A-VCI: A flexible method to efficiently compute vibrational spectra

Marc Odunlami, Vincent Le Bris, Didier Bégué, Isabelle Baraille, Olivier Coulaud

A-VCI: A flexible method to efficiently compute vibrational spectra

Marc Odunlami,[1,a)] Vincent Le Bris,[1] Didier Bégué,[1] Isabelle Baraille,[1] and Olivier Coulaud[2]
[1]*Université de Pau et des Pays de l'Adour, CNRS, Institut des Sciences Analytiques et de Physico-Chimie pour l'Environnement et les Matériaux, UMR5254, 64000 Pau, France*
[2]*HiePACS Project-Team, Inria Bordeaux Sud-Ouest, 200, Avenue de la Vieille Tour, 33405 Talence Cedex, France*

(Received 24 February 2017; accepted 15 May 2017; published online 5 June 2017)

The adaptive vibrational configuration interaction algorithm has been introduced as a new method to efficiently reduce the dimension of the set of basis functions used in a vibrational configuration interaction process. It is based on the construction of nested bases for the discretization of the Hamiltonian operator according to a theoretical criterion that ensures the convergence of the method. In the present work, the Hamiltonian is written as a sum of products of operators. The purpose of this paper is to study the properties and outline the performance details of the main steps of the algorithm. New parameters have been incorporated to increase flexibility, and their influence has been thoroughly investigated. The robustness and reliability of the method are demonstrated for the computation of the vibrational spectrum up to 3000 cm^{-1} of a widely studied 6-atom molecule (acetonitrile). Our results are compared to the most accurate up to date computation; we also give a new reference calculation for future work on this system. The algorithm has also been applied to a more challenging 7-atom molecule (ethylene oxide). The computed spectrum up to 3200 cm^{-1} is the most accurate computation that exists today on such systems. *Published by AIP Publishing.* [http://dx.doi.org/10.1063/1.4984266]

I. INTRODUCTION

The computation of vibrational properties of a polyatomic molecule usually requires solving the time-independent Schrödinger equation, under the Born-Oppenheimer approximation. Within this framework, the vibrational energy levels are solutions of an eigenvalue problem, where the Hamiltonian of the system is discretized in a finite dimensional sub-space of the Hilbert space associated with the system. The variational[1–8] principle states that the larger the subspace is, the more accurate are the eigenpairs.

Several methods can be used for this discretization. The Hamiltonian and the wave functions can be expanded using finite basis representation (FBR)[3,9–17] or sampled on some grid of points in the discrete variable representation (DVR).[18] An alternative approach is the phase space representation. It uses Gaussian functions localized in phase space (von Neumann basis set).[19–21]

The simplest representation uses the harmonic oscillator direct product basis set since they are the exact wavefunctions of the harmonic Hamiltonian. Each function is a product of one-dimensional Hermite polynomials. However, the direct product structure of the basis set makes it unusable for molecular systems of dimension $D > 9$, with $D = 3N - 6$ and N is the number of atoms, due to the exponential growth with D ("curse of dimensionality") of the memory storage cost.[18,22,23]

Some recent works try to overcome this limitation by finding the smallest basis size to be able to deal with larger systems. There are two common strategies for selecting a subset of a direct product basis.[24–30] In the first strategy, a pruning condition is used to remove basis functions,[7,13,16,17,20] while in the second strategy we begin with a small basis and we enlarge it by adding functions that satisfy some conditions.[10,17,19,31–37]

The first strategy is based on a correlated truncation scheme, imposing the pruning condition,

$$\sum_{i=1}^{D} \alpha_i n_i \leq b, \tag{1}$$

where the n_i is the maximal degree of the Hermite polynomial related to the coordinate q_i and b is the convergence parameter. Several choices are possible for α_i depending on how we manage the harmonic frequency of each normal mode. The simplest approach considers $\alpha_i = 1$ (called polyad[20,38–40] or binomial[37] truncation) and then retains all polynomials of total degree lower than b. Therefore, we consider the same discretization in all directions and if the harmonic frequencies cover a large range, this could lead us to select useless vibrational states with high energy and to omit relevant states with lower energy. By using an energy criterion we take into account this difference in the pruning space. Two criteria are classically used: $\alpha_i = \omega_i$ (Ref. 20), where ω_i is the harmonic frequency for the coordinate q_i, $\alpha_i = \left\lfloor \frac{\omega_i}{\omega_{min}} + 0.5 \right\rfloor$ (Ref. 17) or $\alpha_i = \left\lfloor \sqrt{\frac{\omega_i}{\omega_{min}}} + 0.5 \right\rfloor$ (Refs. 17 and 20), where ω_{min} is the lowest harmonic frequency. It is also possible to impose a more general condition,[14–17,41]

$$\sum_{i=1}^{D} g_i(n_i) \leq b. \tag{2}$$

While this condition leads to smaller basis sets than the previous one, the choice of suitable $g_i(n_i)$ functions is difficult, and

a)Electronic mail: marc.odunlami@univ-pau.fr

strongly depends on the studied molecular system. Depending on the energy range considered, these choices lead to different convergence properties when the parameter b is increased.[20] However it is difficult to choose b to reach a given accuracy on the eigenvalues.

In the second approach, we start with a small basis and we iteratively add functions to expand the basis until we get accurate enough eigenpairs. This requires defining for each iteration an admissible set of functions (neighbors), a method to choose which functions to add, and a convergence criterion. The functions are selected from the full direct product space.[10,17,37] These neighbors are defined differently depending on the authors. For example, it is possible to evaluate the contribution of each function to the energy at the second order perturbational level[10,42,43] and only select the most significant ones. An alternative idea is to define the most relevant functions of the basis according to a criterion based on the eigenvectors,[17,19] and then to select their neighbors by adding one to each component of multi-indices.

Concerning the convergence criterion, it is generally based on the differences between the eigenvalues for different values of b in the first approach or between consecutive iterations to stop the basis growth in an iterative process. These two choices seem misleading since small differences between eigenvalues do not ensure that the algorithm has converged (see, for example, several "plateaus" in the convergence curves of Ref. 17, Figs. 4 and 5).

The Adaptative Vibrational Configuration Interaction (A-VCI) algorithm[37] has been recently introduced as a more efficient way to select relevant Hermite functions without any limit on their degree, given an energy range of interest. The potential energy operator [Potential Energy Surface (PES)] is used to build the image space of a given set of functions at each iteration. This iterative approach constructs nested subspaces using a result of perturbation theory for linear operators[44,45] to define an *a posteriori* error estimator. This estimator acts as a condition to build the active space of the configuration interaction process by selecting the most relevant basis elements as well as a trustworthy convergence criterion on both the eigenvalues and the eigenvectors. The A-VCI algorithm is then a robust and general procedure to compute spectra in the case of sum of product bases and operators without any *a priori* chemical and physical information and without defining any arbitrary criterion.

The aim of this paper is to detail the different numerical tools of the A-VCI algorithm. We establish new growth strategies resulting in smaller final basis sets to calculate spectra of molecules with 6 and 7 atoms. The article is organized as follows: in Sec. II we recall the key tools used in the A-VCI algorithm first introduced in the work of Garnier *et al.*[37] for the sake of self-containedness. Section III starts by presenting the algorithm and by explaining how the matrices are built and updated. Then, new families of basis augmentation techniques, called component-wise (CW) and collective component-wise (CCW) strategies, are introduced and lead to a significant CPU and memory cost reduction of the algorithm. In Sec. IV, we illustrate the good convergence properties of the A-VCI algorithm on two medium sized molecules: CH_3CN and C_2H_4O. Our results are compared to the best results known on these molecules. Results reported in the supplementary material are also compared to the best computations that can be obtained using the A-VCI algorithm.

II. NOTATIONS AND GENERAL OVERVIEW

Let introduce the vibrational Hamiltonian $\mathcal{H}$ for a N-atom molecular system,

$$\mathcal{H}(\mathbf{q}) = \mathcal{H}_0(\mathbf{q}) + \mathcal{V}(\mathbf{q}) = \sum_{i=1}^{D} \frac{\omega_i}{2}\left(-\frac{\partial^2}{\partial q_i^2} + q_i^2\right) + \sum_{\|\mathbf{s}\|_1=3}^{S} K_{\mathbf{s}} \prod_{i=1}^{D} q_i^{s_i}, \tag{3}$$

where $\mathcal{H}_0$ is the harmonic Hamiltonian operator, $\mathcal{V}$ is the anharmonic part of the Potential Energy Surface (PES) as a Taylor expansion, S is the maximal degree of the PES, $D = 3N - 6$ is the number of vibrational degrees of freedom of the considered system, $\mathbf{q} = (q_1, q_2, \ldots, q_D)$ are the normal dimensionless coordinates, and $\|\mathbf{s}\|_1$ is the sum of all components of the multi-index $\mathbf{s}$. Each harmonic frequency ω_i and polynomial degree s_i are associated with the q_i coordinate. Moreover, we assume that the PES writes as a sum of products of unidimensional operators.

The solution of the vibrational Schrödinger eigenvalue problem requires the discretization of the Hamiltonian operator $\mathcal{H}$ in a suitable orthonormal basis set. $\mathcal{V}$ can be seen as a perturbation of the Hermitian operator $\mathcal{H}_0$. Consequently a natural choice is to search the eigenpairs of $\mathcal{H}$ in the space $\Pi_{\mathbf{d}}$ spanned by the eigenvectors of $\mathcal{H}_0$ analytically defined by

$$\phi^0_{\mathbf{n}}(\mathbf{q}) = \prod_{i=1}^{D} \psi_{n_i}(q_i), \text{ for all } \mathbf{n} = (n_1, \ldots, n_D), \tag{4}$$

where the quantum number n_i corresponds to ω_i and ψ_{n_i} is the 1-D normalized Hermite function of degree n_i. The discretization space $\Pi_{\mathbf{d}}$ is a direct-product space of size M and is defined by

$$\Pi_{\mathbf{d}} = \left\{ \phi^0_{\mathbf{n}} \,/\, \mathbf{n} \in \prod_{i=1}^{D} [0, d_i] \right\}, \tag{5}$$

where $\mathbf{d} = (d_1, \ldots, d_D)$ and d_i is the maximal degree of the Hermite function with respect to the q_i coordinate.

The discretized Hamiltonian H in $\Pi_{\mathbf{d}}$ is represented by a $M \times M$ matrix, in which each element (i, j) has the form,

$$H_{i,j} = \langle \phi^0_{\mathbf{i}} | \mathcal{H} \phi^0_{\mathbf{j}} \rangle, \tag{6}$$

where the row i (respectively, the column j) of the matrix corresponds to the index of basis function $\phi^0_{\mathbf{i}}$ (respectively, $\phi^0_{\mathbf{j}}$) in $\Pi_{\mathbf{d}}$.

Let B be a subset of $\Pi_{\mathbf{d}}$; we define $\mathcal{H}(B)$ its image space by the operator $\mathcal{H}$, which is spanned by all harmonic functions $\phi^0_{\mathbf{n}}$ such that $\langle \phi^0_{\mathbf{n}} | \mathcal{H} \phi \rangle \neq 0$ for all ϕ in B. As $\mathcal{H}_0$ is diagonal in the basis of the 1-D Hermite function products, the image space can be decomposed as the direct sum of two orthogonal spaces,

$$\mathcal{H}(B) = B \oplus B_R. \tag{7}$$

Let m (respectively, m_R) be the number of elements in B (respectively, B_R), we denote by $\widetilde{H}$ the Hamiltonian matrix of operator $\mathcal{H}$ in space $\mathcal{H}(B)$. Thanks to the

above space decomposition, this matrix is decomposed as follows:

$$\widetilde{H} = \left(\begin{array}{c|c} H & H_R^T \\ \hline H_R & H_C \end{array}\right), \tag{8}$$

where $H = B^T \mathcal{H} B$ is a $(m \times m)$ Rayleigh matrix that approximates the Hamiltonian operator $\mathcal{H}$ in the orthonormal basis B, $H_R = B_R^T \mathcal{H} B$ is a $(m_R \times m)$ matrix, and $H_C = B_R^T \mathcal{H} B_R$ is a $(m_R \times m_R)$ matrix.

As B is strictly included into $\mathcal{H}(B)$ when we search eigenvalues of $\mathcal{H}$, we have to evaluate the error due to the projection in B. In other words, we want to measure the distance between the eigenvalues of H and the eigenvalues of $\widetilde{H}$. To this end, we introduce an *a posteriori* error estimator based on the Bauer-Fike theorem.[44,45] This theorem is a classical tool in spectral perturbation theory to localize eigenvalues and indicates that if ΔH is a symmetric perturbation of a symmetric matrix H then for any eigenpairs $(\widetilde{E}, \widetilde{\mathbf{X}})$ of $H + \Delta H$ with $\|\widetilde{\mathbf{X}}\| = 1$, there is an eigenvalue E of H such that

$$|\widetilde{E} - E| \leq \|\Delta H \widetilde{\mathbf{X}}\| \leq \|\Delta H\|_F, \tag{9}$$

where $\|.\|$ is the usual Euclidian norm and $\|.\|_F$ is the Frobenius norm.

Thanks to the matrix decomposition (8), the matrix $\widetilde{H}$ can be written as

$$\widetilde{H} = \overline{H} + \Delta H = \left(\begin{array}{c|c} H & 0 \\ \hline 0 & 0 \end{array}\right) + \left(\begin{array}{c|c} 0 & H_R^T \\ \hline H_R & H_C \end{array}\right), \tag{10}$$

and it can be viewed as a perturbation of the matrix $\overline{H}$ which is an extension of the matrix H in the image space of B. Let $(E, \mathbf{X})$ be an eigenpair of H and $\overline{\mathbf{X}} = (\mathbf{X}, \mathbf{0}_{m_R})^T$ its extended eigenvector in $\mathcal{H}(B)$. Then, the residual vector $\mathbf{R} = \widetilde{H}\overline{\mathbf{X}} - E\overline{\mathbf{X}}$ satisfies

$$\mathbf{R} = \widetilde{H}\overline{\mathbf{X}} - E\overline{\mathbf{X}} = \overline{H}\,\overline{\mathbf{X}} - E\overline{\mathbf{X}} + \Delta H\overline{\mathbf{X}} = \Delta H\overline{\mathbf{X}} = (\mathbf{0}_m, H_R\mathbf{X})^T. \tag{11}$$

By using (9) and (11), we obtain the *a posteriori* error estimator,

$$|\widetilde{E} - E| \leq \|\mathbf{R}\| = \|H_R\mathbf{X}\|. \tag{12}$$

The distance between E and the target eigenvalue $\widetilde{E}$ of $\widetilde{H}$ is bounded by $\|R\|$. More precisely, the above inequality shows that if E is an eigenvalue of the matrix H and $\|\mathbf{R}\| \leq \varepsilon$, then E is also a good approximation of an eigenvalue $\widetilde{E}$ of the matrix $\widetilde{H}$ in the space $\mathcal{H}(B)$, which is a larger space.

Section III presents in a detailed fashion how this criterion can be used to control the behavior of the A-VCI algorithm, and provides more information on how it is implemented.

III. THE A-VCI ALGORITHM

The A-VCI algorithm introduced in the work of Garnier *et al.*[37] is an iterative procedure that computes both a minimal basis set in the discretized $\Pi_{\mathbf{d}}$ space and the eigenpairs of the vibrational Hamiltonian discretized in this basis.

The main components of such an iterative procedure are a discretization space and a starting space, a method to enlarge the current basis of the active space and finally, a robust criterion to assess the convergence.

In this section, we briefly recall the main steps of the A-VCI algorithm, and then we describe these different elements and how we efficiently use them in the algorithm.

A. Algorithm

Algorithm 1 describes the full A-VCI process to compute both the first F eigenpairs of the Hamiltonian in the discretized space $\Pi_{\mathbf{d}}$ and the final basis set adjusted to the required accuracy.

After selecting an approximation space $\Pi_{\mathbf{d}}$, an initial basis $B^{(0)}$ belonging to this space must be explicitly defined (line 1). We then construct the sparse structure of the matrices $H^{(0)}$ and $H_R^{(0)}$, that is to say only the indices of the rows and the columns of non-zero elements (line 2). During this step, we also build the basis of admissible nodes $B_R^{(0)}$, which is needed for the sparse structure of the matrix $H_R^{(0)}$. The iterative procedure begins by calculating the coefficients of the Hamiltonian matrix (line 3), and then the first F eigenpairs (line 4) are

Algorithm 1. Adaptive vibrational configuration interaction (A-VCI) algorithm.

Result: The first F eigenpairs of the discretized Hamiltonian and the corresponding minimal basis

begin

1 Define the initial orthonormal basis $B^{(0)}$

2 Build the sparse structure of the matrices $H^{(0)} = {B^{(0)}}^T \mathcal{H} B^{(0)}$ and $H_R^{(0)} = {B_R^{(0)}}^T \mathcal{H} B^{(0)}$

// Start the iterations

for $j \geq 0$ **do**

3 Build the coefficients of the sparse Rayleigh matrix $H^{(j)} = {B^{(j)}}^T \mathcal{H} B^{(j)}$

4 Compute the first F eigenpairs of $H^{(j)}$ denoted $(E_\ell^{(j)}, \mathbf{X}_\ell^{(j)})_{\ell=0}^{F-1}$

5 Build the coefficients of the matrix $H_R^{(j)} = {B_R^{(j)}}^T \mathcal{H} B^{(j)}$

6 Check the convergence

Exit if the algorithm has converged

7 Expand $B^{(j)}$ with a selected orthonormal subset $A^{(j)}$ of $B_R^{(j)}$: $B^{(j+1)} = B^{(j)} \oplus A^{(j)}$

8 Build the sparse structure of the matrices $H^{(j+1)}$ and $H_R^{(j+1)}$

computed by an iterative eigensolver. To check the convergence of the algorithm (line 6), we evaluate for all eigenpairs $(E_\ell^{(j)}, \mathbf{X}_\ell^{(j)})$ the scaled residual norms $\|\mathbf{r}_\ell^{(j)}\| = \|H_R^{(j)}\mathbf{X}_\ell^{(j)}\|/E_\ell^{(j)}$. If the maximal value of these norms is lower than the target threshold ε, the method has converged. This evaluation requires to compute all coefficients of the rectangle matrix $H_R^{(j)}$ (line 5). If the convergence is not reached, we build the new active space by adding directions selected in $B_R^{(j)}$ to the $B^{(j)}$ basis (line 7). Finally, we update the sparse structure of the two matrices, thanks to the new basis elements we add (line 8). According to (12) at the convergence of the algorithm, the eigenpairs computed are also the eigenpairs of the Hamiltonian discretized in $B^{(j)} \cup B_R^{(j)}$ and not only in $B^{(j)}$.

The main parts of the algorithm are presented in Subsections III B–III D. First, we describe how to define the approximation space we work in. Then we explain how we build and update the sparse structure of $H^{(j)}$ and $H_R^{(j)}$. Finally, the last part of this section focuses on how the new directions we add to $B^{(j)}$ are selected.

B. Approximation space definition

In order to compute the eigenvalues of the Hamiltonian operator (3), we first define the approximation space in which the operator will be discretized.

We consider the product space $\Pi_{\mathbf{d}}$ defined in (5). The maximal degree of Hermite d_i in each direction i can be determined in different ways. The first idea is to set a constant value N_{max} in each dimension. In this case each direction is discretized by the same number of functions without taking into account harmonic frequencies. Thus, for small harmonic frequencies the highest state has a lower energy than for large ones. To overcome this problem, we propose a second approach based on an energetic criterion to reach the same 1-D harmonic energy in each direction. Let E_{max} be a given maximal energy above the ground state for the 1-D Hermite functions. Then the maximal degree in the direction i is

$$d_i = 1 + \left\lfloor \frac{E_{max}}{\omega_i} \right\rfloor. \tag{13}$$

This last constraint decreases drastically the number of elements in the approximation space. We can reduce its size even more by introducing a pruning condition. Many pruning conditions exist (see Refs. 13, 16, 17, and 20). Let P_b be the pruning space in $\Pi_{\mathbf{d}}$ defined by

$$P_b = \{\phi_{\mathbf{n}}^0 \in \Pi_{\mathbf{d}} \text{ such that } g(\mathbf{n}) \le b\}. \tag{14}$$

Usual pruning functions, write $g(\mathbf{n}) = \sum_{i=1}^{D} \alpha_i n_i$ but they can be more complex such as vectorial or non-linear conditions. Typically $\alpha_i = 1$ corresponds to the binomial pruning, denoted BP(b), and $\alpha_i = \omega_i$ is the energetic pruning.

C. Matrix operations

The algorithm is based on the following operations on matrices: the construction and update of their structure, and the calculation of their coefficients. In this subsection we describe these three steps.

1. Sparse structure definition

Before describing the different steps of the matrix building algorithm, we recall some definitions on sparse matrices. Due to the choice of the basis elements, the discretized Hamiltonian operator is a sparse matrix. Therefore we only store non-zero elements of H and H_R to optimize the storage. We define by sparse structure the indices (row, column) of the non-zero elements.

The structure of the sparse symmetric matrix H is represented by its graph $G(V, \mathcal{E})$, characterized by a set of nodes V and their connecting edges $\mathcal{E}$. A node of V stands for a row or a column of H (i.e., an element of the basis B). There is an edge between two nodes $(i, j) \in V \times V$ if and only if the coefficient $H_{i,j}$ is not zero.

We denote by $G_{H_R}(W, V, \mathcal{E}_R)$ the bipartite graph of the rectangular matrix H_R. The node sets W and V represent the rows and columns of H_R, respectively, and are connected by the set of edges $\mathcal{E}_R$. The condition $(H_R)_{i,j} \neq 0$ defines an edge between $i \in W$ and $j \in V$, and $G_{H_R}(W, V, \mathcal{E}_R)$ gives the structure of H_R.

Thanks to their graphs, we store the matrices H and H_R in the Compressed Sparse Row (CSR) format.

2. Sparse structure construction

Starting from the basis $B^{(0)}$, the second step of our algorithm (line 2) is to construct the sparse structures of the matrices H and H_R. As the matrix H is symmetric, we only consider its lower triangular part. Moreover, this step also constructs the set of the admissible nodes $B_R^{(0)}$ associated with the initial basis $B^{(0)}$. This means that for each element $\mathbf{n} = (n_1, \ldots, n_D)$ of $B^{(0)}$ we should determine all nodes $\mathbf{m}$ in $\Pi_{\mathbf{d}}$ connected to it through the anharmonic part of the PES operator. As the PES is a sum of products of monomials, we proceed as follows.

First, let $C(\mathbf{n}, \mathbf{s}) = \{\mathbf{m} \text{ such that } \langle \Phi_{\mathbf{n}}(\mathbf{q}) | \mathbf{q}^{\mathbf{s}} \Phi_{\mathbf{m}}(\mathbf{q}) \rangle \neq 0\}$ be introduced as the set of all nodes connected to $\mathbf{n}$ through the monomial $\mathbf{q}^{\mathbf{s}}$, with $\mathbf{s} = (s_1, \ldots, s_D)$. Since the basis functions are products of 1-D functions, we only have to check whether the 1-D integral $\langle \phi_{n_i}(q) | q^{s_i} \phi_{m_i}(q) \rangle$ vanishes or not for all $i = 1, \ldots, D$. For 1-D Hermite functions, these integrals have analytic expressions (see Proposition 2 in the Appendix), and due to the approximation space, we only keep indices lower than d_i. To construct all multi-indices connected to $\mathbf{n} = (n_1, \ldots, n_D)$ with a monomial of the PES, we calculate for each dimension all the integers connected to n_i by using Proposition 1 of the Appendix. The selected multi-indices are reconstructed and sorted by their key. The key of a node $\mathbf{n} = (n_1, \ldots, n_D)$ is defined by

$$key(\mathbf{n}) = n_1 + n_2(d_1 + 1) + \cdots + n_D(d_1 + 1) \ldots (d_{D-1} + 1),$$

where d_i is the maximal Hermite degree in each direction. If needed, a pruning condition can be applied before building the key to decrease the number of connected nodes. It is easy to see that the number of elements in $C(\mathbf{n}, \mathbf{s})$ is bounded by $D(\|\mathbf{s}\|_\infty + 1)$, where $\|\mathbf{s}\|_\infty = \max_{i=1,\ldots,D} s_i$ and the cost to obtain them is $O(D\|\mathbf{s}\|_\infty \ln(D\|\mathbf{s}\|_\infty))$.

Finally, the set of all nodes connected to $\mathbf{n}$ is obtained by iterating on all monomial elements $\mathbf{q}^{\mathbf{s}}$ of the PES,

$$C(\mathbf{n}) = \bigcup_{\mathbf{s}\in Deg(\mathrm{PES})} C(\mathbf{n},\mathbf{s}), \tag{15}$$

where $Deg(\mathrm{PES})$ is the set of the monomial degrees $\mathbf{s}$ in the PES such that $\|\mathbf{s}\|_1 \geq 3$. The total number of nodes connected to $\mathbf{n}$ is bounded by $D\,(S+1)\,N_{\mathrm{PES}}$, where S is the maximal degree of the PES and N_{PES} is the number of anharmonic terms in the PES. This number is also the bound of all non-zero elements on the row associated with the node $\mathbf{n}$ of the Hamiltonian matrix in the full space $\Pi_{\mathbf{d}}$. The complexity of this step is bounded by $O(DS\ln(DS)\,N_{\mathrm{PES}})$.

Thanks to the space decomposition (7), the set of all nodes connected to $\mathbf{n}$ at the beginning of the iterative process splits into two sets as follows:

$$C(\mathbf{n}) = \mathcal{E}(\mathbf{n}) \oplus \mathcal{E}_{\mathcal{R}}(\mathbf{n}),$$

where $\mathcal{E}(\mathbf{n})$ contains the nodes inside $B^{(0)}$ and $\mathcal{E}_{\mathcal{R}}(\mathbf{n})$ the nodes outside $B^{(0)}$ sorted by their key. Since the elements of $\mathcal{E}_{\mathcal{R}}(\mathbf{n})$ belongs to the basis set of admissible nodes $B_R^{(0)}$, we have

$$B_R^{(0)} = \bigcup_{\mathbf{n}\in B^{(0)}} \mathcal{E}_{\mathcal{R}}(\mathbf{n}). \tag{16}$$

As the nodes of $\mathcal{E}_{\mathcal{R}}(\mathbf{n})$ are already sorted by their key, the construction of $B_R^{(0)}$ is just a merge of several arrays. The cost is linear with respect to the maximal length of these arrays.

In this approach, the sparse structure built on the partition space $B^{(0)}\times B_R^{(0)}$ corresponds to the sparse structure of the transposed of H_R. Therefore, the last step consists in transposing this structure to get the final pattern of $H_R^{(0)} = {B^{(0)}}^T \mathcal{H} B_R^{(0)}$.

3. Sparse structure update

Starting at iteration j from a set of nodes $A^{(j)}$ selected in $B_R^{(j)}$, the purpose is to build the new sparse structure $G(B^{(j+1)}, \mathcal{E}^{(j+1)})$ [respectively, $G_{H_R}(B_R^{(j+1)}, B^{(j+1)}, \mathcal{E}_{\mathcal{R}}^{(j+1)})$] of the matrix $H^{(j+1)}$ (respectively, $H_R^{(j+1)}$) without reconstructing them from scratch. To do that, we only compute the structure for the new nodes and merge it with the one built at the previous iteration. We proceed in three steps:

Step 1 Add nodes of $A^{(j)}$ in $B^{(j)}$ to obtain the new basis set $B^{(j+1)} = B^{(j)} \cup A^{(j)}$. Fill $\mathcal{E}^{(j+1)}$ with the old sparse structure $\mathcal{E}^{(j)}$ and fill ${\mathcal{E}_{\mathcal{R}}}^{(j+1)}$ with the old structure ${\mathcal{E}_{\mathcal{R}}}^{(j)}$ without the nodes in $A^{(j)}$.

Step 2 For all elements $\mathbf{a}$ in $A^{(j)}$, we compute the set of nodes connected with $\mathbf{a}$ defined by $C(\mathbf{a}) = \mathcal{E}(\mathbf{a}) \oplus \mathcal{E}_{\mathcal{R}}(\mathbf{a})$, where $\mathcal{E}(\mathbf{a})$ are the nodes in $B^{(j+1)}$ and $\mathcal{E}_{\mathcal{R}}(\mathbf{a})$ are the nodes outside $B^{(j+1)}$. We complete the structure $\mathcal{E}^{(j+1)}$ with $\bigcup_{\mathbf{a}\in A^{(j)}} \mathcal{E}(\mathbf{a})$ and the new set of admissible nodes is $B_R^{(j+1)} = (B_R^{(j)} \setminus A^{(j)}) \cup \mathcal{E}_{\mathcal{R}}(A^{(j)})$, with $\mathcal{E}_{\mathcal{R}}(A^{(j)}) = \bigcup_{\mathbf{a}\in A^{(j)}} \mathcal{E}_{\mathcal{R}}(\mathbf{a})$.

Step 3 Finally, we transpose $\mathcal{E}_{\mathcal{R}}(A^{(j)})$ and add it to ${\mathcal{E}_{\mathcal{R}}}^{(j+1)}$ to obtain the final sparse structure of $H_R^{(j+1)}$.

The most expensive part of this update is step 2 which can easily be parallelized.

4. Matrix coefficient evaluation

Let consider two basis functions $\Phi_{\mathbf{n}}$ and $\Phi_{\mathbf{m}}$, the matrix element $\langle\Phi_{\mathbf{n}}(q)|\mathcal{H}\Phi_{\mathbf{n}}(q)\rangle$ is

$$\langle\Phi_{\mathbf{n}}(q)|\mathcal{H}\Phi_{\mathbf{m}}(q)\rangle = \sum_{i=1}^{D} \omega_i \left(\frac{1}{2} + n_i\right) \delta_{\mathbf{n},\mathbf{m}} + \sum_{\|\mathbf{s}\|_1=3}^{S} K_{\mathbf{s}} \prod_{i=1}^{D} \langle\phi_{n_i}(q_i)|q_i^{s_i}\phi_{m_i}(q_i)\rangle\,.$$

Due to the properties of Hermite functions and the PES in sum of products form, each 1-D integral $\mathcal{M}(s,n,m) = \langle\phi_n(x)|x^s\phi_m(x)\rangle$ has an analytic expression (see the Appendix). They are pre-computed once and stored in the tensor moment $\mathcal{M}$. Then, the matrix coefficients are given by

$$\langle\Phi_{\mathbf{n}}(q)|\mathcal{H}\Phi_{\mathbf{m}}(q)\rangle = \sum_{i=1}^{D} \omega_i \left(\frac{1}{2} + n_i\right) \delta_{\mathbf{n},\mathbf{m}} + \sum_{\|\mathbf{s}\|_1=3}^{S} K_{\mathbf{s}} \prod_{i=1}^{D} \mathcal{M}(s_i,n_i,m_i).$$

It is clear that the complexity to build one matrix element is $O(D\,N_{\mathrm{PES}})$ and then the total complexity to build the Hamiltonian matrix is $O(D\,N_{\mathrm{PES}}\,Z_H)$ and $O(D\,N_{\mathrm{PES}}\,Z_{H_R})$ for H_R. Z_H and Z_{H_R} are the number of non-zero elements of the matrices H and H_R, respectively. However, this construction is highly parallel and its cost is negligible compared to the construction of the sparse structure of the matrix.

D. Basis expansion strategies

The cost of A-VCI is mainly related to the total matrix operation complexity, which depends on the size of the basis involved in this adaptive algorithm. The number of elements in $B^{(j)}$ and $B_R^{(j)}$ at iteration j and the number of nodes connected to them increase with the dimension D, the number of terms of the PES and the required precision. Therefore, the node selection strategy to expand the active space is crucial in terms of accuracy and performance. The fewer the functions we add at each iteration, the smaller will be the final basis and the related memory footprint. However the number of iterations can increase, and this can penalize the computational time to reach the convergence. There is a trade-off between the memory requirement and the convergence speed of the algorithm. Moreover, developing an automatic process to select such nodes is important for the efficiency of the algorithm. In this section we discuss different approaches to enlarge the A-VCI basis (line 7 in Algorithm 1) with respect to the previous constraints.

We remind that the convergence of A-VCI is checked by comparing the norms of the scaled residual vectors with a given threshold denoted ε. Let $K^{(j)} = \{\ell \in \{0,\ldots,F-1\}$ such that $\|\mathbf{r}_\ell^{(j)}\| > \varepsilon\}$ be the list of the $k^{(j)}$ non-converged scaled residues at iteration j. In all cases the expansion strategies will be based on the elements of $K^{(j)}$. Consider a vector $\mathbf{v}$ of size m_R; we denote by $M_\eta(\mathbf{v}) = \{i$ such that $|v_i| > \eta\}$ the space of admissible components of size m_η, and we define the

generalized average of $\mathbf{v}$ with respect to this space by

$$mean(\mathbf{v}, p, \eta) := \sqrt[p]{\frac{1}{m_\eta} \sum_{i=1}^{m_R} |v_i|^p \, \mathbf{1}_{M_\eta(\mathbf{v})}(i)}, \tag{17}$$

where $\mathbf{1}_{M_\eta(\mathbf{v})}$ is the indicator function of space $M_\eta(\mathbf{v})$ [i.e., 1 when i is in $M_\eta(\mathbf{v})$ and 0 otherwise]. We also introduce $N_J(\mathbf{v})$ the space of the indices of the J largest components of $\mathbf{v}$.

1. Component-wise strategies

Garnier *et al.*[37] introduced a component-wise procedure on each non-converged relative residue to expand the $B^{(j)}$ basis based on the usual mean [i.e., $p = 1$ in (17)]. We denote this strategy CW($p = 1$). In this approach, the set of nodes added at iteration j is

$$A^{(j)} = \bigcup_{\ell \in K^{(j)}} \left\{ i \text{ such that } |(\mathbf{r}_\ell^{(j)})_i| > mean(\mathbf{r}_\ell^{(j)}, 1, \varepsilon/\sqrt{m_R}) \right\}. \tag{18}$$

Such procedure introduces a lot of components at each step. Then, both $B^{(j)}$ and $B_R^{(j)}$ grow quickly during the iterations. Another way, denoted CW2, consists in selecting at iteration j the functions of $B_R^{(j)}$ corresponding to the $2(j+1)$ largest components of each non-converged residual vector. Using CW2, the set of nodes enlarging $B^{(j)}$ is

$$A^{(j)} = \bigcup_{\ell \in K^{(j)}} N_{2(j+1)}(\mathbf{r}_\ell^{(j)}), \tag{19}$$

and the maximal number of components added at iteration j is $2(j+1)k^{(j)}$. Therefore the expansion of the $B^{(j)}$ basis is moderate compared to the CW(1) method. Another advantage of this strategy is that the convergence is adaptive since the number of added nodes increases with respect to the iteration number. At the beginning of the iterative process the error is large in any case, so it is not necessary to expand $B^{(j)}$ in every direction. It seems more sensible to add as many elements as possible when we get close to the convergence in order to speed it up and refine it. At the end, the basis contains the most relevant nodes to compute the eigenvalues at the given accuracy. On the other hand, to select the largest components, we sort each residual vector leading to the global cost $O(k^{(j)} m_R \ln(m_R))$ compared to the linear cost $O(k^{(j)} m_R)$ for the former CW(1) strategy.

2. Collective component-wise strategies

The main drawback of component-wise strategies is that basis expansion procedures take into account each residual vector independently. For CW(1) the value of $mean(\mathbf{r}_\ell^{(j)}, 1, \varepsilon/\sqrt{m_R})$ depends on the vector $\mathbf{r}_\ell^{(j)}$, and in CW2 the $2(j+1)$ largest components do not necessarily match with the same level of approximation for two different residues. In fact, it is difficult to say if a relevant component for a given residual vector is useful to decrease the residual components of other eigenvalues. Moreover the component-wise selection can be redundant since the same nodes are possibly chosen from different residual vectors. To overcome these issues we introduce at iteration j the mean residual vector $\mathbf{R_A}^{(j)}$ such that each component is

$$R_{A_i}^{(j)} = \frac{1}{k^{(j)}} \sum_{\ell \in K^{(j)}} |(\mathbf{r}_l^{(j)})_i|, \; i = 1, \ldots, m_R. \tag{20}$$

Every non-converged individual residual vector belonging to $K^{(j)}$ contributes to $\mathbf{R_A}^{(j)}$. The same strategies used for the component-wise selection can be applied to the $\mathbf{R_A}^{(j)}$ vector:

CCW(p): we choose the functions using a general mean criterion on $\mathbf{R_A}^{(j)}$ components. The nodes to add to the A-VCI space are defined by

$$A^{(j)} = \{ i \text{ such that } \mathbf{R_A}_i{}^{(j)} > mean(\mathbf{R_A}^{(j)}, p, \varepsilon/\sqrt{m_R}) \}. \tag{21}$$

CCW2: we select the nodes with the $2(j+1)$ maximal components of $\mathbf{R_A}^{(j)}$ such that

$$A^{(j)} = N_{2(j+1)}(\mathbf{R_A}^{(j)}). \tag{22}$$

The number of directions added by CW(p) or CCW(p) increases rapidly with respect to the required accuracy. This leads to large CPU times in the matrix operations. In such cases, it is crucial to make the computational cost as affordable as possible. In order to reach a target number of nodes to add, one possibility is to adapt the mean by increasing or by decreasing p in (17) instead of sorting the scaled vector to find the largest components (like in CW2 or CCW2). The goal here is to bind the time spent in the update step. Such approach will increase the number of iterations but the size of the final space is smaller than those obtained from other strategies. We define this new method as follows.

aCCW(N_T, δ): **a**utomatic **C**ollective **C**omponent-**W**ise selection, where N_T is the target number of basis vectors (nodes) we want to add and δ is the tolerance criterion we accept on this number. In aCCW we adjust p in CCW(p) such that $|size(A^{(j)})/N_T - 1| \leq \delta$, where $size(A^{(j)})$ is the number of elements of $A^{(j)}$. When the value of p is changed we reconstruct $A^{(j)}$ using (21). The complexity of this procedure is only linear with respect to m_R compared to $m_R \ln(m_R)$ with a sort operation.

IV. RESULTS AND DISCUSSION

The presented method has been tested on the 6-atom acetonitrile molecule, CH_3CN, and on a more challenging system of 7 atoms: the ethylene oxide molecule, C_2H_4O. Two implementations of A-VCI have been used: a Python/C prototype and a full C++ version. In the Python/C software all the basis enlargement methods are implemented whereas the C++ driver only provides the collective strategies. These codes use the iterative ARPACK solver[46] to compute the lowest F eigenvalues of the discretized Hamiltonian matrix at each iteration of A-VCI. The initial guess of the eigensolver is a random vector ($IDO = \mathbf{0}$). Moreover, we set the number of Arnoldi vectors to $NCV = 2F + 1$ and we use the ARPACK default convergence criteria based on the machine precision. Finally, both implementations of A-VCI take advantage of the OpenMP fork-join paradigm for parallelization.

The results are obtained on two platforms:

(i) A 24-core Haswell Intel Xeon E5-2680 processors running at 2.8 GHz with 128 GB of shared memory. The Intel compiler (2016 update 3) with the following options, `-mkl=parallel -march=native`

`-axCORE-AVX2,CORE-AVX-I,AVX -qopenmp`, is used. We refer to it as *plafrim* in the sequel.

(ii) A 12-core Nehalem Intel Xeon L5640 nodes running at 2.27 GHz with 24 GB of shared memory. Python 3.5.1 and the open-source GNU 5.1.0 compiler with the options `-O3 -mtune = native -fopenmp` were used. We refer to it as *pyrene* in the sequel.

A. Acetonitrile molecule, CH_3CN

The convergence of the A-VCI algorithm depends on how we choose the approximation space, the threshold, and the basis expansion method. What would be the most efficient way, for a required precision, to set all these parameters? How can we obtain the smallest basis or the lowest CPU time and memory consumption? In this part, we study this problem in the case of the well-known acetonitrile molecule. The CH_3CN force constants come from Ref. 14 and are based on the quartic PES originally developed by Begue *et al.*[47] The PES is composed of 299 anharmonic terms (108 cubic and 199 quartic terms) computed at the B3LYP/cc-pVTZ level. The harmonic frequencies calculated at the CCSD(T)/cc-pVTZ level are sorted by ascending order as follows:

$$\begin{aligned}\omega_1 = \omega_2 &= 361,\ \omega_3 = 920,\\ \omega_4 = \omega_5 &= 1\,061,\ \omega_6 = 1\,413,\\ \omega_7 = \omega_8 &= 1\,487,\ \omega_9 = 2\,297,\\ \omega_{10} &= 3\,065,\ \omega_{11} = \omega_{12} = 3\,149\ (\mathrm{cm}^{-1}).\end{aligned}$$

First, we compare several approaches to expand the $B^{(j)}$ basis at each iteration j, and then we show how the threshold and the size of the approximation space $\Pi_{\mathbf{d}}$ influence the accuracy of the eigenvalues. Finally, we compute the frequencies up to 3000 cm^{-1} and compare them with a reference calculation.[14]

1. Influence of different strategies to expand the A-VCI space

Here we compare the different strategies to expand the A-VCI space presented in Subsection III D when we compute the first 121 eigenvalues as in Ref. 37. We remind that CW(1) and CW2 are component-wise, whereas CCW(p) and CCW2 are based on the collective residue vector $\mathbf{R_A}^{(j)}$. In this section, the calculations are done with the Python/C driver on *pyrene* and we consider the same degree of approximation in each direction fixed to 30 for the approximation space $\Pi_{\mathbf{d}}$; the threshold is $\varepsilon = 7.5 \times 10^{-3}$. The initial basis $B^{(0)}$ is given by the first 121 functions of $\Pi_{\mathbf{d}}$ (with $\mathbf{d}$ = [7, 6, 2, 2, 2, 1, 1, 1, 1, 0, 0, 0]) sorted by ascending energies in the harmonic approximation.

Table I reports the size and the number of non-zero elements in the matrices H and H_R obtained for different strategies, with the corresponding CPU time. The maximum absolute error, denoted *Err*, is computed with respect to the calculation of Avila and Carrington[14] with a basis of 743 103 elements. The first two rows of Table I provide the results of the component-wise methods while the others are related to collective strategies. The first line corresponds to the A-VCI approach used in Ref. 37, leading not only to the largest basis and CPU time but also to the best accuracy. The basis grows less when expanding it with the nodes which have the largest residual components, with both collective (line 7) and individual (line 2) strategies. In these cases, the error can be large (more than 1 cm^{-1}) and the CPU time is higher than with the CCW methods. We observe that the most efficient strategies are CCW(2) and CCW(3), with equivalent performances. From $p = 4$ the cost of the additional iterations overcomes the gain due to the basis reduction, so rising p from this point becomes no longer interesting. The matrix storage cost for CCW strategies is low, and building the full sparse matrix to perform efficient matrix-vector product is a good option. For instance, the number of non-zero elements divided by the total number of elements (sparsity of the matrix) is 0.45% for $H^{(j)}$ matrix and 3 per million for $H_R^{(j)}$ matrix for the CCW(2) strategy. Table I shows that a trade-off between the basis size and the number of elements added at each iteration must be considered. Thanks to the exponent p of the generalized mean (17), we can balance this trade-off. The higher its value, the fewer are the basis functions added, resulting in more iterations and a smaller final basis.

These results on the benchmark molecule CH_3CN highlight some interesting properties of the discussed basis enlargement methods. First of all, selecting $2(j+1)k^{(j)}$ components to augment the basis at each iteration j is not the best strategy. The number of added functions does not grow quickly enough to get a fast convergence rate of the algorithm. Even though the final basis is small, too many iterations and much CPU time are needed. This is true for both the component-wise CW2 and the collective component-wise CCW2 strategies. The methods using the mean of the residual vector should be considered instead. Among these latter strategies, the component-wise CW(1) used in the work of Garnier *et al.*[37] gives the best

TABLE I. Convergence of A-VCI for CH_3CN with different basis expansion strategies. The used parameters are $F = 121$ and $\varepsilon = 7.5 \times 10^{-3}$. The approximation space $\Pi_{\mathbf{d}}$ is defined by $d_i = 30$, $i = 1 \ldots 6$. The maximum absolute error *Err* is evaluated with respect to Ref. 14. The cost of the algorithm is given by the number of iterations, the sizes of the final spaces, and the computational time. The number of non-zero elements (NNZ) in $H^{(j)}$ and $H_R^{(j)}$ at the convergence is also provided.

Expand strategy	Number of iterations	$B^{(j)}$ size	$B_R^{(j)}$ size	$H^{(j)}$ NNZ	$H_R^{(j)}$ NNZ	*Err* (cm^{-1})	Time (s)
CW(1)	6	86 238	5 421 360	13 166 360	51 581 366	0.305	5637
CW2	19	17 839	1 819 345	1 593 955	10 573 541	1.333	1230
CCW(1)	6	26 206	2 360 202	2 835 020	15 430 825	0.766	711
CCW(2)	7	24 158	2 168 898	2 607 190	14 201 050	0.997	667
CCW(3)	8	22 477	2 064 762	2 357 655	13 212 455	0.998	669
CCW(4)	10	22 432	2 082 831	2 339 942	13 197 787	0.803	723
CCW2	17	17 793	1 723 782	1 744 565	10 385 001	0.999	902

accuracy in few iterations, but requires numerous basis functions and an important computational cost. This behavior will quickly become prohibitive for systems of larger size than CH_3CN. The collective approaches CCW(p) are with no doubt to be privileged over CW(1) since they provide the best balance between the CPU time, the basis size, the number of iterations, and the error on eigenvalues. In CCW, the mean parameter p is the key to control the growth of the basis.

2. Convergence study of the A-VCI algorithm

In order to present the convergence properties of the A-VCI algorithm with respect to the threshold ε and the approximation space $\Pi_{\mathbf{d}}$, we search the 239 lowest eigenvalues of CH_3CN. The corresponding frequency range is around 0–3000 cm^{-1} and has been studied in many papers.[14,16,17] We perform a very accurate A-VCI calculation with a small threshold $\varepsilon_{ref} = 2.5 \times 10^{-4}$ and the CCW(2) strategy to obtain a reference spectrum. The maximal Hermite degrees per direction of the related discretized space $\Pi_{\mathbf{d}}$ corresponding to a maximal energy of 15 000 cm^{-1} are

$$\mathbf{d} = [42, 42, 17, 15, 15, 11, 11, 11, 7, 5, 5, 5].$$

Moreover, the initial basis $B^{(0)}$ is composed of the first 239 elements of $\Pi_{\mathbf{d}}$, and the largest degrees are given by [8, 8, 3, 2, 2, 2, 2, 2, 1, 0, 0, 0]. From now on, we will use these functions as the A-VCI starting space of CH_3CN.

For this computation, the C++ driver on *plafrim* is used. The algorithm reaches the convergence in 11 iterations producing a basis of 2 488 511 elements. We denote the associated eigenvalues by E_i^{ref}, for $i = 0 \ldots F-1$. The corresponding frequencies and their assignments are provided in the supplementary material. The maximal degrees of the final basis are [18, 18, 17, 12, 10, 11, 9, 7, 7, 5, 5, 5]. This final set is embedded in the binomial space BP(23). In Secs. IV A 2 a, IV A 2 b, and IV A 3, we compare our results to this reference calculation.

a. Influence of the threshold on the convergence. First we present the influence of the threshold on the accuracy of the eigenvalues and the basis size. We consider the same parameters as our reference computation described above. Let $Err(E)$ denote the maximum absolute error on all eigenvalues,

$$Err(E) = \max_{i \in \{0, F-1\}} |E_i^{\varepsilon} - E_i^{ref}|, \tag{23}$$

where E_i^{ε} is the eigenvalue obtained with the threshold ε.

Figure 1 shows the evolution of the maximal error on the eigenvalues according to the size of the final basis, and the evolution of the latter with the threshold, on a log-log scale. The maximal error nearly evolves as a decreasing power function of the size of the final basis. We deduce that the maximal error quickly and uniformly diminishes when the size of the final basis increases. Moreover, the behavior of the size of the basis with respect to the threshold parameter ε is similar, making the threshold a good parameter to control the growth of the basis. For this molecule, this figure shows that it is possible to determine an approximate basis size from a required accuracy on the eigenvalues and deduce from this size a maximal threshold value needed to reach this precision. For example, an error below 1 cm^{-1} at least requires a basis size of order 5×10^4, for which the threshold needs to be no more than 7×10^{-3}.

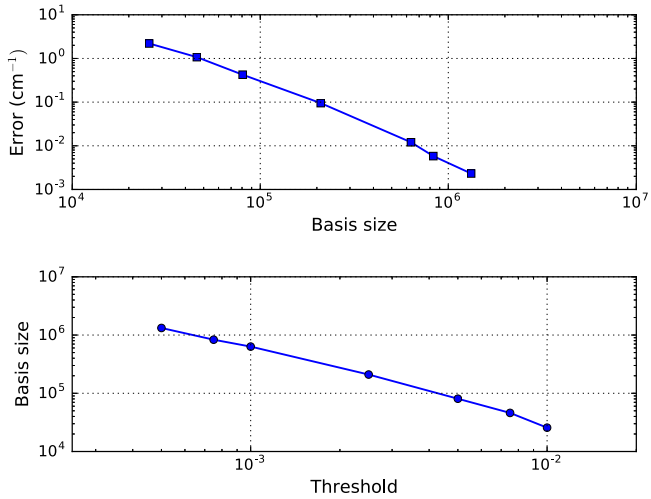

FIG. 1. Influence of the threshold ($\log_{10}$ scale) on the final basis size ($\log_{10}$ scale) and of the basis size on the maximum absolute error ($\log_{10}$ scale) for $E_{max} = 15\,000\ cm^{-1}$.

b. Influence of the truncation parameter of the research space. By using the approximation parameter E_{max} introduced in Sec. III B, we reduce the size of the approximation space and of the research space to find new directions.

Table II reports the maximal Hermite degrees per coordinate corresponding to E_{max} ranging from 8 500 to 15 000 cm^{-1}. Since the ω_{12} harmonic frequency is almost ten times larger than ω_1, we observe a large difference in the 1D-discretization between the coordinates. By increasing the maximal energy, we allow the algorithm to choose from a greater set of terms and thus to accelerate its convergence.

The maximal error of the first 239 eigenvalues with respect to the threshold ε is presented in Fig. 2 for different values of E_{max}. Due to the log-log scales, the curves for E_{max} between 9000 cm^{-1} and 12 000 cm^{-1} are not visible and give an error above 1 cm^{-1}. Except for the green curve $E_{max} = 14\,000\ cm^{-1}$, all the plot lines present a plateau, which means that even if we lower the threshold, we cannot improve the accuracy on the eigenvalues. This plateau decreases with the size of the approximation space. Moreover, the same holds the other way around: for ε higher than 2.5×10^{-3}, there is no need to increase E_{max} above 12 500 cm^{-1}. Finally, the green curve does not have a plateau because for all thresholds, the eigenvectors and the Hamiltonian are well described in the space $\Pi_{\mathbf{d}}$.

TABLE II. Maximal Hermite degrees of the product space $\Pi_{\mathbf{d}}$ corresponding to a given energy above the ground state E_{max}.

E_{max} (cm^{-1})	$\mathbf{d}$ in $\Pi_{\mathbf{d}}$
8 500	[24, 24, 10, 9, 9, 7, 6, 6, 4, 3, 3, 3]
10 000	[28, 28, 11, 10, 10, 8, 7, 7, 5, 4, 4, 4]
11 000	[31, 31, 12, 11, 11, 8, 8, 8, 5, 4, 4, 4]
12 000	[34, 34, 14, 12, 12, 9, 9, 9, 6, 4, 4, 4]
12 500	[35, 35, 14, 12, 12, 9, 9, 9, 6, 5, 4, 4]
13 000	[37, 37, 15, 13, 13, 10, 9, 9, 6, 5, 5, 5]
14 000	[39, 39, 16, 14, 14, 10, 10, 10, 7, 5, 5, 5]
15 000	[42, 42, 17, 15, 15, 11, 11, 11, 7, 5, 5, 5]

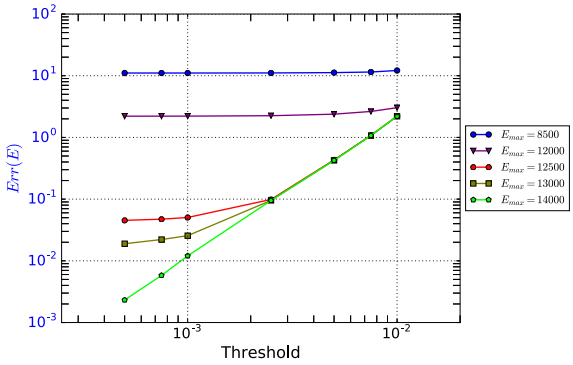

FIG. 2. Influence of the threshold ($\log_{10}$ scale) on the eigenvalues error ($\log_{10}$ scale) for different approximation spaces defined by E_{max}.

3. New results

Our computations highlight a significant gap with the results in Ref. 14, as shown in Fig. 3. Indeed, 35 of the 195 first frequencies have a difference above 0.1 cm^{-1} compared to our reference calculation. This is especially true for energy levels involving the C–C stretching ω_3 mode, namely, frequencies 38, 69, 70, 119–121, 150, 177, and 178 (all attributions are provided in the supplementary material). The 150th frequency is estimated at 2653.053 cm^{-1} in Ref. 14, whereas our reference calculation locates it at 2651.593 cm^{-1}. The difference between the two values calculated in the P_{24} and P_{27} spaces [see (14)] is around 2 cm^{-1} whereas it stays under 1 cm^{-1} for all other frequencies (see Table I in Ref. 14). It seems the convergence of this particular eigenvalue, described by $3\omega_3$ and $4\omega_3$, is hard to reach. On the first 201 frequencies, the mean error is 2.16 cm^{-1} and the RMS error is 1.40 cm^{-1}. However, A-VCI computes 238 frequencies under 3000 cm^{-1} whereas only 201 are reported in Ref. 14, and the highest 6 computed levels of Ref. 14 do not match with the frequencies 196–201 of the A-VCI calculation. We see that A-VCI is able to retrieve the last 6 vibrational states computed by Avila and Carrington[14] and to catch some energy levels missing from their paper, represented by frequencies 196–199, 202–228, 230–232, and

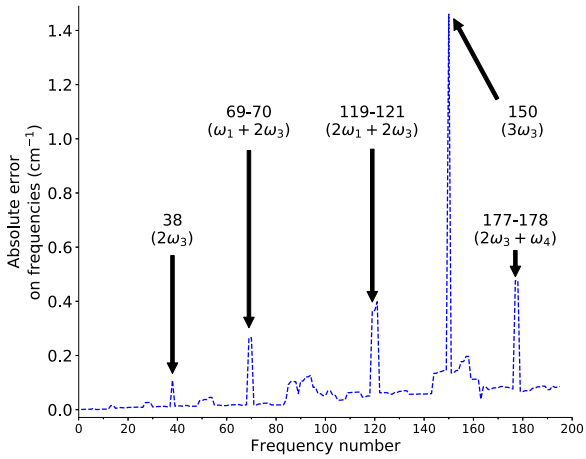

FIG. 3. Absolute error on the first 195 frequencies of CH_3CN computed in Ref. 14 with respect to the A-VCI reference calculation. The annotated frequency numbers correspond to energy states developed on the C-C stretching ω_3 (the main attributions are written in parentheses).

235–237. In particular, the levels 196–199 and 230–232 are developed on the sensitive ω_3 state.

To explain these discrepancies, we perform A-VCI computations in the same pruned basis as in Ref. 14. The corresponding pruning space P_b [see (14)] is defined by the following condition written in our ordering:

$$n_1 + n_2 + 3n_3 + 3n_4 + 3n_5 + 3n_6 + 4n_7 + 4n_8 + 4n_9 + 3n_{10} + 3n_{11} + 3n_{12} \leq b. \quad (24)$$

For this, we extend the algorithm by adding a pruning condition in the direct product search basis $\Pi_{\mathbf{d}}$. The condition only appears when we search the nodes connected to a given basis element [see (15)]. In fact, we discard nodes just before constructing the keys and then the overhead is negligible. By doing this, we are able to find an optimal basis in the pruned space.

Let us consider the 150th frequency, described by $3\omega_3$ and $4\omega_3$, with the largest absolute error. Table III shows its convergence when we increase the size of the pruning space. The A-VCI parameters are $F = 239$, $\varepsilon = 5.0 \times 10^{-3}$, and CCW(2). As shown in Table III, a pruned basis with $b = 41$ is necessary for the A-VCI algorithm to reach the convergence on the 150th frequency. Solving the eigenvalue problem in the corresponding space P_{41} with either a direct or iterative method is difficult due to its size, whereas the A-VCI method succeeds to extract the relevant states in a basis of 82 099 elements. Eventually, the final A-VCI basis in the product space defined by $E_{max} = 12\,500\,cm^{-1}$ counts 79 133 elements and leads to the same results obtained with the pruning condition (24) with $b = 41$. Compared to our reference calculation, the 150th frequency has an error of 0.272 cm^{-1}, and the largest error on all computed eigenvalues is 0.425 cm^{-1}. The full comparison between this latter calculation, the A-VCI reference calculation, and that of Avila and Carrington[14] is provided in the supplementary material.

B. Ethylene oxide molecule, C_2H_4O

We have shown that the A-VCI method accurately computes the spectrum of CH_3CN in a large frequency range of interest. Let us now consider a more difficult problem with the ethylene oxide molecule. To study the vibrational levels of this 7-atom system, we use the PES described in the work of Bégué *et al.*[3] The anharmonic part of this potential obtained at the B3LYP/6-31+G(d,p) level is composed of 180 cubic terms and 445 quartic terms. The 15 harmonic frequencies calculated at the CCSD(T)/cc-pVTZ level are sorted by ascending order,

$$\omega_1 = 815.515,\ \omega_2 = 850.180,\ \omega_3 = 899.564,$$

$$\omega_4 = 1\,052.231,\ \omega_5 = 1\,156.802,\ \omega_6 = 1\,157.906,$$

$$\omega_7 = 1\,174.993,\ \omega_8 = 1\,176.045,\ \omega_9 = 1\,300.108,$$

$$\omega_{10} = 1\,512.350,\ \omega_{11} = 1\,549.074,\ \omega_{12} = 3\,109.459,$$

$$\omega_{13} = 3\,117.878,\ \omega_{14} = 3\,196.560,\ \omega_{15} = 3\,211.265\ (cm^{-1}).$$

In this section, we consider the automatic collective component-wise (aCCW) strategy to expand the basis. All computations were run on *plafrim.*

TABLE III. Absolute error on the 150th frequency of CH_3CN using the A-VCI procedure in different pruned basis P_b. The pruning condition is given by Eq. (24). The number of computed eigenvalues is $F = 239$, and the threshold is $\varepsilon = 5 \times 10^{-3}$. The error is evaluated with respect to the reference calculation described in Sec. IV A 2. The last column corresponds to an A-VCI calculation with the same parameters performed in the product basis $\Pi_{\mathbf{d}}$ defined by $E_{max} = 12\,500\,\mathrm{cm}^{-1}$.

Basis	P_{27}	P_{33}	P_{37}	P_{40}	P_{41}	P_{45}	$\Pi_{\mathbf{d}}(12\ 500)$
Final basis size	58 122	79 409	79 743	81 594	82 099	82 405	79 133
150th frequency (cm^{-1})	2653.405	2652.101	2651.908	2651.884	2651.869	2651.867	2651.862
Error (cm^{-1})	1.815	0.511	0.318	0.294	0.279	0.277	0.272

1. The first 50 eigenvalues

We focus here on the 50 lowest energetic levels of C_2H_4O. The corresponding eigenvalues were computed by Brown and Carrington[17] in a basis of 2 955 289 functions using an adaptive procedure called "*O* Expand." The first difference between the *O* expand method and the A-VCI algorithm is the definition of the neighbors of the basis functions $\phi^0_{\mathbf{n}}$, $\mathbf{n} = (n_1, \ldots, n_D)$. In the first case, they are obtained by increasing one of the indices n_i by 1 for each $i = 1, \ldots, D$. In the other case, the neighbor $\mathbf{m}$ of a basis function $\mathbf{n}$ verifies $\langle \phi^0_{\mathbf{m}} | \mathcal{H} \phi^0_{\mathbf{n}} \rangle \neq 0$. The second difference concerns the selection of functions to add. In the *O* expand method, neighbors are added if a function of the active space is deemed important according to a criterion that focuses on its eigenvector. In the A-VCI method, for each and every function in the active space, the selection of the most important neighbors is based on their residual vectors. Using the *O* expand strategy, Brown and Carrington[17] proved it to be more efficient than with a binomial basis BP(11) of 7 726 160 elements. As far as we know, this *O* expand computation is the most accurate one to date on the first 50 eigenvalues of C_2H_4O. In this part, the maximum absolute error on the A-VCI eigenvalues is denoted *Err* and evaluated with respect to this reference calculation.

The A-VCI initial basis $B^{(0)}$ is made of the first 50 harmonic configurations of the product space defined by the Hermite degrees $\mathbf{d} = [6, 5, 5, 4, 4, 4, 4, 4, 3, 3, 3, 1, 1, 1, 1]$. The approximation spaces $\Pi_{\mathbf{d}}(E_{max})$ and BP(b) with different values of ε are considered. Table IV shows how the choice of these parameters influences the accuracy on eigenvalues, the basis size, and the computational time. The first three lines refer to A-VCI calculations in two approximation spaces $\Pi_{\mathbf{d}}$. These product spaces defined for a maximal energy of 10 000 cm^{-1} and 14 000 cm^{-1} lead to the maximal Hermite degrees $\mathbf{d} = [13, 12, 12, 10, 9, 9, 9, 9, 8, 7, 7, 4, 4, 4, 4]$ and $\mathbf{d} = [18, 17, 16, 14, 13, 13, 12, 12, 11, 10, 10, 5, 5, 5, 5]$, respectively. The last two lines of Table IV present a pruning approach in the binomial spaces BP(11) and BP(16). In this part, we use the aCCW(N_T, Δ) and the CCW(5) basis enlargement methods. The chosen aCCW tolerance is $\delta = 0.6$, and the maximal number of nodes added at each iteration is set according to the required threshold: $N_T = 30\,000$ for $\varepsilon = 5 \times 10^{-3}$ and $N_T = 50\,000$ for $\varepsilon = 2.5 \times 10^{-3}$. For the CCW method we use $p = 5$ to have a larger mean order than for the acetonitrile molecule, for which the optimal values of the p ranges between 1 and 4. This is needed to keep the CCW basis growth rate reasonable since we deal here with a larger system than CH_3CN.

In the considered computations, a threshold of 5×10^{-3} leads to an error of 0.4 cm^{-1} and final bases almost ten times smaller than the *O* expand set of Ref. 17, with less than 350 000 functions. For this level of precision, using the binomial basis BP(11) with the CCW(5) strategy (line 4) is faster than the aCCW expansion method applied in the product space $\Pi_{\mathbf{d}}(10\,000)$ (line 1). This is no longer true when we decrease the threshold to 2.5×10^{-3} to reach an accuracy around 0.04 cm^{-1}. In this case, the $\Pi_{\mathbf{d}}$ spaces give smaller basis and CPU times than the BP(b) spaces. A threshold of 2.5×10^{-3} with $E_{max} = 10\,000\,\mathrm{cm}^{-1}$ is enough to obtain an error below 0.1 cm^{-1}. A larger product space corresponding to $E_{max} = 14\,000\,\mathrm{cm}^{-1}$ leads to a maximum absolute error of 0.047 cm^{-1} in a basis twice smaller than in Ref. 17 (see line 3). This accuracy is confirmed by a mean error of 0.0163 cm^{-1} and an RMS error of 0.0195 cm^{-1}. In this latter calculation, the maximal Hermite degrees per direction of the final basis are [11, 9, 8, 8, 8, 10, 7, 7, 9, 6, 7, 5, 5, 5, 5] and the final basis is embedded in the binomial space BP(17).

Figure 4 presents how the mean order p and the number of points in $A^{(j)}$ fluctuate according to the iterations. The green curve shows how the number of nodes varies with the CCW(5) strategy. At the beginning, a small number of nodes

TABLE IV. Convergence of A-VCI for the 50 lowest eigenvalues of C_2H_4O. Several approximation spaces and thresholds are used with the aCCW and CCW(5) basis expansion strategies. The corresponding basis sizes and number of iterations are reported. The reference calculation for the maximum absolute error is provided in the work of Brown and Carrington.[17]

Space	Threshold	Iterations	*Err*	m	m_R	Strategy	Time (s)
$\Pi_{\mathbf{d}}(10\,000)$	5.0×10^{-3}	11	0.42	325 177	32 469 708	aCCW(30 000, 0.6)	1 406
$\Pi_{\mathbf{d}}(10\,000)$	2.5×10^{-3}	23	0.080	1 240 302	92 205 491	aCCW(50 000, 0.6)	12 190
$\Pi_{\mathbf{d}}(14\,000)$	2.5×10^{-3}	25	0.047	1 362 866	107 840 100	aCCW(50 000, 0.6)	15 116
BP(11)	5.0×10^{-3}	13	0.41	346 033	7216 711	CCW(5)	898
BP(16)	2.5×10^{-3}	27	0.042	1 439 356	105 416 196	aCCW(50 000, 0.6)	17 519

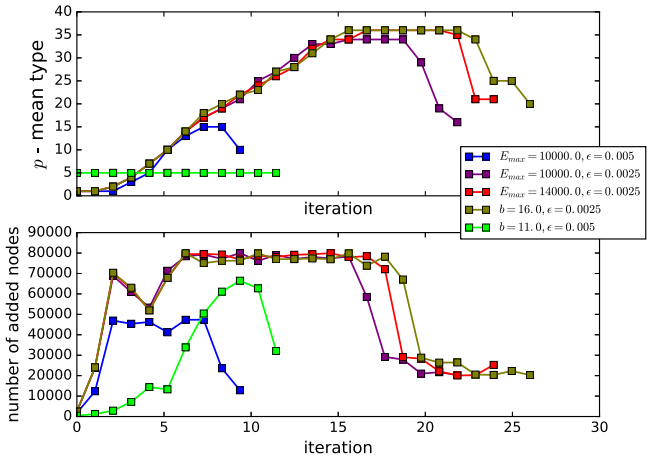

FIG. 4. Evolution of the average type p during the A-VCI iterations for C_2H_4O ($F = 50$). The basis expansion strategy is aCCW, except for the green curve obtained with CCW(5). The used approximation spaces are $\Pi_{\mathbf{d}}(E_{max})$ and BP(b).

are added in the basis, and it increases strongly during the iterations. On the opposite, with the aCCW strategy, we quickly reach the target number of nodes to add, but this strategy needs more iterations and time than CCW to obtain the convergence. Nevertheless, the aCCW strategy generally gives smaller final basis sets than CCW, leading to less memory consumption. The main advantage of aCCW over CCW is the ability to control the number of added nodes depending on the time we can afford to spend at each iteration. The use of aCCW is clearly a good option to address memory issues, especially when dealing with large molecular systems or hundreds of eigenvalues.

2. The first 200 eigenvalues

For the mid-IR spectrum of C_2H_4O, the domain of spectroscopic interest is between 2 800 and 3200 cm^{-1}. In this frequency range, the accurate assignment of active bands is still a challenging matter.[48–50] In the work of Thomas and Carrington,[50] they computed the first 200 eigenvalues by a tensor-type approach and the last calculated frequency was 3233 cm^{-1}. Brown and Carrington[17] showed that the computation of Ref. 50 is not accurate for the first 50 lowest frequencies and would be certainly worse for higher frequencies. The ground state calculated by Thomas and Carrington[50] is 12 461.860 cm^{-1} whereas the O expand basis of Ref. 17 provides a converged value of 12 461.467 cm^{-1}.

In this section, the starting subspace $B^{(0)}$ contains the first 200 elements of $\Pi_{\mathbf{d}}$, with $\mathbf{d}$ = [5, 5, 4, 4, 4, 4, 3, 3, 3, 3, 3, 2, 2, 2, 2] corresponding to the lowest vibrational states, and we expand the basis with the aCCW strategy in order to minimize the memory used by the algorithm. Table V presents a preliminary computation to evaluate which approximation space $\Pi_{\mathbf{d}}$ gives a good accuracy for a threshold of 5×10^{-3}. The ground state obtained is always 12 461.480 cm^{-1} except for $\Pi_{\mathbf{d}}$(12 000). This means that this space is too small to accurately approximate the eigenfunctions. This result is confirmed in Ref. 17 when they compare the error on their eigenvalues with those of Ref. 50. For E_{max} = 12 000 cm^{-1}, the maximum absolute error is 0.300 cm^{-1} (respectively, 23.124 cm^{-1}) while with $E_{max} \geq$ 13 000 cm^{-1} this error stabilizes at 0.258 cm^{-1} (respectively, 23.142 cm^{-1}). These first results also confirm that the computations presented in Ref. 50 have a low accuracy.

According to the previous results, we set E_{max} to 14 000 cm^{-1} to perform a more accurate computation with a threshold of 2.5×10^{-3} and the aCCW strategy with N_T = 400 000 for the targeted number of nodes to add at each iteration and δ = 0.25. For this large computation we decrease the memory footprint by removing the storage of the coefficients of matrix H_R. To do that, we compute twice the coefficients of the matrix to evaluate the scaled residual vectors. For each row of the matrix, all non-zero elements of the matrix are computed and then the components of the scaled residual vectors associated with this row are evaluated to build their norm. Once all the norms are built, for all rows of the vector $\mathbf{R_A}$ we re-compute the non-converged residue rows. Doing that, we drastically decrease the pressure on the memory but we double the computation time for building the scaled residual vector components and add them to obtain (20).

The final basis contains 7 118 214 elements and the size of the related search space is 419 666 917. We reach the convergence in 19 iterations lasting 3 days and 71 min. The total time is split as follows: 67.7% in the ARPACK solver, 28.2% to construct $\mathbf{R_A}$, and only 4.1% to update the structure (line 8 of the Algorithm 1). The matrix-vector products used in the ARPACK solver only take 25% of the total ARPACK time. Figure 5 presents the relative residues at the convergence. We observe that for the smallest 50 eigenvalues, we have a better convergence than in Sec. IV B 1. The maximal error on the 50 eigenvalues computed by Brown and Carrington[17] in a basis of 2 955 289 elements is 0.060 cm^{-1}. As expected, the discrepancies between A-VCI and that of Thomas and Carrington[50] are more important: the maximum absolute error on the 200 computed eigenvalues is 23.504 cm^{-1} (eigenvalue 198), the mean error is 6.97 cm^{-1}, and the RMS error is 8.73 cm^{-1}. Finally, the ground state evaluated

TABLE V. Convergence of the 200 lowest frequencies of C_2H_4O computed with A-VCI in $\Pi_{\mathbf{d}}$, with E_{max} ranging from 12 000 to 15 000 cm^{-1}. The used threshold is 5×10^{-3}.

E_{max} (cm^{-1})	Strategy aCCW(N_T, δ)	Iterations	m	m_R	Time (s)
12 000	(60 000,0.6)	24	1 469 373	108 175 774	44 985
13 000	(60 000,0.6)	26	1 593 865	12 495 044	56 220
13 500	(60 000,0.6)	26	1 593 892	12 497 716	56 190
14 000	(60 000,0.6)	26	1 593 892	124 497 874	56 434
15 000	(60 000,0.6)	26	1 593 892	124 497 898	56 503

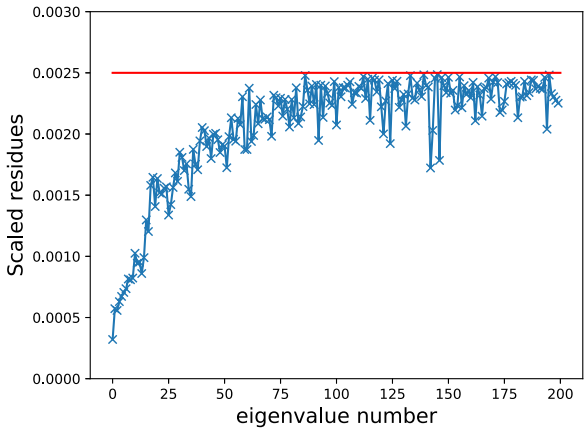

FIG. 5. Scaled residues at the convergence (iteration 19) for C_2H_4O ($\varepsilon = 2.5 \times 10^{-3}$ and $E_{max} = 14\,000\,cm^{-1}$).

at 12 461.473 cm^{-1} by A-VCI is close to the value found by Brown and Carrington.[17] The corresponding eigenvalues, frequencies, and assignments are reported in the supplementary material.

V. CONCLUSION

The A-VCI algorithm[37] is an adaptative procedure to compute iteratively a spectrum at any desired accuracy in the smallest possible VCI basis. It is applied to the resolution of the vibrational Schrödinger equation of two molecular systems (CH_3CN and C_2H_4O), for which sum of product potential energy surfaces, generated by Bégué *et al.*,[3,47] have been used by several authors[14,16,17,20,50,51] to produce the most accurate results currently available. In this paper, we used Hermite function product bases of normal coordinates to represent the Hamiltonian operator, making its calculation straightforward from analytic formulae. However, the algorithm could easily be adapted to other basis functions as soon as the Hamiltonian matrix structure is known; in other, words where its non-zero elements are located.

This iterative method builds nested spaces from an initial active space inside a much larger approximation space. The use of the Bauer-Fike theorem led us to define an *a posteriori* criterion to control the convergence of the algorithm and to select significant basis elements to add at each iteration. Three parameters, namely ε the threshold to control the convergence, E_{max}, to customize the size of the approximation space, and p to adjust the selection criterion at each iteration, have been introduced to increase the flexibility of the method. A full study of the influence of these parameters was carried out. We are now able to efficiently choose these parameters to reach the desired precision for a given PES and a given frequency range. In addition to providing a great accuracy for a large frequency range (0–3000 cm^{-1} for CH_3CN and 0–3200 cm^{-1} for C_2H_4O), the constraints on the parameters can be relaxed to reach an accuracy of $\simeq$1 cm^{-1} on the same energetic range, with very small resulting basis sets. Another novelty of this paper is the introduction of new enlargement techniques to iteratively augment the active basis set. These methods allow us to find a trade-off between the growth rate of the basis (memory cost) and the total number of iterations (CPU cost). Two families of augmentation techniques were developed in this work, namely, the component-wise (CW) and the collective component-wise strategies (CCW). The CW uses a single residual vector in order to iteratively add basis functions to the active space. The CCW combines the residues corresponding to all eigenvectors in a mean residual vector to select the most relevant directions. Our results clearly show that the collective strategies are the most efficient since they offer the best compromise between the computational time and the memory usage for a required accuracy on the calculated eigenvalues. The choice of a particular collective method over another will be guided by the hardware constraints. For memory-bound cases, especially when dealing with large systems as C_2H_4O, we would prefer to limit the number of added basis elements using a CCW, increasing the CPU time. Otherwise, the CCW(p) techniques offer the finest control on the basis expansion and the fastest convergence as soon as the available memory is not a problem.

Our approach to manage the basis elements and the matrix allows us to handle molecular systems up to 7 atoms in three days on a 24-core computer with 128 GB of memory capacity. This limit can be lifted by using a distributed memory programming (Message Passing Interface) to spread the workload over different processing units.

SUPPLEMENTARY MATERIAL

See supplementary material for the results reported for CH_3CN and C_2H_4O molecules, which are compared to the best calculations that can be obtained using the A-VCI algorithm.

ACKNOWLEDGMENTS

We thank G. Avila and T. Carrington for providing us the potential energy surface of the CH_3CN molecule that was used for the comparison tests. A part of this work was funded by the grand CRA-HPC of the Conseil Régional d'Aquitaine. Experiments presented in this paper were carried out using the PlaFRIM experimental computing platform, being developed under the Inria PlaFRIM development action with support from Bordeaux INP, LABRI, IMB, and other entities: Conseil Régional d'Aquitaine, Université de Bordeaux, CNRS, and ANR in accordance to the programme Investissements d'Avenir (see www.plafrim.fr). Some numerical computations were also carried out using the MCIA platform (Mésocentre de Calcul Intensif Aquitain, see www.mcia.univ-bordeaux.fr). Finally we acknowledge the Direction du Numérique of the Université de Pau et des Pays de l'Adour for the computing facilities.

APPENDIX: HERMITE FUNCTIONS

The 1-D normalized Hermite function of degree n writes

$$\psi_n(q) = \left(\frac{1}{\pi}\right)^{1/4} \frac{1}{\sqrt{2^n\, n!}} \cdot e^{-\frac{q^2}{2}} \cdot \mathrm{He}_n(q)\,,$$

where He_n is the normalized Hermite polynomial of degree n. These polynomials are orthogonal for the weight $e^{-\frac{q^2}{2}}$. Moreover we have

$$\int_{\mathbb{R}} \psi_n(x)\psi_m(x)dx = \delta_{n,m}.$$

The sequence of Hermite polynomials satisfies the recursion

$$\mathrm{He}_{n+1}(q) = q\mathrm{He}_n - n\mathrm{He}_{n-1},$$

and ψ_n are the eigenfunctions of the 1-D harmonic Hamiltonian operator defined by

$$\mathcal{H}_0(q) = -\frac{\partial^2}{\partial q^2} + q^2.$$

The properties of Hermite functions provide an analytic formula to calculate the Hamiltonian matrix coefficients as shown in the following propositions.

Proposition 1 (see Ref. 52). The indices m such that $\langle\phi_n(q)|q^s\phi_m(q)\rangle \neq 0$ *are given as follows:*

- *If* $s = 2p$, *then* $m = n + 2l$ *for* $-s/2 \leq l \leq s/2$ *if* $s = 2p$.
- *If* $s = 2p + 1$, *then* $m = n + 2l + 1$ *for* $-(s-1)/2 \leq l \leq (s-1)/2$.

Proposition 2 (see Ref. 52). The tensor moment $\mathcal{M}(s, n, m) = \langle\phi_n(q), q^s\phi_m(q)\rangle$ *has the following analytic expression:*

$$\langle\phi_n(q)|q^s\phi_m(q)\rangle = \begin{cases} 0 & \text{if } s-n-m \text{ is odd}, \\ \dfrac{s!}{2^r}\sqrt{\dfrac{2^{m+n}}{m!n!}} \displaystyle\sum_{p=\max(0,-r)}^{\min(n,m)} C_p^n C_p^m \frac{p!}{2^p(r+p)!} & \text{otherwise}, \end{cases} \tag{A1}$$

with $r = (s-n-m)/2$.

[1] J. Bowman, K. Christoffel, and F. Tobin, J. Phys. Chem. **83**, 905 (1979).
[2] S. Carter and N. Handy, Comput. Phys. Rep. **5**, 117 (1986).
[3] D. Bégué, N. Gohaud, C. Pouchan, P. Cassam-Chenai, and J. Liévin, J. Chem. Phys. **127**, 164115 (2007).
[4] J. Bowman, T. Carrington, and H.-D. Meyer, Mol. Phys. **106**, 2145 (2008).
[5] J. C. Light and T. Carrington, Jr., Adv. Chem. Phys. **114**, 263 (2000).
[6] A. Viel and C. Leforestier, J. Chem. Phys. **112**, 1212 (2000).
[7] H. Romanowski, J. M. Bowman, and L. B. Harding, J. Chem. Phys. **82**, 4155 (1985).
[8] P. Cassam-Chenaï and J. Liévin, Int. J. Quantum Chem. **93**, 245 (2003).
[9] C. Pouchan and K. Zaki, J. Chem. Phys. **107**, 342 (1997).
[10] I. Baraille, C. Larrieu, A. Dargelos, and M. Chaillet, Chem. Phys. **273**, 91 (2001).
[11] G. Rauhut, J. Chem. Phys. **127**, 184109 (2007).
[12] M. Neff and G. Rauhut, J. Chem. Phys. **131**, 124129 (2009).
[13] J. Cooper and T. Carrington, J. Chem. Phys. **130**, 214110 (2009).
[14] G. Avila and T. Carrington, Jr., J. Chem. Phys. **134**, 054126 (2011).
[15] G. Avila and T. Carrington, J. Chem. Phys. **137**, 174108 (2012).
[16] G. Avila and T. Carrington, Chem. Phys. **482**, 3 (2016).
[17] J. Brown and T. Carrington, J. Chem. Phys. **145**, 144104 (2016).
[18] M. J. Bramley and T. Carrington, Jr., J. Chem. Phys. **99**, 8519 (1993).
[19] A. Shimshovitz, Z. Bačić, and D. J. Tannor, J. Chem. Phys. **141**, 234106 (2014).
[20] T. Halverson and B. Poirier, Chem. Phys. Lett. **624**, 37 (2015).
[21] T. Halverson and B. Poirier, J. Phys. Chem. A **119**, 12417 (2015).
[22] U. Manthe and H. Kppel, J. Chem. Phys. **93**, 345 (1990).
[23] M. J. Bramley, J. W. Tromp, T. Carrington, Jr., and G. C. Corey, J. Chem. Phys. **100**, 6175 (1994).
[24] L. Halonen, D. W. Noid, and M. Child, J. Chem. Phys. **78**, 2803 (1983).
[25] S. Carter, S. J. Culik, and J. M. Bowman, J. Chem. Phys. **107**, 10458 (1997).
[26] X.-G. Wang and T. Carrington, J. Phys. Chem. A **105**, 2575 (2001).
[27] H.-G. Yu, J. Chem. Phys. **117**, 2030 (2002).
[28] B. Poirier, J. Theor. Comput. Chem. **02**, 65 (2003).
[29] R. Dawes and T. Carrington, Jr., J. Chem. Phys. **122**, 134101 (2005).
[30] R. Dawes and T. Carrington, Jr., J. Chem. Phys. **124**, 054102 (2006).
[31] Y. Scribano and D. M. Benoit, Chem. Phys. Lett. **458**, 384 (2008).
[32] J. Chang and R. E. Wyatt, J. Chem. Phys. **85**, 1840 (1986).
[33] B. Hartke, Phys. Chem. Chem. Phys. **8**, 3627 (2006).
[34] J. Sielk, H. F. von Horsten, F. Kruger, R. Schneider, and B. Hartke, Phys. Chem. Chem. Phys. **11**, 463 (2009).
[35] S. Machnes, E. Assmat, H. R. Larsson, and D. J. Tannor, J. Phys. Chem. A **120**, 3296 (2016).
[36] J. Brown and T. Carrington, Jr., J. Chem. Phys. **143**, 044104 (2015).
[37] R. Garnier, M. Odunlami, V. Le Bris, D. Bégué, I. Baraille, and O. Coulaud, J. Chem. Phys. **144**, 204123 (2016).
[38] M. Quack, Annu. Rev. Phys. Chem. **41**, 839 (1990).
[39] M. E. Kellman and V. Tyng, Acc. Chem. Res. **40**, 243 (2007).
[40] S. Carter, J. M. Bowman, and A. R. Sharma, AIP Conf. Proc. **1504**, 465 (2012).
[41] G. Avila and T. Carrington, Jr., J. Chem. Phys. **139**, 134114 (2013).
[42] X. Wang, S. Carter, and J. M. Bowman, J. Phys. Chem. A **119**, 11632 (2015).
[43] N. C. Handy and S. Carter, Mol. Phys. **102**, 2201 (2004).
[44] F. Bauer and C. Fike, Numerische Math. **2**, 137 (1960).
[45] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis* (Cambridge University Press, Cambridge, New York, Melbourne, 1994).
[46] R. Lehoucq, D. Sorensen, and C. Yang, *ARPACK Users' Guide* (SIAM, 1998).
[47] D. Begue, P. Carbonniere, and C. Pouchan, J. Phys. Chem. A **109**, 4611 (2005).
[48] R. C. Lord and B. Nolin, J. Chem. Phys. **24**, 656 (1956).
[49] C. Puzzarini, M. Biczysko, J. Bloino, and V. Barone, Astrophys. J. **785**, 107 (2014).
[50] P. S. Thomas and T. Carrington, Jr., J. Phys. Chem. A **119**, 13074 (2015).
[51] A. Leclerc and T. Carrington, J. Chem. Phys. **140**, 174111 (2014).
[52] R. M. Wilcox, J. Chem. Phys. **45**, 3312 (1966).