An intertwined method for making low-rank, sum-of-product basis functions that makes it possible to compute vibrational spectra of molecules with more than 10 atoms

Phillip S. Thomas^{a)} and Tucker Carrington, Jr.^{b)}

Chemistry Department, Queen's University, Kingston, Ontario K7L 3N6, Canada

(Received 14 February 2017; accepted 5 May 2017; published online 31 May 2017)

We propose a method for solving the vibrational Schrödinger equation with which one can compute spectra for molecules with more than ten atoms. It uses sum-of-product (SOP) basis functions stored in a canonical polyadic tensor format and generated by evaluating matrix-vector products. By doing a sequence of partial optimizations, in each of which the factors in a SOP basis function for a single coordinate are optimized, the rank of the basis functions is reduced as matrix-vector products are computed. This is better than using an alternating least squares method to reduce the rank, as is done in the reduced-rank block power method. Partial optimization is better because it speeds up the calculation by about an order of magnitude and allows one to significantly reduce the memory cost. We demonstrate the effectiveness of the new method by computing vibrational spectra of two molecules, ethylene oxide (C_2H_4O) and cyclopentadiene (C_5H_6), with 7 and 11 atoms, respectively. *Published by AIP Publishing.* [<http://dx.doi.org/10.1063/1.4983695>]

I. INTRODUCTION

Given a potential energy surface (PES), it is now rather straightforward to compute the vibrational spectrum of a tri-atomic molecule using a variational method.^{1–3} When the PES is a sum of products (SOPs) and the basis functions are products of univariate functions, potential matrix elements are easily obtained from products of 1D integrals that are evaluated with 1D quadrature. For a general PES, one needs 3D quadrature to calculate potential matrix elements, but this is not difficult on any modern computer. The Hamiltonian matrix whose eigenvalues approximate energy levels is small enough that the standard eigensolvers can be used. For four-atom, five-atom, and six-atom molecules, variational calculations are more difficult, especially for general (e.g., not SOP) PESs. Using a SOP PES obviates the need for multidimensional quadrature, but the basis set is usually large enough that standard eigensolvers are either costly or impossible to use. Both the CPU and the memory costs are too high. When the memory required to store the Hamiltonian matrix is larger than the memory of the computer, a calculation with a standard eigensolver is impossible. In this paper, we present ideas that make it possible to do variational calculations to obtain vibrational spectra of molecules with 11 atoms. They work only if the PES is SOP. For many molecules with five or more atoms, most available PESs are in the SOP form. There are several methods for bringing a general PES into a SOP form.^{4–6}

Probably the simplest way to overcome the memory barrier that one confronts for molecules with more than three atoms is to use an iterative eigensolver.^{7–12} When using an iterative eigensolver, it is necessary only to store vectors

and not the Hamiltonian matrix. In fact, it is also unnecessary to compute elements of the Hamiltonian matrix. With a direct product basis (DPB), the size of the vectors is n^D , where n is the number of basis functions for a single coordinate and D is the number of coordinates. Unfortunately, in the six-atom case, even storing vectors is impossible. One way to deal with the problem is to prune the direct product basis.^{1,3,13–15} If, for example, only basis functions are retained that satisfy $n_1 + n_2 + \dots + n_D \leq n - 1$ then the size of the vectors (basis) is reduced from n^D to $\frac{(D+n-1)!}{D!(n-1)!}$. Although this reduction is substantial, the memory required is still large for molecules with 10 or more atoms. There are vibrational coupled cluster methods that use a basis of selected product functions that have some of the same advantages as pruned basis methods.¹⁶

We have recently shown that it is possible to use a DPB in conjunction with an iterative eigensolver without needing to store n^D numbers to represent each vector.^{17–20} This is done by forcing every basis function to be a SOP,

$$\Psi(q_1, \dots, q_D) \simeq \sum_{i_1=1}^{n_1} \dots \sum_{i_D=1}^{n_D} F_{i_1, \dots, i_D} \prod_{c=1}^D \varphi_{i_c}^c(q_c), \quad (1)$$

where c labels a single coordinate, and

$$F_{i_1, \dots, i_D} = \sum_{r=1}^R s_r^F \prod_{c=1}^D f_{i_c}^{(r,c)} \quad (2)$$

is a tensor of basis coefficients. In mathematical language, it is a tensor in the (canonical polyadic) CP-format and R is called the rank.²¹ The memory cost of storing $F_{i_1, \dots, i_D}$ is RnD , which scales linearly and not exponentially with D . Wavefunctions are obtained by projecting into the space spanned by a set of such SOP basis functions. Storing basis functions and wavefunctions in the CP-format eliminates the memory problem that one encounters when trying to compute vibrational spectra

^{a)}E-mail: Phillip.Thomas@chem.queensu.ca

^{b)}E-mail: Tucker.Carrington@queensu.ca. Fax: 613-533-6669.

of molecules with 6, 7, . . . atoms. In practice, we impose a rank, R , and find that with R between 10 and 100 we are able to obtain accurate solutions to the Schrödinger equation for molecules with fewer than eight atoms. These ideas were implemented in the Reduced-Rank Block Power Method (RRBPM), which uses a shifted power method to generate $F_{i_1, \dots, i_D}$. With the RRBPM, it is possible to compute the lowest 70 eigenstates of CH_3CN using less than 1 GB of memory. There are also multi-configuration time-dependent Hartree methods for computing vibrational spectra.^{22–25} They also use a tensor format to store basis vectors; however, it is a direct product (or Tucker) format, and although the 1D basis functions are optimized, the memory cost scales exponentially with D . The CP format has also been used to reduce the memory cost of storing vectors in a vibrational coupled cluster (VCC) calculation.^{26,27} However, it has not yet been possible to devise a VCC algorithm, in which, like the RRBPM, all vectors are directly computed in the CP format, and that therefore requires storing only vectors in the CP format and exploits its advantages to reduce the CPU cost.

One shortcoming of the RRBPM is that the shifted power method converges slowly, requiring >1000 iterations (i.e., matrix-vector products (MVPs) for each vector in the block) to achieve a moderate accuracy for a six-atom molecule such as CH_3CN . The slow convergence can be mitigated in several ways. One way is to do separate calculations for different symmetries.¹⁹ The second way is to replace the shifted power method with an iterative eigensolver that converges more quickly.^{20,28} The third way is to arrange the coordinates of the molecule into groups in a tree structure and to build the basis hierarchically by solving eigenproblems for subsets of the coordinates.¹⁸ This last strategy, known as the Hierarchical (H-) RRBPM, uses an RRBPM to compute eigenstates at each node of the tree. With a good choice of tree, the H-RRBPM is orders-of-magnitude faster than the ordinary RRBPM. Importantly, since the H-RRBPM also uses the shifted power method, typically fewer than 100 iterations are needed to converge the basis functions of a given node, with convergence being slowest at the top node of the tree where the energy spectrum is densest. The H-RRBPM could be used in combination with either of the previous two strategies. Rakhuba and Oseledets have recently proposed a method that, like the RRBPM, builds a basis by evaluating matrix-vector products and reducing the rank of output vectors.²⁸ Rather than using the CP format, they opt for another tensor format. It is known as tensor-train to mathematicians and as matrix-product states (MPSs) to physicists and chemists. Rather than the power method they use inverse iteration, which converges more quickly. Because it uses MPS, their method has some of the features of the density matrix renormalization group approach, but it, like the RRBPM, builds a basis rather than optimizing a parameterized wavefunction.

The RRBPM, whether applied to the full problem or at the top node of an H-RRBPM tree, has two other important shortcomings. The most severe is the cost of the rank reduction. After each matrix-vector product, one must reduce the rank of the output vector. To do this, we used an implementation of the Alternating Least Squares (ALS) algorithm similar to the one proposed by Beylkin and Mohlenkamp.²⁹ In previous

calculations, most of the CPU time (typically $>90\%$) was spent on rank reduction. Much less severe, but potentially important for molecules with dozens of atoms or if one is using a desktop computer with less than 32 GB of memory, is the memory cost of storing large-rank vectors. Storing a vector before its rank is reduced requires storing RnD numbers. For molecules studied previously (acetonitrile, ethylene oxide), for which the SOP PES has only hundreds of terms, this was not a problem. However, if MVPs for different vectors in the block are evaluated on different processors, then for larger molecules the memory cost might be a problem. The largest rank (R) usually obtained in an RRBPM calculation is a product of the rank specified for the basis vectors (R_ψ) and the number of terms in the SOP PES (T) which might be $T \sim 10\,000$. If $R = R_\psi * 10\,000$ and $R_\psi = 100$ and $D = 50$ and $n = 20$, then ~ 6 GB is required to store one vector.

In this paper, we modify the RRBPM to reduce its CPU cost. In the original RRBPM, after generating a tensor $G_{i_1, \dots, i_D}$,

$$G_{i_1, \dots, i_D} = \sum_{r=1}^{R_G} s_r^G \prod_{c=1}^D g_{i_c}^{(r,c)} \quad (3)$$

from the input vector ${}^{in}F_{i_1, \dots, i_D}$ by evaluating a MVP (here, the rank of G is $T * R_\psi$, where T is the number of terms in the shifted Hamiltonian), one applies ALS to reduce its rank. ALS optimizes ${}^{out}f_{i_c}^{(r,c)}$ so that ${}^{out}F_{i_1, \dots, i_D} \sim G_{i_1, \dots, i_D}$, where

$${}^{out}F_{i_1, \dots, i_D} = \sum_{r=1}^{R_\psi} {}^{out}s_r^F \prod_{c=1}^D {}^{out}f_{i_c}^{(r,c)}. \quad (4)$$

Output vectors ${}^{out}f_{i_c}^{(r,c)}$ are determined first for $c = 1$ and then for $c = 2, \dots, D$, and this cycle is repeated N_{ALS} times to obtain a final set of ${}^{out}f_{i_c}^{(r,c)}$. ${}^{out}f_{i_c}^{(r,c)}$ are then re-named ${}^{in}f_{i_c}^{(r,c)}$, and another MVP is computed. This process is repeated for all vectors in a block, and the Schrödinger equation is projected onto the basis obtained after N_{pow} MVPs.

In this paper, each of the MVPs of the RRBPM is replaced by D MVPs. After evaluating D MVPs for each vector in the block, we obtain a new block of vectors. After each of the D MVPs, rather than using ALS to optimize ${}^{out}f_{i_c}^{(r,c)}$ for all coordinates, we optimize ${}^{out}f_{i_c}^{(r,c)}$ for a single coordinate. This greatly reduces the cost of the optimization. Due to the incomplete optimization, the individual basis vectors in the block may not be as close to eigenvectors as those of the RRBPM. However, the energy levels we compute are *more* accurate because (1) the optimization is so cheap that we can use a larger rank (R_ψ); (2) even if the optimization is poor, repeated applications of the shifted Hamiltonian will push all vectors in the block in the right direction; (3) it is not important that individual basis vectors be close to eigenvectors, what matters is the space spanned by the vectors in the block. We are intertwining the optimization of ${}^{out}f_{i_c}^{(r,c)}$ for coordinates $c = 1, 2, \dots, D$ with the evaluation of MVPs.

The intertwining has three advantages. First, fewer systems of linear equations must be solved to make the final basis. This reduces the calculation time by about an order of magnitude and makes it possible to use larger ranks. Second, it reduces the cost of setting up the linear systems that must be

solved to do the optimizations done to make new blocks of vectors. Third, the new method can be implemented so that there is no need to generate large-rank CP-vectors. This considerably reduces the memory cost of calculations if a Hamiltonian with many terms is used or if many states are computed in parallel.

II. INTERTWINING MATRIX-VECTOR PRODUCTS WITH RANK REDUCTION

A. CP format and matrix-vector products

The RRBPM uses sum-of-product basis functions,

$$\Psi(q_1, \dots, q_D) \simeq \sum_{i_1=1}^{n_1} \dots \sum_{i_D=1}^{n_D} F_{i_1, \dots, i_D} \prod_{c=1}^D \varphi_{i_c}^c(q_c), \quad (5)$$

where $\varphi_{i_c}^c(q_c)$ are 1-D basis functions depending on coordinate c , and

$$F_{i_1, \dots, i_D} = \sum_{r=1}^{R_\psi} s_r^{\mathbf{F}} \prod_{c=1}^D f_{i_c}^{(r,c)}, \quad (6)$$

where each term in the sum over r is weighted by a normalization coefficient $s_r^{\mathbf{F}}$. We require the Hamiltonian operator to be in the sum-of-product form,

$$\hat{H} = \sum_{m=1}^T \prod_{c=1}^D \hat{h}_{m,c}, \quad (7)$$

where $\hat{h}_{m,c}$ is an operator depending only on coordinate c . A matrix-vector product is¹⁷

$$\begin{aligned} & (\mathbf{G})_{i'_1, i'_2, \dots, i'_D} \\ &= (\mathbf{H}\mathbf{F})_{i'_1, i'_2, \dots, i'_D} \\ &= \sum_{i_1, i_2, \dots, i_D} \left(\sum_{m=1}^T \prod_{c=1}^D \hat{h}_{m,c} \right)_{i'_1, i'_2, \dots, i'_D; i_1, i_2, \dots, i_D} \sum_{r=1}^{R_\psi} s_r^{\mathbf{F}} \prod_{c=1}^D f_{i_c}^{(r,c)} \\ &= \sum_{m=1}^T \sum_{r=1}^{R_\psi} s_r^{\mathbf{F}} \prod_{c=1}^D \sum_{i_c} (\hat{h}_{m,c})_{i'_c, i_c} f_{i_c}^{(r,c)} \\ &= \sum_{m=1}^T \sum_{r=1}^{R_\psi} s_r^{\mathbf{F}} \prod_{c=1}^D \sum_{i_c} g_{i_c}^{(r,c)}. \end{aligned} \quad (8)$$

In this paper, we denote large-rank vectors $\mathbf{G}$ (rank $\gg R_\psi$) and small-rank vectors $\mathbf{F}$ (rank = R_ψ). The matrix-vector product costs $O(TR_\psi n^2 D)$ operations.

B. Reduced-rank block power method

The RRBPM algorithm used in this work is similar to the one in Ref. 17 (Alg. 1). In Alg. 1, a block of $\mathcal{B}$ guess vectors $\mathcal{F} = \{\mathbf{F}_k\}$ ($k = 1 \dots \mathcal{B}$) is iteratively improved until the set of vectors converges to the eigenvectors associated with the $\mathcal{B}$ smallest eigenvalues. Calls to RRPM, in step 1, take most of the CPU time. In Subsection II C, we describe how to improve the RRBPM by intertwining RRPM steps a and b. All of the steps in Alg. 1 which generate a large-rank $\mathbf{G}_k$ vector must be followed by rank-reduction and normalization. The $\mathbf{G}_k$ vector in the RRPM step a has rank TR_ψ , whereas the $\mathbf{G}_k$ vectors in steps 2a and 3d-i have at most rank $\mathcal{B}R_\psi$. We use Alternating

Algorithm 1. Main steps of the RRBPM algorithm.

Input: $\mathcal{F}$, matrix of guess vectors; output: $\mathcal{F}$, matrix of eigenvectors

---

For $\alpha = 1 \dots N_{\text{cyc}}$:

(1) Power iterations

For $k = 1 \dots \mathcal{B}$:

Call RRPM($\mathbf{F}_k$) (see below)

(2) Gram-Schmidt orthogonalization:

For $k = 2 \dots \mathcal{B}$

(a) compute $\mathbf{G}_k \leftarrow \text{in} \mathbf{F}_k - \sum_{k'=1}^{k-1} \langle \text{in} \mathbf{F}_{k'}^T, \text{in} \mathbf{F}_k \rangle \cdot \text{in} \mathbf{F}_{k'}$

(b) reduce rank $\text{out} \mathbf{F}_k^{(R=R_\psi)} \leftarrow \mathbf{G}_k^{(R=\mathcal{B}R_\psi)}$

(c) normalize $\mathbf{F}_k \leftarrow \frac{\mathbf{F}_k}{\|\mathbf{F}_k\|}$

(3) Vector update

(a) compute $\mathbf{H}^{(\mathcal{F})} = \mathcal{F}^T \mathbf{H} \mathcal{F}$ and $\mathbf{S} = \mathcal{F}^T \mathcal{F}$

(b) solve $\mathbf{H}^{(\mathcal{F})} \mathbf{U} = \mathbf{S} \mathbf{U}$

(c) rename $\text{in} \mathcal{F} \leftarrow \mathcal{F}$

(d) for $k = 1 \dots \mathcal{B}$

(i) compute $\mathbf{G}_k \leftarrow \sum_{k'=1}^{\mathcal{B}} \mathbf{U}_{k'} \text{in} \mathbf{F}_{k'}$

(ii) reduce rank $\text{out} \mathbf{F}_k^{(R=R_\psi)} \leftarrow \mathbf{G}_k^{(R=\mathcal{B}R_\psi)}$

(iii) normalize $\mathbf{F}_k \leftarrow \frac{\mathbf{F}_k}{\|\mathbf{F}_k\|}$

---

subroutine RRPM($\mathbf{F}_k$): Reduced-Rank Power Method

Input: $\mathbf{F}_k$, old basis vector; output: $\mathbf{F}_k$, new basis vector;

for $j = 1 \dots N_{\text{pow}}$:

(a) matrix-vector product $\mathbf{G}_k \leftarrow (\mathbf{H} - E_s \mathbf{1}) \text{in} \mathbf{F}_k$

(b) reduce rank $\text{out} \mathbf{F}_k^{(R=R_\psi)} \leftarrow \mathbf{G}_k^{(R=(T+1)R_\psi)}$

(c) normalize $\mathbf{F}_k \leftarrow \frac{\mathbf{F}_k}{\|\mathbf{F}_k\|}$

Least Squares (ALS)²⁹ to reduce the rank (unless there are only two coordinates, in which case we reduce the ranks using singular value decomposition).³⁰ The ALS procedure finds a small-rank vector, $\mathbf{F}_k$, which closely approximates a large-rank vector, $\mathbf{G}_k$, by iteratively improving $\mathbf{F}_k$ one coordinate at a time.

For each coordinate c , let us define the following matrices of inner products:

$$B_{r',r}^c = \langle f^{(r',c)}, f^{(r,c)} \rangle, \quad (9a)$$

$$P_{r',r}^c = \langle g^{(r',c)}, f^{(r,c)} \rangle, \quad (9b)$$

$$B_{r',r}^{\neq c} = \prod_{c' \neq c} B_{r',r}^{c'}, \quad (9c)$$

$$P_{r',r}^{\neq c} = \prod_{c' \neq c} P_{r',r}^{c'}, \quad (9d)$$

$$B_{r',r} = \prod_{c=1}^D B_{r',r}^c, \quad (9e)$$

$$P_{r',r} = \prod_{c=1}^D P_{r',r}^c. \quad (9f)$$

Note that elements of the matrices $\mathbf{B}^{\neq c}$ and $\mathbf{P}^{\neq c}$ are products of inner products for all coordinates *except* coordinate c . $\mathbf{P}^{\neq c}$ is used to construct right-hand sides of a linear system,

$$b_r^{(i_c,c)} = \sum_{r'=1}^{R_G} s_{r'}^{\mathbf{G}} g_{i_c}^{(r',c)} P_{r',r}^{\neq c}. \quad (10)$$

Algorithm 2. Alternating least squares algorithm.

Input: CP-format vectors $\mathbf{F}$ and $\mathbf{G}$, with ranks $R_\psi < R_G$;
Output: $^{out}\mathbf{F}$, such that $\|^{out}\mathbf{F} - \mathbf{G}\|$ is minimized

(1) Initialization:
construct $\mathbf{B}$ and $\mathbf{P}$ matrices

(2) for $\alpha = 1 \dots N_{ALS}$:
for $c = 1 \dots D$: (Loop over coordinates)
(a) downdate: $B_{r',r}^{\pm c} \leftarrow B_{r',r}^{\pm c} / B_{r',r}^c$ for all r', r
(b) downdate: $P_{r',r}^{\pm c} \leftarrow P_{r',r}^{\pm c} / P_{r',r}^c$ for all r', r
(c) compute $b_{i_c,c}^{(i_c,c)}$ for all i_c, r (Eq. (10))
(d) solve linear system for $\mathbf{x}^{(i_c,c)}$ (Eq. (11));
replace $f_{i_c}^{(r,c)} \leftarrow x_r^{(i_c,c)}$ for all i_c, r
(e) normalize $s_r^{\mathbf{F}} \leftarrow \|\mathbf{f}^{(r,c)}\|$; $\mathbf{f}^{(r,c)} \leftarrow \frac{\mathbf{f}^{(r,c)}}{\|\mathbf{f}^{(r,c)}\|}$ for all r ;
(f) update: $B_{r',r}^{\pm c} \leftarrow B_{r',r}^{\pm c} \cdot B_{r',r}^c$ for all r', r
(g) update: $P_{r',r}^{\pm c} \leftarrow P_{r',r}^{\pm c} \cdot P_{r',r}^c$ for all r', r

---

The linear systems are solved for $\mathbf{x}^{(i_c,c)}$,

$$\mathbf{B}^{\pm c} \mathbf{x}^{(i_c,c)} = \mathbf{b}^{(i_c,c)}. \quad (11)$$

The solutions $x_r^{(i_c,c)}$ replace $f_{i_c}^{(r,c)}$. We then normalize each $\mathbf{f}^{(r,c)}$ so that $s_r^{\mathbf{F}} = \|\mathbf{f}^{(r,c)}\|$ and $normalized \mathbf{f}^{(r,c)} = \frac{\mathbf{f}^{(r,c)}}{\|\mathbf{f}^{(r,c)}\|}$. The process of applying Eqs. (9a)–(11) to compute new $\mathbf{f}_{i_c}^{(r,c)}$ for coordinates $c = 1 \dots D$ in sequence constitutes one full ALS sweep. Our ALS implementation is summarized in Alg. 2.

For each ALS sweep and each c , steps 2a and 2f of Alg. 2 require computing a $\mathbf{B}^c$ matrix, which costs $\mathcal{O}(n_c R_\psi^2)$ operations. Likewise, constructing the $\mathbf{P}^c$ matrices in steps 2b and 2g costs $\mathcal{O}(n_c R_G R_\psi)$ operations. Building the right-hand sides in step 2c costs $\mathcal{O}(n_c R_G R_\psi)$, and solving the linear system in step 2d costs $\mathcal{O}(R_\psi^3)$ operations. The total cost of ALS is therefore $\mathcal{O}(D(n_c R_\psi^2 + n_c R_G R_\psi))$ to initialize and $\mathcal{O}(N_{ALS} D(2n_c R_\psi^2 + 3n_c R_G R_\psi + R_\psi^3))$ to perform the sweeps in step 2. In an RRBPM calculation, the ALS rank-reduction dominates the cost. Most costly are the steps 2b, 2c, and 2g of Alg. 2 that require constructing and manipulating $\mathbf{P}$ matrices. Steps 2b, 2c, and 2g are more expensive than step 2d because $R_G = TR_\psi$ is large since there are usually many terms, T , in the Hamiltonian. For a single value of α and c , if one compares the cost of steps 2c and 2d with $n = 10$, $R_\psi = 100$, and $T = 1000$, one finds that solving the linear system in step 2d costs only $\mathcal{O}(R_\psi^3) \sim 10^6$ operations, which is much less than the $\mathcal{O}(n_c R_G R_\psi) \sim 10^8$ operations required to build the right hand sides in step 2c. Moreover, as the number of atoms in the molecule increases, we expect T to increase faster than R_ψ (the number of quartic terms scales as D^4), which means that ALS steps 2b, 2c, and 2g will dominate the cost of using the RRBPM on large molecules.

C. Intertwining: Reducing the cost by using partial optimization

In the call to RRBPM in step 1 of Alg. 1, each vector is replaced with a new vector obtained by repeating the ALS procedure N_{pow} times. Each call to RRBPM requires making $\mathbf{P}^{\pm c}$ matrices and the right hand sides $\mathbf{b}^{(i_c,c)}$ (Eq. (10)) after every MVP and solving linear equations $N_{pow} N_{ALS} D$ times for each vector and this is costly. In this paper, we use a modified

Algorithm 3. Intertwined power method.

Input: vector in block, $\mathbf{F}$, with rank R_F
Output: improved vector in block, $\mathbf{F}$, with rank R_F

(1) Initialization:
(a) compute matrix-vector product: $(\mathbf{H} - E_s \mathbf{1}) \mathbf{F} = \mathbf{G}$
(b) construct $\mathbf{B}$ and $\mathbf{P}$ matrices

(2) for $\alpha = 1 \dots N_{sweep}$:
For $c = 1 \dots D$: (Loop over coordinates)
(a) downdate: $B_{r',r}^{\pm c} \leftarrow B_{r',r}^{\pm c} / B_{r',r}^c$ for all r', r
(b) downdate: $P_{r',r}^{\pm c} \leftarrow P_{r',r}^{\pm c} / P_{r',r}^c$ for all r', r
(c) compute $b_{i_c,c}^{(i_c,c)}$ for all i_c, r (Eq. (10))
(d) solve linear system for $\mathbf{x}^{(i_c,c)}$ (Eq. (11));
replace $f_{i_c}^{(r,c)} \leftarrow x_r^{(i_c,c)}$ for all i_c, r
(e) normalize $s_r^{\mathbf{F}} \leftarrow \|\mathbf{f}^{(r,c)}\|$; $\mathbf{f}^{(r,c)} \leftarrow \frac{\mathbf{f}^{(r,c)}}{\|\mathbf{f}^{(r,c)}\|}$ for all r ;
if $c = D$: normalize $\mathbf{F} \leftarrow \frac{\mathbf{F}}{\|\mathbf{F}\|}$
(f) update MVP:
For $m = 1 \dots T$:
For $l = 1 \dots R_\psi$:
 $r' \leftarrow (m-1)R_\psi + l$
 $\mathbf{g}^{(r',c)} \leftarrow s_l^{\mathbf{F}} \mathbf{h}_{m,c} \mathbf{f}^{(l,c)}$
(g) update: $B_{r',r}^{\pm c} \leftarrow B_{r',r}^{\pm c} \cdot B_{r',r}^c$ for all r', r
(h) update: $P_{r',r}^{\pm c} \leftarrow P_{r',r}^{\pm c} \cdot P_{r',r}^c$ for all r', r

---

version of the RRBPM (Alg. 1) where the call to the RRBPM in step 1 is replaced with Alg. 3. In Alg. 3, for each α , we calculate a new vector, obtained after completion of the loop over c , by evaluating D MVPs and after each of them, rather than using the ALS to optimize $^{out}\mathbf{f}_{i_c}^{(r,c)}$ for *all* coordinates, we optimize $^{out}\mathbf{f}_{i_c}^{(r,c)}$ for a *single* coordinate. What we call a “sweep” is completed when $^{out}\mathbf{f}^{(r,c)}$ for all the coordinates have been replaced. In terms of equations, for $c = c'$, the shifted Hamiltonian is applied to a small-rank vector

$$^{in}F_{i_1, \dots, i_D} = \sum_{r=1}^{R_\psi} \prod_{c=1}^D s_r^{\mathbf{F}} {}^{in}\mathbf{f}_{i_c}^{(r,c)} \quad (12)$$

to make a large-rank vector

$$G_{i_1, \dots, i_D} = \sum_{r'=1}^{R_G} \prod_{c=1}^D s_{r'}^{\mathbf{G}} g_{i_c}^{(r',c)}, \quad (13)$$

which is replaced with a small-rank vector

$$^{out}F_{i_1, \dots, i_D} = \sum_{r=1}^{R_\psi} \prod_{c=1}^D s_r^{\mathbf{F}} {}^{out}\mathbf{f}_{i_c}^{(r,c)}, \quad (14)$$

where for all c except $c = c'$, $^{out}\mathbf{f}_{i_c}^{(r,c)} = {}^{in}\mathbf{f}_{i_c}^{(r,c)}$. The key idea is that $^{out}F_{i_1, \dots, i_D}$ is not fully optimized.

The most important feature of Alg. 3 is that after every MVP $\mathbf{f}^{(r,c)}$ is updated only for one value of c and therefore to evaluate the next MVP it is only necessary to compute $\mathbf{g}^{(r',c)} = \mathbf{h}_{m,c} \mathbf{f}^{(r,c)}$ for one value of c . Steps 2a–2e improve the set of $\mathbf{f}$ vectors for coordinate c ; in step 2f, one multiplies the new $\mathbf{f}^{(r,c)}$ with a 1D matrix in order to update $\mathbf{g}^{(r',c)}$. Because $\mathbf{f}^{(r,c)}$ is updated only for one c , there is no need to compute $\mathbf{B}$ and $\mathbf{P}$ matrices after completing one sweep before beginning another. In Alg. 1, $\mathbf{g}^{(r',c)}$ are replaced in batches, i.e., after each MVP, *all* of the $\mathbf{g}^{(r',c)}$ are modified.

In Alg. 3, $\mathbf{g}^{(r',c)}$ are sequentially replaced in a rolling fashion, i.e., $\mathbf{g}^{(r',c=1)}, \mathbf{g}^{(r',c=2)}, \dots, \mathbf{g}^{(r',c=D)}; \mathbf{g}^{(r',c=1)}, \mathbf{g}^{(r',c=2)}, \dots, \mathbf{g}^{(r',c=D)}, \dots$

Alg. 3 has several important advantages. First, to make a final basis vector, it requires solving linear equations only $N_{\text{sweep}}D$ times (with $N_{\text{sweep}} \approx N_{\text{pow}}$ in Alg. 1) and thereby reduces the number of linear solves by a factor of N_{ALS} . In previous papers, we used values of N_{ALS} between 10 and 100. Second, Alg. 3 significantly reduces the initialization cost because one needs to compute $\mathbf{B}$ and $\mathbf{P}$ matrices from scratch only once (at the beginning of a cycle), before entering the loop over α in step 2. By contrast, in the call to RRPM of Alg. 1, one must compute $\mathbf{B}$ and $\mathbf{P}$ from scratch for each value of j , at the beginning of ALS. Third, after optimizing $f_{i_c}^{(r,c)}$ for coordinate c in Alg. 3, evaluating a MVP requires changing only $\mathbf{g}^{(r',c)}$; $\mathbf{g}^{(r',c')}$ with $c' \neq c$ are re-used. This reduces the cost of the MVPs because in the RRBPM, each time ALS is used, *all* of the $\mathbf{g}_{i_c}^{(r',c)}$ are re-computed.

A better optimization might speed up convergence, but it is not necessary and would be costly. When α is small, there is certainly no need for an excellent optimization because $G_{i_1, \dots, i_D}$ itself may be a poor basis vector (i.e., far from an eigenvector). When α is large, there might be some advantage to doing an excellent optimization because $G_{i_1, \dots, i_D}$ itself is a better basis vector (i.e., close to an eigenvector). Despite the partial optimization of Alg. 3, each time we apply the shifted Hamiltonian we drive the space spanned by basis vectors toward the space spanned by the $\mathcal{B}$ eigenvectors with the smallest eigenvalues. The partial optimization of ${}^{\text{out}}F_{i_1, \dots, i_D}$ introduces noise into the basis vectors, which slows convergence, but one can compensate for it by increasing the rank (R_ψ) of ${}^{\text{out}}F_{i_1, \dots, i_D}$.

The cost per vector is

$$\mathcal{O}\left(N_{\text{pow}}\left(TRn^2D + D(n_cR_\psi^2 + n_cRG R_\psi) + N_{\text{ALS}}D(2n_cR_\psi^2 + 3n_cRG R_\psi + R_\psi^3)\right)\right)$$

for the ordinary power method, but only

$$\mathcal{O}\left(TRn^2D + D(n_cR_\psi^2 + n_cRG R_\psi) + N_{\text{sweep}}D(2n_cR_\psi^2 + 3n_cRG R_\psi + R_\psi^3 + TRn^2)\right)$$

for the intertwined power method. The intertwined algorithm essentially reduces the cost by a factor of N_{ALS} compared to the ordinary power method. In addition, it only requires one to build $\mathbf{B}$ and $\mathbf{P}$ from scratch once per call, in contrast to RRPM, which builds $\mathbf{B}$ and $\mathbf{P}$ from scratch N_{pow} times per call, where $N_{\text{pow}} \sim 10$.

D. Avoiding large-rank vectors

Both the RRBPM and the intertwined RRBPM (I-RRBPM) of Alg. 3 store, in memory, large-rank vectors $G_{i_1, \dots, i_D}$ (see Eq. (3)). This is done by simultaneously storing *all* $\mathbf{g}_{i_c}^{(r',c)}$. There are three types of large-rank vectors: vectors generated by computing a MVP, vectors created when Gram-Schmidt (GS) is applied to $\mathcal{F}$, and vectors one obtains from the vector update step. The rank of the first type is largest. It is $R_G = TR_\psi$, and because the number of terms in the Hamiltonian is large when D is large, the memory cost of storing $G_{i_1, \dots, i_D}$ of the first type can limit the size of the molecule to

Algorithm 4. Intertwined power method, low-memory version.

Input: vector in block, $\mathbf{F}$, with rank R_F
Output: improved vector in block, $\mathbf{F}$, with rank R_F

(1) Initialization:
construct $\mathbf{B}$ and $\mathbf{P}$ matrices

(2) for $\alpha = 1 \dots N_{\text{sweep}}$:
for $c = 1 \dots D$: (Loop over coordinates)
(a) downdate: $B_{r',r}^{\neq c} \leftarrow B_{r',r}^c / B_{r',r}^c$ for all r', r
(b) Call Alg. 5 to calculate $P_{r',r}^{\neq c}$ for all r', r ; and $b_r^{(i_c,c)}$ for all i_c, r (Eq. (10))
(c) solve linear system for $\mathbf{x}^{(i_c,c)}$ (Eq. (11)); replace $f_{i_c}^{(r,c)} \leftarrow x_r^{(i_c,c)}$ for all i_c, r
(d) normalize $s_r^{\mathbf{F}} \leftarrow \|\mathbf{f}^{(r,c)}\|$; $\mathbf{f}^{(r,c)} \leftarrow \frac{\mathbf{f}^{(r,c)}}{\|\mathbf{f}^{(r,c)}\|}$ for all r ;
if $c = D$: normalize $\mathbf{F} \leftarrow \frac{\mathbf{F}}{\|\mathbf{F}\|}$
(e) update: $B_{r',r} \leftarrow B_{r',r}^{\neq c} \cdot B_{r',r}^c$ for all r', r
(f) Call Alg. 6 to compute $P_{r',r}$ for all r', r

---

which one can apply the RRBPM or I-RRBPM. The memory cost of storing the other two types is lower. Parallelizing by computing different vectors in the block on different processors exacerbates the problem because it increases the number of $\mathbf{G}_k$ vectors that must be stored. To parallelize, we use OPENMP. It would be nice to be able to compute spectra of molecules with about 10 atoms on a standard desktop computer. This is possible with the low-memory I-RRBPM of this section. The algorithm that avoids large-rank vectors generated by the MVPs is summarized in Alg. 4.

To use the I-RRBPM idea, one must modify $f_{i_c}^{(r,c)}$ sequentially for $c = 1, 2, \dots, D$. This is done by solving a system of linear equations for each c . To make the linear system, one must compute, sequentially for each c , the RHS $b_r^{(i_c,c)}$ and the matrix $B_{r',r}^{\neq c}$. A simple way to make $B_{r',r}^{\neq c}$ is to keep $B_{r',r}$ in memory and to build $B_{r',r}^{\neq c}$, by dividing out (called downdating in Alg. 4) $B_{r',r}^c$ and then after modifying $f_{i_c}^{(r,c)}$, update $B_{r',r}$ by multiplying by the modified $B_{r',r}^c$. This is implemented in steps 2a and 2e of Alg. 4. Making $B_{r',r}^{\neq c}$ does not require much memory because it does not involve $\mathbf{g}_{i_c}^{(r',c)}$. One has to be more careful when making a RHS $b_r^{(i_c,c)}$ because it depends on *all* $\mathbf{g}_{i_c}^{(r',c)}$ (see Eq. (10)) and it is the memory cost of storing *all* $\mathbf{g}_{i_c}^{(r',c)}$ that may be large. $b_r^{(i_c,c)}$ is calculated in Alg. 5. To avoid storing all $\mathbf{g}_{i_c}^{(r',c)}$, we calculate $\mathbf{g}_{i_c}^{(r',c)}$ and $P_{r',r}^{\neq c}$ and sum their

Algorithm 5. Pseudo-code for downdating $\mathbf{P}$ matrices and computing right-hand sides $\mathbf{b}^{(i_c,c)}$ without generating a long vector $\mathbf{G}$.

Input: CP-format vector, $\mathbf{F}$, with rank R_F , matrix $\mathbf{P}$
Output: matrix $\mathbf{P}^{\neq c}$, where $P_{r',r}^{\neq c} = P_{r',r} / P_{r',r}^c$; right-hand sides, $\mathbf{b}^{(i_c,c)}$

Initialize $b_r^{(i_c,c)} \leftarrow 0$ for all i_c, r
For $m = 1 \dots T$:
For $l = 1 \dots R_\psi$:
 $r' \leftarrow (m-1)R_\psi + l$
 $\mathbf{g}^{(r',c)} \leftarrow \mathbf{h}_{m,c} \mathbf{f}^{(l,c)}$
For $r = 1 \dots R_\psi$:
 $P_{r',r}^{\neq c} \leftarrow P_{r',r} / \langle \mathbf{g}^{(r',c)}, \mathbf{f}^{(r,c)} \rangle$
 $b_r^{(i_c,c)} \leftarrow b_r^{(i_c,c)} + \mathbf{f}_l^{(r',c)} P_{r',r}^{\neq c}$ for all i_c
Discard $\mathbf{g}^{(r',c)}$

Algorithm 6. Pseudo-code for updating $\mathbf{P}^{\neq c}$ matrices without generating a long vector $\mathbf{G}$.

Input: CP-format vector, $\mathbf{F}$, with rank $R_{\mathbf{F}}$, matrix $\mathbf{P}^{\neq c}$

Output: matrix $\mathbf{P}$, where $P_{r',r} = P_{r',r}^{\neq c} \cdot P_{r',r}^c$

For $m = 1 \dots T$:

For $l = 1 \dots R_{\psi}$:

$r' \leftarrow (m-1)R_{\psi} + l$

$\mathbf{g}^{(r',c)} \leftarrow \mathbf{h}_{m,c} \mathbf{f}^{(l,c)}$

For $r = 1 \dots R_{\psi}$:

$P_{r',r} \leftarrow P_{r',r}^{\neq c} \cdot \langle \mathbf{g}^{(r',c)}, \mathbf{f}^{(r,c)} \rangle$

discard $\mathbf{g}^{(r',c)}$

---

products, one r' at a time. $P_{r',r}^{\neq c}$ is calculated from $P_{r',r}$ by dividing out $P_{r',r}^c$ which in turn is made from $\mathbf{g}_{i_c}^{(r',c)}$. After the contribution from a particular value of r' has been added, $\mathbf{g}_{i_c}^{(r',c)}$ is discarded. It is only necessary to store $P_{r',r}$. Alg. 5 requires evaluating a 1D MVP for each r' . To update in Alg. 6, one needs $P_{r',r}^c$ for all r' , and because we do not store $\mathbf{g}_{i_c}^{(r',c)}$ for all r' , these must be calculated. That requires a second 1D MVP for each r' . In the initialization step of Alg. 4, $\mathbf{P}$ is computed by multiplying $\mathbf{P}^c$ for $c = 1 \dots D$ (Eq. 9(f)). This is accomplished without generating large-rank vectors by defining $\mathbf{P}_{r',r}^0 \equiv 1$ for all r', r , and calling Alg. 6 D times, once for each value of c .

The cost of Algs. 4–6 is $\mathcal{O}(D(n_c R_{\psi}^2 + n_c R_G R_{\psi} + TRn^2) + N_{\text{sweep}} D(2n_c R_{\psi}^2 + 3n_c R_G R_{\psi} + R_{\psi}^3 + 2TRn^2))$, where $R_G = TR_{\psi}$. This is less efficient than the ordinary intertwined power method in Alg. 3. In Alg. 3, only one MVP is computed for each α, c pair in step 2f. In Alg. 4, two MVPs are needed per α, c pair; they are evaluated in the calls to Alg. 5 and Alg. 6 in steps 2b and 2f, respectively. The factor of 2 in the term $2TRn^2$ above is due to the cost of the extra MVP. The memory cost of the low-memory intertwined power method is dominated by storing the $\mathbf{P}$ matrix, which costs $\mathcal{O}(TR_{\psi}^2)$. This is less than the full-memory version of the intertwined power method, which stores $\mathbf{P}$ in addition to the $\mathbf{G}_k$ vector.

Calculating $\mathbf{H}^{(\mathcal{F})} = \mathcal{F}^T \mathbf{H} \mathcal{F}$ to update vectors (step 3a in Alg. 1) also involves computing large-rank $\mathbf{G}_k$ vectors with rank TR_{ψ} . Not only is it expensive to store $\mathbf{G}_k = \mathbf{H} \mathbf{F}_k$ but it is also expensive to compute the inner products $\langle \mathbf{F}_{k'}, \mathbf{G}_k \rangle$ needed to obtain the elements of $\mathbf{H}^{(\mathcal{F})}$. To reduce the memory cost in this step, we use ALS (Alg. 2) but implement it without computing or storing large-rank vectors $\mathbf{G}_k$. To do this, we re-calculate $\mathbf{g}_{i_c}^{(r',c)}$, which requires storing $\mathbf{f}_{i_c}^{(r,c)}$ and doing 1D MVPs to update, downdate, and compute RHSs. The updating is done with Alg. 6, and the downdate and RHS calculation is done with Alg. 5. In Alg. 5, the RHSs are again obtained by (see Eq. (10)) summing terms one r' at a time.

We avoid also storing $\mathbf{G}_k$ created during the Gram-Schmidt and vector update phases (steps 2 and 3 in Alg. 1, respectively). The memory cost of storing these $\mathbf{G}_k$ vectors is lower than those obtained by evaluating MVPs because steps 2 and 3 in Alg. 1 generate vectors with a maximum rank of $\mathcal{B}R_{\psi}$, but it can still be significant if the block size is large. It is possible to implement ALS so that it is not necessary to store $\mathbf{G}_k$ vectors in order to compute ${}^{\text{out}}\mathbf{F}_k$. To perform a Gram-Schmidt

orthogonalization, we want

$${}^{\text{out}}\mathbf{F}_k \approx \mathbf{G}_k = {}^{\text{in}}\mathbf{F}_k - \sum_{k'=1}^{k-1} \langle {}^{\text{in}}\mathbf{F}_{k'}, {}^{\text{in}}\mathbf{F}_k \rangle \cdot {}^{\text{in}}\mathbf{F}_{k'}, \quad (15)$$

and to perform a vector update, we want

$${}^{\text{out}}\mathbf{F}_k \approx \mathbf{G}_k = \sum_{k'=1}^B \mathbf{U}_{k'k} {}^{\text{in}}\mathbf{F}_{k'}^{\text{old}}.$$

In both cases, we can make ${}^{\text{out}}\mathbf{F}_k$ without building $\mathbf{G}_k$. For a single c , to calculate ${}^{\text{out}}\mathbf{f}_k^{(r,c)}$, one needs $\mathbf{B}^{\neq c}$ and the RHS $\mathbf{b}^{(i_c,c)}$. The memory cost of computing $\mathbf{B}^{\neq c}$ is low because it does not require $\mathbf{g}_{i_c}^{(r',c)}$. To use Eq. (10) to compute $\mathbf{b}^{(i_c,c)}$, it is not necessary to store all $\mathbf{g}_{i_c}^{(r',c)}$ because contributions for each r' can be added to the sum and then discarded. The factors in the terms of the vector to be reduced are $\mathbf{f}_{i_c}^{(r',c)}$, that are already stored, multiplied by numerical coefficients. In the GS case, these coefficients are inner products of ${}^{\text{in}}\mathbf{F}_k$ vectors. In the vector update case, the coefficients are elements of $\mathbf{U}$. The coefficients are multiplied with the $\mathbf{f}_{i_c}^{(r',c)}$ factors for the first ($c = 1$) coordinate when we construct the RHS $\mathbf{b}^{(i_c,c)}$ vectors of the linear system.

III. RESULTS AND DISCUSSION

In this section, we report vibrational energy levels of two molecules computed using a combination of the intertwining idea of Sec. II C and the H-RRBPM¹⁸ that we refer to as the Hierarchical Intertwined Reduced-Rank Block Power Method (HI-RRBPM). All calculations described below use the fast version (Alg. 3) instead of the low-memory version (Alg. 4). We use the intertwined power method only for nodes in the tree with $D > 2$. For nodes with $D = 2$, we use the ordinary power method with singular value decomposition (SVD) to reduce the rank since SVD reduction is faster and more accurate than any ALS-type method. As in previous papers, we parallelize over vectors in the blocks, although other parallelization schemes are possible. Energy levels tend to decrease as more matrix-vector products are computed and in general increasing the rank decreases levels. However, as noted previously,²⁰ if the rank is small energy levels oscillate. As a consequence, it is not always true that every level decreases when the rank is increased.

The potential energy surfaces we use are “semi-diagonal” quartic Taylor expansions of the potential about the minimum energy geometry.³¹ This simple form of the potential is convenient to use, although our method is compatible with any sum-of-product potential. The Hamiltonian is

$$\begin{aligned} \hat{H} = & \frac{\omega_c}{2} \left(\sum_{c=1}^D -\frac{\partial^2}{\partial q_c^2} + q_c^2 \right) \\ & + \frac{1}{6} \sum_{c_1=1}^D \sum_{c_2=1}^D \sum_{c_3=1}^D \phi_{c_1 c_2 c_3}^{(3)} q_{c_1} q_{c_2} q_{c_3} \\ & + \frac{1}{24} \sum_{c_1=1}^D \sum_{c_2=1}^D \sum_{c_3=1}^D \sum_{c_4=1}^D \phi_{c_1 c_2 c_3 c_4}^{(4)} q_{c_1} q_{c_2} q_{c_3} q_{c_4}, \quad (16) \end{aligned}$$

where we neglect all $\pi^t \mu \pi$ terms in the kinetic energy operator (KEO) and the potential-like $\sum_{\alpha} \mu_{\alpha, \alpha}$ term.³² In our calculations, the 1D basis functions, $\varphi_{i_c}^c(q_c)$ in Eq. (5), are eigenfunctions of 1D cut Hamiltonians obtained by keeping only the $\frac{\omega_c}{2p_c^2}$ term in the KEO and setting $q_{c'} = 0, c' \neq c$. These eigenfunctions are obtained by solving each 1D cut Hamiltonian in a harmonic oscillator basis chosen large enough to converge the levels of interest.

A. Ethylene oxide (C_2H_4O)

We test the HI-RRBPM by computing the lowest 200 vibrational states of the ethylene oxide (EO) molecule (Fig. 1), which has 7 atoms and 15 vibrational degrees of freedom (DOF). As in our previous paper, we use the PES of Bégue, Gouhaud, and Pouchan³³ although newer potentials^{34–37} are available. The harmonic frequencies of the PES of Ref. 33 were computed at the CCSD(T)/cc-pVTZ level and the anharmonic constants at the B3LYP/6-31+G(d,p) level. We obtained the PES directly from one of the authors (PC-C). It includes eight force constants missing from the supplementary material of Ref. 33 but published in Ref. 38.

The trees used in the hierarchical calculations in this paper are shown in Figure 2. They are different from the tree used in Ref. 18. We obtained the tree in Figure 2(a) by adding a layer (third from the top) to the tree used earlier (Figure 11 in Ref. 18) and by increasing the basis sizes slightly on the second-to-highest layer. The new tree is structured so that $D = 2$ for all nodes except the top. This ensures that the intertwining algorithm is used only for the top-node calculation. The basis sizes on the second-to-highest layer (84/70/70/70 in Figure 2(a), cf. 64/64/64/64 in Ref. 18), were also modified so that the basis of each node includes complete polyads. For instance, the 70 states computed for the $q_6 - q_{13} - q_1 - q_9$ contracted node include all states with $v_6 + v_{v_{13}} + v_{v_1} + v_{v_9} \leq 4$ and no states with $v_6 + v_{v_{13}} + v_{v_1} + v_{v_9} > 4$. Since v_6, v_{13}, v_1 , and v_9 are similar, one finds a large energy gap between the 70th state, the last state in the $\sum v = 4$ polyad, and the 71st state, the first state in the $\sum v = 5$ polyad. Choosing block sizes in this way accelerates the convergence of the power method. The tree in Figure 2(b) was constructed with similar considerations, but larger basis sizes are used than in the tree in Figure 2(a). With the trees arranged in this manner, the top-node calculation accounts for virtually all of the CPU time. For example, in the EO calculation with rank $R_\psi = 240$, the entire

FIG. 1. Ethylene oxide.

FIG. 2. Trees used in calculations on ethylene oxide: (a) medium tree and (b) large tree. Numbers appearing at vertices are basis sizes; the placement of the primitive coordinates in the leaf nodes is shown at the bottom of the tree.

calculation takes ca. 3.5 days, all but 4 min of which is spent on the top node. The top node is the most expensive because most of the terms in the Hamiltonian are applied there. In Figure 2(b), each node in the third layer includes *all* states that can be computed from the direct-product basis of its sub-nodes in the second layer. For nodes in the third layer, the DP basis is small and it is therefore better, when one retains all states, to diagonalize the Hamiltonian matrix with a direct eigensolver than to use a power method.

When is it cheaper to build and diagonalize the matrix rather than to use a power method? If there are N_{DP} basis functions in the direct-product basis of a particular node, the total cost of computing basis functions by diagonalization is the cost of making the $N_{DP} \times N_{DP}$ matrix plus the cost of the diagonalization itself. The cost of diagonalizing the Hamiltonian matrix, $\mathcal{O}(N_{DP}^3)$, is much smaller than the cost of building the matrix. To construct the matrix, one must evaluate N_{DP} matrix-vector products. Computing each element of $\langle \mathbf{F}_{k'}, \mathbf{H} \mathbf{F}_k \rangle$, $k' = 1 \dots \mathcal{B}$, $k = 1 \dots \mathcal{B}$, requires evaluating an inner product between two CP-format vectors, $\mathbf{F}_{k'}$ and $\mathbf{G}_k = \mathbf{H} \mathbf{F}_k$, with ranks R_ψ and TR_ψ , respectively. We reduce the cost by using the ideas in Section II D to compute $\mathbf{H} \mathbf{F}_k$ and reduce its rank without storing the large-rank $\mathbf{G}_k$ vector. This procedure does introduce errors in the Hamiltonian matrix elements, but the errors are small and are no larger than the errors that one obtains from imposing a low rank on the vectors. Building the matrix requires N_{DP} MVPs. Contrast this with the cost of evaluating the $N_{cyc} N_{sweep} \mathcal{B}$ matrix-vector products required to use the power method to compute $\mathcal{B}$ desired states. Therefore if N_{DP} is less than $N_{cyc} N_{sweep} \mathcal{B}$, it is certainly faster to compute the full direct-product Hamiltonian matrix than to apply the RRBPM to $\mathcal{B}$ vectors. At the top of the tree, matrix diagonalization is out of the question: one would need 2.1×10^8 MVPs to construct the

TABLE I. Parameters for HI-RRBPM calculations on ethylene oxide.

Parameter	Value			
Calculation:	A	B	C	D
Tree	a	b	a	a
R_ψ	120	120	240	320
$N_{ALS;\psi}$ ^a				1
N_{cyc}				20
N_{sweep}				10

^aFor rank reductions in Gram-Schmidt, vector updates, and $\mathbf{H}^{(\mathcal{F})} = \mathcal{F}^T \mathbf{H} \mathcal{F}$ steps.

direct-product Hamiltonian whereas the power method only requires $20 \times 10 \times 200 = 40\,000$ MVPs. Moreover, the direct-product basis Hamiltonian matrix contains 4.5×10^{16} entries and is thus too large to store, and diagonalizing it would require $\mathcal{O}(10^{25})$ operations.

Parameters used in the calculations are summarized in Table I. We use the same parameters for all calculations except the rank, R_ψ , which we vary to converge the energy levels. Following the Gram-Schmidt orthogonalization and the vector update for the top node, the rank is reduced with ALS and N_{ALS} is set to 1 (lower nodes use SVD for all rank-reductions). For the top node, intertwining is used to reduce the rank after evaluating a MVP.

Energy levels computed using the HI-RRBPM are shown in Table II, where levels from the H-RRBPM “E” calculation of Ref. 18 and from a vibrational mean field configuration interaction (VMFCI)(11 300) calculation are shown for comparison.³⁹ The number in brackets after VMFCI is the cutoff wavenumber used to make the final basis.⁴⁰ The published VMFCI(10 721) results are in Ref. 33. In Table II, assignments are based on HI-RRBPM “D” wavefunctions while symmetry

TABLE II. Lowest 200 vibrational energy levels (cm^{-1}) computed for ethylene oxide using HI-RRBPM, in comparison to literature values. We report the zero-point energy (ZPE) and differences between other levels and the ZPE. Parameters specifying calculations “A,” “B,” “C,” and “D” are in Table I. Wall times in this work were obtained using AMD Opteron 6386 SE processors running at 2.8 GHz; the number of processors used is shown in parentheses next to the wall time for each calculation.

Sym	VMFCI	H-RRBPM	HI-RRBPM				Assignment
	(11 300)	E	A	B	C	D	
A_1	12 463.65	12 461.86	12 461.90	12 461.81	12 461.70	12 461.65	ZPE
B_2	793.37	793.29	793.33	793.12	793.14	793.10	ν_{15}
B_1	822.75	822.10	822.17	822.16	822.03	822.00	ν_{12}
A_1	879.04	878.40	878.45	878.47	878.36	878.33	ν_5
A_2	1 018.96	1 017.68	1 017.83	1 017.64	1 017.56	1 017.49	ν_8
A_1	1 122.96	1 122.11	1 122.25	1 121.80	1 121.99	1 121.94	ν_4
B_1	1 125.18	1 124.65	1 124.63	1 124.14	1 124.41	1 124.37	ν_{11}
B_2	1 147.09	1 146.44	1 146.38	1 146.13	1 146.22	1 146.19	ν_{14}
A_2	1 149.02	1 148.64	1 148.61	1 148.35	1 148.45	1 148.40	ν_7
A_1	1 271.50	1 271.12	1 271.25	1 271.24	1 270.98	1 270.94	ν_3
B_1	1 468.34	1 467.93	1 468.09	1 467.85	1 467.79	1 467.72	ν_{10}
A_1	1 496.65	1 496.23	1 496.35	1 496.15	1 495.85	1 495.74	ν_2
A_1	1 589.38	1 591.22	1 590.03	1 589.16	1 589.25	1 589.09	$2\nu_{15}$
A_2	1 612.92	1 612.46	1 612.25	1 612.00	1 611.76	1 611.63	$\nu_{15} + \nu_{12}$
A_1	1 642.86	1 641.65	1 641.69	1 641.69	1 641.32	1 641.21	$2\nu_{12}$
B_2	1 671.94	1 671.65	1 671.29	1 671.07	1 670.91	1 670.78	$\nu_{15} + \nu_5$
B_1	1 696.71	1 695.57	1 695.47	1 695.47	1 695.23	1 695.13	$\nu_5 + \nu_{12}$
A_1	1 756.21	1 755.22	1 755.18	1 755.15	1 754.87	1 754.81	$2\nu_5$
B_1	1 808.96	1 807.52	1 808.23	1 807.49	1 807.03	1 806.81	$\nu_{15} + \nu_8$
B_2	1 835.47	1 833.54	1 834.14	1 833.94	1 833.07	1 832.89	$\nu_{12} + \nu_8$
A_2	1 891.96	1 890.39	1 891.04	1 890.86	1 890.05	1 889.81	$\nu_5 + \nu_8$
A_2	1 910.19	1 910.06	1 911.14	1 910.37	1 909.24	1 908.90	$\nu_{15} + \nu_{11}$
B_2	1 912.38	1 911.78	1 912.84	1 911.85	1 911.11	1 910.80	$\nu_{15} + \nu_4$
B_1	1 930.21	1 930.35	1 931.35	1 930.71	1 929.51	1 929.15	$\nu_{15} + \nu_7$
A_1	1 940.01	1 939.87	1 940.55	1 940.15	1 939.24	1 938.92	$\nu_{15} + \nu_{14}$
B_1	1 941.82	1 941.55	1 942.28	1 941.72	1 940.88	1 940.58	$\nu_{12} + \nu_4$
A_1	1 945.80	1 945.56	1 945.99	1 945.49	1 944.95	1 944.71	$\nu_{12} + \nu_{11}$
A_2	1 963.62	1 962.73	1 962.99	1 962.66	1 962.26	1 962.04	$\nu_{12} + \nu_{14}$
B_2	1 971.83	1 971.11	1 971.26	1 970.94	1 970.76	1 970.60	$\nu_{12} + \nu_7$
A_1	1 999.34	1 998.13	1 998.58	1 997.94	1 997.86	1 997.73	$\nu_5 + \nu_4$
B_1	2 000.76	1 999.88	2 000.11	1 999.65	1 999.55	1 999.39	$\nu_5 + \nu_{11}$
B_2	2 022.57	2 021.59	2 021.81	2 021.41	2 021.29	2 021.12	$\nu_5 + \nu_{14}$
A_2	2 025.19	2 024.46	2 024.56	2 024.29	2 024.13	2 024.00	$\nu_5 + \nu_7$
A_1	2 036.08	2 033.45	2 033.56	2 032.63	2 032.91	2 032.78	$2\nu_8$
B_2	2 061.51	2 061.39	2 061.97	2 061.79	2 060.93	2 060.74	$\nu_{15} + \nu_3$
B_1	2 087.49	2 086.87	2 087.28	2 087.05	2 086.57	2 086.38	$\nu_{12} + \nu_3$

TABLE II. (*Continued.*)

Sym	VMFCI	H-RRBPM	HI-RRBPM				Assignment
	(11 300)	E	A	B	C	D	
A_2	2 132.05	2 129.74	2 130.64	2 129.78	2 129.14	2 128.77	$\nu_8 + \nu_4$
B_2	2 138.11	2 136.36	2 136.96	2 136.34	2 135.49	2 135.21	$\nu_8 + \nu_{11}$
A_1	2 141.80	2 140.94	2 141.17	2 141.17	2 140.71	2 140.60	$\nu_5 + \nu_3$
B_1	2 155.72	2 153.95	2 154.70	2 153.95	2 153.27	2 152.89	$\nu_8 + \nu_{14}$
A_1	2 167.73	2 166.12	2 166.66	2 166.11	2 165.60	2 165.40	$\nu_8 + \nu_7$
A_1	2 237.65	2 240.00	2 238.12	2 235.85	2 236.86	2 236.66	$2\nu_4$
B_1	2 243.80	2 245.63	2 245.45	2 243.34	2 244.27	2 243.99	$\nu_4 + \nu_{11}$
A_2	2 247.97	2 248.85	2 249.99	2 250.02	2 247.86	2 247.45	$\nu_{15} + \nu_{10}$
A_1	2 250.86	2 264.16	2 251.54	2 250.26	2 250.92	2 250.81	$2\nu_{11}$
B_2	2 266.28	2 267.58	2 268.59	2 267.68	2 266.63	2 266.13	$\nu_7 + \nu_{11}$
B_2	2 269.43	2 270.18	2 270.39	2 269.52	2 268.91	2 268.67	$\nu_{14} + \nu_4$
A_2	2 269.27	2 270.34	2 270.86	2 269.81	2 269.29	2 268.95	$\nu_{14} + \nu_{11}$
A_2	2 276.17	2 277.09	2 277.12	2 276.32	2 276.08	2 275.77	$\nu_7 + \nu_4$
B_2	2 284.40	2 285.89	2 286.69	2 285.50	2 284.94	2 284.58	$\nu_{15} + \nu_2$
A_1	2 289.57	2 289.13	2 289.51	2 289.46	2 288.71	2 288.50	$\nu_{12} + \nu_{10}$
A_2	2 291.52	2 291.40	2 292.08	2 291.43	2 290.94	2 290.67	$\nu_3 + \nu_8$
A_1	2 292.21	2 303.56	2 292.36	2 291.58	2 291.58	2 291.41	$2\nu_{14}$
B_1	2 294.73	2 304.48	2 295.00	2 294.04	2 294.18	2 294.02	$\nu_7 + \nu_{14}$
A_1	2 297.09	2 306.59	2 296.92	2 295.90	2 296.50	2 296.40	$2\nu_7$
B_1	2 312.71	2 312.47	2 312.60	2 312.37	2 311.57	2 311.26	$\nu_{12} + \nu_2$
B_1	2 346.58	2 346.02	2 346.22	2 346.06	2 345.73	2 345.49	$\nu_5 + \nu_{10}$
A_1	2 372.12	2 371.51	2 372.03	2 372.17	2 371.00	2 370.74	$\nu_5 + \nu_2$
B_2	2 382.86	2 390.19	2 390.15	2 383.58	2 388.33	2 388.00	$3\nu_{15}$
A_1	2 390.76	2 391.54	2 391.28	2 391.00	2 389.79	2 389.51	$\nu_3 + \nu_4$
B_1	2 392.13	2 391.88	2 392.34	2 391.99	2 391.25	2 391.01	$\nu_3 + \nu_{11}$
B_1	2 404.62	2 407.45	2 407.10	2 404.41	2 405.68	2 405.27	$2\nu_{15} + \nu_{12}$
B_2	2 416.22	2 415.92	2 416.42	2 415.91	2 415.31	2 415.12	$\nu_3 + \nu_{14}$
A_2	2 416.72	2 416.32	2 416.67	2 416.29	2 415.90	2 415.73	$\nu_3 + \nu_7$
B_2	2 430.50	2 430.45	2 430.83	2 429.49	2 429.54	2 429.22	$\nu_{15} + 2\nu_{12}$
B_1	2 460.42	2 459.56	2 462.70	2 458.49	2 460.17	2 458.83	$3\nu_{12}$
A_1	2 467.01	2 469.86	2 474.80	2 466.65	2 469.33	2 468.36	$2\nu_{15} + \nu_5$
B_2	2 483.52	2 482.81	2 483.24	2 482.63	2 482.29	2 482.05	$\nu_{10} + \nu_8$
A_2	2 486.14	2 485.88	2 486.99	2 485.04	2 485.10	2 484.74	$\nu_{15} + \nu_5 + \nu_{12}$
A_2	2 510.34	2 509.13	2 510.00	2 509.34	2 508.45	2 507.98	$\nu_2 + \nu_8$
A_1	2 512.47	2 512.05	2 516.28	2 510.75	2 515.86	2 513.46	$\nu_5 + 2\nu_{12}$
A_1	2 538.02	2 537.64	2 538.14	2 537.54	2 537.53	2 537.41	$2\nu_3$
B_2	2 548.58	2 548.67	2 550.79	2 547.35	2 548.93	2 547.55	$\nu_{15} + 2\nu_5$
B_1	2 569.28	2 568.81	2 569.65	2 567.58	2 568.29	2 567.33	$2\nu_5 + \nu_{12}$
B_1	2 585.75	2 585.35	2 586.12	2 585.36	2 584.71	2 584.32	$\nu_{10} + \nu_4$
A_1	2 589.46	2 589.29	2 590.03	2 589.20	2 588.64	2 588.24	$\nu_{10} + \nu_{11}$
A_2	2 599.54	2 602.32	2 603.66	2 603.79	2 597.83	2 597.09	$2\nu_{15} + \nu_8$
A_2	2 601.69	2 603.45	2 605.07	2 604.55	2 600.79	2 600.34	$\nu_{10} + \nu_{14}$
B_2	2 603.53	2 604.07	2 605.83	2 605.62	2 602.61	2 602.28	$\nu_{10} + \nu_7$
A_1	2 615.39	2 615.80	2 618.38	2 617.91	2 614.57	2 613.87	$\nu_2 + \nu_4$
B_1	2 618.85	2 618.52	2 619.63	2 619.49	2 617.66	2 617.02	$\nu_2 + \nu_{11}$
A_1	2 621.28	2 620.48	2 622.40	2 622.03	2 618.87	2 618.09	$\nu_{15} + \nu_{12} + \nu_8$
A_1	2 631.55	2 630.68	2 633.22	2 629.95	2 631.75	2 630.88	$3\nu_5$
A_2	2 643.15	2 643.17	2 644.20	2 643.53	2 642.36	2 641.75	$\nu_2 + \nu_7$
B_2	2 643.48	2 643.50	2 644.61	2 644.14	2 642.46	2 641.91	$\nu_2 + \nu_{14}$
A_2	2 650.27	2 647.84	2 649.26	2 649.22	2 646.86	2 646.24	$2\nu_{12} + \nu_8$
B_1	2 680.29	2 679.77	2 682.25	2 680.35	2 678.79	2 677.43	$\nu_{15} + \nu_5 + \nu_8$
B_1	2 697.64	2 700.08	2 705.69	2 701.16	2 697.39	2 696.02	$2\nu_{15} + \nu_{11}$
B_2	2 704.29	2 702.16	2 706.41	2 703.29	2 702.57	2 701.38	$\nu_5 + \nu_{12} + \nu_8$
A_1	2 703.48	2 705.90	2 711.34	2 705.45	2 703.02	2 701.69	$2\nu_{15} + \nu_4$
A_2	2 715.21	2 717.83	2 723.95	2 718.84	2 714.95	2 713.94	$2\nu_{15} + \nu_7$
B_2	2 725.03	2 725.32	2 728.24	2 726.45	2 725.85	2 723.68	$\nu_{15} + \nu_{12} + \nu_{11}$
A_2	2 731.66	2 733.76	2 736.98	2 740.74	2 732.82	2 730.67	$\nu_{15} + \nu_{12} + \nu_4$
B_2	2 737.56	2 740.10	2 739.17	2 742.98	2 738.47	2 736.72	$2\nu_{15} + \nu_{14}$

TABLE II. (*Continued.*)

Sym	VMFCI	H-RRBPM	HI-RRBPM				Assignment
	(11 300)	E	A	B	C	D	
B_1	2737.96	2741.18	2742.30	2743.73	2738.73	2737.37	$\nu_{10} + \nu_3$
A_1	2747.23	2747.36	2750.19	2748.31	2745.92	2744.96	$\nu_{15} + \nu_{12} + \nu_7$
B_1	2754.35	2754.49	2756.90	2755.04	2754.29	2752.71	$\nu_{15} + \nu_{12} + \nu_{14}$
A_1	2760.02	2760.53	2764.07	2766.04	2759.13	2758.29	$2\nu_{12} + \nu_4$
A_2	2763.83	2762.10	2766.10	2768.07	2762.02	2761.35	$2\nu_5 + \nu_8$
B_1	2763.28	2762.62	2768.07	2769.23	2762.23	2761.37	$2\nu_{12} + \nu_{11}$
A_1	2764.81	2767.68	2769.24	2771.66	2764.35	2763.88	$\nu_2 + \nu_3$
B_2	2778.08	2776.81	2780.87	2777.59	2777.14	2775.59	$2\nu_{12} + \nu_{14}$
A_2	2786.85	2786.84	2790.83	2788.64	2786.73	2787.72	$\nu_{15} + \nu_5 + \nu_{11}$
B_2	2788.85	2788.26	2796.66	2789.21	2788.06	2788.41	$\nu_{15} + \nu_5 + \nu_4$
A_2	2792.07	2790.87	2797.51	2791.27	2791.19	2790.40	$2\nu_{12} + \nu_7$
B_1	2805.89	2805.94	2809.71	2806.50	2807.43	2809.98	$\nu_{15} + \nu_5 + \nu_7$
A_1	2813.86	2813.77	2819.22	2814.90	2813.63	2813.50	$\nu_5 + \nu_{12} + \nu_{11}$
A_1	2818.25	2815.50	2825.73	2816.01	2818.17	2817.12	$\nu_{15} + \nu_5 + \nu_{14}$
B_2	2822.93	2818.40	2827.82	2823.13	2819.09	2817.46	$\nu_{15} + 2\nu_8$
B_1	2815.20	2819.35	2829.39	2825.22	2819.29	2817.70	$\nu_5 + \nu_{12} + \nu_4$
A_2	2834.86	2833.72	2835.65	2833.99	2836.82	2834.71	$\nu_5 + \nu_{12} + \nu_{14}$
B_2	2843.35	2842.21	2842.55	2842.08	2842.96	2842.15	$\nu_5 + \nu_{12} + \nu_7$
B_1	2847.79	2843.25	2846.56	2843.66	2843.22	2842.50	$\nu_{12} + 2\nu_8$
A_1	2854.33	2857.47	2860.43	2855.60	2854.13	2853.47	$2\nu_{15} + \nu_3$
A_1	2874.30	2873.13	2873.70	2872.71	2872.86	2873.05	$2\nu_5 + \nu_4$
B_1	2874.81	2873.68	2876.64	2873.62	2874.94	2873.88	$2\nu_5 + \nu_{11}$
A_2	2874.44	2874.70	2881.14	2875.07	2877.18	2874.54	$\nu_{15} + \nu_{12} + \nu_3$
B_2	2896.62	2895.28	2896.07	2895.14	2896.86	2895.89	$2\nu_5 + \nu_{14}$
A_2	2899.84	2898.63	2899.91	2898.61	2899.15	2898.62	$2\nu_5 + \nu_7$
A_1	2905.43	2902.37	2909.54	2905.60	2903.43	2901.60	$\nu_5 + 2\nu_8$
A_1	2902.67	2903.39	2910.83	2906.85	2904.43	2902.18	$2\nu_{12} + \nu_3$
B_1	2910.86	2912.48	2914.20	2911.25	2911.59	2911.04	ν_9
A_1	2924.52	2920.93	2923.38	2919.09	2920.15	2917.53	$\nu_{15} + \nu_8 + \nu_{11}$
B_1	2922.14	2921.36	2926.79	2922.83	2920.65	2917.90	$\nu_{15} + \nu_8 + \nu_4$
A_1	2916.69	2926.67	2935.38	2934.52	2924.13	2923.02	ν_1
B_2	2931.82	2932.42	2935.86	2936.10	2934.55	2932.84	$\nu_{15} + \nu_5 + \nu_3$
B_2	2941.73	2940.11	2950.85	2943.66	2941.12	2937.33	$\nu_{12} + \nu_8 + \nu_4$
A_2	2946.78	2945.96	2955.31	2951.28	2945.16	2942.52	$\nu_{15} + \nu_8 + \nu_{14}$
B_2	2951.93	2951.19	2962.41	2955.61	2952.43	2949.06	$\nu_{15} + \nu_8 + \nu_7$
A_2	2954.90	2953.39	2965.66	2957.53	2955.04	2951.56	$\nu_{12} + \nu_8 + \nu_{11}$
B_1	2954.55	2954.67	2969.82	2961.09	2956.24	2954.44	$\nu_5 + \nu_{12} + \nu_3$
A_1	2956.38	2960.45	2978.82	2961.70	2959.84	2958.32	$2\nu_{10}$
A_1	2971.25	2970.19	2983.78	2973.70	2968.63	2967.45	$\nu_{12} + \nu_8 + \nu_{14}$
B_1	2985.81	2982.90	2985.47	2983.69	2983.87	2981.94	$\nu_{12} + \nu_8 + \nu_7$
B_1	2993.97	2998.85	3000.54	2997.51	2996.59	2995.63	$\nu_2 + \nu_{10}$
A_2	3005.26	3001.73	3013.90	3003.92	3001.69	3000.44	$\nu_5 + \nu_8 + \nu_4$
A_1	3000.08	3005.90	3018.92	3004.68	3003.65	3002.34	$2\nu_2$
B_2	3009.46	3007.93	3028.11	3016.14	3009.29	3006.30	$\nu_5 + \nu_8 + \nu_{11}$
A_1	3011.16	3011.64	3032.55	3019.18	3010.24	3009.47	$2\nu_5 + \nu_3$
B_2	3020.67	3025.57	3033.22	3024.71	3022.80	3020.05	$\nu_{15} + 2\nu_{11}$ ^a
B_1	3028.81	3026.66	3035.05	3028.56	3026.25	3024.79	$\nu_5 + \nu_8 + \nu_{14}$
A_2	3027.65	3029.99	3042.77	3030.02	3026.98	3025.44	$\nu_{15} + \nu_4 + \nu_{11}$
A_2	3029.75	3031.89	3045.47	3032.13	3028.75	3027.95	ν_6
B_2	3035.82	3037.35	3047.57	3035.91	3037.58	3033.86	$\nu_{15} + 2\nu_4$ ^a
A_1	3039.04	3041.79	3049.07	3039.54	3039.05	3035.70	$\nu_5 + \nu_8 + \nu_7$
B_1	3038.05	3043.43	3052.84	3046.25	3040.13	3038.36	$2\nu_{15} + \nu_{10}$
B_2	3041.99	3046.18	3056.30	3046.47	3041.32	3040.21	ν_{13}
B_1	3049.21	3052.14	3059.30	3048.02	3049.66	3048.42	$\nu_{12} + 2\nu_{11}$ ^b
A_1	3047.88	3054.71	3061.21	3050.41	3050.54	3048.70	$\nu_{15} + \nu_7 + \nu_{11}$
A_2	3055.15	3054.90	3069.00	3055.09	3052.21	3048.87	$3\nu_8$
B_1	3055.92	3059.55	3070.07	3059.59	3058.91	3057.21	$\nu_{15} + \nu_{14} + \nu_{11}$
A_1	3057.68	3061.32	3071.45	3061.04	3061.71	3060.39	$\nu_{12} + \nu_4 + \nu_{11}$

TABLE II. (Continued.)

Sym	VMFCI	H-RRBPM	HI-RRBPM				Assignment
	(11 300)	E	A	B	C	D	
A_1	3 061.06	3 065.37	3 077.15	3 069.47	3 066.80	3 061.69	$\nu_{15} + \nu_{14} + \nu_4$
B_1	3 064.98	3 069.01	3 079.92	3 074.17	3 069.46	3 066.08	$\nu_{15} + \nu_7 + \nu_4$
B_2	3 066.76	3 069.52	3 081.24	3 076.92	3 071.96	3 066.94	$\nu_{15} + \nu_{12} + \nu_{10}$
B_1	3 069.49	3 079.01	3 085.87	3 077.25	3 073.61	3 069.95	$\nu_{12} + 2\nu_4^b$
A_1	3 068.03	3 079.80	3 087.90	3 079.26	3 075.33	3 071.18	$2\nu_{15} + \nu_2$
B_2	3 073.86	3 082.54	3 091.53	3 084.88	3 075.83	3 072.99	$\nu_{15} + 2\nu_{14}^c$
A_2	3 073.86	3 084.47	3 093.22	3 086.12	3 079.16	3 074.73	$\nu_{12} + \nu_{14} + \nu_4^d$
B_1	3 078.94	3 084.85	3 096.10	3 090.57	3 084.71	3 079.51	$\nu_{15} + \nu_3 + \nu_8$
B_2	3 084.92	3 087.10	3 100.08	3 090.96	3 087.53	3 084.54	$\nu_{12} + \nu_{14} + \nu_{11}$
A_2	3 084.02	3 089.09	3 101.81	3 094.57	3 088.76	3 085.65	$\nu_{15} + \nu_{12} + \nu_2$
A_2	3 087.97	3 095.81	3 109.45	3 096.20	3 091.32	3 088.11	$\nu_{15} + \nu_7 + \nu_{14}$
B_2	3 089.06	3 099.08	3 110.45	3 098.78	3 095.50	3 088.58	$\nu_{15} + 2\nu_7^c$
B_2	3 096.23	3 100.35	3 114.11	3 102.13	3 103.41	3 098.61	$\nu_{12} + \nu_7 + \nu_4$
A_2	3 102.56	3 105.14	3 120.94	3 104.78	3 105.01	3 102.95	$\nu_{12} + \nu_7 + \nu_{11}$
B_2	3 104.44	3 105.55	3 125.34	3 106.60	3 107.51	3 103.16	$\nu_{12} + \nu_3 + \nu_8$
B_1	3 105.97	3 108.65	3 127.67	3 106.91	3 110.65	3 104.30	$\nu_{12} + 2\nu_{14}$
B_1	3 109.55	3 114.93	3 128.20	3 110.84	3 111.75	3 109.15	$2\nu_{12} + \nu_{10}$
A_1	3 116.35	3 116.00	3 129.81	3 114.59	3 115.43	3 112.80	$\nu_5 + 2\nu_{11}^e$
A_1	3 112.92	3 120.45	3 131.77	3 115.53	3 120.70	3 115.58	$\nu_{12} + \nu_7 + \nu_{14}$
B_1	3 118.38	3 122.67	3 132.70	3 120.44	3 122.67	3 119.83	$\nu_5 + \nu_4 + \nu_{11}$
B_1	3 122.34	3 127.29	3 136.78	3 123.52	3 124.84	3 121.56	$\nu_{12} + 2\nu_7$
A_1	3 127.59	3 129.90	3 139.27	3 124.07	3 131.31	3 125.67	$\nu_5 + 2\nu_4^e$
A_2	3 126.44	3 130.56	3 140.66	3 130.28	3 132.82	3 125.96	$\nu_{15} + \nu_5 + \nu_{10}$
A_1	3 129.22	3 139.09	3 151.45	3 132.42	3 136.56	3 127.61	$2\nu_{12} + \nu_2$
A_1	3 146.79	3 139.62	3 152.82	3 139.95	3 142.00	3 134.62	$2\nu_8 + \nu_4$
B_2	3 143.48	3 142.25	3 155.87	3 143.02	3 143.03	3 141.97	$\nu_5 + \nu_7 + \nu_{11}$
B_2	3 146.04	3 143.82	3 156.69	3 145.39	3 145.07	3 142.77	$\nu_5 + \nu_{14} + \nu_4$
A_2	3 143.09	3 145.01	3 162.41	3 147.67	3 153.11	3 143.98	$\nu_5 + \nu_{14} + \nu_{11}$
B_1	3 156.68	3 146.79	3 164.03	3 149.10	3 153.84	3 146.30	$2\nu_8 + \nu_{11}$
A_2	3 153.20	3 152.25	3 174.43	3 152.83	3 161.45	3 154.15	$\nu_5 + \nu_7 + \nu_4$
A_2	3 159.94	3 158.81	3 175.98	3 160.48	3 161.79	3 160.67	$\nu_5 + \nu_3 + \nu_8$
B_2	3 160.91	3 162.78	3 176.97	3 165.42	3 164.62	3 162.47	$\nu_{15} + \nu_5 + \nu_2$
B_2	3 170.99	3 163.26	3 178.24	3 168.26	3 166.98	3 163.50	$2\nu_8 + \nu_{14}$
A_1	3 163.64	3 164.91	3 182.14	3 168.38	3 167.78	3 163.70	$\nu_5 + \nu_{12} + \nu_{10}$
A_1	3 168.19	3 176.05	3 185.42	3 171.97	3 169.85	3 164.77	$\nu_5 + 2\nu_{14}$
B_1	3 170.35	3 177.70	3 199.02	3 174.72	3 172.75	3 169.44	$\nu_5 + \nu_7 + \nu_{14}$
A_1	3 173.84	3 179.74	3 200.53	3 179.51	3 180.75	3 171.54	$\nu_5 + 2\nu_7$
A_1	3 174.28						$4\nu_{15}$
A_2	3 175.82	3 180.48	3 202.69	3 181.47	3 180.89	3 175.51	$\nu_{15} + \nu_3 + \nu_{11}$
B_2	3 179.58	3 181.53	3 204.69	3 186.16	3 184.78	3 179.51	$\nu_{15} + \nu_3 + \nu_4$
A_2	3 190.21	3 181.70	3 211.72	3 190.70	3 186.92	3 182.09	$2\nu_8 + \nu_7$
B_1	3 186.27	3 186.81	3 213.07	3 192.41	3 190.88	3 189.11	$\nu_5 + \nu_{12} + \nu_2$
A_2	3 195.37						$3\nu_{15} + \nu_{12}$
B_1	3 197.44	3 197.59	3 216.72	3 203.60	3 200.97	3 198.95	$\nu_{15} + \nu_3 + \nu_7$
A_1	3 206.44	3 205.15	3 218.86	3 207.75	3 207.33	3 205.73	$\nu_{12} + \nu_3 + \nu_{11}$
B_1	3 206.78	3 205.96	3 224.78	3 210.92	3 208.70	3 206.84	$\nu_{12} + \nu_3 + \nu_4$
A_1	3 210.84	3 210.19	3 226.41	3 212.78	3 212.35	3 209.19	$\nu_{15} + \nu_3 + \nu_{14}$
B_1	3 223.58	3 222.74	3 237.39	3 222.90	3 228.31	3 225.03	$2\nu_5 + \nu_{10}$
A_2	3 229.36	3 226.46	3 245.47	3 227.28	3 235.14	3 227.09	$\nu_{12} + \nu_3 + \nu_{14}$
B_2	3 235.64	3 232.63	3 251.72	3 232.82	3 241.80	3 235.46	$\nu_{12} + \nu_3 + \nu_7$
	512	6.1	3.2	3.9	9.2	14.6	Memory (GB)
	4.6 d (32) ^f	14 d (63)	12 h (32)	53 h (28)	3.5 d (32)	8.7 d (32)	CPU time

^aBoth levels assigned to $\nu_{15} + 2\nu_4$ in VMFCI.^bThese assignments are inverted in VMFCI relative to the HI-RRBPM assignment.^cThese assignments are inverted in VMFCI relative to the HI-RRBPM assignment.^dAssigned to $\nu_{15} + \nu_7 + \nu_{14}$ in VMFCI.^eThese assignments are inverted in VMFCI relative to the HI-RRBPM assignment.^fVMFCI calculations performed on Intel E5-4650L processors (2.60 GHz).

labels are taken from the VMFCI calculation. In the table, the HI-RRBPM “D” levels are listed in the ascending order and assigned by examining the wavefunctions. The VMFCI levels are ordered so that their assignments match with those in the last column. The HI-RRBPM “A”–“C” levels are sorted in the ascending order without attempting to assign them, although most of these levels can be assigned by inspection.

Comparing the HI-RRBPM “A,” “C,” and “D” levels, one sees that increasing the rank improves the levels, with lower levels decreasing by $<1\text{ cm}^{-1}$ and higher levels decreasing by as much as 30 cm^{-1} as the rank is increased from $R_\psi = 120$ to $R_\psi = 320$. The HI-RRBPM “A” levels with $R_\psi = 120$ are worse than those of the earlier H-RRBPM “E” calculation with the same rank.¹⁸ This is due to the poorer quality of the cheaper optimization used to reduce the rank in the HI-RRBPM “A” calculation. In the H-RRBPM “E” calculation, 100 ALS iterations were used for every rank reduction, and it is therefore not surprising that the H-RRBPM “E” levels are more accurate. The HI-RRBPM calculation uses the equivalent of *one* ALS iteration per rank reduction. Since HI-RRBPM is much faster, one can compensate for the poorer optimization by choosing larger ranks. Even the HI-RRBPM “D” calculation, with $R_\psi = 320$, requires less time (8.7 days on 32 processors) and gives more accurate levels than H-RRBPM “E,” which required 14 days on 64 processors. Rank-reduction was the rate-limiting step in the earlier H-RRBPM calculations on ethylene oxide, whereas computing the inner products required for the vector updates (step 3a in Alg. 1) is rate-limiting for the HI-RRBPM calculations. Comparing HI-RRBPM “A” and “B,” one sees that enlarging the basis also improves the levels, with higher levels generally improving more than lower ones. The improvement in energy levels obtained by enlarging the basis is less significant than that obtained by increasing the rank, indicating that the levels are mostly well-converged with respect to the basis. Increasing the rank from 240 to 320 (calculations “C” and “D”) decreases the higher energies more because intrinsic coupling is more important for higher states.

All HI-RRBPM levels up to 2200 cm^{-1} above the HI-RRBPM zero-point energy (ZPE) are lower than the corresponding VMFCI levels. At energies $>3000\text{ cm}^{-1}$ above the ZPE, however, many VMFCI levels are lower than those from

FIG. 3. Cyclopentadiene.

the HI-RRBPM “A”–“C” calculations. Only ten of the HI-RRBPM “D” levels are higher than VMFCI, most notably the ν_1 fundamental. HI-RRBPM and VMFCI are similar in that they are both variational, which means that energy levels decrease as the number of basis functions increases. However, HI-RRBPM also exploits the low-rank structure of the wavefunctions to keep the memory cost low; as a result, only $<15\text{ GB}$ is required by our largest (“D”) calculation. In comparison, VMFCI(11 300) requires 512 GB. With the low memory version, we need only 5.3 GB for the “D” calculation. We also note that VMFCI(11 300) computed a larger block of 236 states although only 200 states are listed in Table II.

B. Cyclopentadiene (C_5H_6)

In Subsection III A, we demonstrate that with the HI-RRBPM it is possible to compute energy levels for a seven-atom molecule to ca. 1 cm^{-1} precision. Is it possible to compute accurate vibrational spectra for larger molecules? With the original RRBPM, it is possible to compute levels of a 6-atom molecule, and with the H-RRBPM, calculations are possible for a 7-atom molecule. Further improvements ought to make it possible to tackle larger molecules. Indeed, with the HI-RRBPM, it is possible to compute the lowest 192 levels of the cyclopentadiene molecule (CPD, Figure 3), which has 11 atoms and 27 vibrational degrees of freedom. This includes

FIG. 4. Tree used in calculations on cyclopentadiene. Numbers appearing at vertices are basis sizes; the placement of the primitive coordinates in the leaf nodes is shown at the bottom of the tree.

all fundamental levels except for those of the C-H stretch vibrations. We obtained the force field from the supplementary material of Ref. 41. This PES is a quartic force field with 27 harmonic constants and 771 and 1124 cubic and quartic anharmonic constants, respectively. Including kinetic energy terms, the Hamiltonian has the form of Eq. (16) and has 1949 terms.

Figure 4 depicts the tree used in the CPD HI-RRBPM calculations. In lower layers, we group into node coordinates with similar frequencies and types of motion. Although better trees are probably possible, we have not attempted to improve the tree. As with CH₃CN and EO, grouping modes by the frequency/type-of-motion is intuitive and does not preclude computing accurate energy levels. We also constructed bases in higher layers of the tree in Figure 4 with the same strategies used for the EO trees: we diagonalize and truncate instead of using the power method when the direct-product basis is small, and we combine modes in groups-of-two until we reach the second-to-highest layer. The final contraction combines six nodes, each containing between 3 and 6 coordinates.

We performed six HI-RRBPM calculations with R_ψ values between 60 and 360. Parameters for the calculations are given in Table III. Calculations “A,” “E,” and “F” were done sequentially. Since the top-node calculation is expensive when the rank is large, (cf. “E” and “F”) it is useful to begin the top-node calculation using guess vectors optimized in a less expensive calculation. We therefore generated guess vectors for the top-node “E” calculation by adding random terms with small coefficients to the final vectors from the “A” calculation. We generated guess vectors for the top-node of “F” from the

TABLE III. Parameters for HI-RRBPM calculations on cyclopentadiene.

Parameter	A	B	C	D	E	F
R_ψ	60	100	150	240	300	360
N_{CPU}	32 ^a	32 ^b	32 ^b	64 ^b	32 ^a	32 ^a
$N_{ALS;\psi}$ ^c	1	1	1	1	1	1
N_{cyc}	20	20	20	20	20 ^d	10 ^e
N_{sweep}	10	10	10	10	10	10
Memory (GB)	9.7	15.9	24.6	83.9	56.9	72.4
Wall time	43.0 h	32.2 d	61.1 d	50.2 d	31.3 d	22.8 d

^aIntel Xeon E5-2670 0 processors running at 2.6 GHz.

^bAMD Opteron 6386 SE processors running at 2.8 GHz.

^cFor rank reductions in Gram-Schmidt, vector updates, and $\mathbf{H}^{(\mathcal{F})} = \mathcal{F}^T \mathbf{H} \mathcal{F}$ steps.

^dContinuation of “A” calculation with a larger rank.

^eContinuation of “E” calculation with a larger rank.

final vectors from “E” in a similar manner. Instead of recomputing the basis in lower layers of the tree, the “E” and “F” calculations simply reuse the basis computed in “A.” Although the “A” calculation used a smaller rank ($R_\psi = 60$), this value is large enough to compute a good basis for all nodes below the top. In the “A,” “B,” “C,” and “D” calculations, every node has the same R_ψ .

Fundamental vibrational levels of CPD are shown in Table IV. Other levels are in the [supplementary material](#). As one increases the rank R_ψ from 60 to 360, all HI-RRBPM levels progressively decrease. As expected, larger ranks are required to converge higher levels. The change in energies between the “E” and “F” columns is a rough indicator of convergence and ranges from 0.3 to 1.5 cm⁻¹. On this basis,

TABLE IV. Fundamental vibrational levels (cm⁻¹) computed for cyclopentadiene using HI-RRBPM, in comparison to VPT2 values.

Sym	VPT2	HI-RRBPM						Assignment
		A	B	C	D	E	F	
A ₁		19911.13	19907.80	19906.52	19904.74	19904.28	19903.98	ZPE
B ₁	342	318.55	317.19	315.50	314.00	313.95	313.64	ν_{19}
A ₂	516	509.48	508.06	507.48	506.50	506.41	506.25	ν_{14}
B ₁	666	654.48	653.09	650.44	648.21	647.87	647.58	ν_{18}
A ₂	701	688.11	686.26	684.91	684.21	684.00	683.81	ν_{13}
B ₂	802	802.47	800.48	800.02	798.94	798.61	798.32	ν_{27}
A ₁	802	805.68	802.35	802.07	800.72	800.59	800.38	ν_{10}
B ₁	888	877.30	874.93	872.79	870.78	870.41	870.07	ν_{17}
A ₁	906	909.50	907.21	907.12	905.87	905.79	905.43	ν_9
B ₁	933	924.11	919.85	922.24	918.03	918.15	917.59	ν_{16}
A ₂	932	925.85	921.89	921.74	918.31	918.88	919.33	ν_{12}
B ₂	949	948.21	945.21	944.12	943.35	942.84	942.51	ν_{26}
A ₁	991	1005.74	996.10	994.20	992.81	992.00	991.61	ν_8
A ₂	1088	1084.87	1083.21	1080.38	1078.48	1078.19	1077.84	ν_{11}
B ₂	1093	1091.48	1089.46	1088.58	1086.18	1085.89	1085.41	ν_{25}
A ₁	1112	1112.64	1110.12	1108.80	1106.84	1106.88	1106.48	ν_7
B ₂	1221	1227.81	1221.25	1219.36	1214.97	1213.94	1213.06	ν_{24}
B ₂	1285	1288.22	1283.87	1282.35	1280.77	1283.69	1283.08	ν_{23}
A ₁	1361	1363.22	1355.17	1350.45	1347.37	1346.65	1345.91	ν_6
A ₁	1371	1375.40	1370.90	1368.86	1366.29	1366.00	1365.49	ν_5
A ₁	1501	1523.47	1521.22	1503.45	1499.54	1499.19	1498.45	ν_4
B ₂	1592	1659.00	1615.76	1601.14	1592.35	1591.29	1590.09	ν_{22}

we estimate that lower levels are converged to within ~ 2 or 3 cm^{-1} , and higher levels are converged to within $\sim 5\text{--}10\text{ cm}^{-1}$. The fact that the “F” wavenumber of ν_{12} is slightly larger than its “E” counterpart is an indication of noise resulting from imperfect rank reduction. The noise decreases as the rank is increased.

To the best of our knowledge, there are no variational vibrational level calculations on CPD with which to compare our results. At best, one can compare with the fundamental levels of the second-order vibrational perturbation theory (VPT2) calculation in Ref. 41. In our largest (“F”) calculation, all fundamentals except for ν_8 are lower than the VPT2 values. A few fundamentals ($\nu_{10}, \nu_9, \nu_8, \nu_{23}, \nu_{22}$) are within 2 cm^{-1} of the VPT2 values, while several others ($\nu_{19}, \nu_{18}, \nu_{13}, \nu_{17}, \nu_{16}, \nu_{12}, \nu_{11}, \nu_6$) are lower than the VPT2 values by more than 10 cm^{-1} . We cannot compare our zero-point energy (ZPE) with the VPT2 ZPE because it is unknown.

The memory cost of the largest (“F”) calculation is $\sim 72\text{ GB}$. It is large because we are parallelizing over 32 processors and simultaneously storing the large-rank vectors obtained after evaluating MVPs and not using the low-memory version of the intertwining idea. One could decrease the memory cost by parallelizing over fewer processors or by using the low-memory version of intertwining (Alg. 4). If one uses the low-memory approaches to avoid storing *all* large-rank vectors, but parallelizes over 32 processors then storing $\mathbf{P}$ matrices used to calculate RHSs in Algs. 4–6 and to compute $\mathcal{F}^T \mathbf{H} \mathcal{F}$ steps becomes memory-limiting, and the memory cost is reduced to ca. 16 GB .

IV. CONCLUSION

Methods that build a basis by evaluating matrix-vector products and storing basis vectors in a tensor format drastically reduce the memory required to compute vibrational spectra.^{17–20,28} They enable one to use a direct product basis without dealing with huge matrices or vectors. When using an iterative eigensolver and a direct product basis to compute a spectrum, it is common to store vectors (at least two) with n^D components. Although iterative eigensolvers do obviate the need to store and compute huge Hamiltonian matrices, their usefulness is limited if one needs to store vectors with n^D components in memory. If n , the number of basis functions for a single coordinate, is 10 and $D = 12$, it is not possible to store the vectors in the memory of most modern computers. The memory cost scales exponentially with D . When the Hamiltonian is a SOP, it is possible, using tensor methods,^{17–20,28} to make a basis whose vectors are all in the CP format so that the memory cost is only $\mathcal{O}(RnD)$; it scales *linearly* with D . Such methods would be simple, but for the fact that R , the rank of the CP tensor, increases rapidly as the basis is generated. It is therefore imperative that the rank of basis vectors be reduced. Reducing the rank of basis vectors is costly and the cost is larger when the number of terms in the Hamiltonian is larger. In previous calculations, $>90\%$ of the CPU time is used for rank reduction. In this paper, we reduce the cost of the rank reduction by about an order of magnitude. This makes calculations possible for a molecule with 11 atoms.

To reduce the cost, we use a partial optimization method. This means that we introduce errors into the basis vectors. The error is small enough that it is still possible to obtain a good basis by applying powers of the shifted Hamiltonian. The partial optimization is implemented by (1) intertwining the evaluation of matrix-vector products and the optimization of a single factor, in all of the terms, and (2) replacing each factor only once per sweep. We have demonstrated that with these ideas it is possible to compute vibrational energy levels of cyclopentadiene. When we parallelize the calculation, we increase the memory cost because we must store a $\mathbf{P}$ matrix, the number of elements of which is the product of the rank of the vector being reduced and the target rank, on each processor. Despite this increased memory cost, using 32 processors, the cyclopentadiene calculation requires only 16 GB . The intertwining idea therefore makes it possible to compute spectra of molecules with a sum-of-product PES and as many as 11 atoms on a modern desktop computer.

SUPPLEMENTARY MATERIAL

See [supplementary material](#) for the full list of the levels of cyclopentadiene.

ACKNOWLEDGMENTS

This work was supported by the Natural Sciences and Engineering Research Council of Canada. Calculations were done on computers purchased with money from the Canada Foundation for Innovation.

- ¹G. D. Carney, L. L. Sprandel, and C. W. Kern, *Adv. Chem. Phys.* **37**, 305 (1978).
- ²J. Tennyson, *Comput. Phys. Rep.* **4**, 1 (1986).
- ³S. Carter and N. Handy, *Comput. Phys. Commun.* **51**, 49 (1988).
- ⁴A. Jackle and H.-D. Meyer, *J. Chem. Phys.* **104**, 7974 (1996).
- ⁵S. Manzhos and T. Carrington, *J. Chem. Phys.* **129**, 224104 (2008).
- ⁶G. Avila and T. Carrington, Jr., *J. Chem. Phys.* **143**, 044106 (2015).
- ⁷E. Matyus, G. Czako, B. T. Sutcliffe, and A. G. Csaszar, *J. Chem. Phys.* **127**, 084102 (2007).
- ⁸C. Iung and C. Leforestier, *J. Chem. Phys.* **102**, 8453 (1995).
- ⁹R. Chen, G. Ma, and H. Guo, *J. Chem. Phys.* **114**, 4763 (2001).
- ¹⁰M. J. Bramley and T. Carrington, Jr., *J. Chem. Phys.* **99**, 8519 (1993).
- ¹¹H.-G. Yu and J. T. Muckerman, *J. Mol. Spectrosc.* **214**, 11 (2002).
- ¹²S. W. Huang and T. Carrington, *Chem. Phys. Lett.* **312**, 311 (1999).
- ¹³X.-G. Wang and T. Carrington, *J. Phys. Chem. A* **105**, 2575 (2001).
- ¹⁴G. Avila and T. Carrington, *J. Chem. Phys.* **131**, 174103 (2009).
- ¹⁵G. Avila and T. Carrington, Jr., *J. Chem. Phys.* **134**, 054126 (2011).
- ¹⁶P. Seidler, M. B. Hansen, and O. Christiansen, *J. Chem. Phys.* **128**, 154113 (2008).
- ¹⁷A. Leclerc and T. Carrington, *J. Chem. Phys.* **140**, 174111 (2014).
- ¹⁸P. S. Thomas and T. Carrington, Jr., *J. Phys. Chem. A* **119**, 13074 (2015).
- ¹⁹A. Leclerc and T. Carrington, Jr., *Chem. Phys. Lett.* **644**, 183 (2016).
- ²⁰A. Leclerc, P. S. Thomas, and T. Carrington, Jr., *Mol. Phys.* (published online 2016).
- ²¹T. G. Kolda and B. W. Bader, *SIAM Rev.* **51**, 455 (2009).
- ²²H.-D. Meyer, F. L. Quere, C. Leonard, and F. Gatti, *Chem. Phys.* **329**, 179 (2006).
- ²³L. J. Doriel, F. Gatti, C. Iung, and H.-D. Meyer, *J. Chem. Phys.* **129**, 224109 (2008).
- ²⁴Multidimensional Quantum Dynamics: MCTDH Theory and Application, edited by H.-D. Meyer, F. Gatti, and G. Worth (Wiley-VCH, Weinheim, 2009).
- ²⁵H.-D. Meyer, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2**, 351 (2012).
- ²⁶I. H. Godtliebsen, M. B. Hansen, and O. Christiansen, *J. Chem. Phys.* **142**, 024105 (2015).

- ²⁷I. H. Godtlielsen, B. Thomsen, and O. Christiansen, *J. Phys. Chem. A* **117**, 7267 (2013).
- ²⁸M. Rakhuba and I. Oseledets, *J. Chem. Phys.* **145**, 124101 (2016).
- ²⁹G. Beylkin and M. J. Mohlenkamp, *SIAM J. Sci. Comput.* **26**, 2133 (2005).
- ³⁰G. H. Golub and C. F. V. Loan, *Matrix Computations*, 3rd ed. (John Hopkins University Press, Baltimore, MD, 1996).
- ³¹I. M. Mills, *Vibration-rotation Structure in Asymmetric and Symmetric Top Molecules* (Academic Press, New York, 1972), Vol. 1, p. 115.
- ³²J. K. G. Watson, *Mol. Phys.* **15**, 479 (1968).
- ³³D. Bégué, N. Gouhaud, C. Pouchan, P. Cassam-Chenai, and J. Liévin, *J. Chem. Phys.* **127**, 164115 (2007).
- ³⁴P. Seidler, E. Matito, and O. Christiansen, *J. Chem. Phys.* **131**, 034115 (2009).
- ³⁵P. Seidler and O. Christiansen, *J. Chem. Phys.* **131**, 234109 (2009).
- ³⁶P. Seidler, T. Kaga, K. Yagi, O. Christiansen, and K. Hirao, *Chem. Phys. Lett.* **483**, 138 (2009).
- ³⁷C. Puzzarini, M. Biczysko, J. Bloino, and V. Barone, *Astrophys. J.* **785**, 107 (2014).
- ³⁸J. Brown and T. Carrington, Jr., *J. Chem. Phys.* **145**, 144104 (2016).
- ³⁹P. Cassam-Chenai, private communication 11 January (2016).
- ⁴⁰P. Cassam-Chenai and J. Liévin, *J. Comput. Chem.* **27**, 627 (2006).
- ⁴¹E. Cané and A. Trombetti, *Phys. Chem. Chem. Phys.* **11**, 2428 (2009).