basis functions are given by[4]

$$V_{nn'} = \left[(1 + \delta_{v_1 v_2}\delta_{j_1 j_2})\ (1 + \delta_{v'_1 v'_2}\delta_{j'_1 j'_2})\right]^{-1/2} \times \ (V_{\alpha\alpha'} + (-1)^{j_1+j_2+j_{12}+l} V_{\bar{\alpha}\alpha'}) \quad , \tag{5}$$

where $V_{\alpha\alpha'}$ is a potential matrix element calculated using the basis functions for the formally distinguishable molecules. To evaluate $V_{\alpha\alpha'}$ it is convenient to expand the interaction potential function V in terms of angular functions for which the integrals over angles are easily performed:

$$V = \sum_{q_1 q_2 \mu} v_{q_1 q_2 \mu}\ (r, r_1, r_2)\ \gamma_{q_1 q_2 \mu}\ (\hat{r}_1, \hat{r}_2) \tag{6}$$

where

$$\gamma_{q_1 q_2 \mu} = \frac{4\pi}{\left[2(1 + \delta_{\mu 0})\right]^{1/2}} \left[Y_{q_1 \mu}\ (\hat{r}_1)\ Y_{q_2 -\mu}\ (\hat{r}_2) + Y_{q_1 -\mu}(\hat{r}_1)\ Y_{q_2 \mu}(\hat{r}_2)\right] \quad , \tag{7}$$

$\vec{r}_i$ is a vector indicating the bond length and direction of the bond axis of molecule i measured in the frame of reference where the z axis is along $\vec{r}$, and $Y_{lm}$is a spherical harmonic. Our vibrational-rotational-orbital basis functions take the form

$$\phi_\alpha = (R_1 R_2)^{-1}\chi_{v_1 j_1}(R_1)\chi_{v_2 j_2}(R_2) \sum_{m_1 m_2 m_{12} m_l} (j_1 m_1 j_2 m_2 \mid j_1 j_2 j_{12} m_{12})$$

$$\times\ (j_{12}m_{12}lm_l \mid j_{12}lJM)\ Y_{j_1m_1}(\hat{R}_1)\ Y_{j_2m_2}(\hat{R}_2)\ Y_{lm_l}(\hat{r})\quad , \tag{8}$$

where χ_{vj} is a vibrational wave function, (. . . . | . . . .) denotes a Clebsch-Gordon coefficient, and $\vec{R}_i$ is a vector indicating the bond length and direction of molecule i in the frame of reference where the Z axis is along a laboratory-fixed direction. Then the potential matrix elements are given by

$$V_{\alpha\alpha'} = \sum_{q_1q_2\mu} < v_1j_1v_2j_2 \mid v_{q_1q_2\mu} \mid v'_1j'_1v'_2j'_2 > \times\ < j_1j_2j_{12}lJM \mid \gamma_{q_1q_2\mu} \mid j'_1j'_2j'_{12}l'JM > \tag{9}$$

where the first factor is a vibrational integral given by

$$< v_1j_1v_2j_2 \mid v_{q_1q_2\mu} \mid v'_1j'_1v'_2j'_2 > = \int dR_1 \int dR_2\ \overset{*}{\chi}_{v_1j_1}(R_1)\ \chi^*_{v_2j_2}(R_2)$$

$$\times\ v_{q_1q_2\mu}(r, R_1, R_2)\, \chi_{v'_1j'_1}(R_1)\ \chi_{v'_2j'_2}(R_2) \tag{10}$$

and the second factor is an angular integral, which can be easily evaluated in terms of sums and products of vector coupling coefficients.[4] We evaluate Eq. (10) using a recently proposed quadrature scheme[13] using 6 points per dimension. The $\{\chi_{vj}\}$ vibrational wave functions are determined by the linear variational method in a basis of harmonic oscillator functions for the HF diatomic potential given by Murrell and Sorbie[14] which is a fit to RKR data. We do not neglect vibrational-rotational coupling. The functions $v_{q_1q_2\mu}$ are evaluated as needed by numerically performing the three-dimensional integral

$$v_{q_1q_2\mu} = \frac{1}{4\pi} \int_{-1}^{1} d[\cos(\phi_1 - \phi_2)]\ \left[1 - \cos^2(\phi_1 - \phi_2)\right]^{-1/2}$$

$$\times \int_{1}^{1} d(\cos\theta_1) \int_{-1}^{1} d(\cos\theta_2)\ \overset{*}{\gamma}_{q_1q_2\mu} V \tag{11}$$

where θ_i and ϕ_i are the inclination and azimuthal angle for $\vec{r}_i$.

The formalism for the rigid rotator calculations is formally identical to that given above with $v_1 = v_2 = 0$ except that integrals over R_1 and R_2 are replaced by functional values at $R_1 = R_e$ and $R_2 = R_e$, where R_e is the equilibrium intermolecular (1.733 a_0) of HF.

The calculations yield transition probabilities $P_{v_1j_1v_2j_2 \to v'_1j'_1v'_2j'_2}$ or, for the rigid rotator case, $P_{j_1j_2 \to j'_1j'_2}$. In the present article we concentrate on a single initial state, $(v_1j_1v_2j_2) = (1010)$ for the vibrating rotator case and $(j_1j_2) = (00)$ for the rigid rotator case. For convenience we sometimes consider probabilities summed over subsets of final rotational states. Thus we define

$$P^{R}_{j'_{\text{sum}}} = \sum_{\substack{j'_1 j'_2 \\ j'_1 + j'_2 = j'_{\text{sum}}}} P_{00 \to j'_1 j'_2} \tag{12}$$

for the rigid rotator case and

$$P^{VV}_{j'_{\text{sum}}} = \sum_{\substack{j'_1 j'_2 \\ j'_1 + j'_2 = j'_{\text{sum}}}} P_{1010 \to 2 j'_1 0 j'_2} \tag{13}$$

for the vibrating rotator case. The total V-V probability, obtained by summing the previous probability over j'_{sum}, is called P^{VV}.

For both kinds of calculations we consider three values of the initial relative translational energy E_{rel}, which equals $\hbar^2 k^2_{n_0}/(2\mu_{rel})$. In order to utilize the efficiencies possible in R-matrix propagation when results are to be calculated at more than one energy we propagate the solutions through a given step for all three energies before proceeding to the next step.[4]

All transition probabilities presented in this article are well converged with respect to increasing the integration range or decreasing the step size, with the largest source of error being the truncation to finite N. The integration range is from 2 to 150 a_0, using a fixed step size of 0.06 a_0 from 2 to 10.1 a_0 and a variable step size thereafter. The variable-step size algorithm is described in Ref. 4; in the notation of that reference we use EPS = 0.01 and a maximum step size of 3 a_0 is reached by the end of the integration. A total of 297-303 integration steps are taken in each case, depending on the expansion basis.

We note that R-matrix propagation algorithm involves the transformation at every integration step into a basis which is adiabatic with respect to the coordinate r. An expansion of the wave function ψ_{n_0} in terms of the adiabatic basis functions is more rapidly convergent than the expansion of Eq. (1). This is especially true for large r. In the present calculations we use the algorithm described in Ref. 4 with parameter EPSRED = 0.01 to reduce the number of terms in the adiabatic expansion. This reduction has a negligible effect on the accuracy of our calculations. As an example of this reduction, the wave function for the lowest-energy calculation with N = 948 is expanded in terms of 948 adiabatic basis functions for r < 51 a_0, but then the expansion is reduced to 893 terms over the range r = 51 to 150 a_0

3. POTENTIAL ENERGY FUNCTION

The original interaction potential of Alexander and DePristo[3b] is a fit to the *ab initio* SCF data of Yarkony *et al.*[3a] in a representation involving space-fixed angular functions. They wrote, for both diatoms at their equilibrium separations,

$$V = (4\pi)^{3/2} \sum_{\lambda_1\lambda_2\lambda} U^e_{\lambda_1\lambda_2\lambda}(r) Y_{\lambda_1\lambda_2\lambda}(\hat{r}, \hat{R}_1, \hat{R}_2) \tag{14}$$

where

$$Y_{\lambda_1\lambda_2\lambda} = \sum_{m_1m_2m} (\lambda_1 m_1 \lambda_2 m_2 / \lambda_1 \lambda_2 \lambda m)\ Y_{\lambda_1 m_1}(\hat{R}_1)\ Y_{\lambda_2 m_2}(\hat{R}_2)\ Y^*_{\lambda m}(\hat{r}) \tag{15}$$

and they truncated the sum in Eq. (14) to 6 terms, namely those with $(\lambda_1\lambda_2\lambda)$ equal to (000), (112), (011), (123), (101), and (213). The coefficients $U^e_{\lambda_1\lambda_2\lambda}$ for the fifth and sixth terms are equal by symmetry to minus those for the third and fourth; and the first four coefficients are shown in Fig. 1. We note that the factor before the summation in Eq. (14) is chosen so that U^e_{000} is the spherical average of the interaction potential. The potential of Ref. 3b was used unmodified in the rigid rotator calculations.

Note that this potential has only 6 terms in the laboratory-frame expansion (14), but it gives rise to 9 terms in the body-frame expansion (6), namely those with $q_1 + q_2 \leq 3$ except for $(q_1q_2\mu)$ equal to (020), (030), (200), and (300).

The calculations of Yarkony *et al.* were single-configuration SCF calculations with a double-zeta-plus-polarization set. To gain some idea of the reasonableness of this data we compare the coefficients of Alexander and DePristo to the analogous ones we computed for the Redmon-Binkley potential. The Redmon-Binkley is a multiparameter fit, including 2-body, 3-body, and 4-body terms, to fourth-order Møller-Plesset perturbation theory calculations with a double-zeta-core, triple-zeta-valence-plus-polarization basis set. The four coefficients retained by Alexander and DePristo, as evaluated from the Redmon-Binkley potential, are shown in Fig. 2. The coefficients from the two potentials show qualitatively similar behavior.

For the vibrating molecule calculations, it was necessary to introduce vibrational dependence into the interaction potential. This was done by extending an approximation of Gianturco *et al.*[3e] for the vibrational dependence of the short-range repulsive interactions and by using accurate R_i- dependent multipole moments to simulate the vibrational dependence of long-range electrostatic interactions.

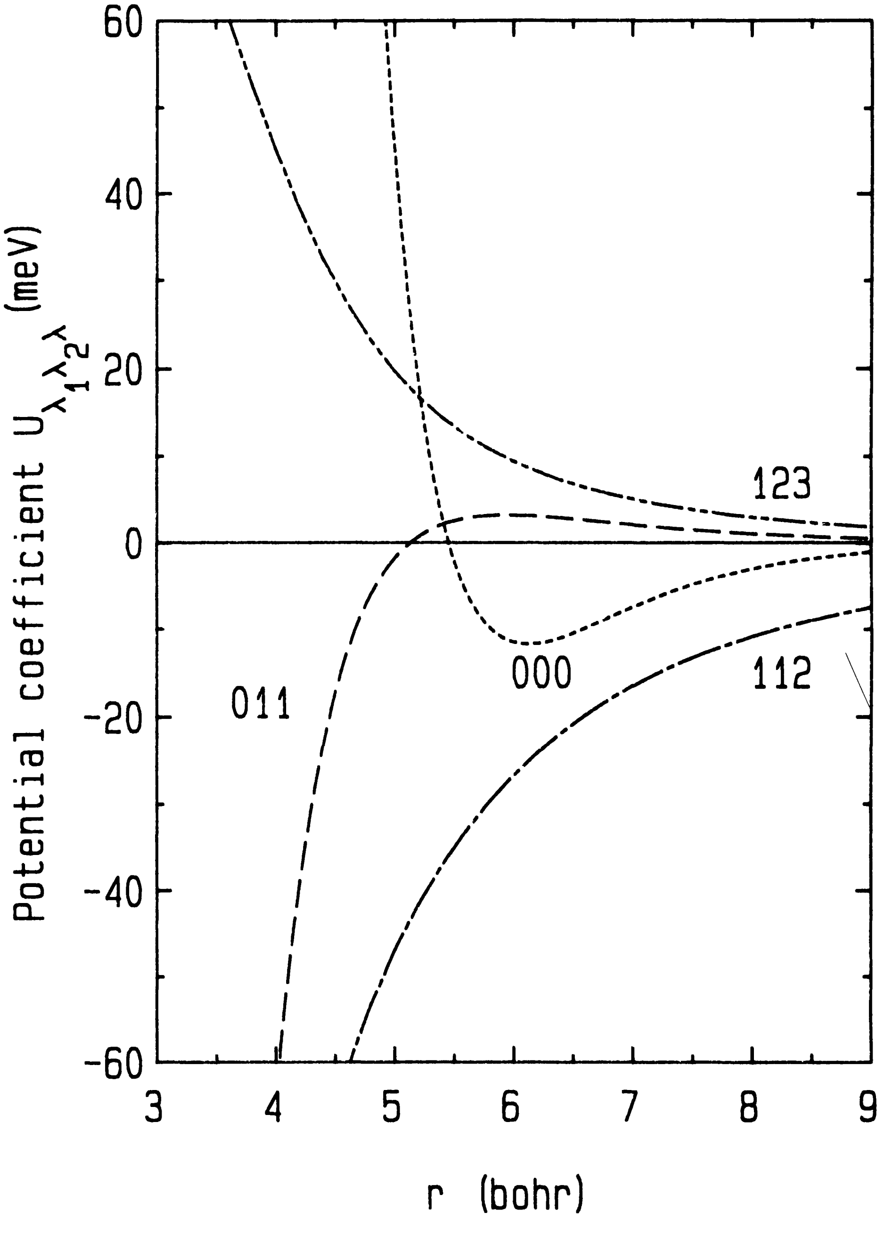

Figure 1. $U_{\lambda_1\lambda_2\lambda}(r, R_1, R_2)$ for the modified Alexander-DePristo potential, with $R_1 = R_2 = R_e$, as functions of the distance between the molecular centers of mass. Short dashed line, U_{000}; long dashed line, U_{011} ; long-short dashed line, U_{112}; long-short-short dashed line, U_{123}.

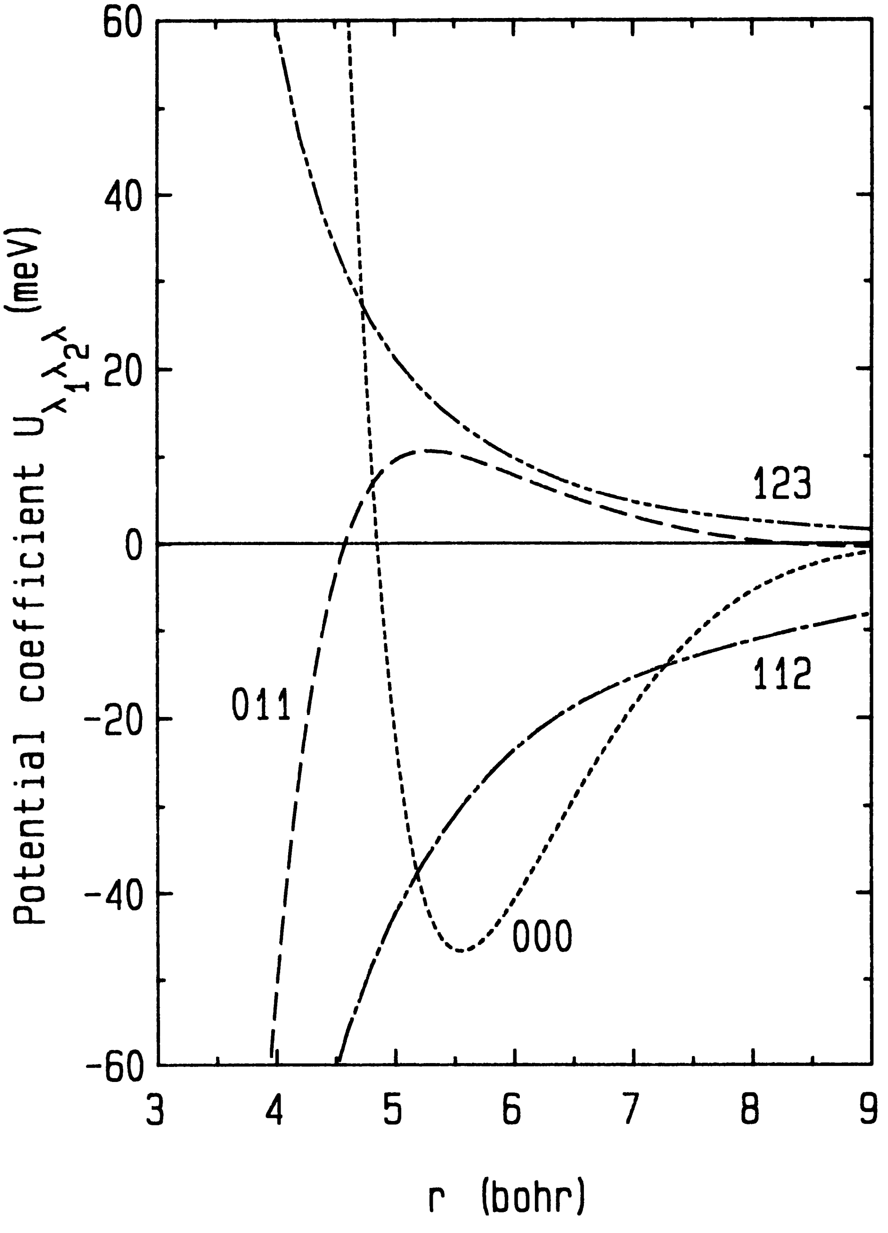

Figure 2. Same as Fig.1 except for the potential of ref.3*l*

Gianturco *et al.*[3e] wrote

$$U_{\lambda_1\lambda_2\lambda} = U^e_{\lambda_1\lambda_2\lambda}(r)\ \exp\left[-\alpha_{\lambda_1\lambda_2\lambda}(R_1 + R_2 - 2R_e) \right] \tag{16}$$

with

$$\alpha_{\lambda_1\lambda_2\lambda} = -\left[U^e_{\lambda_1\lambda_2\lambda}(r_0) \right]^{-1} \frac{dU^e_{\lambda_1\lambda_2\lambda}}{dr} \Big|_{r=r_0} \tag{17}$$

for $(\lambda_1\lambda_2\lambda)$ = (000) and we employ this for all $(\lambda_1\lambda_2\lambda)$. Gianturco *et al.*[3e] suggested that r_0 be the translational classical turning point. We set r_0 equal to 4.1 a_0 because that is approximately the largest distance close to a typical classical translational turning point for which none of the $U^e_{\lambda_1\lambda_2\lambda}$ are passing through zero or near a local extremum. In the original fit[3b] the coefficients U^e_{112} and U^e_{123} take the form

$$U^e_{112} = -c_1e^{-c_2r} - c_3e^{-c_4r} - (2/15)^{\frac{1}{2}}\mu^2 r^{-3} \tag{18}$$

$$U^e_{123} = -(c_5/r - c_6)\, e^{-c_7r} + (1/7)^{\frac{1}{2}}\mu\,\theta\, r^{-4} \tag{19}$$

where the c_i are constants, and μ and θ are the dipole and quadrupole moments, which were set equal to vibrationally averaged values of 0.716 and 1.93 a.u., respectively. We replaced the multipole moments in these expressions with the linear functions

$$\mu = \mu_e + \beta_{\lambda_1\lambda_2\lambda}(R_i - R_e) \tag{20}$$

$$\theta = \theta_e + \gamma_{\lambda_1\lambda_2\lambda}(R_i - R_e) \tag{21}$$

Table I. Parameters for the interaction potential (in a.u.)

λ_1	λ_2	λ	$\alpha_{\lambda_1\lambda_2\lambda}$	$\beta_{\lambda_1\lambda_2\lambda}$	$\gamma_{\lambda_1\lambda_2\lambda}$
0	0	0	2.38		
0	1	1	2.42		
1	0	1	2.42		
1	1	2	0.738	0.838	
1	2	3	0.850	0.918	3.00
2	1	3	0.850	0.918	3.00
				μe	θe
				0.7066[a]	1.64[b]

[a] From Ref. 15.

[b] From Ref. 16.

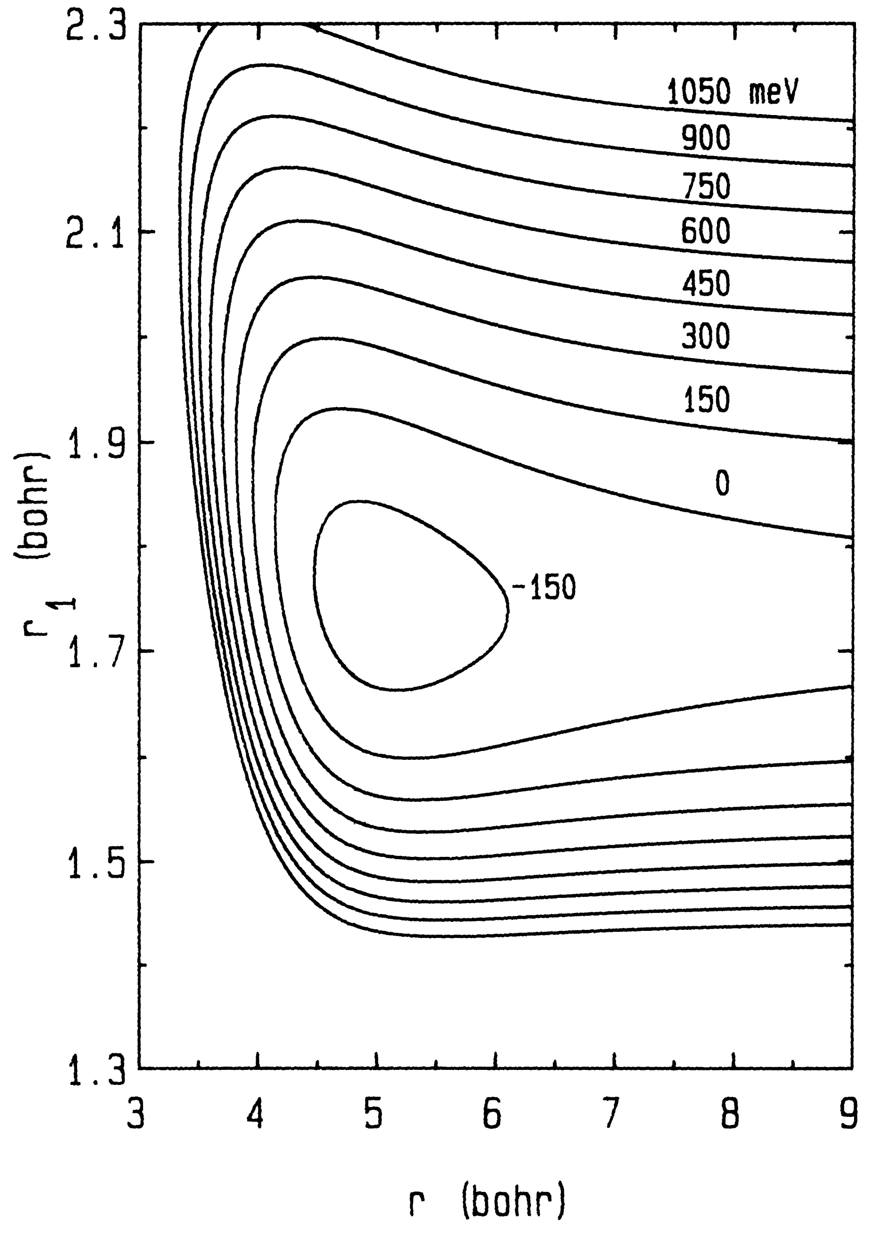

Figure 3. Total potential based on modified Alexander-DePristo interaction potential plus Sorbie-Murrell diatom potentials, as a function of r and r_1 for the collinear configuration with the H of molecule one hydrogen bonded to the F of molecule two. The contours run from 150 to 1050 meV in steps of 150 meV.

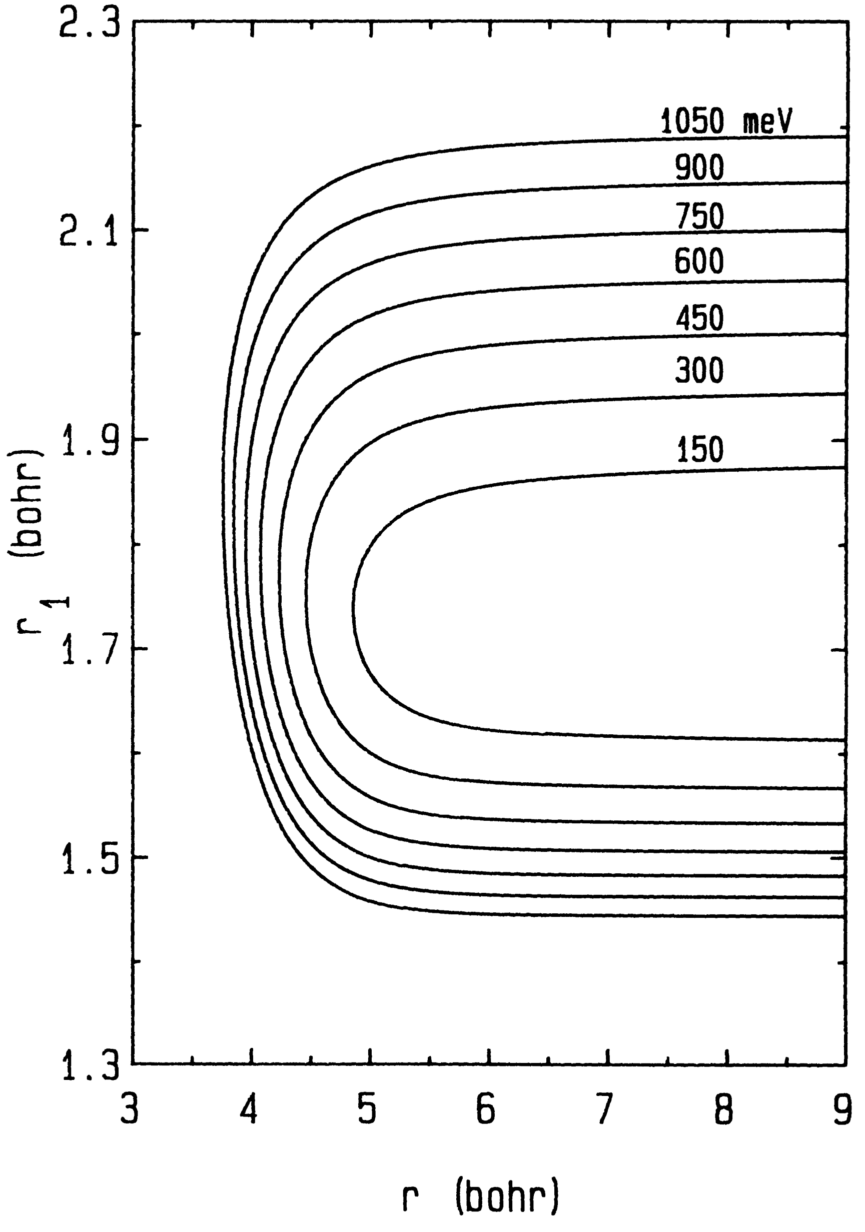

Figure 4. Same as Fig.3 except for $\theta_1 = 90$ ° and $\theta_2 = 0$ °. The contours run from 150 to 1050 meV.

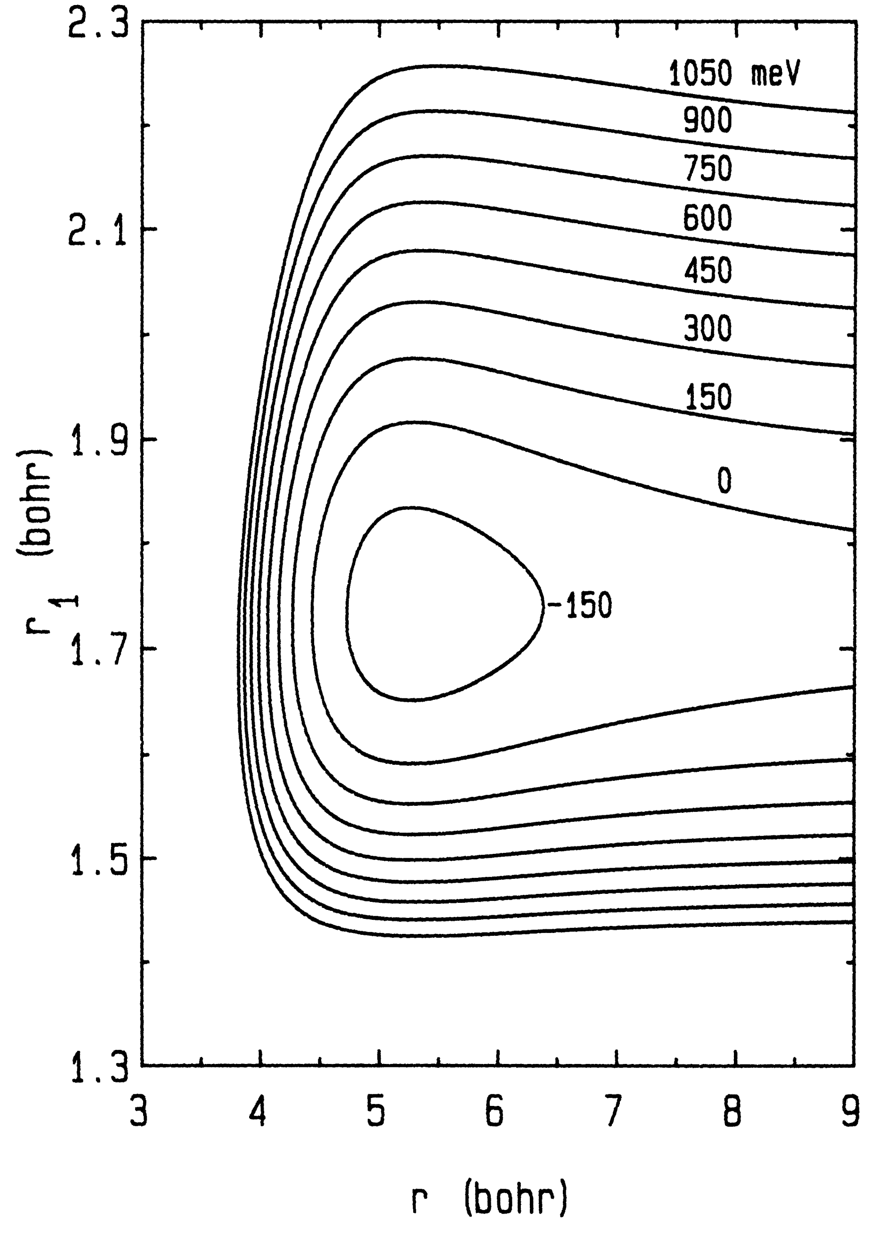

Figure 5. Same as Fig.3 except for the potential of ref.31.

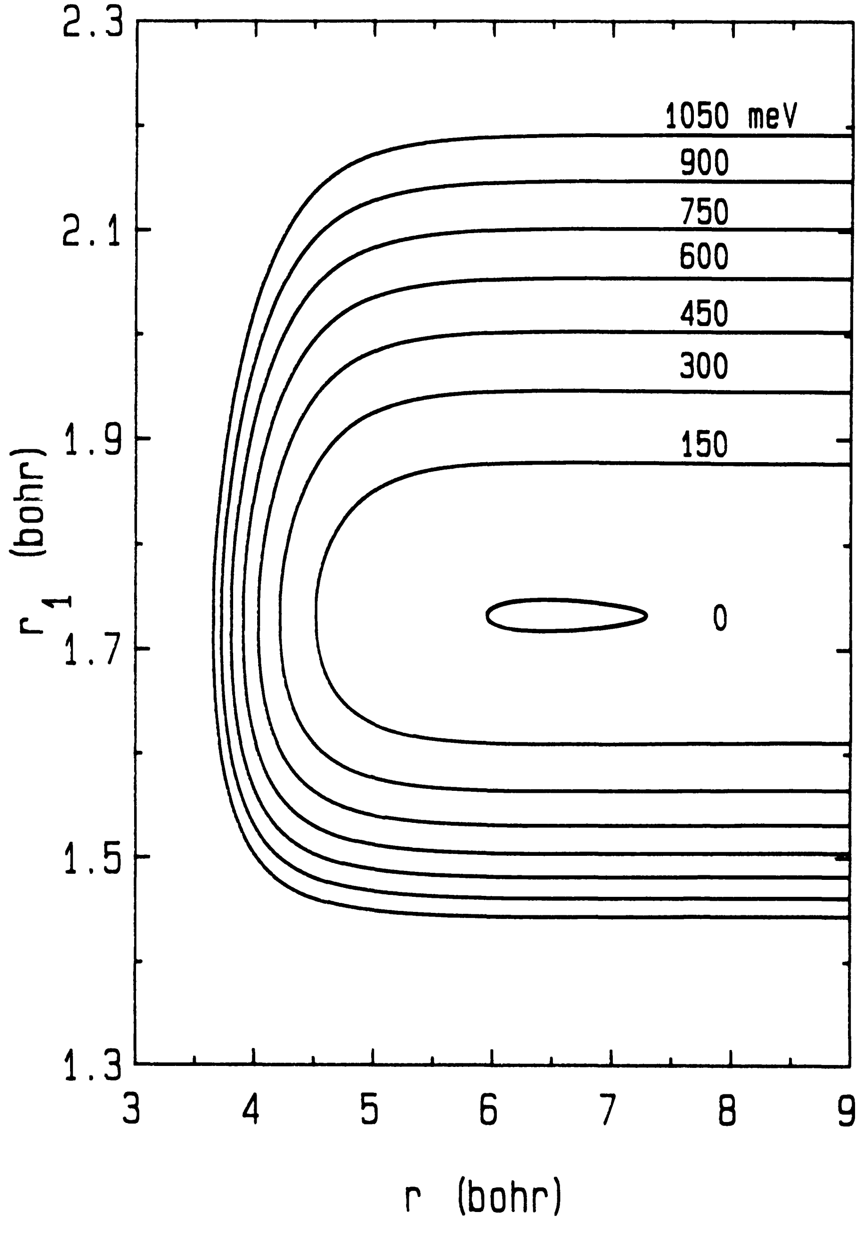

Figure 6. Same as Fig.4 except for the potential of ref.31 and contour levels run from 150 to 1050 meV.

where μ_e and θ_e are values corresponding to the equilibrium internuclear separation[15,16] and the $\beta_{\lambda_1\lambda_2\lambda}$ and $\gamma_{\lambda_1\lambda_2\lambda}$ are constants. The constants β_{112}, β_{123}, and γ_{123} were chosen to that $(d/dR_i)\exp(-\alpha_{\lambda_1\lambda_2\lambda}R_i)\,\mu(R_i)$ and $(d/dR_i)\exp(-\alpha_{\lambda_1\lambda_2\lambda}R_i)\,\theta(R_i)$ reproduce the experimental value of $d\mu/dR = 0.3169$ a.u.[15] and the theoretical value of $d\theta/dR_i = 1.601$ a.u.,[17] respectively. The parameters for the modification of the Alexander and DePristo interaction potential are given in Table I.

In order to illustrate the character of the potential energy surfaces resulting from the approximations (16)-(21) we present contour maps of the modified Alexander - DePristo (MAD) potential to see if the interaction potential is physically reasonable. Figure 3 shows a contour map for a linear hydrogen bonded arrangement (F-H...F-H), and Fig. 4 shows a similar map for a configuration with the F of molecule 2 oriented toward the center of mass of molecule 1, $\theta_1 = 90^0$, and $\theta_2 = 0^0$. (Because the center of mass of a monomer is closer to the F than to the bond midpoint, this resembles an "L"). The quantity plotted in the figures is the total potential defined as the sum of the interaction potential and the two diatomic potentials, with the zero of energy at 2HF at R_e. Figures 5 and 6 present comparable maps for the full Redmon-Binkley (RB) potential. These maps illustrate one inherent weakness of the approximation in (16), namely it does not seem to predict the character of the vibrational force equally validly for different orientations of approach. Thus the vibrational force of the MAD potential is opposite to that of the RB potential for the collinear case (Figs. 3 and 5), but the two potentials are very similar for the L case (Figs. 4 and 6). In the real world, perpendicular approaches are more likely than collinear ones, but the extent of rotational participation in V-V energy transfer may depend sensitively on the orientational dependence of the vibrational force.

Another set of attributes of the two potentials that can be compared are the energy and geometry of the van der Waals minima. For the MAD potential the van der Waals dimer is bound by 225 meV, with $r_e = 4.97\ a_0$, $R_{1e} = 1.749\ a_0$, and $R_{2e} = 1.743\ a_0$. For the RB potential the corresponding quantities are 274 meV, 5.04 a_0, 1.742 a_0, and 1.737 a_0. The agreement is reasonable, which gives us further confidence that the MAD potential is physically reasonable.

Although the vibrational forces of the MAD potential are not quantitatively accurate, it was thought to provide a reasonable and portable test potential for which converged V-V energy transfer calculations would be useful as a benchmark, and so we proceeded to carry out such calculations.

4. ROTATIONAL ENERGY TRANSFER IN COLLISIONS OF RIGID ROTATORS

In this section we discuss the results of our rigid rotator calculations. First, however, it is necessary to specify our choice of basis functions. We carried out a series of calculations for $J=0$ and $\eta=+1$ with basis functions chosen by one of two rules. The first rule was to include all channels coupled by total angular momentum, identical particle interchange symmetry, and parity restrictions which had $j_{sum} \leq j_{sum,max}$, where $j_{sum} = j_1 + j_2$, while the second rule used $\max(j_1,j_2) \leq j_{max}$.

Table II shows the convergence of $P^R_{j'_{sum}}$ for both schemes as a function of the maximum quantum number index and N, the number of channels. The convergence of two of the state-to-state transition probabilities is illustrated in Figs. 7-9. Table II and Figs. 7-9 show that convergence is slow, especially by the standard of previous calculations. [The largest basis set used previously for this potential by Alexander[8b] corresponds to $j_{max} = 3$ (their basis B4). For total angular momentum zero this basis yields $n=20$.]

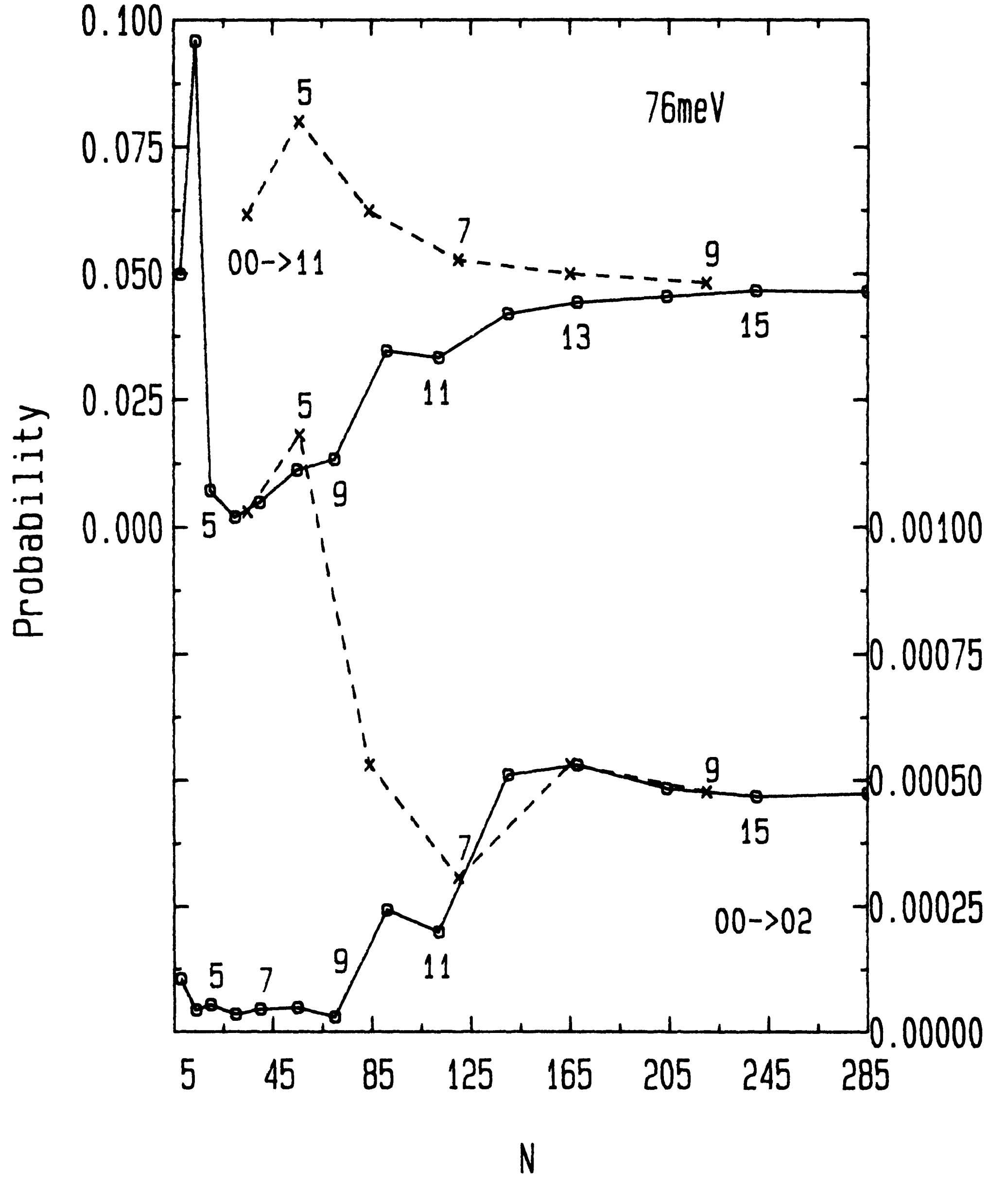

Figure 7. Convergence of the rotational energy transfer probabilities $P_{00\rightarrow 11}$ and $P_{00\rightarrow 02}$ in rigid rotator calculations for E_{rel} = 76 meV. Solid line and □, $j_{max,sum}$ rule; dashed line and x, j_{max} rule.

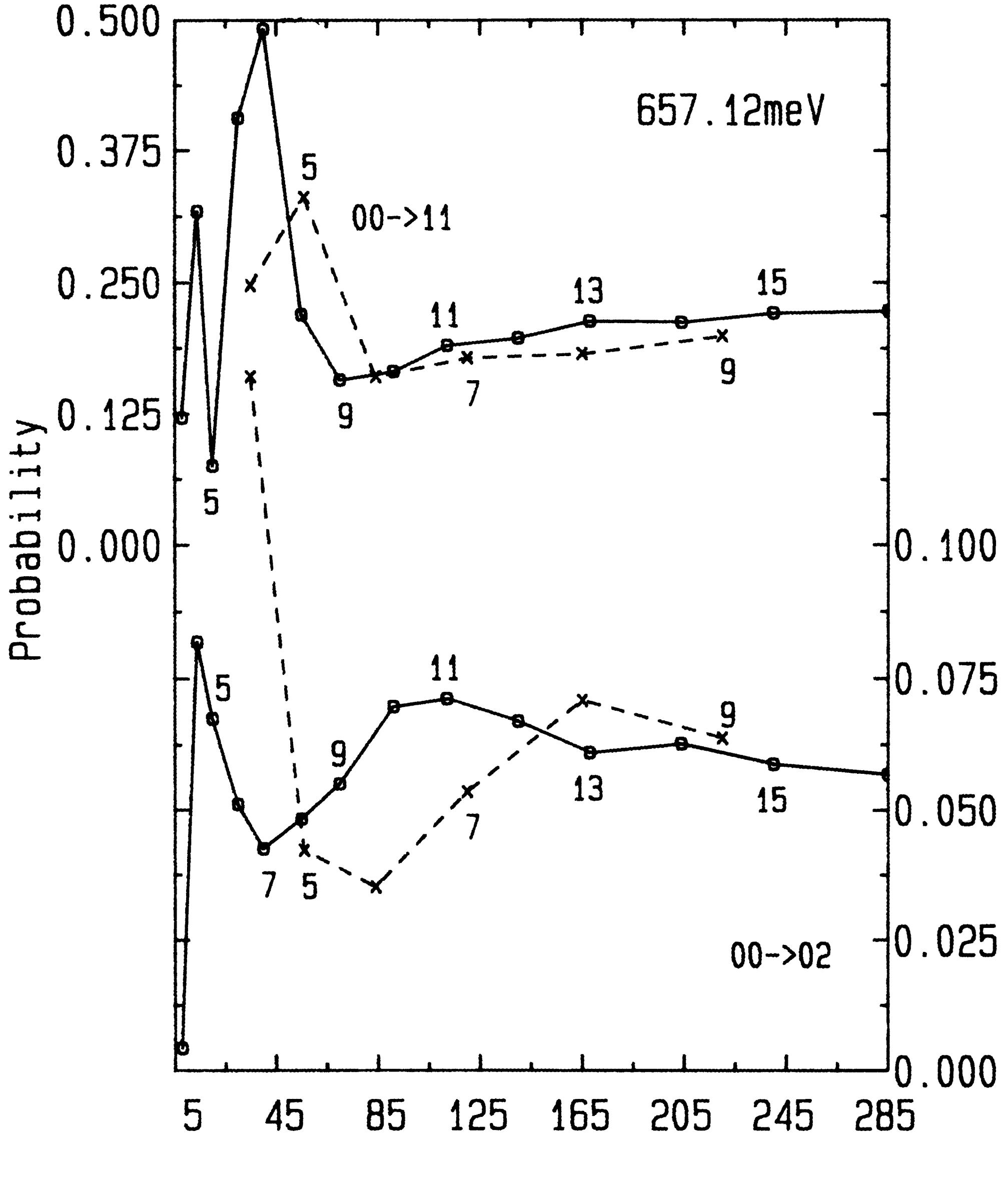

Figure 8. Same as Fig. 7 except for E_{rel} = 657 meV.

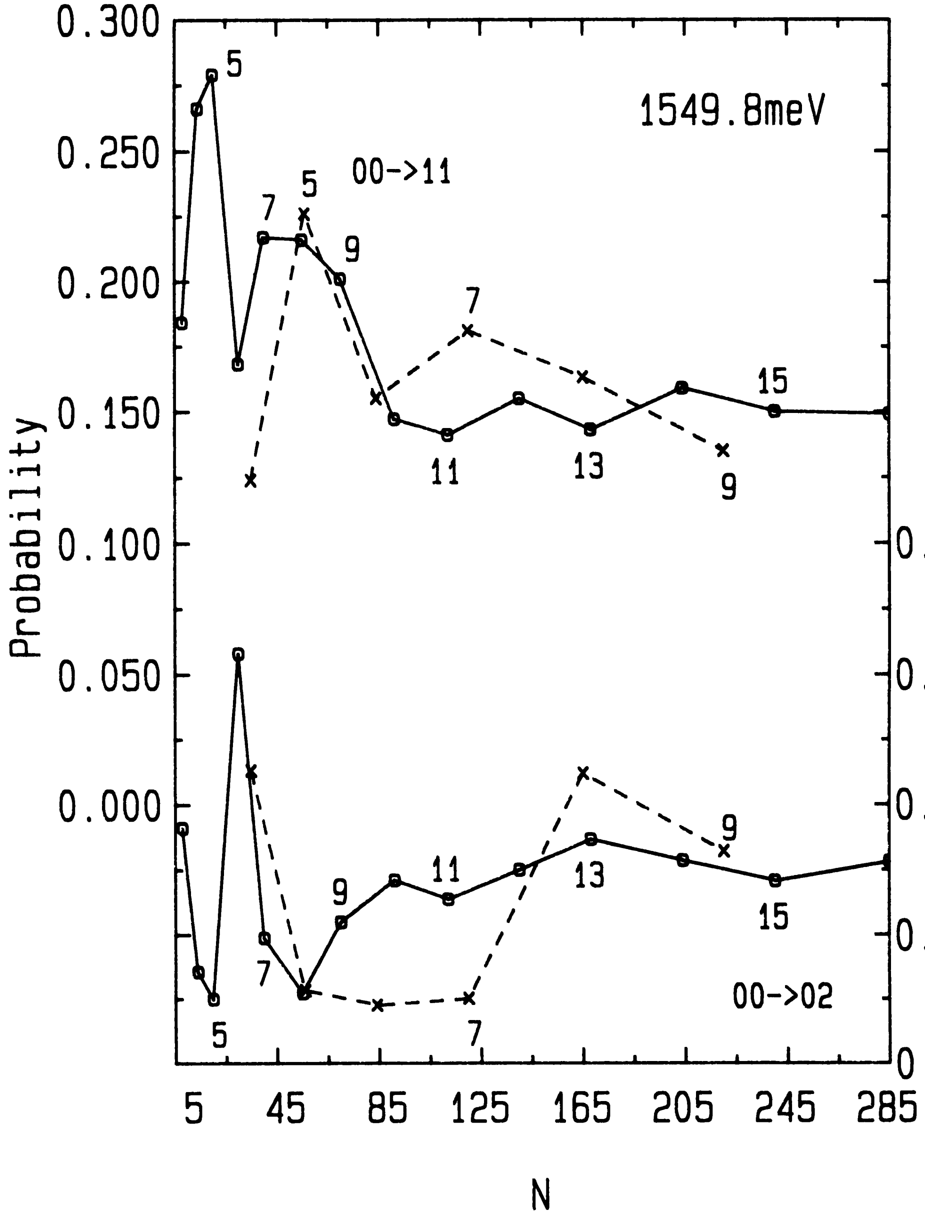

Figure 9. Same as Fig. 7 except for E_{rel} = 1549.8 meV.

Table II. Rotationally summed probabilities for rigid rotator calculations.

		E_{rel} = 0.076 eV			0.657 eV			1.550 eV		
index	N	P_0^R	P_1^R	P_2^R	P_0^R	P_1^R	P_2^R	P_0^R	P_1^R	P_2^R
$j_{sum,max}$										
3	8	0.948	0.002	0.050	0.819	0.045	0.125	0.634	0.025	0.275
4	14	0.900	0.002	0.096	0.479	0.044	0.399	0.361	0.037	0.302
5	20	0.987	0.001	0.007	0.753	0.038	0.082	0.434	0.105	0.304
6	30	0.991	0.002	0.002	0.358	0.042	0.457	0.050	0.141	0.326
7	40	0.992	0.001	0.005	0.252	0.020	0.534	0.037	0.031	0.266
8	55	0.980	0.002	0.011	0.367	0.043	0.267	0.038	0.019	0.243
9	70	0.977	0.003	0.013	0.473	0.018	0.212	0.104	0.011	0.256
10	91	0.950	0.002	0.035	0.468	0.018	0.234	0.120	0.001	0.218
11	112	0.951	0.003	0.033	0.423	0.024	0.261	0.104	0.004	0.205
12	140	0.940	0.003	0.042	0.430	0.021	0.264	0.106	0.009	0.230
13	168	0.937	0.003	0.045	0.417	0.017	0.274	0.097	0.009	0.230
14	204	0.935	0.004	0.046	0.420	0.019	0.274	0.101	0.007	0.238
15	240	0.934	0.004	0.047	0.414	0.017	0.280	0.106	0.011	0.221
16	285	0.934	0.004	0.047	0.414	0.019	0.280	0.102	0.011	0.227
j_{max}										
4	35	0.924	0.001	0.063	0.372	0.018	0.380	0.075	0.026	0.236
5	56	0.891	0.001	0.081	0.368	0.007	0.373	0.098	0.025	0.254
6	84	0.912	0.003	0.063	0.514	0.008	0.195	0.115	0.023	0.238
7	120	0.925	0.003	0.053	0.468	0.018	0.232	0.139	0.011	0.206
8	165	0.929	0.003	0.050	0.435	0.021	0.252	0.076	0.004	0.276
9	220	0.932	0.004	0.049	0.426	0.020	0.262	0.100	0.025	0.217

Table II shows that the present calculations are converged for both basis-set selection schemes at all three energies, with convergence being achieved typically at N $\cong$ 200. At the lowest energy the two basis-set selection schemes converge at about the same rate, but at higher energies the $j_{sum,max}$ rule leads to convergence at a smaller N and hence is more efficient.

Table III gives a set of converged state-to-state transition probabilities for the rigid rotator calculations at each of the three energies. The table shows that the elastic scattering probability decreases from 93% to 10% as the energy is increased. The first-order dipole-dipole transition, (00) $\rightarrow$ (11), has the largest inelastic transition at each energy; these probabilities are in the range 5-22%. The (00) $\rightarrow$ (02), (00) $\rightarrow$ (22), and (00) $\rightarrow$ (32) transitions have probabilities in the 2-8% range at the two highest energies. None of the other transitions has a probability in excess of 5% at any of the energies.

5. V-V ENERGY TRANSFER IN CALCULATIONS WITH NO COORDINATE OR DIMENSIONAL RESTRICTIONS AND NO DYNAMICAL APPROXIMATIONS

For the vibrating molecule calculations we again considered $J=0$ and $\eta=+1$, and chose rotational channels by the j_{sum} rule and vibrational channels using an analogous v_{sum} rule which was previously shown to be efficient for reduced dimensionality calculations.[11d] In particular, for each v_{sum}, where $v_{sum} \equiv v_1+v_2$, we included all channels allowed by total angular momentum, interchange symmetry, and parity restrictions which had $j_{sum} \leq j_{sum,max}$ where $j_{sum,max}$ depends on v_{sum}. We use the same value of $j_{sum,max}$ for $v_{sum} \leq 2$, and smaller values for larger v_{sum}. The compositions of the basis sets are summarized in Table IV. Results were calculated for the process HF($v_1=1, j_1=0$) + HF($v_2=1, j_2=0$) $\rightarrow HF(v'_1=2, j'_1) + HF(v'_2=0, j'_2)$. Table V summarizes the convergence of the transition probabilities with respect to increasing the size of the basis set. First consider the calculations with $j_{sum,max}=8$, 9, 10 and 11 for $v_{sum} \leq 2$ and $j_{sum,max}=6$, 7, 8 and 9 for $v_{sum}=3$, excluding channels with $v_{sum}>3$. These calculations involved 400, 530, 694, and 880 channels, respectively.

Table III. Rigid rotator transition probabilities from calculations with $j_{sum,max} = 16$.[a]

$j_1j_2 \rightarrow j_1j_2$	76[b]	657[b]	1550[b]
0,0 → 0,0	0.934	0.414	0.102
0,0 → 1,0	0.004	0.019	0.011
0,0 → 1,1	0.046	0.223	0.149
0,0 → 2,0	<0.001	0.057	0.08
0,0 → 2,1	0.004	0.019	0.007
0,0 → 3,0	0.001	0.002	0.003
0,0 → 2,2	0.010	0.031	0.076
0,0 → 3,1	<0.001	0.019	0.003
0,0 → 3,2	<0.001	0.077	0.022
0,0 → 4,0	<0.001	0.007	0.005
0,0 → 4,1	<0.001	0.020	0.012
0,0 → 3,3	<0.001	0.027	0.020
0,0 → 4,2	<0.001	0.009	0.03
0,0 → 5,0	c	0.001	0.005
0,0 → 5,1	c	0.010	0.008
0,0 → 4,3	c	0.018	0.04
0,0 → 5,2	c	0.008	0.04
0,0 → 4,4	c	0.005	0.050
0,0 → 6,0	c	<0.001	0.004
0,0 → 5,3	c	0.004	0.011
0,0 → 6,1	c	0.001	0.005
0,0 → 6,2	c	0.001	0.004
0,0 → 5,4	c	0.006	0.027
0,0 → 6,3	c	0.002	0.028
0,0 → 7,0	c	<0.001	0.001
0,0 → 7,1	c	0.001	0.005
0,0 → 5,5	c	0.006	0.05
0,0 → 7,2	c	0.001	0.004
0,0 → 6,4	c	0.002	0.02
0,0 → 7,3	c	0.001	0.007
0,0 → 8,0	c	<0.001	0.001
0,0 → 6,5	c	0.002	0.019
0,0 → 8,1	c	<0.001	0.002
0,0 → 7,4	c	0.001	0.02
0,0 → 8,2	c	<0.001	0.003
0,0 → 8,3	c	<0.001	0.003
0,0 → 6,6	c	0.003	0.04
0,0 → 7,5	c	<0.001	0.011
0,0 → 9,1	c	<0.001	0.001
0,0 → 8,4	c	<0.001	0.003
0,0 → 7,6	c	0.001	0.008
0,0 → 9,3	c	<0.001	0.002
0,0 → 8,5	c	<0.001	0.010
0,0 → 9,4	c	<0.001	0.001
0,0 → 7,7	c	0.001	0.02
0,0 → 8,6	c	<0.001	0.01
0,0 → 9,5	c	<0.001	0.002
0,0 → 10,3	c	<0.001	0.001
0,0 → 8,7	c	<0.001	0.003
0,0 → 10,4	c	<0.001	0.002
0,0 → 9,6	c	<0.001	<0.01
0,0 → 10,5	c	<0.001	<0.01
0,0 → 8,8	c	<0.001	<0.01
0,0 → 9,7	c	<0.001	<0.01

[a] Only transitions with probabilities greater than 0.0005 are included. Transitions are listed in order of increasing excitation energy.
[b] E_{rel} in meV.
[c] Closed channels.

Comparison of the 880-channel calculations to the 694-channel calculations show convergence of P^{VV} to better than 0.7% at all the energies considered here. In Table V, 13 of the probabilities for the 694-channel and 880-channel calculations agree to within 0.004; for the other two probabilities the discrepancies are 0.006 and 0.011. These calculations demonstrate that the rotational basis of basis set 3 is converged for $v_{sum} \leq 3$. It is interesting to note that the convergence is attained at

$j_{sum,max} = 10$ for these V-V calculations, whereas the rigid rotator calculations discussed above required $j_{sum,max}$ values of about 12-14 in order to converge the results. Next we consider calculations that employ the converged rotational basis of basis set 3 for $v_{sum} \leq 3$ and also include channels with $v_{sum} > 3$ in order to converge the vibrational basis. Two additional basis sets were considered, one with $j_{sum,max} = 6$ for $v_{sum} = 4$, and no $v_{sum} \geq 5$ states, which yields 824 channels, and another in which two additional j_{sum} states are included for $v_{sum} = 4$ and 5 ($j_{sum} = 7$ and 8 for $v_{sum} = 4$ and $j_{sum} = 0$ and 1 for $v_{sum} = 5$), yielding our largest basis of 948 channels. Values of $P^{VV}_{j_{sum}}$ and P^{VV} calculated from these additional basis sets are also given in Table V. Comparison of the 824-channel calculations to the 948-channel calculations shows convergence of P^{VV} to better than 1.3% at all energies considered here. If we also consider $P^{VV}_{j_{sum}}$, we see that for 10 of the probabilities in Table V, the 824-channel and 948-channel calculations differ by 0.002 or less and for the other probabilities the discrepancies are between 0.006 and 0.019. This convergence is considered acceptable. Furthermore since the comparison of bases 3 and 4 shows that $j_{sum,max} = 8$ is sufficient to converge the $v_{sum} = 3$ channels, it should also be sufficient for $v_{sum} = 4$.

Table IV. Basis sets for V-V energy transfer calculations.

	$j_{sum,max}$				
Basis	$v_{sum} \leq 2$	$v_{sum} = 3$	$v_{sum} = 4$	$v_{sum} = 5$	N
1	8	6	...	...	400
2	9	7	...	...	530
3	10	8	...	...	694
4	11	9	...	...	880
5	10	8	6	...	824
6	10	8	8	1	948

The state-to-state transition probabilities $P_{1010 \rightarrow 2j'_1 0j'_2}$ for the four largest basis sets are given in Table VI. As for the results in Table V, if basis 3 agrees with basis 4 and basis 5 with basis 6, then basis 6 should be considered converged. For the most part probabilities with both $j'_1 \leq 1$ and $j'_2 \leq 1$ are well converged.

The most striking aspect of the rotational energy distributions in Table VI is the peaking of the V-V energy transfer cross sections at a final state that is nonresonant by 21 meV and the fact that transition probabilities to other channels with smaller translational energy defects are smaller by factors of thirty or more. It would be interesting to learn the sensitivity of this rotational distribution to the nature of the potential.

6. COMPUTATIONAL CONSIDERATIONS

The close coupling calculations were performed using a code vectorized separately for the Cray-1 and for the Control Data Corporation Cyber 205 pipelined vector

Table V. Partially summed transition probabilities $P^{VV}_{j'\mathrm{sum}}$.

E_{rel} (meV)	open channels	Basis	N	open-channel basis functions	P^{VV}_0	P^{VV}_1	P^{VV}_2	P^{VV}_3	P^{VV}
2.455	1548	1	400	162	0.698	0.045	0.058	0.022	0.823
		2	530	207	0.791	0.032	0.017	0.003	0.843
		3	694	264	0.826	0.031	0.003	0.002	0.863
		4	880	327	0.827	0.031	0.005	0.006	0.868
		5	824	264	0.875	0.034	0.003	0.002	0.914
		6	948	264	0.881	0.035	0.003	0.002	0.921
29	1671	1	400	184	0.873	0.036	0.009	0.004	0.923
		2	530	229	0.904	0.036	0.001	0.005	0.948
		3	694	286	0.906	0.038	0.009	0.005	0.960
		4	880	349	0.902	0.039	0.011	0.008	0.962
		5	824	286	0.925	0.037	0.012	0.005	0.980
		6	948	286	0.924	0.037	0.014	0.005	0.981
76	1888	1	400	239	0.901	0.048	0.013	0.006	0.970
		2	530	284	0.854	0.048	0.045	0.006	0.955
		3	694	341	0.815	0.045	0.065	0.008	0.937
		4	880	404	0.804	0.046	0.068	0.008	0.933
		5	824	341	0.760	0.033	0.085	0.009	0.891
		6	948	341	0.741	0.031	0.093	0.009	0.880

Table VI. State-to-state transition probabilities $P_{1010\rightarrow 2j'_1 0 j'_2}$.

j'_1	j'_2	ΔE[a]	Basis	2.455[b]	29[b]	76[b]
0	0	21	3	0.83	0.91	0.81
			4	0.83	0.90	0.80
			5	0.87	0.92	0.76
			6	0.88	0.92	0.74
1	0	17	3	0.026	0.030	0.022
			4	0.024	0.032	0.023
			5	0.028	0.026	0.015
			6	0.028	0.026	0.014
0	1	16	3	0.005	0.009	0.023
			4	0.006	0.006	0.023
			5	0.006	0.011	0.017
			6	0.006	0.011	0.016
1	1	12	3	0.0008	0.008	0.064
			4	0.0007	0.010	0.067
			5	0.0005	0.011	0.084
			6	0.0006	0.0125	0.092
2	0	7	3	0.002	0.0009	0.0006
			4	0.004	0.0007	0.0007
			5	0.002	0.0009	0.0005
			6	0.002	0.0009	0.0006
0	2	6	3	0.0001	0.0002	0.0003
			4	0.0005	0.0004	0.0004
			5	0.0002	0.0002	0.0003
			6	0.0002	0.0002	0.0003

[a] Translational energy defect in meV (21 meV = 172 cm^{-1}).

[b] E_{rel} in meV (76 meV = 613 cm^{-1}).

computers. On both machines we used single precision (64-bit) arithmetic for all calculations. The rigid-rotator production runs were carried out on the Cray-1. The largest rigid-rotator calculation, involving 285 channels, required approximately 1.7 hours of CPU time. Relative to unvectorized calculations on a Digital Equipment Corporation VAX 11/780 with scalar floating point accelerator, we achieved enhancements in computation speed of a factor of about 540 for the whole computation for this calculation. The vibrating rotator production runs were carried out on the two-pipeline Cyber 205, with the largest calculation, with 948 channels, requiring about 17.5 hours of computer time. In this case we estimate a speed enhancement over the VAX of about a factor of 1700, so that the calculation would have required 30,000 hours (1300 days) to execute on the VAX. It is instructive to give more details on how we arrive at the estimated speed enhancement and also to estimate how many millions of floating point operations per second (MFLOPS) we are obtaining. In our initial tests,[4] we compared times on our VAX to the Cyber 205 and the Cray-1 for a test case having $N=101$. Additional tests of the matrix routines on the VAX indicate that for $N>50$, the time for a calculation on the VAX scales with N as $N^{3.0}$. Using this factor to scale the execution time for a full calculation with $N=101$, we estimate for a three-energy calculation taking 297 steps with $N=948$ a time of 3.0×10^4 hours. We estimate speed in MFLOPS as follows. Most of time (about 95% for the $N=948$ run and about 92-94% for other N for this potential) used for our calculations is spent on matrix manipulations; in particular for the first energy in each of 297 steps we must diagonalize an NxN matrix, multiply three NxN matrices, and solve two sets of linear equations for N different unknown vectors, while for second and third energies in each of 297 steps we must perform two NxN matrix multiplications and solve a set of linear equations for N different unknown vectors. The number of operations to perform a matrix multiplication is $2N^3$, the number to perform a linear equation solution is $8N^3/3$ ($2N^3/3$ for LU decomposition,[18a] N^2 per vector for forward elimination,[18b] and N^2 per vector for back substitution[18b]), and we will take the number to perform a matrix diagonalization[18c] to be $10N^3$ (which may be a slight, 5-10%, underestimate for our problem), so that the number of operations will be about $28N^3$ per step, which, when combined with our actual execution time for the $N=948$ run, yields a lower bound for the operation rate of 112 MFLOPS. This compares favorably to the maximum theoretically obtainable value of 200 MFLOPS on a two-pipeline machine running in single precision performing linked triads.[19a]

The speed enhancements of the vector pipelined supercomputers relative to the scalar VAX would increase with increasing N since the number of scalar operations increases as N^3 whereas we found that for the largest N values considered here, the CPU time was increasing proportional to $N^{2.7}$ on the Cray-1 at $N=200$-285, and proportional to $N^{2.6}$ on the Cyber 205, at $N=400$-948.

For $N=285$ we estimate that the ratio of the CPU time for the Cyber 205 to CPU time for the Cray-1 is about 2, which is the same as we observed previously[4] for $N=101$. We note, however, that the Cray-1 version of our code used here is more efficient than the one described previously[4] because we have implemented the technique of FORTRAN DO-loop unrolling[20,21] for matrix multiplications and linear equation solutions. This increases the rate of execution on the Cray-1 because it decreases the number of unnecessary transfers between main memory and vector registers. Such transfers can become a bottleneck because the Cray-1 has only one data bus between main memory and the vector registers. For the new version of the Cray-1 code, the Cray-1 and Cyber 205 CPU times are almost the same for $N=101$; however, the Cyber 205 continues to become more efficient as N is further increased whereas the Cray-1 is closer to its asymptotic speed. Further details of our experience with loop unrolling on the Cray-1 are given in an appendix to this chapter. For eigenanalysis steps in the R matrix propagation calculations we use the EISPACK[22] routines of the Cray SCILIB[23] library. On the Cyber 205 we use the

MAGEV library[24] for both eigenanalysis and linear algebra. Loop unrolling is less useful on the Cyber 205 because it is a memory-to-memory machine, i.e., there are no vector registers, and thus the same data transfer bottleneck does not exist.

In separate calculations we estimated that for R matrix propagation calculations, the Cyber 205 and Cray X-MP/48 machine (using a single processor) lead to comparable execution times for $N = 101$. For these tests the linear algebra on the Cray X-MP/48 was carried out by LINPACK[25] routines with the BLAS[26] replaced by their FORTRAN equivalents, as discussed for the Cray-1 in Ref. 4. Since the Cray X-MP/48 has three data buses between main memory and vector registers, we anticipate that the effect of DO-loop unrolling would not be large.

7. CONCLUSIONS

We have shown that it is possible to converge three-dimensional calculations of V-V energy transfer for diatom-diatom collisions, although a very large number of channels are required. Although the present calculations, being restricted to zero total angular momentum, cannot be compared to experiment, they provide benchmarks for testing approximate dynamical theories that can more easily be applied to all total angular momenta.

In future work we plan to study the dependence of the results on the nature of the intermolecular potential.

8. ACKNOWLEDGEMENTS

We are grateful to R. W. Numrich and T. F. Walsh for administrative assistance that facilitated these calculations, to J. J. Dongarra for sharing his improved code for the decomposition step of linear equations solution, and to David Cochrane for comments on the manuscript. This work was supported in part by the National Science Foundation under grant no. CHE83-17944, by the University of Minnesota Supercomputer Institute under grant SCIH001, by the Control Data Corporation, and by Cray Research, Inc.

APPENDIX. Loop Unrolling

As mentioned in Sect. 6, we found significant enhancements in performance on the Cray-1 computer when we incorporated the loop unrolling techniques suggested by Dongarra and Eisenstat.[20,21] Dongarra has also found significant enhancements on a variety of advanced scientific computers.[27] Since these techniques offer significant time savings for widely used matrix algebraic procedures, they should be of general interest, and in this appendix we report on further details of our experience with them. We will compare times for various levels of loop unrolling to times for optimized library routines. On the Cray-1, we will compare to routines from the SCILIB[23] library; in particular, we give reference times for the matrix multiplier MXM, the linear equation solver MINV, which uses Gaussian elimination, and the routines SGEFA and SGESL, which are SCILIB versions of LINPACK[25] codes, which perform the LU decomposition of a matrix and use the LU decomposition of a matrix to solve linear equations, respectively. On the CYBER 205 we will compare to the MAGEV library[24] matrix multiplier MXMPY. The University of Minnesota Supercomputer Institute (UMSI) Cray-1 has COS 1.12 as its operating system and has two versions of the FORTRAN compiler, CFT 1.11 and CFT 1.13. We will discuss execution times for codes compiled by both compilers in what follows, and we will see that the latter compiler often produces faster code;

however, for the production runs discussed in Sect. 6 above we used version 1.11 of the compiler because version 1.13 does not interfere properly with version 1.12 of the operating system in our production context. The UMSI Cyber 205 has the VSOS operating system, and we use the FORTRAN 2.1.6 compiler for the timings in this section.

It should be noted that the Cray-1 is able to hold a maximum of 64 elements in one vector register. Thus all vector operations occur with vectors of length 64 or less. When FORTRAN programs have vectors longer than 64 elements, the compiler automatically breaks the longer vector into shorter vectors containing no more than 64 elements, and it operates on each of the shorter vectors separately. The inclusion of vector lengths greater than 64 in the discussion below would complicate the presentation without adding any important new ideas. Thus, we will discuss the operation of the Cray computers as if the vector registers were infinitely long or all vectors were of length 64 or less, but we give timing examples involving longer vectors.

A. Matrix multiplication

The basic idea of loop unrolling can be illustrated clearly for the operation of matrix multiplication. Consider the following FORTRAN code which multiplies an NxM matrix $\underset{\sim}{A}$ times an MxP matrix $\underset{\sim}{B}$ to produce an NxP matrix $\underset{\sim}{C}$:

```
  DO 1 I1=1,P
  DO 1 I2=1,N
1    C(I2,I1)=0                                              (A1)
  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).
```

The Cray-1 compiler will vectorize the innermost DO-loop over the I variable by giving the instructions: (i) load the vector with N elements starting with A(1,K) from main memory into vector register V_0 , (ii) multiply the elements of the vector register V_0 times the scalar B(K,J) and store the results into vector register V_1, (iii) load the vector with N elements starting with C(1,J) from main memory into vector register V_2, (iv) add the elements of the vector register V_2 to the elements of the vector register V_1 and store the result in vector register V_3, and (v) store the elements of the vector register V_3 back into main memory. The Cray-1 is able to "chain" many of these operations together so that they form one continuous pipeline.[19b] For two operations to chain, an instruction must be issued at the "chain time slot," which is the functional unit time plus two clock periods. When this is done, the result from one functional unit is immediately sent to another functional unit as an operand. Chaining is essentially equivalent to connecting two functional units together as one continuous unit. For algorithm (A1), it is possible for step (i) to be chained with step (ii) and step (iii) to be chained with step (iv). The CPU time required to complete a vectorized operation on a vector of length N is $N\tau + T_{\text{overhead}}$, where τ is a cycle time and T_{overhead} is an overhead time which is independent of N. For two vectorized operations that are chained, the CPU time is not $2(N\tau + T_{\text{overhead}})$ but rather $N\tau + T'_{\text{overhead}}$. Normally τ is the clock period, but if memory bank conflicts occur τ will be larger. Thus if N is large enough so that T'_{overhead} can be neglected and no memory bank conflicts occur, the time for one pass through steps (i)-(v) of algorithm (A1) will be about 3N clock periods (whereas without chaining it would be about 5N clock periods). Unfortunately, however, the

FORTRAN compiler is not always able to properly recognize potential chaining opportunities. It appears that the CFT 1.11 compiler is not able to chain steps (iii) and (iv) whereas the CFT 1.13 compiler schedules instructions so that steps (iii) and (iv) do chain. It turns out, for this particular case, that comparable savings can be achieved by placing parentheses around C(I,J) so that it is loaded first and so that the loading of A(I,K) is chained to both the multiplication and the addition. However, even more savings can be achieved by loop unrolling as discussed next.

The key to further efficiency in accomplishing the objectives of algorithm (A1) is to observe that the vector which is stored back into main memory in step (v) is the same vector which will be loaded from memory in step (ii) of the next pass (with K incremented) through steps (i)-(v). Ideally we would like to have the compiler eliminate this unnecessary step and maximize chaining by having it generate the instructions: (i′) load the vector with N elements starting with C(1,J) from main memory into vector register V_2; (ii′) load the vector with N elements starting with A(1,K) from main memory into vector register V_0: (iii′) multiply the elements of the vector register V_0 times the scalar B(K,J) and store the results into vector register V_1; (iv′) add the elements of vector register V_1 to the elements in vector register V_α and store the results into vector register V_β; then repeat steps (ii′)-(iv′) M times, successively increasing K so that B(K,J) is different each time and using $\alpha=2$, $\beta=3$ for even K and $\alpha=3$, $\beta=2$ for odd K; and (v′) store the elements of vector register V_β into main memory. In this scheme steps (ii′)-(iv′) can all be chained together so that the time is about N clock periods, or about 1/3 of the previous example. One way one could attempt to cause the FORTRAN compiler to recognize this additional chaining possibility would be to replace the J, K, and I DO-loops of algorithm (A1) by

```
   DO 2 J=1,P                                                    (A2)
   DO 2 I=1,N
2     C(I,J)=(...(C(I,J))+A(I,1)*B(1,J))+...+A(I,M)*B(M,J).
```

Table VII. Times in central processor seconds to multiply two square matrices of order 300.

	execution time (CPU seconds)	
unrolling depth	Cray-1	Cyber 205
1	1.63 (1.31)[a]	0.51
2	1.03 (0.87)	0.51
4	0.75 (0.71)	0.51
8	0.63 (0.56)	0.52
16	0.58 (0.52)	0.51
32	0.55 (0.53)	0.50
MXM	0.37 (0.37)	...
MXMPY	...	0.43

[a] Times in parenthesis are for the CFT 1.13 compiler, while the other Cray-1 times are for the CFT 1.11 compiler.

This approach suffers from the drawback that it is necessary to have a separate program for each value of M. A more general yet still partially optimized code would be:

```
   DO 2 J=1,P                                                    (A3)
   DO 2 K=1,M,NU
   DO 2 I=1,N
2     C(I,J)=(...(C(I,J))+A(I,K)*B(K,J))+...+A(I,K+NU-1)*B(K+NU-1,J)
```

where it is assumed here that M is an integral multiple of NU. The procedure employed in algorithm (A3) is called unrolling the K loop to a depth of NU. Table VII contains some CPU times for various depths of unrolling the K loop in the example above for the two Cray-1 compilers mentioned above. It also contains CPU times for performing the same matrix multiplications with subprogram MXM of the Cray SCILIB library. (The last column of this table will be explained in the next paragraph.) For the case in the table the largest speedup obtained with algorithm (A3) and for the CFT 1.11 compiler is observed for unrolling to a depth of 32, which is the largest amount of unrolling we considered, while for CFT 1.13 the largest speedup occurs when unrolling to a depth of 16. It can be seen from the table that even for CFT 1.11 unrolling to depths greater than 32 will not significantly increase the speedup factor. The code produced by CFT 1.13 runs consistently 4-20% times faster than that produced by CFT 1.11. The speedup factor of about three for the programs with loops unrolled as compared to the original program is in accordance with the analysis above. It is interesting to note that the MXM routine is about 1.4 times faster than the fastest FORTRAN program. It is, however, unfortunate that MXM is coded in a restrictive manner that requires that the orders, i.e., N, M, and P, of the matrices involved must correspond to their first dimension. This makes MXM unusable for our application because during the course of our calculation we change the order of the matrices which we need to multiply. This arises because the adiabatic basis set size is reduced at large r as discussed at the end of Sect. 2. Another reason we do not use MXM is that our application requires multiplications of the form

$$\underset{\sim}{C} = \underset{\sim}{A}^T \underset{\sim}{B} \tag{22}$$

with the elements of $\underset{\sim}{C}$ stored in the same space as the elements of $\underset{\sim}{B}$. With our own FORTRAN code we can easily implement multiplications of the form of (22) without explicitly constructing $\underset{\sim}{A}^T$. In this manner we avoid wasting the time necessary to form $\underset{\sim}{A}^T$ as well as the extra memory required to store $\underset{\sim}{A}^T$. For these two reasons we do not use MXM, and the DO-loop unrolling procedures described above are indeed very helpful. In particular the version of our code which we used to perform the rigid rotator calculations used matrix multiplication routines unrolled to a depth of 8.

Table VII also shows CPU times for algorithm (A3) as run on the two-pipeline Cyber 205 and compares them to times obtained with subprogram MXMPY from the Cyber 205 MAGEV library. On the Cyber 205 the effect of DO-loop unrolling is much smaller than one the Cray-1; the difference in performance with various values of NU is on the order of two percent, which is understandable because the memory-to-register transfers discussed above are not required on this machine. (The Cyber 205 feeds the vector pipeline directly from memory without requiring a prior transfer into vector registers.)

B. Solution of linear equations

We now turn to the problem of solving linear equations. The problem is written in the form

$$\underset{\sim}{A}\vec{x} = \vec{b} \tag{23}$$

where $\underset{\sim}{A}$ is a nonsingular, nonsymmetric NxN matrix and $\vec{b}$ and $\vec{x}$ are column vectors; $\underset{\sim}{A}$ and $\vec{b}$ are known, and the problem is to solve for $\vec{x}$. The usual method involves first decomposing $\underset{\sim}{A}$ into its so called LU form, i.e., determining matrices $\underset{\sim}{L}$ and $\underset{\sim}{U}$ such that

$$\underset{\sim}{U} = \underset{\sim}{L}^{-1}\underset{\sim}{A} \tag{24}$$

where $\underset{\sim}{U}$ has zeros below its diagonal (see e.g., Ref. 18d or 28 for explicit definitions of $\underset{\sim}{L}$ and $\underset{\sim}{U}$). Usually the matrices $\underset{\sim}{L}$ and $\underset{\sim}{L}^{-1}$ are not explicitly formed, however information is retained to allow the easy determination of products of the form of Eq. (24). The solution continues by forming $\underset{\sim}{L}^{-1}\vec{b}$, which is called the forward elimination step, followed by

$$\vec{x} = \underset{\sim}{U}^{-1}\underset{\sim}{L}^{-1}\vec{b} \ . \tag{25}$$

which is called the back substitution step. The number of arithmetic operations to perform the LU decomposition is $0(N^3)$ and the number of arithmetic operations to compute $\vec{x}$ (the forward elimination and back substitution steps) is $0(N^2)$. Thus in cases where there is only one vector $\vec{b}$ most of the time will be spent in the decomposition step, while in cases where there are N vectors $\vec{b}$, about equal time will be spent on the decomposition and on the forward elimination and back substitution steps.

1. Decomposition step

Dongarra[20] has pointed out that it is possible to re-organize the usual sequence of operations in the decomposition step in order to cast it into a form in which FORTRAN DO-loop unrolling as discussed above can be used to improve the performance. An example of a code re-organized this way is (this code is base on one given to us by Dongarra, but differs in that subroutine calls to matrix kernels are replaced by in-line code)

```
      DO 1 J=1,N                                        (A4)
       IF (J.EQ.1) GO TO 2
       DO 3 K=1, J-1
       DO 3 L=1, N-J+1
3         A(J+L-1,J)+A(J+L-1,J)+A(K,J)*A(J+L-1,K)
2      T=ABS(A(J,J))
       K=J
       IF (J.EQ.N) GO TO 4
       DO 5 I=J+1,N
          IF (ABS(A(I,J)).LE.T) GO TO 5
          T=ABS(A(I,J))
          K=I
```

```
    5     CONTINUE
    4  IPVT(J)=K
       IF (T.EQ.0) STOP
       IF (K.EQ.J) GO TO 6
       DO 7 M=1,N
          T=A(J,M)
          A(J,M)=A(K,M)
    7     A(K,M)=T
    6  A(J,J)=1./A(J,J)
       IF (J.EQ.N) GO TO 1
       IF (J.EQ.1) GO TO 8
       DO 9 K1=1,J-1
       DO 9 L1=1, N-J
    9     A(J,J+L1)=A(J,J+L1)+A(J,K1)*A(K1,J+L1)
    8  T=-A(J,J)
       DO 10 I1=J+1,N
   10     A(J,I1)=T*A(J,I1)
    1  CONTINUE
```

In this example the K loop and the K1 loop would be unrolled, i.e., the first loop would become, assuming NU divides J-1.

```
       DO 3 K=1,J-1,NU                                          (A5)
       DO 3 L=1,N-J+1
    3     A(J+L-1,J)=(...(A(J+L-1,J))+A(K,J)*A(J+L-1,K))+...
    $     +A(K+NU-1,J)*A(J+L-1,K+NU-1)
```

and similarly for the K1 loop. Of course in an actual code provision would have to be made for the case when NU is not an integral divisor of J-1. It should be noted that the L1 loop causes the second index of the array A to vary fastest. This causes

Table VIII. Times in central processor seconds on the Cray-1 to solve linear equations of order 300 in which there are 300 right hand side vectors.

unrolling depth	time for decomposition step	time for forward elimination and back substitution steps	total time
1	0.62 (0.56)[a]	1.38 (1.30)	2.00 (1.87)
2	0.46 (0.39)	1.05 (0.88)	1.51 (1.27)
4	0.35 (0.33)	0.77 (0.72)	1.11 (1.05)
8	0.30 (0.27)	0.64 (0.57)	0.94 (0.84)
16	0.28 (0.25)	0.57 (0.51)	0.85 (0.75)
32	0.26 (0.24)	0.54 (0.53)	0.80 (0.78)
LINPACK[b]	0.52 (0.49)	1.58 (1.49)	2.10 (1.99)
MINV[c]	...	...	1.78 (1.78)

[a] Times in parentheses are for the CFT 1.13 compiler while the other times are for the CFT 1.11 compiler.

[b] Using SCILIB routines SGEFA for decomposition step and SGESL for substitution step.

[c] SCILIB routine.

the memory to be accessed in a nonsequential manner. This can be a problem on the Cray-1 if the first dimension of the array A is an integral multiple of half the number of memory banks. If that occurs, memory bank conflicts will arise which will greatly slow down the calculation. Some times for the decomposition step for various levels of unrolling are given in Table VIII along with the time for the Cray-1 SCILIB routine SGEFA. The maximum speedup, a factor of about 2.3 for both CFT 1.11 and CFT 1.13 compilers, occurs for unrolling to a depth of 32, which is the maximum unrolling depth we considered. It can be seen from the table that unrolling to depths greater than 32 will not significantly increase the speedup factor. The CFT 1.13 compiler produces code which runs 6-15% faster than the CFT 1.11 compiler.

2. Forward elimination and back substitution step

Both the forward elimination and back substitution steps can also benefit from loop unrolling.[21] For a single vector $\vec{b}$ the code to determine $\vec{x}$ from the decomposition performed above (the values of $\vec{b}$ in the array B are overwritten with the values of $\vec{x}$) can be written (this code was given to us by Dongarra)

```
        DO 1 K=1, N                                        (A6)
        L=IPVT(K)
        T=B(L)
        B(L)=B(K)
  1     B(K)=T
        DO 2 K1A=1,N-1
        T=B(K1A)*A(K1A,K1A)
            DO 3 I=K1A+1,N
  3         B(I)=B(I)-A(I,K1A)*T
  2     B(K1A)=T
        B(N)=B(N)*A(N,N)
        DO 4 K2A=N,2,-1
        T=B(K2A)
        DO 4 I1=1,K2A-1
  4     B(I1)=B(I1)+A(I1,K2A)*T
```

The K1A loop and K2A loops can be unrolled, e.g., unrolling the first loop to a depth of two yields

```
        NREM=MOD(N,2)                                          (A7)
        M=N-NREM
            DO 2 K1A=1,M,2
            T1=B(K1A)*A(K1A,K1A)
            B(K1A)=T1
            T2=(B(K1A+1)-A(K1A+1,K1A)*T1)*A(K1A+1,K1A+1)
            B(K1A+1)=T2
                DO 3 I=K1A+2,N
  3             B(I)=((B(I))-A(I,K1A)*T1)-A(I,K1A+1)*T2
  2         CONTINUE
        IF (NREM.EQ.0) GO TO 5
        B(N)=B(N)*A(N,N)
  5     CONTINUE
```

A similar procedure can be used to unroll the K2A loop in (A6). However, if the number, NRHS, of vectors $\vec{b}$ is large, it is more efficient to re-organize the substitution steps in (A6) to yield

```
        DO 1 K=1,N                                                    (A8)
        L=IPVT(K)
        IF (L.EQ.K) GO TO 1
            DO 11 J=1, NRHS
            T=B(L,J)
            B(L,J)=B(K,J)
11          B(K,J)=T
 1      CONTINUE
        DO 3 I=1,N
            DO 2 K1B=1,I-1
            DO 2 J1=1,NRHS
 2          B(I,J1)=B(I,J1)-A(I,K1B)*B(K1B,J1)
        DO 3 J2=1,NRHS
 3      B(I,J2)=B(I,J2)*A(I,I)
        DO 4 I1=N,1,-1
        DO 4 K2B=I1+1,N
        DO 4 J3=1,NRHS
 4      B(I1,J3)=B(I1,J3)+A(I1,K2B)*B(K2B,J3)
```

In algorithm (A8) the K1B loop and the K2B loop are the ones to be unrolled. This version is faster than algorithm (A7) because the vectorized loops have constant lengths, i.e., vectors of length NRHS are always used, whereas in algorithm (A7) the vector lengths range from 1 to N. It should be noted that the second index of the array B is varied fastest in the J, J1, J2, and J3 loops; since this constitutes nonsequential access from memory, memory bank conflicts may pose a problem. Table VIII shows times for the forward elimination and back substitution steps for algorithm (A8) with NRHS = N using various levels of loop unrolling along with the time for the Cray-1 SCLIB routine SGESL. For the CFT 1.11 compiler, unrolling to a depth of 32, which is the maximum considered, gives the largest speedup — a factor of about 2.6, while for the CFT 1.13 compiler the largest speedup is a factor of 2.4 and occurs when unrolling to a depth of 16. As before, we see from the table that further unrolling will not produce significant additional speedups for the CFT 1.11 compiler. The CFT 1.13 compiler produces code which runs 2-16% faster than the CFT 1.11 compiler for a given unrolling depth. Overall for both decomposition and for the forward elimination and back substitution steps we obtain a factor of about 2.6 over using the SCILIB routines SGEFA and SGESL and a factor of about 2.2 over using the SCILIB routine MINV.

In the code used in the rigid rotator calculations reported here, we used algorithm (A4) and (A6) unrolled to a depth of 8. Additional performance enhancements of our code would be obtained by using algorithm (A4) and (A8) unrolled to a depth of 32.

It should be noted that since both algorithm (A4) and (A8) involve varying the second index of an array fastest, these algorithms are not well suited for the Cyber 205.

We have found that the larger the amount of loop unrolling, the better the performance on the Cray-1. In the standard linear algebraic examples considered here, the algebraic expressions to be evaluated in the innermost loop are relatively simple, and we have seen that unrolled loops are evaluated very efficiently on the Cray-1. Although this need not be the case for more complicated examples, in our application a large fraction of the CPU time is spent on these standard linear algebraic steps. Possible difficulties in extending the techniques presented here to more complicated expressions is that the number of instructions may eventually exceed the size of the instruction buffer, causing chaining to be inhibited, and the compiler may not be able to determine the optimum utilization of the vector registers and so be forced to save temporary data in main memory.

References

1. R.B. Bernstein, "Atom-Molecule Collision Theory," Plenum, New York, 1979.
2. S.R. Leone, J. Phys. Chem. Ref. Data 11, 953 (1982).
3. An incomplete survey of HF-HF potential energy surfaces and related calculations includes (a) D.R. Yarkony, S.V. O'Neil, H.F. Schaefer III, C.P. Baskin, and C.F. Bender, J. Chem. Phys. **60**, 855 (1974); (b) M.H. Alexander and A.E. DePristo, J. Chem. Phys. **65**, 5009 (1976); (c) L.L. Poulsen, G.D. Billing, and J.I. Steinfeld, J. Chem. Phys. **68**, 5121 (1978); (d) M.L. Klein, I.R. McDonald, and S.F. O'Shea, J. Chem. Phys. **69**, 63 (1978); (e) F.A. Gianturco, U.T. Lamanna, and F. Battiglia, Int. J. Quantum Chem. 19, 217 (1981); (f) R.L. Redington, J. Chem. Phys. **75**, 4417 (1981); (g) A.E. Barton and B.J. Howard, Faraday Disc. Chem. Soc. **73**, 45 (1982); (h) J.T. Brobjer and J.N. Murrell, Mol. Phys. **50**, 885 (1983); (i) M.E. Cournoyer and W.L. Jorgensen, Mol. Phys. **51**, 119 (1984); (j) D.W. Michael, C.E. Dykstra, and J.M. Lisy, J. Chem Phys. **81**, 5998 (1984); (k) D.W. Schwenke and D.G. Truhlar, J. Chem. Phys. **82**, 2418 (1985); and (l) M.J. Redmon and J.S. Binkley, unpublished.
4. D.W. Schwenke and D.G. Truhlar, in "Supercomputer Applications," edited by R.W. Numrich, Plenum, New York, 1985, p. 215.
5. D.W. Schwenke and D.G. Truhlar, paper presented at the 1985 Cray Science and Engineering Symposium, Bloomington, Minnesota, April 14-17, 1985, University of Minnesota Supercomputer Institute report UMSI85/5.
6. L.D. Thomas, J. Chem. Phys. **76**, 4925 (1982).
7. J.F. McNutt and M.J. Redmon, unpublished.
8. (a) A.E. DePristo and M.H. Alexander, J. Chem. Phys. **66**, 1334 (1977); (b) M.H. Alexander, J. Chem. Phys. **73**, 5135 (1980); (c) H.K. Haugen, W.H. Pence, and S.R. Leone, J. Chem. Phys. **80**, 1839 (1984); (d) P.H. Vohralik and R.E. Miller, J. Chem. Phys. **83**, 1609 (1983).
9. D.W. Schwenke, D. Thirumalai, D.G. Truhlar, and M.E. Coltrin, J. Chem. Phys. **78**, 3078 (1983).
10. J.C. Light and R.B. Walker, J. Chem. Phys. **65**, 4272 (1976).
11. (a) N.A. Mullaney and D.G. Truhlar, Chem. Phys. **39**, 91 (1979); (b) D.G. Truhlar, N.M. Harvey, K. Onda, and M.A. Brandt, in "Algorithms and Computer Codes for Atomic and Molecular Quantum Scattering Theory," Vol. 1, edited by L. Thomas, National Resource for Computation in Chemistry, Lawrence Berkeley Laboratory, Berkeley, CA, 1979, p. 220; (c) N.M. Harvey and D.G. Truhlar, Chem. Phys. Lett. **74**, 252 (1980); (d) D. Thirumalai and D.G. Truhlar, J. Chem. Phys. **76**, 5287 (1982).
12. D.U. Webb and K.N. Rao, J. Mol. Spectry. **28**, 121 (1968).
13. D.W. Schwenke and D.G. Truhlar, Comp. Phys. Comm. **34**, 57 (1984).
14. J.N. Murrell and K.S. Sorbie, J. Chem. Soc. Faraday Trans. II **70**, 1552 (1974).
15. R.N. Siko and T.A. Cool, J. Chem. Phys. **65**, 117 (1976).
16. K.D. Huber and G. Herzberg, "Constants for Diatomic Molecules" Van Nostrand, Princeton, New York, 1979.
17. D. Maillard and B. Silvi, Mol. Phys. **40**, 933 (1980).
18. G.H. Golub and C.F. Van Loan, "Matrix Computations," Johns Hopkins University Press, Baltimore, Maryland, 1983, (a) p. 69, (b) p. 53, (c) p. 282, (d), chapter 4. Note that the authors count operations in terms of a unit that corresponds to a floating point add plus a floating point multiply. We have converted the operation count to the more usual definition by which each add or multiply counts as an operation.
19. R.W. Hockney and C.F. Jesshope, "Parallel Computers," Adam Hilger Ltd., Bristol, England, 1981, (a) p. 124, (b) p.80.
20. J.J. Dongarra and S.C. Eisenstat, ACM Trans. Math. Software **10**, 221 (1984).
21. J.J. Dongarra, L. Kaufman, and S. Hammarling, Argonne National Laboratory, Mathematics and Computer Science Division, Technical Memorandum No. 46, Jan., 1985, unpublished.
22. B.T. Smith, J.M. Boyle, J.J. Dongarra, B.S. Garbow, Y. Ikebe, V.C. Klema, and C.B. Moler, "Matrix Eigensystem Routines-EISPACK Guide," Springer-Verlag, New York, 1976.

23. Cray-1 and Cray X-MP Computer Systems Library Reference Manual SR-0014, Cray Research, Inc., Mendota Heights, Minnesota, 1984.
24. Mathematical-Geophysical Vector Library for the Cyber 205, Control Data Corporation, unpublished.
25. J.J. Dongarra, C.B. Moler, J.R. Bunch, and G.W. Stewart, "LINPACK User' Guide," Society for Industrial and Applied Mathematics, Philadelphia, 1979.
26. C. Lawson, R. Hanson, D. Kincaid, and F. Krogh, ACM Trans. Math. Software **5**, 308 (1979).
27. J.J. Dongarra, Argonne National Laboratory, Mathematics and Computer Science Division, Technical Memorandum No. 23, revised version of Dec., 1984.
28. L.W. Johnson and R.D. Riess, "Numerical Analysis," 2nd ed., Addison-Wesley, Reading, Massachusetts, 1982, chapter 2.