Computing vibrational energy levels using a canonical polyadic tensor method with a fixed rank and a contraction tree

Sangeeth Das Kallullathil[a] and Tucker Carrington Jr.[b]

Chemistry Department, Queen's University,

Kingston, Ontario K7L 3N6, Canada

(Dated: May 2, 2023)

Abstract

In this paper, we use the previously introduced CP-Multiple Shift Block Inverse Iteration (MSBII) eigensolver [J. Chem. Phys. 155, 234105 (2021)] in conjunction with a contraction tree to compute vibrational spectra. The CP-MSBII eigensolver uses the Canonical Polyadic (CP) format. The memory cost scales linearly with the number of coordinates. A tensor in CP format represents a wavefunction constrained to be a sum of products (SOP). An SOP wavefunction can be made more and more accurate by increasing the number of terms, the rank. When the required rank is large, the runtime of a calculation in CP format is long, although the memory cost is small. To make the method more efficient we break the full problem into pieces using a contraction tree. The required rank for each of the sub-problems is small. To demonstrate the effectiveness of the ideas, we computed vibrational energy levels of acetonitrile (12-D) and ethylene oxide (15-D).

[a] Electronic address: 17sdk2@queensu.ca
[b] Electronic address: Tucker.Carrington@queensu.ca

I. INTRODUCTION

Methods for computing the vibrational spectrum of a molecule can be divided into two types: those which require that the potential energy surface (PES) have a special form and those that can be used with a general PES. A common special form is the sum-of-product (SOP) form and there are many methods designed to work with SOP PESs.[1–14] In this paper we modify our previous canonical polyadic (CP) multiple shift block inverse iteration (MSBII) method[15] for computing a spectrum with a SOP PES by incorporating a tree structure and an intertwining idea[5]. For a given molecule, the most accurate possible PES will not be a SOP. Methods which rely on the PES having SOP form are nonetheless useful because: (1) any PES can, in principle, be re-written as a SOP and (2) for molecules with more than about 6 atoms it is often true that the best available PES is a SOP (e.g. a "force field")[16]. There are several methods for converting a general PES into SOP form[17–22]. A spectrum is usually computed with a variational method and in most cases the cost of a variational calculation with a SOP PES depends linearly on the number of terms in the SOP.

Many variational methods designed to be used with a SOP PES exploit some sort of compact tensor representation of wavefunctions. The simplest methods represent multi-dimensional wavefunctions in a direct product basis, which is built from products of univariate functions. They require storing coefficient vectors with n^D elements, n is a representative number of basis functions for a single coordinate and D is the dimensionality or the number of coordinates. When computing a vibrational spectrum, $D = 3N_{atoms} - 6$, where N_{atoms} is the number of atoms in the molecule and n is frequently about 10. For molecules with more than about 5 atoms, n^D is so large that vectors cannot be stored in computer memory. The most popular tensor format is called the Tucker format and is used in multi-configuration time-dependent Hartree (MCTDH) methods[2,23]. The corresponding basis is a direct product basis with optimized univariate functions. The optimization reduces the number of coefficients from n^D to $(n_{opt})^D$. The matrix product states or tensor train format is sometimes used to compute spectra by solving the time-dependent Schrödinger equation and Fourier transforming[12]. It is also used in density-matrix renormalization group methods, which are typically used when only a small number of energies (as in electronic structure calculations) are required[25]. DMRG has also been applied to vibrational problems[26,27]. Recent

methods using the MPS format and applied to vibrational problems combine DMRG ideas with iterative eigensolvers to calculate many vibrational states[13,28]. The FEAST-DMRG combination[13] is similar to the original MSBII[15], but uses a different tensor format and FEAST[29] rather than multiple shift block inverse iteration. An important difference between DMRG methods and CP methods is that DMRG methods store and manipulate matrices whereas CP methods store and manipulate vectors. Larsson has proposed tensor network methods that use DMRG-type methods to optimize parameters and a hierarchical Tucker format[14]. There are also tensor methods using CP format[3–5,15,24]. A CP vector is written,

$$\mathbf{F} = \sum_{\ell=1}^{R} \bigotimes_{c=1}^{D} \mathbf{f}_{c,\ell} \ , \tag{1}$$

where $\otimes$ indicates an outer product[30–32]. In terms of components this equation becomes,

$$F_{i_1,i_2,..,i_d} = \sum_{\ell=1}^{R} \prod_{c=1}^{D} f_{i_c}^{(c,\ell)} \tag{2}$$

The memory required to store one CP tensor scales as DRn, i.e., linearly with D. In a standard direct-product-basis calculation the memory cost scales as n^D. In all CP methods, the number of terms, called the rank and denoted R, is a critical parameter. If R is large then despite the fact that the memory cost scales only linearly with D the runtime of a CP calculation is so long that it not practical. Some of these methods require reducing the rank of CP tensors generated during the calculation[3–5]. We recently proposed a CP method without rank reduction[15].

Tensor format methods reduce the memory cost of variational methods and make it possible to solve the Schrödinger equation for the vibration of 12 or more coupled coordinates. However, the cost of calculations can be further reduced by breaking the problem up into pieces each of which has a subset of the total number of coordinates. The utility of this idea in the MCTDH context was recognized long ago[33–35]. It has also been exploited in the CP context[4,5]. When the coupling between groups of coordinates is weaker than the coupling of coordinates within the groups, it is always beneficial. Of course, the tree complicates the writing of the necessary computer code. There is recent new research about how to choose the groups of coordinates[36]. When the coordinates are divided into successively larger groups, it is helpful to organize the groups into a tree[34] with nodes in layers. Nodes in

higher layers contain more coordinates. Assuming that wavefunctions for all of the groups are represented in the same format, the final full-dimensional wavefunctions are represented in a hierarchical tensor format[37]. In this paper we propose a hierarchical version of our earlier CP-MSBII approach.

The CP methods we have used in our group are all built on the idea that eigenvalues of a Hamiltonian matrix in a direct product basis can be computed by projecting into a space spanned by basis vectors. We first proposed reduced-rank block power (RRBP) based methods in which the vectors are obtained by evaluating matrix-vector products (MVPs) with the matrix that represents the Hamiltonian in a primitive direct product basis, $\mathbf{H}$. Because the Hamiltonian is a SOP operator, $\mathbf{H}$ is a sum of Kronecker products of $n \times n$ matrices that represent the factors in the terms of the SOP. When $\mathbf{H}$ is applied to a CP tensor, a new CP tensor is obtained.

$$\mathbf{HF} = \left[\sum_{t}^{T}\bigotimes_{c=1}^{D}\mathbf{h}_c^t\right]\left[\sum_{\ell=1}^{R}\bigotimes_{c=1}^{D}\mathbf{f}_{c,\ell}\right] = \sum_{\ell}^{RT}\bigotimes_{c=1}^{D}\left[\mathbf{h}_c^t\mathbf{f}_{c,\ell}\right]. \tag{3}$$

It is a also sum of outer products of vectors for each coordinate. The rank of the new CP tensor is RT, where T is the number of terms in the Hamiltonian. In RRBPM methods, an alternating least squares (ALS) method was used to reduce the rank of the new vector. In an ALS method, parameters (in this case elements of $\mathbf{f}_{c,\ell}$ vectors) for all but one coordinate are fixed and parameters for a single coordinate are optimised. When using ALS, linear systems with an $R \times R$ matrix are solved. In Refs. [3,4] vectors for all the coordinates are optimized after evaluating a MVP. In Ref. [5], the optimization of the factors was intertwined with the MVPs and, after each MVP, vectors for only one coordinate were optimized. In effect, we used vectors for coordinate c to accelerate the convergence of vectors for coordinates $j > c$. In this paper we do something similar to accelerate the convergence of the CP-MSBII method.

In Ref. [15], we make the basis into which the Hamiltonian matrix is projected not by using the shifted (block) power method but by solving linear systems[38]

$$(\mathbf{H} - \sigma_v\mathbf{I})\mathbf{F}_v = \mathbf{s}_v\ . \tag{4}$$

In this equation, $\mathbf{s}_v$ and $\mathbf{F}_v$ are tensors in CP format. Basis vectors are made from the $\mathbf{F}_v$. $\mathbf{F}_v$ is a linear combination of the eigenvectors of $\mathbf{H}$ whose eigenvalues are close to σ_v. A basis composed of $\mathbf{F}_v$ vectors computed with σ_v values distributed between E_{min} and

E_{max} is ideal for calculating eigenvalues in this range. Very similar ideas are responsible for the success of Filter Diagonalization[39–41], FEAST[13], and Rational Krylov methods[42]. In standard Filter Diagonalization, FEAST, and Rational Krylov methods, to store $\mathbf{F}_v$ it is necessary to store vectors with as many elements as the size of $\mathbf{H}$. That is not the case whenever a tensor format is used, e.g. CP format in this paper and in 15 or MPS format in Refs. 28 and 13. In CP-MSBII, Eq. (4) is solved with an ALS method that constrains the rank of $\mathbf{F}_v$ to be R. There are no vectors with rank larger than R and no need to reduce the rank of vectors or orthogonalize vectors. In terms of memory and runtime, the CP-MSBII method is more efficient than the previous method, RRBPM[3]. However, it is less efficient than a hierarchical version of the RRBPM[4] that breaks the full problem up into sub-problems and uses the RRBPM for each of them. Breaking the problem up of course reduces the total cost. In this paper we break the problem up into sub-problems and solve each of them not with the RRBPM, but with CP-MSBII. In addition, we implement an intertwining idea to accelerate the convergence of vectors for coordinates $j > c$, by using vectors already computed for other coordinates. The hierarchical/intertwining approach of this paper is significantly faster than the pure CP-MSBII method of Ref. [15].

II. CP MULTIPLE SHIFT BLOCK INVERSE ITERATION (CP-MSBII)

In practice, we choose B sets of S σ_v^b values and solve,

$$(\mathbf{H} - \sigma_v^b \mathbf{I})^{-P} \mathbf{s}_v^b = \mathbf{F}_v^b \ . \tag{5}$$

Each set is labelled by b and $b = 1, \ldots, B$. A set includes values labelled by v and $v = 1, \ldots, S$. A set is referred to as a block. For a given value of b, the S σ_v^b values are chosen to be between E_{min} and E_{max}. As explained in the introduction, $\mathbf{F}_v^b$ is a filtered vector that has significant overlap with only a few eigenvectors. P in this paper is typically about three. Using $P > 1$ yields a better basis because it decreases the number of eigenvectors that have significant overlap with $\mathbf{F}_v^b$. We compute eigenvalues by projecting into the space spanned by the BS filtered vectors. FEAST-DMRG[13] uses basis vectors that are sums of filtered vectors, rather than filtered vectors.

$(\mathbf{H} - \sigma_v^b \mathbf{I})^{-P}$ is applied to $\mathbf{s}_v^b$ by solving P sets of linear equations. $\mathbf{s}_v^b$ and $\mathbf{F}_v^b$ are in CP format,

$$\mathbf{F}_v^b = \sum_{\ell=1}^{R} \bigotimes_{c=1}^{D} \mathbf{f}_{c,\ell} \ , \tag{6}$$

and

$$\mathbf{s}_v^b = \sum_{\ell=1}^{R} \bigotimes_{c=1}^{D} \mathbf{g}_{c,\ell} \ . \tag{7}$$

and

$$(\mathbf{H} - \sigma_v^b \mathbf{I}) = \sum_{t=1}^{T+1} \bigotimes_{c=1}^{D} \mathbf{h}_c^t. \tag{8}$$

$\mathbf{f}_{c,\ell}$ and $\mathbf{g}_{c,\ell}$ actually depend on v and b but to avoid cluttered notation that dependence is not indicated. In Eq. (8), some of the matrices $\mathbf{h}_c^t$ are identity matrices; the others represent p_c, q_c, q_c^2, q_c^3 and q_c^4.

An alternating least square (ALS) algorithm is used to approximately solve the linear systems. It converts the $n^D \times n^D$ problem in Eq. (5) into a set of $nR \times nR$ problems. The $\ell = 1, ..., R$ vectors $\mathbf{f}_{j,\ell}$ for a single coordinate j are stacked next to each other to form a $n \times R$ matrix which we call $\boldsymbol{\mathcal{F}}_j$. We obtain values for all elements of $\boldsymbol{\mathcal{F}}_j$ by fixing $\boldsymbol{\mathcal{F}}_c$, $c \neq j$ and solving

$$\left[\sum_{t=1}^{T+1} \left(\bigodot_{c\neq j}^{D} (\boldsymbol{\mathcal{F}}_c^T \mathbf{h}_c^t \boldsymbol{\mathcal{F}}_c) \right) \otimes \mathbf{h}_j^t \right] vec(\boldsymbol{\mathcal{F}}_j) = \sum_{\ell''=1}^{R} \left(\bigodot_{c\neq j}^{D} (\boldsymbol{\mathcal{F}}_c^T \mathbf{g}_{c,\ell''}) \right) \otimes \mathbf{g}_{j,\ell''} \ . \tag{9}$$

$\bigodot$ is a Hadamard product, i.e. element-wise multiplication. $vec(\boldsymbol{\mathcal{F}}_j)$ is obtained by reshaping $\boldsymbol{\mathcal{F}}_j$ into an $nR \times 1$ vector. The recipes we use to choose the start vectors and the σ_v^b are explained in Ref. [15]. Eq. (9) is solved for each of the coordinates in turn. The calculation of $\mathcal{F}_j$ for all coordinates $j = 1, 2, ..., D$ is referred to as a "sweep". N_{ALS} sweeps are done to compute $\mathbf{F}_v^b$. In most of our calculations N_{ALS}=1. Solutions obtained from ALS are only approximate (even if $N_{ALS} \gg 1$), but because $\mathbf{F}_v^b$ is only a basis vector, numerically exact eigenvalues of $\mathbf{H}$ can nonetheless be obtained. The solution of Eq. (9) is parallelized over σ_v^b.

Let $\mathcal{V}$ be the matrix whose columns are the filtered vectors,

$$\mathcal{V} = \left[\mathbf{F}_1^1, \mathbf{F}_2^1, ..., \mathbf{F}_S^B\right], \tag{10}$$

We solve the generalized eigenvalue problem

$$\mathbf{H}^{\mathcal{V}}\mathbf{Y} = \mathbf{S}^{\mathcal{V}}\mathbf{Y}\mathbf{E} \ , \tag{11}$$

where $\mathbf{H}^{\mathcal{V}} = \mathcal{V}^T\mathbf{H}\mathcal{V}$ and $\mathbf{S}^{\mathcal{V}} = \mathcal{V}^T\mathcal{V}$. The matrix of eigenvectors of $\mathbf{H}$ is

$$U = \mathcal{V}\mathbf{Y} \ . \tag{12}$$

Near linear dependence is mitigated using singular value decomposition in the standard fashion.

A detailed algorithm is given for CP-MSBII in Ref. [15]. Linear equations are solved for all D coordinates and outside the loop over coordinates is a loop over sweeps. In the RRBPM there is a similar loop over sweeps that is eliminated by intertwining[5].

Values of the P, N_{ALS}, S, B, and R are important. All of these parameters are interrelated. For example, if P is larger then the filtered vectors have overlap with fewer eigenvectors of $\mathbf{H}$ and two filtered vectors may have significant overlap with only one and the same eigenvector; this can cause the algorithm to fail to compute some eigenvalues unless S is increased.

III. HIERARCHICAL CP MULTIPLE SHIFT BLOCK INVERSE ITERATION (H-CP-MSBII)

In the H-CP-MSBII, we divide coordinates into groups that are associated with nodes and solve a Schrödinger equation for each node. Linked with each node is a node Hamiltonian and a set of its eigenfunctions. The nodes are organized into a tree. An example of a tree for ethylene oxide is given in Figure 1. In Figure 1, a node is labelled with the size of its eigenfunction basis. Each node has children. Eigenfunctions for a node are computed in a direct product basis whose functions are products of eigenfunctions of the node's children. For a given node, the number of eigenfunctions retained is a convergence parameter. As the size of the node bases increases the full multi-D (multi-dimensional) wavefunctions at the top of the tree converge. Retaining too few node eigenfunctions will introduce error into all higher nodes in the tree (including the top node which has the final full-dimensional wavefunctions). The tree in Figure 1 has 6 layers. Figure 1 represents a multi-layer contraction. Simpler contraction trees with only two layers are more common[43].

The CP-MSBII algorithm can only be used with a direct product basis. In figure 1, many of the direct product basis functions are products of multivariate functions. It is cheaper to use the CP-MSBII to compute eigenfunctions of a node than to compute the full-dimensional wavefunction because for a single node the dimensionality is reduced. As a consequence, there are fewer levels, the spacing of energy levels is more uniform and it is possible to obtain converged levels with smaller values of P, S, B, N_{ALS}, and R.

In this paper we make no attempt to optimize the trees. We simply use trees from previous papers[4,5]. In the tree of Figure 1, the individual coordinates are at the bottom of the tree and 1-D Schrödinger equations are solved to determine the bases in the second layer. It is crucial that coordinates for which coupling is important be combined into nodes near the bottom of the tree. We use dimensionless normal coordinates and neglect the $\boldsymbol{\pi}^T\boldsymbol{\mu}\boldsymbol{\pi}$ term and the potential-like term in the kinetic energy operator[44].

A node Hamiltonian is made by discarding all terms from the kinetic energy operator for coordinates not in the node and putting $q_i = 0$ in the potential energy for all coordinates that are not in the node. In some cases, a node has a single child, this means that functions (perhaps not all of them) of the child node are transferred to the parent node.

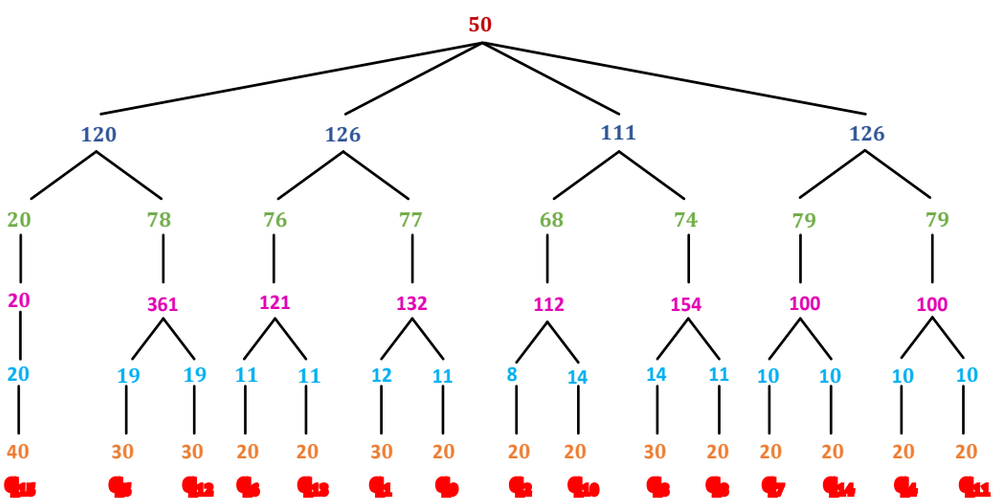


FIG. 1: Tree structure with basis size for the ethylene oxide molecule

A node in a tree is labelled with its layer number, ℓ. The nodes in a layer are numbered

from left to right and labelled with n. In the third layer of the tree in Figure 1, the Hamiltonian of the node for which 361 functions are retained is written H_2^3. We denote the basis size of the node associated with ℓ and n by η_n^ℓ. Each node Hamiltonian can be written as a sum of terms for the children and one coupling term. For example,

$$H_2^3 = H_2^2 + H_3^2 + {}^{couple}H_{2,3}^3 \ . \tag{13}$$

In the basis of eigenfunctions of its children

$$\mathbf{H}_2^3 = \mathbf{E}_2^2 \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{E}_3^2 + {}^{couple}\,\mathbf{H}_{2,3}^3 \ . \tag{14}$$

In Eq. (14), each term is a $19^2 \times 19^2$ matrix (see Figure 1), but because each term is a Kronecker product or a sum of Kronecker products, matrices this large are not stored. $\mathbf{E}_2^2$ is a diagonal matrix whose diagonal elements are the smallest eigenvalues of the 30×30 matrix that represents H_2^2. Every term in ${}^{couple}H_{2,3}^3$ depends on coordinates of at least two children. ${}^{couple}\mathbf{H}_{2,3}^3$ is the matrix representing ${}^{couple}H_{2,3}^3$ in the direct product basis formed from 19 eigenfunctions of H_2^2 and 19 eigenfunctions of H_3^2. When a coupling term has two or more factors that are powers of coordinates in a single child then the product of the factor matrices is not symmetric and must be symmetrized.

A key parameter in the cost of a CP-MSBII calculation is the rank, R. R is the number of terms in a filtered vector. The rank of the final wavefunctions obtained by solving the generalized eigenvalue problem is BSR. The greater the coupling or the entanglement, the larger R must be. In practice, the required value of R is large if T, the number of terms in the Hamiltonian is large. A great advantage of a hierarchical approach is that for each of the sub-problems solved with CP-MSBII, T is small. For $\mathbf{H}_2^3$, $T = 2 + {}^{couple}T_{2,3}^3$, where ${}^{couple}T_{2,3}^3$ is the number of terms in ${}^{couple}\mathbf{H}_{2,3}^3$. ${}^{couple}T_{2,3}^3$ can be reduced by using the factorization scheme outlined in Ref. [4]. Using factorization means that two terms in ${}^{couple}H_{2,3}^3$ such as $q_5^2 q_{12}^2$ and $q_5^1 q_{12}^2$ can be combined (reducing ${}^{couple}T_{2,3}^3$) into a single term $(q_5^2 + q_5^1) q_{12}^2$.

Information from calculations in layer ℓ of the tree is passed to layer $\ell + 1$ in the form of matrices representing q_c, q_c^2, $q_c^3 \forall c$ and the diagonal matrices representing node Hamiltonians in layer ℓ in the basis of their eigenvectors. We keep just one set of matrices in memory, after they are used they are discarded. For example, in the third layer of the tree in Figure 1, matrices representing q_5 and q_{12} in the basis of the 361 eigenfunctions of H_2^3 are computed from $\mathbf{q}_5^{3,2} = \mathbf{U}_{3,2}^T (\mathbf{q}_5^{2,2} \otimes \mathbf{I}^{2,3}) \mathbf{U}_{3,2}$, where $\mathbf{q}_5^{3,2}$ is the matrix that represents q_5 in the basis of

eigenfunctions of H_2^3, $\mathbf{U}_{3,2}$ is the matrix whose columns are selected eigenvectors of $\mathbf{H}_2^3$, $\mathbf{q}_5^{2,2}$ is the matrix representing q_5 in the basis of eigenvectors of $\mathbf{H}_2^2$, and $\mathbf{I}^{2,3}$ is an identity matrix with as many rows and columns as $\mathbf{H}_3^2$.

How does one compute and store matrices like $\mathbf{U}_{3,2}$? $\mathbf{U}_{\ell,n}$ has $\prod_{n\in\{K_n^\ell\}} \eta_n^{\ell-1}$ rows; $\{K_n^\ell\}$ is the set of nodes that are the children of the node labelled by ℓ, n. Near the top of the tree, storing $\mathbf{U}_{\ell,n}$ would be costly. However, we calculate $\mathbf{U}_{3,2}$ as a product of a matrix of filtered vectors for $\mathbf{H}_2^3$ (see Eq. (10)) and a matrix of eigenvectors of a generalized eigenvalue problem (see Eq. (11)). The matrix of eigenvectors of the generalized eigenvalue problem is small and is easily stored. The columns of the matrix of filtered vectors are stored in CP format with a memory cost that scales linearly in the number of children of the node. As an alternative, one might store $\mathbf{U}_{3,2}$ in CP format but the rank of the vectors that are its columns is large.

IV. INTERTWINING

In this paper, we modify the CP-MSBII method in two ways. First, we use it in a hierarchical fashion, which is explained in Section III. Second, we intertwine updating the right side of Eq. (9) with the calculation of the $\boldsymbol{\mathcal{F}}_j$. In Ref. [15], in a single sweep, $\boldsymbol{\mathcal{F}}_j$ $\forall j$ were computed without changing the right side of Eq. (9). From the $\boldsymbol{\mathcal{F}}_j$, we make the filtered vectors $\mathbf{F}_v^b$. We expect $\mathbf{F}_v^b$ to have significant overlap with eigenvectors of $\mathbf{H}$ whose eigenvalues are close to σ_v^b. The overlaps will be bigger, if the right side, i.e. $\mathbf{s}_v^b$, is itself close to being an eigenvector of $\mathbf{H}$ whose corresponding eigenvalue is close to σ_v^b. We therefore update the right side of Eq. (9) as we loop over the coordinates, labelled by j. The new algorithm for solving Eq. (9) is given in Algorithm 1. It replaces the function CPII in Algorithm 1 in Ref [15]. The matrices $\mathbf{M}_{\forall j}^t$ and $\mathbf{P}_{\forall j}$ are defined in Ref. [15]. There are two differences between the new and the old algorithm. First, in Algorithm 1 there is no loop over α from $\alpha = 1$ to $\alpha = N_{ALS}$. Second, in step 2. (h), the right side is updated by replacing g_{j,ℓ^v} with f_{j,ℓ^v}. Intertwining enables us to reduce P.

Algorithm 1 Function $[\mathbf{F}] = \mathrm{CPII}(\mathbf{s},\sigma)$

1. Initialize

(a) Assign $(\mathbf{f}_{1,\ell}, ..., \mathbf{f}_{D,\ell}) \leftarrow (\mathbf{g}_{1,\ell}, ..., \mathbf{g}_{D,\ell}) \quad \forall\ \ell$

(b) Construct $\mathbf{M}^t_{\forall c}$: $M^t_{\forall c}(\ell', \ell) = \prod_{c=1}^{D} (\mathbf{f}^T_{c,\ell'} \mathbf{h}^t_c \mathbf{f}_{c,\ell}) \quad \forall\ \ell, \ell'$

(c) Construct $\mathbf{P}_{\forall c}$: $P_{\forall c}(\ell', \ell'') = \prod_{c=1}^{D} (\mathbf{f}^T_{c,\ell'} \mathbf{g}_{c,\ell''}) \quad \forall\ \ell, \ell'$

2. Solve for CP parameters

for $j = 1$ to D

(a) Compute $\mathbf{M}^t_{\neq j}$: $M^t_{\neq j}(\ell', \ell) = M^t_{\forall c}(\ell', \ell) / M^t_j(\ell', \ell) \quad \forall\ \ell, \ell'$

(b) Compute $\mathbf{P}_{\neq j}$: $P_{\neq j}(\ell', \ell'') = P_{\forall c}(\ell', \ell'') / P_j(\ell', \ell'') \quad \forall\ \ell', \ell''$

(c) Compute $\mathbf{A}$: $A(\ell', i'_j, \ell, i_j) = \sum_{t=1}^{T+1} M^t_{\neq j}(\ell', \ell) h^t_j(i'_j, i_j) \quad \forall\ \ell', i'_j, \ell, i_j$

(d) Compute $\mathbf{b}$: $b(\ell', i'_j) = \sum_{\ell''=1}^{R} P_{\neq j}(\ell', \ell'')\ g^{(j,\ell'')}_{i'_j} \quad \forall\ \ell', i'_j$

(e) Solve the linear system for $\mathbf{x}$: $A(\ell', i'_j, \ell, i_j) x(\ell, i_j) = b(\ell', i'_j) \quad \forall\ \ell, i_j$

(f) Update $\mathbf{f}_{j,\ell}$ by replacing $f^{(j,\ell)}_{i_j} \leftarrow x(\ell, i_j) \quad \forall\ \ell, i_j$

(g) Compute $\mathbf{M}^t_j$: $M^t_j(\ell', \ell) = \mathbf{f}^T_{j,\ell'} \mathbf{h}^t_j \mathbf{f}_{j,\ell} \quad \forall\ \ell, \ell', t$

(h) Compute $\mathbf{P}_j$: $P_j(\ell', \ell'') = \mathbf{f}^T_{j,\ell'} \mathbf{f}_{j,\ell''} \quad \forall\ \ell'', \ell'$

(i) Update $\mathbf{M}^t_{\forall c}$ with new parameters $M^t_{\forall c}(\ell', \ell) = M^t_{\neq j}(\ell', \ell) M^t_j(\ell', \ell) \quad \forall\ \ell, \ell', t$

(j) Update $\mathbf{P}_{\forall c}$ with new parameters $P_{\forall c}(\ell', \ell'') = P_{\neq j}(\ell', \ell'') P_j(\ell', \ell'') \quad \forall\ \ell'', \ell'$

3. Normalize $\mathbf{F} \leftarrow \frac{\mathbf{F}}{\|\mathbf{F}\|}$

V. CHOOSING THE PARAMETERS

To use the CP-MSBII method of Ref. [15], one must choose values of N_{ALS}, P, B, R, and S. To use the H-CP-MSBII method of this paper, all these parameters, except N_{ALS}, must be chosen for each of the nodes in the contraction tree. In this section, we outline our systematic way of choosing the parameters. We use the same P, B, and R for all the nodes. It is surely possible to decrease the cost of the calculations by using different values for different nodes. For example, it is probably possible to use a smaller P for nodes with less closely spaced levels. Some compromises must be made to avoid the need to choose too many parameter values. To simplify the parameter choices we take values that are good enough for the most difficult nodes. For calculations in this paper, we choose $P = 2$ and $P = 3$. In Ref.

[15], larger P values were necessary. Increasing P makes the filtered vectors better basis vectors in the sense that fewer vectors are required to obtain converged energies. Making B larger increases the size of the basis used to solve the generalized eigenvalue problems. We increase R to achieve convergence. R values required with a tree are much smaller than those required with no tree.

It is important that S be different for different nodes. The number of node eigenfunctions that we must retain is linked to the S value used to compute those eigenfunctions. For the node labelled by ℓ, n, it is crucial that S be larger when η_n^ℓ is large. The S σ_v^b values for a given node are chosen to be within limits $E_{\rm min}$ and $E_{\rm max}$ that are determined by the energies of the children of the node. $E_{\rm min}$ and $E_{\rm max}$ define an energy window in which we seek converged energies. Sums of energies of the children nodes are sorted into a list in increasing order. $E_{\rm min}$ is somewhat less than ${\rm Min}_{\rm n}^\ell$, where ${\rm Min}_{\rm n}^\ell$ is the sum of the ground state energies of the children nodes,

$$E_{\rm min} = {\rm Min}_{\rm n}^\ell - [{\rm Max}_{\rm n}^\ell - {\rm Min}_{\rm n}^\ell]/\eta_{\rm n}^\ell \ , \tag{15}$$

where ${\rm Max}_{\rm n}^\ell$ is the η_n^ℓ'th element in the list. $E_{\rm max}$ is,

$$E_{\rm max} = (1+a_1)[{\rm Max}_{\rm n}^\ell - {\rm Min}_{\rm n}^\ell] + {\rm Min}_{\rm n}^\ell \ . \tag{16}$$

$a_1 < 1$ is large enough to ensure that the basis of filtered vectors is sufficient for converging all levels up to and including the largest eigenvalue of the node Hamiltonian. It is required because filtered vectors have significant overlaps with $\mathbf{H}$ eigenvectors whose eigenvalues are slightly larger than the largest σ_v^b. a_1 is about 0.3 or 0.5 and is different for different molecules, depending on the density of states.

Once S is determined, we need a recipe for placing the σ_v^b between $E_{\rm max}$ and $E_{\rm min}$. We build a set of σ_v^b iteratively. In the first iteration, there are η_n^ℓ σ_v^b equal to the energies in the list of sums of energies of the children nodes between $E_{\rm max}$ and $E_{\rm min}$. In subsequent iterations, we add additional σ_v^b between the closest pairs of σ_v^b in the previous iteration. The total number of additional σ_v^b is not the same for each node. Instead, the number of additional σ_v^b is $a_2\eta_n^\ell$. a_2 is between 0.4 and 0.7 and different for different molecules and larger when the density of states is greater. Increasing by a percentage in this way ensure that more σ_v^b are added for nodes for which η_n^ℓ is larger. We add a second set of additional σ_v^b between the closest pairs of σ_v^b in the top 30% of the window.

VI. RESULTS AND DISCUSSION

We have done test calculations on acetonitrile (12-D) and ethylene oxide (15-D). We use the same SOP PESs used in previous papers [4,5,10,45,46]. We use the contraction trees in Refs. [4,5].

A. Acetonitrile (CH_3CN)

The lowest 50 energy levels of acetonitrile were calculated using the H-CP-MSBII method. The tree is shown in Figure 2. The parameters used for the calculation are presented in Table I. The lowest 5 levels and the highest 5 levels are shown in Table II.

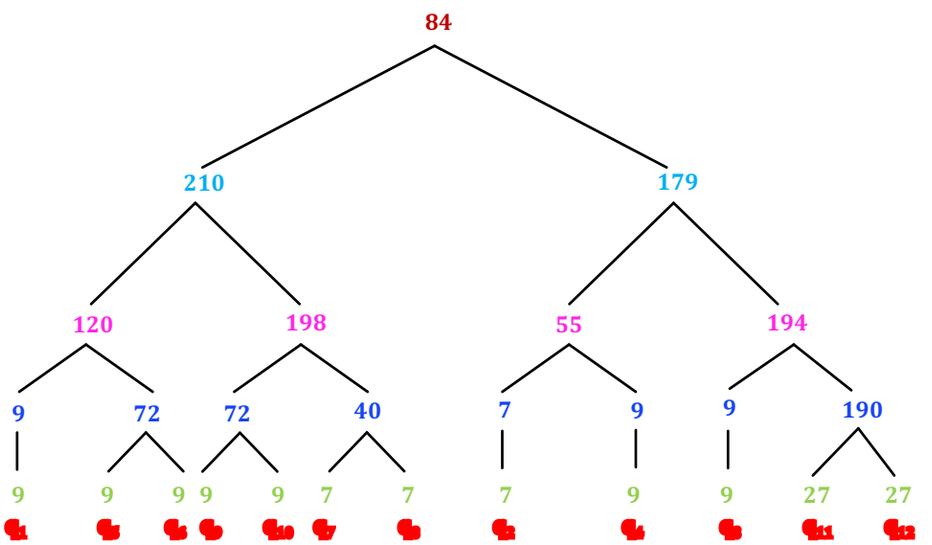


FIG. 2: Tree and basis sizes for acetonitrile

TABLE II: Some vibrational energy levels (cm^{-1}) of CH_3CN calculated using H-CP-MSBII. The Zero Point Energy (ZPE) and differences of other levels and the ZPE are listed. Reference values are taken from Ref. [45]

Level	Sym.	Reference	Rank=20, B=2	E-E_{ref}
ZPVE		9837.4073	9837.420	0.013
ν_{11}	E	360.991	360.99	0.00
		360.991	360.99	0.00
$2\nu_{11}$	E	723.181	723.21	0.03
		723.181	723.21	0.03
⋮	⋮	⋮	⋮	⋮
$\nu_7+\nu_{11}$	E	1844.33	1844.47	0.14
		1844.33	1844.51	0.18
$\nu_7+\nu_{11}$	A_1	1844.69	1844.87	0.18
$\nu_4+\nu_9$	E	1931.547	1931.74	0.20
$\nu_4+\nu_9$	E	1931.547	1931.88	0.24

TABLE I: Parameters for the calculation on acetonitrile

Parameter	Value(s)
Rank	20
n_c (primitive)	9, 7, 9, 9, 9, 9, 7, 7, 9, 9, 27, 27
P	2
B	2
a_1	0.3
a_2	0.4

The H-CP-MSBII calculation is much more efficient than the previous CP-MSBII calculation. Nothing is done to enforce degeneracy and the splitting between degenerate components is there a good measure of the accuracy of the calculation. The CP-MSBII calculation of the same energy levels required a rank of 400, and achieved about the same accuracy. The H-CP-MSBII method requires a rank of 15. The H-CP-MSBII calculation requires 1

GB of memory and a factor of about 24 less CPU time than the calculation in Ref. 15.

B. Ethylene oxide (C_2H_4O)

We also computed the lowest 50 energy levels of ethylene oxide (EO) for which $D = 15$. The SOP PES has 655 terms, many more than the PES for CH_3CN. There are 180 cubic terms and 445 quartic terms. The larger the number of terms, the higher is the required rank. The tree for the EO molecule, shown in Figure 1. Eigenvectors of nodes with only two children are computed with a direct eigensolver. CP-MSBII is used at the top of the tree. The lowest and highest 5 energy levels are presented in Table IV, and compared with values from VMFCI, HI-RRBPM, and HRRBPM calculations. VMFCI uses successive contractions and eigenfunctions and matrices are stored for each contraction[10]. HI-RRBPM and HRRBPM are RRBPM methods that use a tree, the former is with intertwining. The parameters of the H-CP-MSBII method used in this calculation are given in Table III.

TABLE III: Parameters for the calculation on ethylene oxide

Parameter	Value(s)
Rank	20
n_c (primitive)	40, 30, 30, 20, 20, 30, 20, 20, 20, 30, 20, 20, 20, 20, 20
P	3
B	30
a_1	0.5
a_2	0.7

TABLE IV: Some vibrational energy levels (cm^{-1}) of ethylene oxide calculated using H-CP-MSBII. The first level is the Zero Point Energy (ZPE) and other levels are differences with the ZPE.

Symmetry	Assignment	VMCI[5,10]	HI-RRBPM[5]	H-CP-MSBII
A_1	ZPE	12463.65	12461.65	12461.73

B_2	ν_{15}	793.37	793.10	793.29
B_1	ν_{12}	822.75	822.00	822.20
A_1	ν_5	879.04	878.33	878.58
A_2	ν_8	1018.96	1017.49	1017.82
⋮	⋮	⋮	⋮	⋮
B_2	$\nu_7+\nu_{11}$	2266.28	2266.13	2266.49
B_2	$\nu_{14}+\nu_4$	2269.43	2268.67	2269.09
A_2	$\nu_{14}+\nu_{11}$	2269.27	2268.95	2269.40
A_2	$\nu_7+\nu_4$	2276.17	2275.77	2276.29
B_2	$\nu_{15}+\nu_2$	2284.40	2284.97	2284.97

VII. CONCLUSION

When a PES has SOP form, various tensor methods can be used to compute the vibrational spectrum of a molecule. All tensor methods have the advantage that they reduce the memory cost of computing a spectrum. The memory cost of a simple direct product calculation scales as n^D and this makes calculations impossible for molecules with more than 5 atoms. A CP tensor method requires a SOP PES and represents wavefunctions as SOPs. The memory cost of CP tensor methods scales linearly with D. All CP methods compute wavefunctions by projecting into a basis of functions each of which has CP format and R terms. R is the key convergence parameter. The larger the coupling or the entanglement, the larger R must be. Our previous CP-MSBII[15] uses CP basis vectors, but obviates the need for rank reduction and orthogonalization. It was shown to be much more efficient than the simple RRBPM of Ref. [3]. Naturally, applying CP-MSBII to the full problem is not more efficient than dividing the full problem into sub-problems organized into a tree and solving them with the RRBPM. The obvious next step is to develop a version of CP-MSBII that works with a tree. This we have done in this paper. Because the dimensionality of the sub-problems is small, each of them has a Hamiltonian with fewer terms than the original Hamiltonian and the eigenvalue spectrum of each of them is less dense. Consequently, the CP-MSBII calculations for the sub-problems require a smaller R and are much less costly.

Another advantage of the tree is that it makes it easier to deal with holes in a PES, which can otherwise cause problems[13].

Using the H-CP-MSBII, we have successfully calculated the vibrational energy levels of acetonitrile (12-D). Levels as accurate as those obtained with CP-MSBII are obtained. The required rank is reduced from 400 to 15. In addition, 50 energy levels of ethylene oxide (15-D) were computed. Further improvements are certainly possible, but we have demonstrated that using CP-MSBII in conjunction with a contraction tree works well.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgements

The financial support of the Natural Sciences and Engineering Research Council is gratefully acknowledged. We thank Compute Canada for providing access to its computers, the University of Paris-Saclay for hospitality, Phillip Thomas for discussions and Robert Wodraszka for comments on the manuscript.

[1] H. Romanowski, and J. M. Bowman, J. Chem. Phys. **82**, 4155 (1985).

[2] M. H. Beck, A. Jaeckle, G. A. Worth, and H. D. Meyer, Phys. Rep. **324**, 1 (2000).

[3] A. Leclerc and T. Carrington, J. Chem. Phys. **140**, 174111 (2014).

[4] P. S. Thomas and T. Carrington, J. Chem. Phys. **119**, 13074 (2015).

[5] P. S. Thomas and T. Carrington, J. Chem. Phys. **146**, 204110 (2017).

[6] P. S. Thomas, T. Carrington, J. Agarwal, and H. F. Schaefer, J. Chem. Phys. **149**, 064108 (2018).

[7] J. Brown and T. Carrington, J. Chem. Phys. **145**, 144104 (2016).

[8] B. Poirier, J. Theor. Comput. Chem. **2**, 65 (2003).

[9] M. B. Hansen, M. Sparta, P. Seidler, D. Toffoli, and O. Christiansen, J. Chem. Theory Comput. **6**, 235 (2010).

[10] N. Gohaud, D. Bégué, C. Pouchan, P. Cassam-Chenai, and J. Liévin, J. Chem. Phys. **127**, 164115 (2007).

[11] S. N. Yurchenko, W. Thiel, and P. Jensen, J. Mol. Spectrosc. **245**, 126 (2007).

[12] S. M. Greene and V. S. Batista, J. Chem. Theory Comput. **13**, 4034 (2017).

[13] A. Baiardi, A. Kelemen, and M. Reiher, J. Chem. Theory Comput. **18**, 415 (2022).

[14] H. R. Larsson, J. Chem. Phys. **151**, 204102 (2019).

[15] S. Kallullathil and T. Carrington, J. Chem. Phys. **155**, 234105 (2021).

[16] R. C. Fortenberry, X. Huang, A. Yachmenev, W. Thiel, and T. J. Lee, Chem. Phys. Lett. **574**, 1 (2013).

[17] A. Jäckle and H.-D. Meyer, J. Chem. Phys. **109**, 3772 (1998).

[18] R. L. Panadés-Barrueta and D. Peláez, J. Chem. Phys. **153**, 234110 (2020).

[19] M. Schröder, J. Chem. Phys. **152**, 024108 (2020).

[20] G. Avila and T. Carrington, Jr., J. Chem. Phys. **143**, 044106 (2015).

[21] S. Manzhos and T. Carrington, Jr., J. Chem. Phys. **125**, 194105 (2006).

[22] W. Koch and D. H. Zhang, J. Chem. Phys. **141**, 021101 (2014).

[23] U. Manthe, H.-D. Meyer, and L. S. Cederbaum, J. Chem. Phys. **97**, 3199 (1992).

[24] J. Jerke and B. Poirier, J. Chem. Phys. **148**, 104101 (2018).

[25] U. Schollwöck, Ann. Phys. **326**, 96 (2011).

[26] A. Baiardi, C. J. Stein, V. Barone, and M. Reiher, J. Chem. Theory Comput. **13**, 3764 (2017).

[27] A. Baiardi, C. J. Stein, V. Barone, and M. Reiher, J. Chem. Phys. **150**, 094113 (2019).

[28] M. Rakhuba and I. Oseledets, J. Chem. Phys. **145**, 124101 (2016).

[29] E. Polizzi, Phys. Rev. B **79**, 115112 (2009).

[30] T. G. Kolda and B. W. Bader, SIAM. Rev. **51**, 455 (2009).

[31] G. Beylkin and M. J. Mohlenkamp, SIAM J. Sci. Comput. **26**, 2133 (2005).

[32] G. Beylkin and M. J. Mohlenkamp, PNAS **99**, 10246 (2002).

[33] H. Wang and M. Thoss, J. Chem. Phys. **119**, 1289 (2003).

[34] U. Manthe, J. Chem. Phys. **128**, 164116 (2008).

[35] O. Vendrell and H.-D. Meyer, J. Chem. Phys. **134**, 044135 (2011).

[36] D. Mendive-Tapia, H.-D. Meyer, and O. Vendrell, J. Chem. Phys. **19**, 1144 (2023).

[37] D. Kressner and C. Tobler, ACM Trans. Math. Softw. **40**, 1 (2014).

[38] S. H. Crandall, Proc. R. Soc. Lond. A **207**, 416 (1951).

[39] M. R. Wall and D. Neuhauser, J. Chem. Phys. **102**, 8011 (1995).

[40] V. A. Mandelshtam and H. S. Taylor, J. Chem. Phys. **106**, 5085 (1997).

[41] V. A. Mandelshtam, J. Chem. Phys. **108**, 24 (1998).

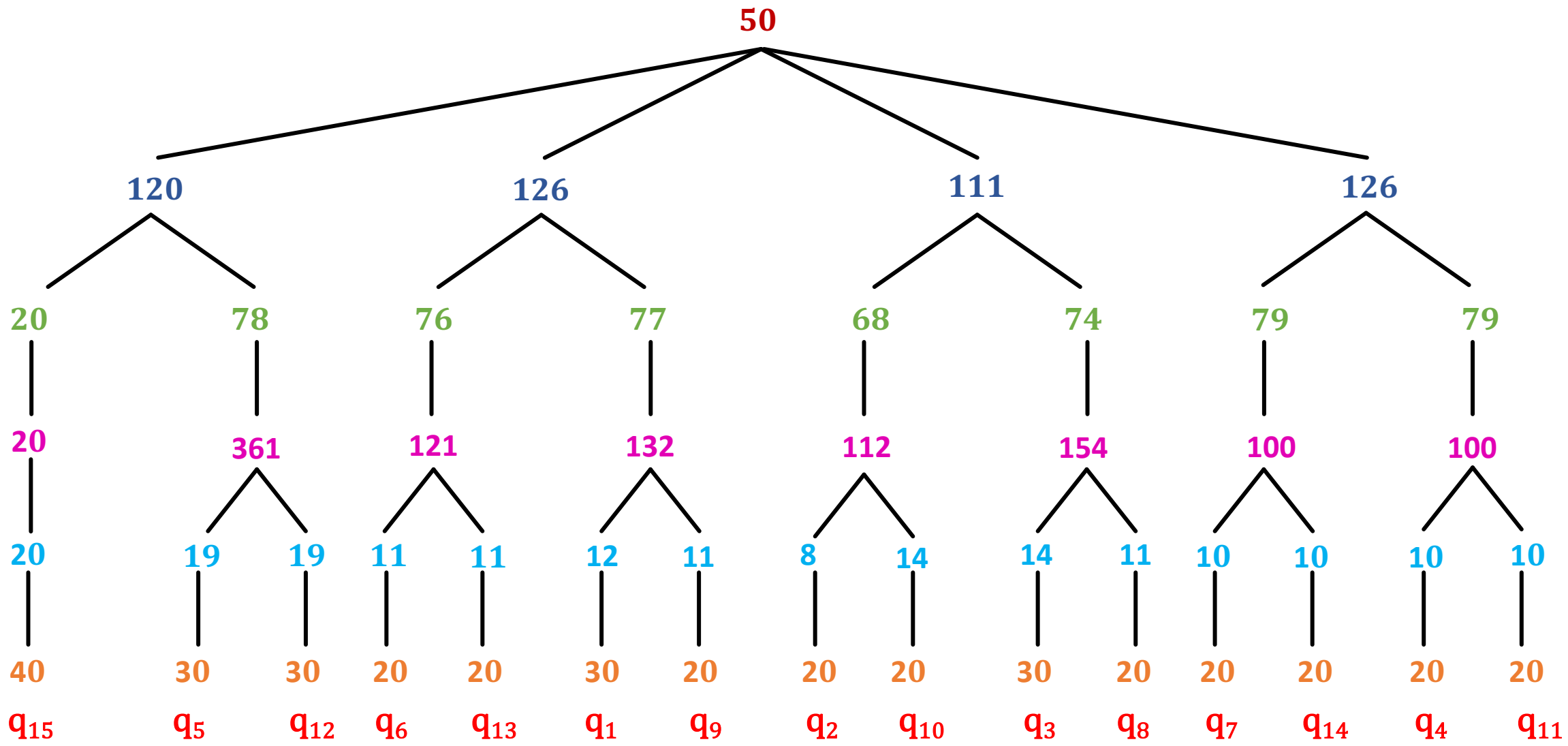
[42] A. Ruhe, Linear Algebra Appl. **58**, 391 (1984).

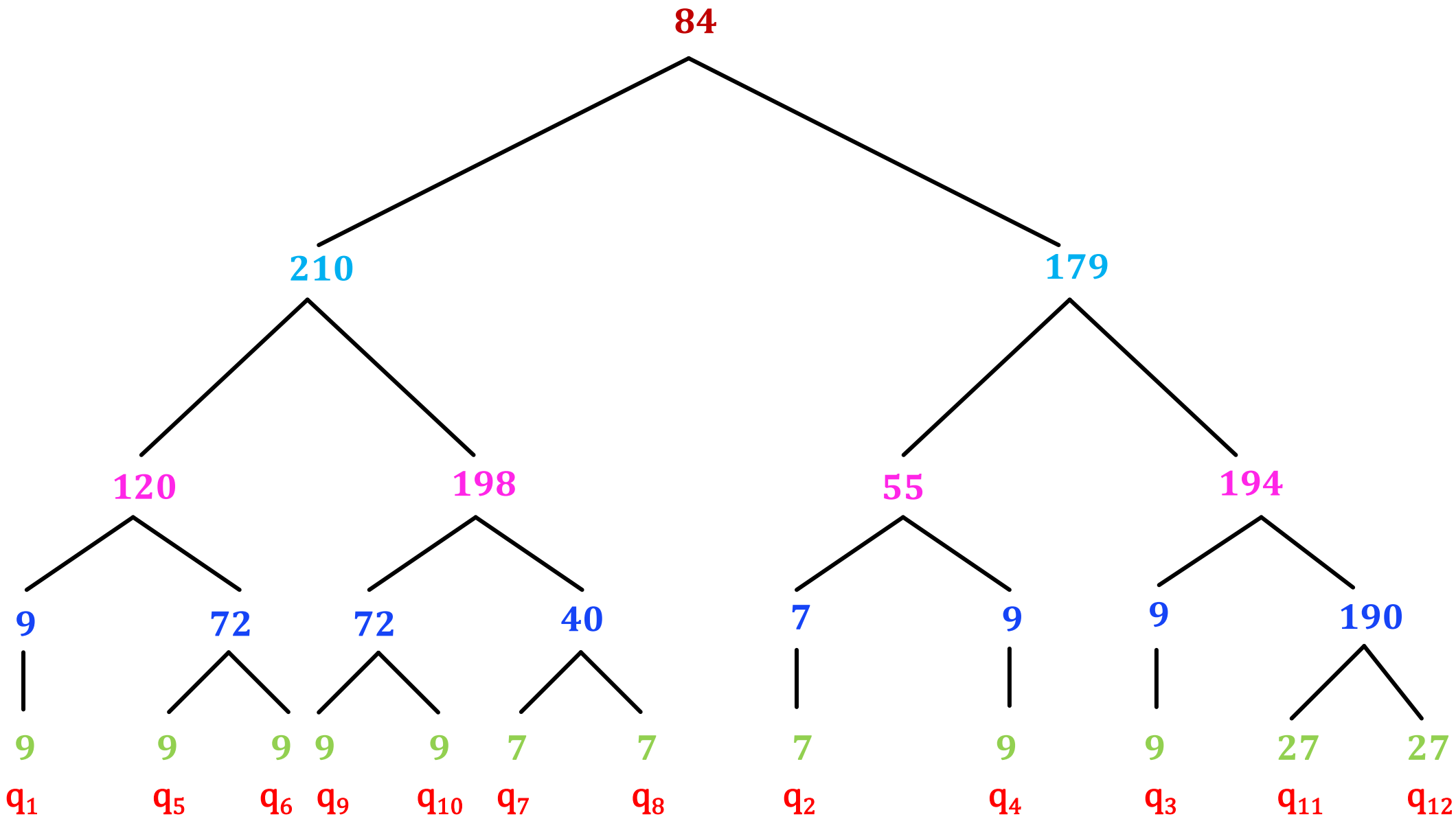
[43] T. Carrington, Jr., J. Chem. Phys. **146**, 120902 (2017).

[44] J. K. G. Watson, Mol. Phys. **15**, 479 (1968).

[45] G. Avila and T. Carrington, J. Chem. Phys. **134**, 054126 (2011).

[46] D. Bégué, P. Carbonnière, and C. Pouchan, J. Phys. Chem. A **109**, 4611 (2005).


50
120
126
111
126
20
78
76
77
68
74
79
79
20
361
121
132
112
154
100
100
20
19
19
11
11
12
11
8
14
14
11
10
10
10
10
40
30
30
20
20
30
20
20
20
30
20
20
20
20
q_{15}
q_5
q_{12}
q_6
q_{13}
q_1
q_9
q_2
q_{10}
q_3
q_8
q_7
q_{14}
q_4
q_{11}


84
210
179
120
198
55
194
9
72
72
40
7
9
9
190
9
9
9
9
7
7
7
9
9
27
27
q1
q5
q6
q9
q10
q7
q8
q2
q4
q3
q11
q12