Comparison of different eigensolvers for calculating vibrational spectra using low-rank, sum-of-product basis functions

Arnaud Leclerc, Phillip Thomas, Tucker Carrington

Comparison of different eigensolvers for calculating vibrational spectra using low-rank, sum-of-product basis functions.

Arnaud Leclerc[a], Phillip S. Thomas[b], Tucker Carrington[b,*]

[a] *Université de Lorraine, UMR CNRS 7565 SRSMC, 1 boulevard Arago 57070 Metz, France*
[b] *Chemistry Department, Queen's University, Kingston, Ontario K7L 3N6, Canada*

Abstract

Vibrational spectra and wavefunctions of polyatomic molecules can be calculated at low memory cost using low-rank sum-of-product (SOP) decompositions to represent basis functions generated using an iterative eigensolver. Using a SOP tensor format does not determine the iterative eigensolver. The choice of the interative eigensolver is limited by the need to restrict the rank of the SOP basis functions at every stage of the calculation. We have adapted, implemented and compared different reduced-rank algorithms based on standard iterative methods (block-Davidson algorithm, Chebyshev iteration) to calculate vibrational energy levels and wavefunctions of the 12-dimensional acetonitrile molecule. The effect of using low-rank SOP basis functions on the different methods is analyzed and the numerical results are compared with those obtained with the reduced rank block power method introduced in J. Chem. Phys. 140, 174111 (2014). Relative merits of the different algorithms are presented, showing that the advantage of using a more sophisticated method, although mitigated by the use of reduced-rank sum-of-product functions, is noticeable in terms of CPU time.

1. Introduction

Understanding the internal motion of the nuclei of a polyatomic molecule is an important problem in molecular physics. Given a potential energy surface, the most general approach to compute vibrational spectra is to calculate eigenvalues and eigenvectors of a matrix representing the Hamiltonian operator in some basis. When a direct product basis, with functions that are products

*Corresponding author
Email addresses: Arnaud.Leclerc@univ-lorraine.fr (Arnaud Leclerc), Phillip.Thomas@chem.queensu.ca (Phillip S. Thomas), Tucker.Carrington@queensu.ca (Tucker Carrington)

of functions of a single coordinate, is used, the computational cost increases exponentially with the number of internal degrees of freedom ($D = 3N - 6$ for a molecule with N atoms). This problem is well-known as the "curse of dimensionality". For example, to use a direct product basis to calculate the vibrational spectrum for a molecule with 6 atoms, one would need about 8×10^{15} GB to store the matrix, assuming there are 10 basis functions for each coordinate. If one uses an iterative eigensolver [1, 2, 3, 4, 5] it is possible to compute a spectrum without storing (or computing) the Hamiltonian matrix [6, 7, 8, 9, 10, 11]. One must only store a few vectors, however a single vector requires about 8000 GB. Iterative methods only require the ability to apply the Hamiltonian matrix to vectors.

Different strategies have been employed to drastically reduce the memory cost of storing these vectors. To some extent they can be used together. These include: pruning the basis set by retaining only some of the direct product basis functions [12, 13, 14, 15, 16, 17, 18, 19, 20]; using basis functions localized in the classically allowed regions [21, 22, 23, 24]; making contracted basis functions by diagonalizing reduced-dimension Hamiltonians for strongly coupled coordinates [25, 26, 27]. All of these strategies aim to reduce the number of required basis functions. Instead, one can work with a direct product basis, but obviate the need to store vectors with n^D components, where n is a representative number of basis functions for a single coordinate. This can be achieved by representing basis vectors with low-rank tensors. One can think of the multi-configuration time-dependent Hartree (MCTDH) method as representing wavefunctions in the Tucker tensor format [28, 29, 30]. The Tucker format does not defeat the curse of dimensionality because it is a direct product representation. The advantage of MCTDH is the optimization of the 1D basis functions. There is also a multi-layer MCTDH method using what mathematicians call a hierarchical-Tucker format [31, 32]. There are several other tensor formats that can be used, such as the Matrix Product States [33] (equivalent to the Tensor Train format [34, 35]) or the Canonical decomposition (CP format) [30, 36], which we exploit in this article.

Tensors can be used to compute vibrational energy levels in two ways. One way to use tensors is to represent the desired eigenvectors in tensor format and to optimize elements of the tensors. The well-known density matrix renormalization group (DMRG) method is of this type. Another way is to compute eigenvalues by projecting into a space each of whose vectors is in a tensor format. The vectors may be calculated using a standard iterative algorithm. In this article we use this second way. It can only be used if the Hamiltonian matrix is also low rank. The tensor format we use is CP and we therefore require that the Hamiltonian be a sum of products. Whenever the Hamiltonian is applied to a vector, the rank (i.e. the number of terms) increases and must be reduced to keep the memory cost within acceptable limits. This reduction step also requires optimization and is the computational bottleneck. For both ways, the optimization is usually done with a variant of the alternating least squares algorithm. The idea of using a basis of vectors in a tensor format to compute vibrational levels was introduced in Ref. [37], where the Reduced

Rank Block Power Method (RRBPM) was presented. It was later shown that the RRBPM is compatible with symmetry partitioning of the basis set, which facilitates computing and assigning levels [38]. A hierarchical version of the RRBPM, taking advantage of successive contractions for strongly coupled coordinates was proposed in Ref. [39]. Making intermediate basis functions using a tree structure greatly improved the accuracy and reduced the required CPU time. The H-RRBPM of Ref. [39] uses the two-layer RRBPM of Ref. [37] at each layer.

In this article, we test a method similar to the original RRBPM but with a better eigensolver. We do not use the hierarchical version, but the ideas we introduce could be used in conjunction with it. The power method used in [37] is a crude and simple iterative method. It is well known to converge slowly, especially if the density of states is high. The CPU time required to obtain converged eigenvalues can thus be quite long. The power method was originally used not only because it is simple but also because the intermediate vectors it generates become increasing similar to eigenvectors which are assumed to be "reducible". Here reducible means that little error is incurred when the rank is reduced. In this article, we show that it is possible to use more elaborate eigensolvers. In section 2, we briefly recall the main ideas of the RRBPM and introduce the modifications required to use two other eigensolvers. One is a Chebyshev filter method and the other is a block-Davidson method. These algorithms are then used to calculate vibrational eigenstates of acetonitrile (CH_3CN), a 12D problem, in section 3.

2. Reduced-rank Iterative Methods for calculating spectra

2.1. Sum-of-product format for functions and operators

In all calculations in this article basis vectors are in CP format [30]. Eigenvectors of the Hamiltonian matrix are obtained as linear combinations of the CP basis vectors and hence they also are in CP format. In general, one can expand a multidimensional function in a direct product basis,

$$F(q_1,\ldots,q_D) \simeq \sum_{i_1=0}^{n_1-1} \cdots \sum_{i_D=0}^{n_D-1} F_{i_1 i_2 \ldots i_D} \prod_{k=1}^{D} \theta_{i_k}^{k}(q_k), \tag{1}$$

where $\{\theta_{i_k}^{k}(q_k),\ i_k = 0 \ldots (n_k-1)\}$, is a set of basis functions associated with coordinate q_k, $k = 1 \ldots D$. The expansion coefficient is in CP format if

$$F_{i_1 i_2 \ldots i_D} = \sum_{\ell=1}^{R} \prod_{k=1}^{D} f_{i_k}^{(\ell,k)} \ , \tag{2}$$

where the $f_{i_k}^{(\ell,k)}$, $i_k = 0 \ldots (n_k-1)$ are components of one-dimensional vectors $\mathbf{f}^{(\ell,k)}$ which generally appear only once each in the expansion. There is no need

for them to be orthogonal or normalized. If $F_{i_1 i_2 \ldots i_D}$ is in CP format then $F(q_1, \ldots, q_D)$ is a sum of products (SOP),

$$F(q_1, \ldots, q_D) = \sum_{\ell=1}^{R} \prod_{k=1}^{D} \phi^{(\ell,k)}(q_k) = \sum_{\ell=1}^{R} \prod_{k=1}^{D} \left(\sum_{i_k=0}^{n_k-1} f_{i_k}^{(\ell,k)} \theta_{i_k}^{k}(q_k) \right) . \qquad (3)$$

The important idea underlying all tensor-based methods is: in general the memory cost of storing $F_{i_1 i_2 \ldots i_D}$ scales as n^D but the memory cost of storing the right side of Eq. (2) scales as RDn [37, 39].

Vectors generated by the iterative eigensolver from a vector in CP format will themselves only be in CP format if the Hamiltonian is a SOP. Throughout this article, we shall assume that the potential energy surface is known and expressed as a SOP. The kinetic energy operator (KEO) is often a SOP. In this article we use normal coordinates and neglect the $\pi - \pi$ term [40] so that the KEO is a sum of terms each of which depends on a single coordinate,

$$\hat{H}(q_1, \ldots, q_D) = \sum_{k=1}^{T} \prod_{j=1}^{D} \hat{h}_{kj}(q_j), \qquad (4)$$

where $\hat{h}_{kj}$ is a one-dimensional operator acting in a Hilbert space associated with coordinate q_j. The application of $\hat{H}$ on vectors is the crucial step in every iterative method. When vectors are in CP format and the operator is a SOP, the application of $\mathbf{H}$ requires only one-dimensional operations, this can be seen by multiplying Eq. (4) and Eq. (3) [37]. Only the small $(n_j \times n_j)$ Hamiltonian matrices representing $\hat{h}_{kj}(q_j)$ have to be computed and stored in memory. In the following subsections, we describe several methods to make a small basis of low-rank SOP basis functions with which one can compute low-lying levels of molecules with more than five atoms.

2.2. Reduced-rank Block Power method

In the power method, the matrix $\mathbf{H}$ is applied recursively to a start vector $\mathbf{F}_0$, to calculate the eigenvector associated with the largest eigenvalue [41, 42]. To obtain several eigenvalues one uses a block of vectors. To calculate the lowest eigenvalues, the matrix is also shifted by $-\sigma\mathbf{I}$. The RRBPM uses a shifted block power method and stores $\mathbf{H}$ as a SOP of small matrices and the vectors in CP format. Each matrix-vector product increases the rank of a vector by a factor of T. The rank also increases, to a lesser extent, when vectors in the block are orthogonalized and updated, see Ref. [37]. The rank must be reduced after each operation which increases it. As in Ref. [37, 38], an old (large-rank) vector $\mathbf{F}^{\text{old}}$ is reduced from rank R^{old} (typically a few thousands) to rank R^{new} (a few tens) using an alternating-least-squares (ALS) algorithm. For consistency of comparison we will use the same implementation of ALS as in [37, 38], based on [36]. ALS is an iterative process in which linear equations are solved to find new vectors ${}^{\text{new}}f_{i_k}^{(\ell,k)}$ which minimize the difference $\| \mathbf{F}^{\text{new}} - \mathbf{F}^{\text{old}} \|$. The ALS reduction is a crucial step in the reduced rank methods described below and is

responsible for a majority of the calculation time. The main reduction has to be performed after the matrix-vector product which is responsible for the most important increase of the rank. The algorithm can be summarized as follows:

1. Generate initial guess for the block of SOP eigenvectors $\boldsymbol{\mathcal{F}} = (\mathbf{F}_{k=1} \dots \mathbf{F}_{k=B})$
2. For $m = 1 \dots N_{rs}$ (with $N_{rs} \simeq 10$)
   (a) For $k = 1 \dots B$, $\mathbf{F}_k^{m+1} \leftarrow (\mathbf{H} - \sigma\mathbf{I})\mathbf{F}_k^m$.
   (b) Reduce the ranks using alternating least squares.
3. Orthogonalize the vectors, make a matrix $\mathbf{H}' = \boldsymbol{\mathcal{F}}^t\mathbf{H}\boldsymbol{\mathcal{F}}$ representing $\hat{H}$ in this SOP basis set and compute the overlap matrix $\mathbf{S} = \boldsymbol{\mathcal{F}}^t\boldsymbol{\mathcal{F}}$.
4. Solve the generalized eigenvalue problem to obtain eigenvalues and eigenvectors.
5. Reduce the rank, update the vectors with eigenvectors; go back to step 2.

In the above algorithm N_{rs} is the number of power iterations done between two updates of the basis vectors using reduced eigenvectors of $\mathbf{H}'$. Note that the basis size does not grow during the calculation; in step 2 (a) the basis at step $(m+1)$ replaces the basis at step m. The energy shift is $\sigma = (E_B + E_{max})/2$, where E_{max} is the highest eigenvalue of $\mathbf{H}$ determined by doing a few unshifted power iterations [37]. The memory cost scales as

$$\mathcal{O}(BTRDn) \tag{5}$$

where B is the size of the computed subspace, T the number of terms in the Hamiltonian, R the reduction rank (we keep it fixed for simplicity), D the number of coordinates and n the number of basis functions per coordinate. The cost of performing matrix vector products in the subspace scales as

$$\mathcal{O}(N_{\mathrm{pow}}BTRDn^2) \tag{6}$$

where N_{pow} is the number of power iterations. The overall computational cost of rank reductions using ALS [36] scales as

$$\mathcal{O}(N_{\mathrm{pow}}N_{\mathrm{ALS}}BD(R^3 + nTR^2)) \tag{7}$$

with N_{ALS} as the number of internal iterations for the least squares procedure used to reduce the rank. The memory cost and the computational cost of reductions thus scale linearly with D.

2.3. Reduced-rank Chebyshev filter technique

There are many ways to push a block of vectors towards the space spanned by the eigenvectors with the lowest eigenvalues. The shifted block power method does push a block of vectors in the right direction, however, the space it generates includes contributions from eigenvectors whose corresponding eigenvalues are not in the set of the lowest B. It might be better to use a filter that excludes vectors outside the block. Here, we apply a sequence of filter functions to each vector in the block, where a different filter is used for each vector. The filter

for the k^{th} vector is designed to exclude all vectors corresponding to eigenvalues above the k^{th}. The filters we use are carefully chosen polynomials, $\mathbf{F}_{k,purified} = P_m^k(\mathbf{H})\mathbf{F}_k$, where P_m^k is a polynomial of degree m.

Following Saad [43, 41], we use a Chebyshev polynomials of the first kind, $P_m^k(\mathbf{H}) = T_m\left(\tilde{\mathbf{H}}_k\right)$, where $\tilde{\mathbf{H}}_k = \frac{\mathbf{H}-d_k\mathbf{I}}{c_k}$ denotes a Hamiltonian that has been shifted and scaled. We choose the scaling parameter c_k and the shift parameter d_k as

$$\begin{aligned} c_k &= \frac{E_{max} - E_{k+1}}{2} \\ d_k &= \frac{E_{max} + E_{k+1}}{2} \ . \end{aligned} \tag{8}$$

This choice means that for $\mathbf{F}_k$, the contribution of eigenvectors corresponding to the eigenvalues $E_{k+1} \ldots E_{max}$, which are mapped into the interval $[-1, 1]$, is decreased and the contribution of eigenvectors corresponding to the eigenvalues $E_1 \ldots E_k$, which are mapped to < -1, is increased [43, 41]. This is true because if $-1 \leq x \leq 1$, $-1 \leq T_m(x) \leq 1$ and if x is outside of $[-1, 1]$, $T_m(x)$ is the most rapidly growing of all m^{th}-order polynomials. This filter will therefore accentuate contributions from eigenvectors whose eigenvalues are less than E_{k+1}. Using a different filter for each vector in the block makes the vectors more reducible. Each filter accentuates contributions from eigenvectors with smaller eigenvalues more than eigenvectors with larger eigenvalues. For instance, if there are $B = 80$ states in the block, the filter for the B^{th} vector accentuates the E_0 eigenvector more than the E_B eigenvector. As a result, it is necessary to orthogonalize the k^{th} vector to vectors $1 \ldots k-1$ after applying the filter.

To compute parameters c_k and d_k, one needs an estimate of eigenvalues E_{k+1} and E_{max}. We estimate E_{max} by doing a few (non-shifted) power iterations. If the E_{max} estimate is too low, then the true value of E_{max} will be mapped to a value larger than one and vectors in the block will become "contaminated" with contributions from the eigenvector corresponding to E_{max}, ruining convergence. To avoid this we "pad" our estimate of E_{max} by adding $0.01(E_{max} - E_{min})$. For $k = B$, we do not have an estimate of $E_{k+1} = E_{B+1}$ since E_{B+1} lies outside of the block. In this case we use the estimate $E_{B+1} \approx 2E_B - E_{B-1}$ where E_B and E_{B-1} are from the previous iteration. If the actual spacing between E_B and E_{B+1} is smaller than the spacing between E_B and E_{B-1}, E_B will converge slowly.

To apply the polynomial filter $P_m(H)$ one must evaluate m matrix-vector products. This is done recursively. Beginning with $\mathbf{F}_k^{(0)} \equiv \mathbf{F}$ and $\mathbf{F}_k^{(1)} = \tilde{\mathbf{H}}_k\mathbf{F}_k^{(0)}$, the other filtered vectors are obtained from

$$\mathbf{F}_k^{(j)} = 2\tilde{\mathbf{H}}_k\mathbf{F}_k^{(j-1)} - \mathbf{F}_k^{(j-2)} \qquad (j = 2 \ldots m) \tag{9}$$

Augmenting the degree of the filter by one requires one additional matrix-vector product and one vector-vector addition. After $\mathbf{F}_k^{(j)}$ is generated, it is reduced with ALS. Each matrix-vector product is done by exploiting the CP format of

the vector. As explained by Saad [43, 41], the polynomial filter can magnify the coefficient of the ground state eigenvector beyond the overflow limit if high Chebyshev orders are used. The use of a scaling parameter σ_j is necessary to prevent this. The algorithm is:

1. Generate an initial block of vectors $\boldsymbol{\mathcal{F}} = (\mathbf{F}_1 \dots \mathbf{F}_B)$, initial eigenvalue estimates, and an initial spectral range.
2. For $k = 1 \dots B$:
   (a) Set c_k and d_k as described in Eq. (8); set $\sigma_1 = \frac{c_1}{d_1 - E_1}$; compute $\mathbf{F}_k^{(1)} = \sigma_1 \tilde{\mathbf{H}}_k \mathbf{F}_k^{(0)}$
   (b) For $j = 2 \dots m$:
      i. Compute $\sigma_j = \left(\frac{2}{\sigma_1} - \sigma_{j-1} \right)^{-1}$
      ii. Chebyshev iteration: $\mathbf{F}_k^{(j)} = \sigma_j \left(2\tilde{\mathbf{H}}_k \mathbf{F}_k^{(j-1)} - \sigma_{j-1} \mathbf{F}_k^{(j-2)} \right)$
      iii. Reduce the rank of $\mathbf{F}_k^{(j)}$ using ALS
3. Orthogonalize the vectors, make the matrix $\mathbf{H}' = \boldsymbol{\mathcal{F}}^t \mathbf{H} \boldsymbol{\mathcal{F}}$ and overlap matrix $\mathbf{S} = \boldsymbol{\mathcal{F}}^t \boldsymbol{\mathcal{F}}$.
4. Solve the generalized eigenvalue problem to obtain eigenvalues and eigenvectors; update vectors in $\boldsymbol{\mathcal{F}}$; reduce ranks; back to step 2.

The computational cost of filtering a subspace of dimension B is dominated by the cost of evaluating m matrix-vector products and doing m rank reductions, i.e.

$$\mathcal{O}(mBTRDn^2) + \mathcal{O}(mN_{\mathrm{ALS}}BD(R^3 + nTR^2)). \tag{10}$$

We typically use polynomials of degree $m = 10$.

2.4. Reduced-rank Block Davidson method

The original Davidson algorithm begins with a start vector and builds a space adapted to the calculation of a single eigenvalue by adding one vector at a time [3]. The best estimate of the desired eigenvalue is obtained by projecting the eigenvalue problem into the space spanned by the Davidson vectors. Davidson uses a form of preconditioning to favor the convergence of the desired eigenvalue. We use a block version of Davidson [4]. From one iteration to the next the power method replaces the previous basis with a new basis of the same size. The size of the matrix does not increase as the calculation proceeds. The size of the block Davidson basis does increase during the iteration because at each iteration B vectors are added to the basis. B new vectors are generated and then orthogonalized with respect to vectors already in the basis. The basis is then augmented with the orthogonalized vectors, a generalized eigenvalue problem is solved and eigenvectors with the lowest eigenvalues are used either to compute the B new vectors to be added at the next iteration (see algorithm), or as new start vectors. The block Davidson algorithm we use is essentially a CP version of the one in Ref. [4, 5]. We restart the algorithm every N_{rs} iterations. The algorithm to compute B eigenvectors is:

1. Preparation:
   (a) Define an initial subspace $\boldsymbol{\mathcal{F}} = (\mathbf{F}_1 \dots \mathbf{F}_B)$; set $B' = B$.
   (b) Make the matrix $\mathbf{H}' = \boldsymbol{\mathcal{F}}^t \mathbf{H} \boldsymbol{\mathcal{F}}$ and overlap matrix $\mathbf{S} = \boldsymbol{\mathcal{F}}^t \boldsymbol{\mathcal{F}}$.
   (c) Solve the eigenvalue problem, reduce the ranks of the eigenvectors.
   (d) Select B eigenvectors $\psi_{\mathbf{m}}$ corresponding to the lowest eigenvalues E_m, $m = 1 \dots B$.
2. For $j = 1 \dots N_{cycle}$:
   (a) For $k = 1 \dots N_{rs}$:
      i. Compute the residuals $\mathbf{q}_m = (\mathbf{H} - E_m \mathbf{I})\psi_m$; reduce the rank.
      ii. Precondition: Compute (in an approximate way, see text below) the new vectors $\mathbf{F}_{B'+m} = (E_m - \mathbf{H}_0)^{-1} \mathbf{q}_m$ for $m = 1 \dots B$.
      iii. Orthogonalize with respect to previous vectors in $\boldsymbol{\mathcal{F}}$, reduce the rank.
      iv. Set $B' \leftarrow B' + B$; augment $\mathbf{H}'$; augment $\mathbf{S}$;
      v. Diagonalize, reduce the ranks of the eigenvectors.
      vi. Select B eigenvectors ψ_m corresponding to the lowest eigenvalues E_m, $m = 1 \dots B$.
   (b) Restart: set $B' = B$, update the basis set, keeping only the first B eigenvectors approximations; update $\mathbf{H}'$ and $\mathbf{S}$.

A good starting block is important for the convergence of the Davidson algorithm. We use rank-one eigenvectors of the uncoupled, separable part of the Hamiltonian as start vectors.

The main difficulty is applying the preconditioner in step (2(a)ii). In our calculations $\mathbf{H}_0$ is diagonal. The corresponding operator is the separable part of the Hamiltonian. With this choice, $\mathbf{H}_0$ is naturally represented as a sum of products. $(\mathbf{E}_m - \mathbf{H}_0)^{-1}$ is also diagonal in the direct product basis. However, it is not in low-rank sum-of-product form. The corresponding operator can be written as a sum of $\prod_k n_k$ terms of rank one by using a spectral expansion of $(E_m - \mathbf{H}_0)^{-1}$ in the direct product basis set,

$$\sum_{i_1 \dots i_D} \frac{1}{E_m - E^0_{i_1 \dots i_D}} |\Theta_{i_1 \dots i_D}\rangle\langle\Theta_{i_1 \dots i_D}| \tag{11}$$

where $\{E^0_{i_1 \dots i_D}, \Theta_{i_1 \dots i_D}\}$ are the eigenpairs of $\mathbf{H}_0$. We do not use $(\mathbf{E}_m - \mathbf{H}_0)^{-1}$ because applying it to a vector would increase it rank by a factor of $\prod_k n_k$. Instead, we replace diagonal elements of $(\mathbf{E}_m - \mathbf{H}_0)^{-1}$ with $E_{0,\text{cut-off}}$, if the $i_1 \dots i_D$ diagonal element of $\mathbf{H_0}$ is larger than $E_{0,\text{cut-off}}$. Denote this modified matrix $\mathbf{M}_{J_0}$, where J_0 is the set of indices, $i_1 \dots i_D$, for which $E_{0,i_1 \dots i_D} \leq E_{0,\text{cut-off}}$. There are N_{lr} (lr means "low rank") elements in J_0. $\mathbf{M}_{J_0}$ is a matrix whose rank is $\prod_k n_k$. It can be written as a sum of two matrices,

$$\mathbf{M}_{J_0} = \mathbf{M}^{lr}_{J_0} + \lambda \mathbf{I} \tag{12}$$

where $\lambda = \frac{1}{E_m - E_{0,\text{cut-off}}}$ and $\mathbf{M}^{lr}_{J_0}$ is a diagonal matrix whose $i_1 \dots i_D$ diagonal element is $\left(\frac{1}{E_m - E^0_{i_1 \dots i_D}} - \lambda\right)$, if $i_1 \dots i_D \in J_0$ and zero otherwise. Applying

$\mathbf{M}_{J_0}^{lr}$ to a vector increases its rank by a factor of N_{lr}. Applying $\mathbf{I}$ to a vector does not change its rank. Therefore applying $\mathbf{M}_{J_0}$ to a vector increases its rank by a factor of $N_{lr}+1$. An ALS reduction must be done immediately after applying $\mathbf{M}_{J_0}$. If step (2(a)ii) is ignored, the iteration still converges because the Davidson space becomes identical to a Lanczos subspace (but with vectors different from those we would obtain using the Lanczos method, which does not include a diagonalization step).

One important advantage of the Davidson algorithm is that the selection criteria (step 2(a)vi) can take several forms. Here we will always select the eigenvectors corresponding to the lowest eigenvalues. We could also choose some overlap criteria or specify a given spectral window.

The memory cost depends on the number of iterations between restarts, N_{rs}. The maximum memory cost is reached just before restarting and scales as $\mathcal{O}(N_{rs}BTRDn)$. The calculation cost scales as the number of matrix-vector products, equal to $(2BN_{cycle}N_{rs})$ because there are B matrix-vector products for calculating the residuals and B more for making or augmenting the $\mathbf{H}'$ matrix, for each value of the j and k indices. This has to be multiplied by the cost of one product and one reduction, Eqs. (6) and (7). This estimate does not include the cost of applying the preconditioner which significantly affects the overall cost if the cut-off is large.

2.5. Parallelization

There are two parallelization strategies. Both the Davidson and the Chebyshev eigensolvers can be parallelized over vectors in the block (different vectors in a block are computed on different threads) or the operations required to compute a single vector can be parallelized. The first strategy is easier to implement but requires more memory because one must store many high rank vectors generated by doing MVP. The second strategy allows one to store, one at a time, the high-rank vectors arising from matrix-vector products. In practice, we usually parallelize over vectors in the block, as is done in Ref. [37]. Some of the calculations have been done sequentially in order to facilitate CPU time comparisons.

3. Numerical results on the Acetonitrile molecule

In this section we show that the three eigensolvers presented in section 2 all work well and all give accurate eigenvalues for a 12-D problem. In this paper, we use the eigensolvers only in two layer calculations. Since we do not exploit the hierarchical (contraction) ideas of Ref. [39], we do not demand that the energies be as accurate as in [39]. The comparison in this paper is a relative comparison of the different reduced-rank eigensolvers. If the Chebyshev and/or Davidson eigensolver is more efficient than the shifted power method it can be used in conjunction with the hierarchical idea. The most demanding part of the hierarchical calculation is computing eigenfunctions of the node at the top of the tree, which typically requires many RRBPM iterations. A more efficient eigensolver could therefore be used to improve the H-RRBPM.

3.1. Hamiltonian and basis set

We compute the acetonitrile (CH_3CN) vibrational spectrum. This is a six-atom molecule and we calculate eigenvalues and eigenstates of a 12D quartic Hamiltonian. The normal-coordinate Hamiltonian is

$$\begin{aligned}\hat{H}(q_1,\ldots,q_{12}) &= -\frac{1}{2}\sum_{i=1}^{12}\omega_i\frac{\partial^2}{\partial q_i^2}+\frac{1}{2}\sum_{i=1}^{12}\omega_i q_i^2+\frac{1}{6}\sum_{i=1}^{12}\sum_{j=1}^{12}\sum_{k=1}^{12}\phi^{(3)}_{ijk}q_iq_jq_k \\ &+\frac{1}{24}\sum_{i=1}^{12}\sum_{j=1}^{12}\sum_{k=1}^{12}\sum_{\ell=1}^{12}\phi^{(4)}_{ijk\ell}q_iq_jq_kq_\ell, \qquad (13)\end{aligned}$$

with the same assumptions as in [13]. In Eq. (13), coordinates q_1 to q_4 are non-degenerate, coordinates q_5 to q_{12} are members of doubly-degenerate pairs. The potential coefficients are those used in [13] and are based on the constants reported in [44]. The direct product basis set is a product of 1D anharmonic eigenfunctions. The 1D functions are obtained by diagonalizing 1D uncoupled Hamiltonians that are obtained by setting all but one normal coordinate equal to zero. This preliminary calculation has been done in a basis of harmonic oscillator eigenfunctions. The Hamiltonian operator is then factorized following [39] to minimize the number of terms that need to be applied to each vector. After factorization, there are 216 terms in $\hat{H}$.

3.2. Numerical results

To compare the three reduced rank eigensolvers, we list differences between energy levels and the zero-point energy (ZPE), and corresponding errors after 20 and 100 matrix-vector products (MVP), see Table 1. The number of MVP is roughly proportional to the cost of the calculation. It is not actually the MVP itself that is costly, but the rank reduction that is done after each MVP. All the calculations have been done with the same bases (identical to those of [13, 37]), the same initial block made of $B = 32$ eigenvectors of the separable approximation to the Hamiltonian. We have used the same reduction rank $R = 50$ for all the calculations and a fixed number of ALS iterations, $N_{ALS} = 15$ for rank reductions. RRBPM calculations were done with an energy shift of $\sigma = 170000$ cm^{-1} [37]. Diagonalization and vector updates were done every 10 iterations. When using the Reduced Rank Block Chebyshev (RRBC) method, diagonalizations and vector updates were done every 10 Chebyshev iterations. The value of E_{max} used to compute the filter parameters was padded by adding 3173.7 cm^{-1}. The Reduced Rank Block Davidson (RRBD) method is restarted every $N_{rs} = 4$ iterations to reduce CPU cost. The preconditioning step (Eq. (11) in section 2.4) is applied within an active subspace made of the first 500 basis functions of the direct product basis set.

In the third column of table 1, we show differences between levels computed using the RRBD method and the corresponding ZPE, for the first 32 vibrational states of acetonitrile. These results are obtained after 200 matrix-vector products. Increasing the number of MVP causes them to oscillate. The oscillations

could be reduced by increasing the target rank, R. In columns 4-9 we report errors with respect to the Smolyak results of Ref. [13] for the RRBP, RRBC and RRBD methods. In the RRBPM and the RRBC columns of table 1, differences are given for a fixed number of matrix-vector products (either 20 MVP or 100 MVP). The number of MVP is given per computed vector, i.e. we count all the MVP and divide by B.

From table 1, one concludes that all methods converge to the true eigenvalues. It is also clear that the RRBC and RRBD methods converge substantially faster than the RRBPM. All errors of RRBD calculations are smaller than their RRBPM counterparts, both after 20 matrix-vector products and after 100 matrix-vector products. The same is true after 100 matrix-vector product for the reduced-rank Chebyshev method, but some errors are larger and some are smaller than those of the RRBPM if we compare the energies after only 20 matrix-vector products. It should be noted that errors are calculated by taking differences of energy differences and this means that the error in the zero-point energy also contributes to the error estimates given in table 1. For all three eigensolvers, the highest two energy levels in the block have larger errors (about 4 cm^{-1}). Except these two states, all errors in energy differences are between $[0.01, 1.32]$ cm^{-1} for the Chebyshev results, $[0.08, 1.76]$ cm^{-1} for the block-Davidson method whereas the RRBPM errors are between $[0.13, 6.25]$, after 100 MVP.

Convergence curves for two representative levels, the first excited state ν_{11} and the 13^{th} state $\nu_4 + \nu_{11}$, are given in Fig. 1. Convergence curves for the power method are almost monotonic but decreases slowly. The Davidson results exhibit first a global descent but then small oscillations. The Davidson oscillations occur because error increases when the algorithm is restarted with a new, smaller block of vectors, and decreases as number of vectors in space again increases. The first few MVP decrease the RRBD error very quickly.

The RRBC algorithm converges much more quickly than the RRBPM during the first tens of iterations for all energy levels considered. However, later on oscillations set in, due to the imposed reduction rank. For the RRBPM, the oscillations do not appear for all levels, but for the levels that do oscillate they are typically less than 0.01 cm^{-1} in magnitude. For the RRBC algorithm the oscillations are larger, having magnitudes of $\sim$0.1 cm^{-1} for all levels. For both solvers the oscillations can be dampened by increasing the rank.

In the RRBD method there are also orthogonalization and preconditioning steps which do not influence the number of MVP but do increase the CPU cost. Therefore, it is also important to compare the cost of the three calculations. In table 2, we report ratios of RRBC and RRBD CPU times to the RRBPM CPU time. These speed-up ratios are computed using the CPU time, with no parallelization, required to achieve a difference of less than 2 cm^{-1} with respect to Smolyak results for the first ten, for the first twenty, and for the first twenty-five energy levels. The RRBD method reduces the CPU cost by factor of 3-4 for all three groups of states. The CPU time ratios could be further improved by omitting the orthogonalization step and the associated reduction (step 2(a)iii of the Davidson algorithm in subsection 2.4), which are not essential because

Table 1: First 32 levels (from which the ZPE has been subtracted) (cm^{-1}) and differences with Smolyak results [13], after a fixed number of matrix-vector products (mean number per computed vector). Comparison of three reduced-rank eigensolvers: reduced-rank block power method (RRBPM), reduced-rank block Chebyshev method (RRBC) and reduced-rank block Davidson method (RRBD). The results in column 3 are those of the Davidson calculation after 200 matrix-vector products.

Assign.	Sym.	Energy	Error after 20 mvp			Error after 100 mvp		
		level	RRBPM	RRBC	RRBD	RRBPM	RRBC	RRBD
ZPE		9837.498						
ν_{11}	E	361.08	0.39	1.59	0.16	0.13	0.05	0.08
		361.15	0.48	1.57	0.18	0.17	0.02	0.10
$2\nu_{11}$	E	723.25	0.88	3.14	0.20	0.25	0.13	0.11
		723.63	1.12	2.75	0.53	0.57	0.25	0.36
$2\nu_{11}$	A_1	724.35	1.30	3.02	0.61	0.64	0.17	0.50
ν_4	A_1	900.78	5.59	3.73	0.27	1.75	0.19	0.11
ν_9	E	1034.40	9.39	3.92	0.31	1.79	0.21	0.19
		1034.74	10.04	3.84	0.37	1.84	0.20	0.23
$3\nu_{11}$	A_1	1087.27	6.28	2.87	1.10	2.95	0.49	0.60
$3\nu_{11}$	A_2	1087.40	6.32	2.69	1.29	3.10	0.57	0.64
$3\nu_{11}$	E	1088.55	6.51	3.25	1.27	3.30	0.50	0.52
		1088.63	6.59	3.06	1.29	3.50	0.52	0.73
$\nu_4 + \nu_{11}$	E	1260.12	8.04	3.81	1.02	2.86	0.25	0.43
		1260.26	8.05	3.51	1.11	2.92	0.29	0.46
ν_3	A_1	1390.79	5.77	3.85	1.29	2.45	0.57	1.05
$\nu_9 + \nu_{11}$	E	1395.50	11.23	3.74	1.08	2.76	0.23	0.60
		1395.64	11.77	3.37	1.30	3.39	0.76	1.29
$\nu_9 + \nu_{11}$	A_2	1396.46	11.78	3.40	1.18	3.21	0.63	1.76
$\nu_9 + \nu_{11}$	A_1	1398.56	10.66	3.73	1.77	3.10	0.64	1.02
$4\nu_{11}$	E	1452.04	8.27	3.24	1.22	3.94	0.69	0.62
		1452.04	8.37	2.96	1.30	4.02	1.12	0.97
$4\nu_{11}$	E	1454.33	8.85	2.17	1.52	4.89	0.61	0.83
		1454.51	9.03	1.81	1.59	5.68	0.74	1.13
$4\nu_{11}$	A_1	1454.79	9.47	2.09	2.04	5.36	0.98	1.57
ν_7	E	1483.57	5.48	5.39	0.54	1.14	0.02	0.20
		1483.61	6.89	5.38	0.58	1.59	0.01	0.26
$\nu_4 + 2\nu_{11}$	E	1620.57	11.28	2.75	1.21	4.26	0.29	0.45
		1621.41	11.66	2.35	1.95	4.87	0.71	1.23
$\nu_4 + 2\nu_{11}$	A_1	1622.25	13.26	1.04	2.49	6.25	1.32	1.51
$\nu_3 + \nu_{11}$	E	1753.08	8.77	1.23	4.34	5.94	4.27	3.91
		1753.18	8.81	0.79	4.50	6.01	4.42	4.11

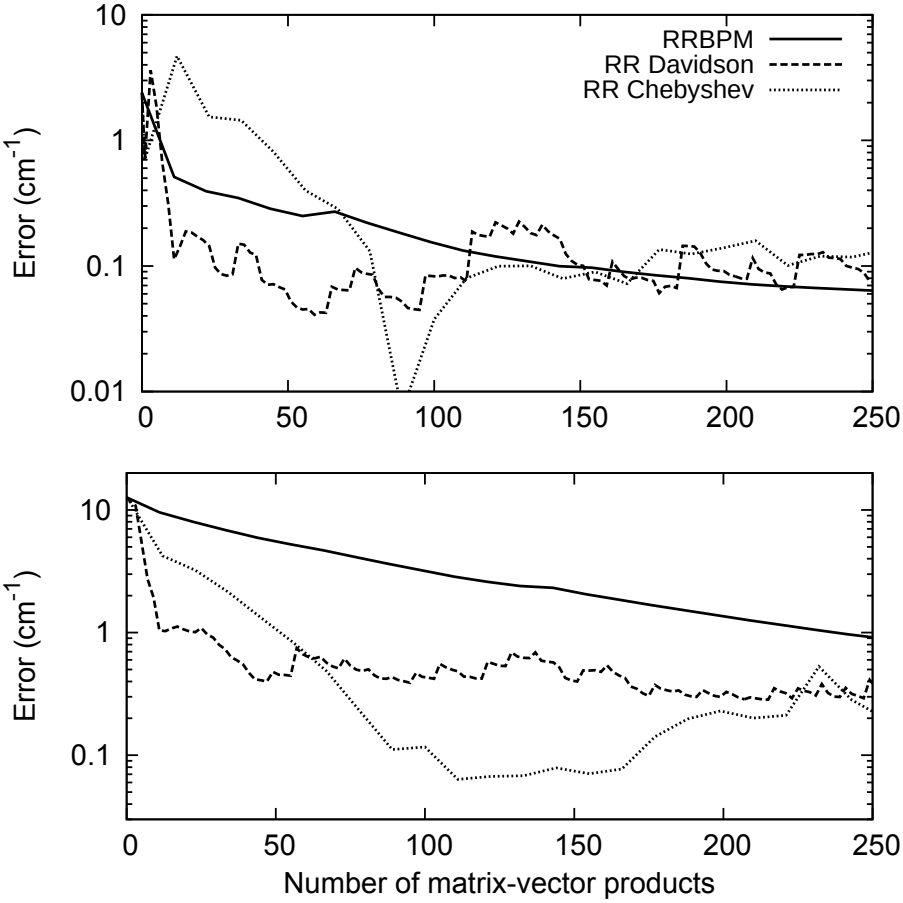

Figure 1: Convergence curves for two levels: ν_{11} (upper curves) and $\nu_4 + \nu_{11}$ (lower curves), using RRBPM (solid line), RRBD method (dashed line) and RRBC method (dotted line). The horizontal axis is the number of matrix-vector products per computed vector and the errors are calculated with respect to results of [13].

Table 2: Speed-up ratios with respect to RRBPM [37], based on the CPU time required to achieve frequencies within 2 cm^{-1} of the Smolyak values, for the first 10, 20, 25 levels above the ground state, for the reduced rank Chebyshev and Davidson methods described in section 2. The last column is obtained by omitting the orthogonalization step in the Davidson algorithm. The speed-up ratios are computed using CPU times from sequential calculations.

Number of converged levels	$\frac{T_{cpu}(\mathrm{RRCheb})}{T_{cpu}(\mathrm{RRBPM})}$	$\frac{T_{cpu}(\mathrm{RRBDav.})}{T_{cpu}(\mathrm{RRBPM})}$	$\frac{T_{cpu}(\mathrm{RRBDav.})}{T_{cpu}(\mathrm{RRBPM})}$ no orthog.
first 10	0.295	0.249	0.217
first 20	0.218	0.293	0.287
first 25	0.180	0.321	0.279

we diagonalize using a generalized eigenvalue algorithm anyway. The RRBC method is a bit more efficient than the Davidson method, the former being 4-5 times faster than the RRBPM. The largest speed-up ratio is obtained when comparing convergence for larger numbers of eigenvalues because the RRBPM generally fails to give the largest eigenvalues in the wanted block. We conclude that both reduced rank iterative methods presented in section 2 converge faster than the RRBPM with similar speed-up ratios, with the Chebyshev method being slightly faster.

All three eigensolvers require little memory and the memory cost scales linearly with dimensionality. The RRBD method has the highest memory cost, since the subspace size increases during the calculation. However, due to the use of CP format, the memory requirement is so low that it is not important. One reduced CP-vector with rank 50 takes 55 kB. After multiplication by $\mathbf{H}$, the rank becomes temporarily larger (approximately 10000) with a memory cost of 12 MB per vector.

4. Conclusion

The memory cost of variational calculations has limited them to molecules with fewer than about six atoms. Modern methods all use an iterative algorithm, based on evaluating matrix-vector products, to compute eigenvalues and eigenvectors and require storing only a few vectors in memory. Nonetheless, if a simple direct product basis is used the memory cost of such calculations is prohibitive because each vector has n^D components. Although it is possible to avoid direct product bases, they have the advantage of being simple and easy to use. It is therefore important to explore ideas that make it possible to use a direct product basis without storing n^D numbers. For potentials in SOP form this is possible if one uses an iterative eigensolver in conjunction with tensor rank reduction. One uses an iterative eigensolver to generate basis vectors, reduces their rank, and then computes eigenvalues by projecting into the space spanned by the reduced basis vectors [37]. Many variants of this idea are possible [35]. It is clear that the block power method used in Ref. [37] is not optimal. In this paper we have assessed the advantages of two other iterative eigensolvers. Both the Cheybshev and the Davidson methods significantly reduce the CPU time. The memory cost of the RRBD method is greater than that of the RRBPM of Ref. [37] due to the growth of subspace. The memory cost of the RRBC method is essentially the same as that of the RRBPM. However, the memory cost is very low (and scales linearly with D) compared to that of standard iterative direct product calculation. The RRBPM remains the simplest method to implement. Improved eigensolvers can be coupled with the hierarchical ideas of [39] that use successive contractions and intermediate diagonalizations.

Acknowledgments

Some of the calculations were done on computers purchased with a grant for the Canada Foundation for Innovation. This research was funded by the

Natural Sciences and Engineering Research Council of Canada. The PMMS (Pôle Messin de Modélisation et de Simulation) is gratefully acknowledged for providing us with computer time.

[1] C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, J. of Research of the Nat. Bur. Of Standards 45 (4) (1950) 255.

[2] J. K. Cullum, R. A. Willoughby, Lanczos algorithms for large symmetric eigenvalue computations: Vol. I: Theory, SIAM Classics in Applied Mathematics, 2002.

[3] E. Davidson, The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices, J. Comp. Phys. 17 (1975) 87.

[4] B. Liu, The simultaneous expansion for the solution of several of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices, Numerical Algorithms in Chemistry: Algebraic Method, Lawrence Berkeley Laboratory, University of California, California (1978) 49–53.

[5] F. Ribeiro, C. Iung, C. Leforestier, A jacobi-wilson description coupled to a block-davidson algorithm: An efficient scheme to calculate highly excited vibrational levels, J. Chem. Phys. 123 (5) (2005) 054106.

[6] M. J. Bramley, T. Carrington Jr, A general discrete variable method to calculate vibrational energy levels of three and four atom molecules, J. Chem. Phys. 99 (11) (1993) 8519–8541.

[7] J. M. Bowman, T. Carrington, H.-D. Meyer, Variational quantum approaches for computing vibrational energies of polyatomic molecules, Mol. Phys. 106 (16-18) (2008) 2145–2182.

[8] M. Bramley, J. Tromp, T. Carrington, G. Corey, Efficient calculation of highly excited vibrational energy levels of floppy molecules: The band origins of h+ 3 up to 35 000 cm- 1, J. Chem. Phys. 100 (1994) 6175.

[9] D. Neuhauser, Circumventing the heisenberg principle: A rigorous demonstration of filter-diagonalization on a licn model, J. Chem. Phys. 100 (7) (1994) 5076–5079.

[10] C. Iung, C. Leforestier, Direct calculation of overtones: Application to the cd3h molecule, J. Chem. Phys. 102 (21) (1995) 8453–8461.

[11] X.-G. Wang, T. Carrington Jr, A contracted basis-lanczos calculation of vibrational levels of methane: Solving the schrödinger equation in nine dimensions, J. Chem. Phys. 119 (1) (2003) 101–117.

[12] R. Dawes, T. Carrington, How to choose one-dimensional basis functions so that a very efficient multidimensional basis may be extracted from a direct product of the one-dimensional functions: Energy levels of coupled systems with as many as 16 coordinates, J. Chem. Phys. 122 (2005) 134101.

[13] G. Avila, T. Carrington, Using nonproduct quadrature grids to solve the vibrational schrÃűdinger equation in 12d, J. Chem. Phys. 134 (5) (2011) –.

[14] D. Lauvergnat, A. Nauts, Quantum dynamics with sparse grids: A combination of smolyak scheme and cubature. application to methanol in full dimensionality, Spectrochim. Acta, Part A (2014) 18.

[15] T. Halverson, B. Poirier, Large scale exact quantum dynamics calculations: Ten thousand quantum states of acetonitrile, Chem. Phys. Lett. 624 (2015) 37–42.

[16] R. Wodraszka, T. Carrington, Using a pruned nondirect product basis in conjunction with the multi-configurational time-dependent hartree (mctdh) method, J. Chem. Phys. 145 (2016) 044110–1–044110–13.

[17] G. Avila, T. Carrington, Nonproduct quadrature grids for solving the vibrational schrödinger equation, J. Chem. Phys. 131 (2009) 174103.

[18] L. Halonen, D. Noid, M. Child, Local mode predictions for excited stretching vibrational states of hccd and h 12c 13ch, J. Chem. Phys. 78 (5) (1983) 2803–2804.

[19] J. M. Bowman, S. Carter, X. Huang, Multimode: a code to calculate rovibrational energies of polyatomic molecules, International Reviews in Physical Chemistry 22 (3) (2003) 533–549.

[20] X.-G. Wang, T. Carrington, The utility of constraining basis function indices when using the lanczos algorithm to calculate vibrational energy levels, The Journal of Physical Chemistry A 105 (12) (2001) 2575–2581.

[21] M. J. Davis, E. J. Heller, Semiclassical gaussian basis set method for molecular vibrational wave functions, J. Chem. Phys. 71 (1979) 3383.

[22] T. Halverson, B. Poirier, Accurate quantum dynamics calculations using symmetrized gaussians on a doubly dense von neumann lattice, J. Chem. Phys. 137 (2012) 224101.

[23] J. Brown, T. Carrington, Assessing the utility of phase-space-localized basis functions: exploiting direct product structure and a new basis function selection procedure, J. Chem. Phys. 144 (2016) 244115–1–244115–10.

[24] A. Shimshovitz, D. J. Tannor, Phase-space approach to solving the time-independent schrödinger equation, Phys. Rev. Lett. 109 (7) (2012) 070402.

[25] S. Carter, N. C. Handy, A variational method for the determination of the vibrational (j = 0) energy levels of acetylene, using a hamiltonian in internal coordinates, Comput. Phys. Commun. 51 (1988) 49.

[26] M. J. Bramley, N. C. Handy, Efficient calculation of rovibrational eigenstates of sequentially bonded four atom molecules, J. Chem. Phys. 98 (1993) 1378.

[27] X.-G. Wang, T. Carrington, New ideas for using contracted basis functions with a lanczos eigensolver for computing vibrational spectra of molecules with four or more atoms, J. Chem. Phys. 117 (2002) 6923.

[28] H.-D. Meyer, F. Gatti, G. A. Worth (Eds.), Multidimensional Quantum Dynamics: MCTDH Theory and Applications, Wiley-VCH, Weinheim, 2009.

[29] M. H. Beck, A. Jäckle, G. A. Worth, H.-D. Meyer, The multiconfiguration time-dependent hartree (mctdh) method: a highly efficient algorithm for propagating wavepackets, Phys. Rep. 324 (2000) 1.

[30] T. G. Kolda, B. W. Bader, Tensor decompositions and applications, SIAM review 51 (3) (2009) 455–500.

[31] H. Wang, M. Thoss, Multilayer formulation of the multiconfiguration time-dependent hartree theory, J. Chem. Phys. 119 (2003) 1289–1299.

[32] U. Manthe, A multilayer multiconfigurational time-dependent hartree approach for quantum dynamics on general potential energy surfaces, J. Chem. Phys. 128 (16) (2008) 164116.

[33] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Annals of Physics 326 (1) (2011) 96 – 192, january 2011 Special Issue.

[34] I. V. Oseledets, Tensor-train decomposition, SIAM J. Sci. Comput. 33 (5) (2011) 2295–2317.

[35] M. Rakhuba, I. Oseledets, Calculating vibrational spectra of molecules using tensor train decomposition, arXiv preprint arXiv:1605.08422.

[36] G. Beylkin, M. J. Mohlenkamp, Algorithms for numerical analysis in high dimensions, SIAM J. Sci. Comput. 26 (6) (2005) 2133–2159.

[37] A. Leclerc, T. Carrington, Calculating vibrational spectra with sum of product basis functions without storing full-dimensional vectors or matrices, J. Chem. Phys. 140 (17) (2014) 174111.

[38] A. Leclerc, T. Carrington, Using symmetry-adapted optimized sum-of-products basis functions to calculate vibrational spectra, Chem. Phys. Lett. 644 (2016) 183–188.

[39] P. S. Thomas, T. Carrington Jr, Using nested contractions and a hierarchical tensor format to compute vibrational spectra of molecules with seven atoms, J. Phys. Chem. A 119 (52) (2015) 13074–13091.

[40] J. K. Watson, Simplification of the molecular vibration-rotation hamiltonian, Mol. Phys. 15 (5) (1968) 479–490.

[41] Y. Saad, Numerical Methods for Large Eigenvalue Problems, 2nd Edition, SIAM Classics in Applied Mathematics, 2011.

[42] G. Strang, Introduction to applied mathematics, Wellesley Cambridge Press, Wellesley, Massachusetts, 1986.

[43] Y. Saad, Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems, Mathematics of Computation 42 (166) (1984) 567–588.

[44] D. Begue, P. Carbonnière, C. Pouchan, Calculations of vibrational energy levels by using a hybrid ab initio and dft quartic force field: Application to acetonitrile, J. Phys. Chem. A 109 (2005) 4611–4616.