Received: 10 August 2021 | Revised: 20 September 2021 | Accepted: 20 September 2021

DOI: 10.1002/jcc.26764

Journal of COMPUTATIONAL CHEMISTRY WILEY

FULL PAPER

Advances in vibrational configuration interaction theory - part 2: Fast screening of the correlation space

Tina Mathea | Taras Petrenko | Guntram Rauhut

Institute for Theoretical Chemistry, University of Stuttgart, Stuttgart, Germany

Correspondence
Guntram Rauhut, Institute for Theoretical Chemistry, University of Stuttgart,
Pfaffenwaldring 55, D-70569 Stuttgart,
Germany.
Email: rauhut@theochem.uni-stuttgart.de

Funding information
Deutsche Forschungsgemeinschaft, Grant/Award Number: Ra 656/25-1; Studienstiftung des Deutschen Volkes

Abstract

For larger molecules, the computational demands of configuration selective vibrational configuration interaction theory (cs-VCI) are usually dominated by the configuration selection process, which commonly is based on second order vibrational Møller-Plesset perturbation (VMP2) theory. Here we present two techniques, which lead to substantial accelerations of such calculations while retaining the desired high accuracy of the final results. The first one introduces the concept of configuration classes, which allows for a highly efficient exploitation of the analogs of the Slater-Condon rules in vibrational structure calculations with large correlation spaces. The second approach uses a VMP2 like vector for augmenting the targeted vibrational wavefunction within the selection of configurations and thus avoids any intermediate diagonalization steps. The underlying theory is outlined and benchmark calculations are provided for highly correlated vibrational states of several molecules.

KEYWORDS
configuration selection, vibrational configuration interaction theory, vibrational structure theory

1 | INTRODUCTION

Vibrational configuration interaction (VCI) theory allows for the accurate calculation of state energies,[1–5] but usually on cost of considerable computational demands. In part 1 of this article series (cf. Ref. [6]) we reported about the acceleration of configuration-selective VCI calculations arising from a rigorous exploitation of the antisymmetry of the spectroscopic ζ-constants, the analogs of the Slater-Condon rules in finite basis vibrational structure theory and the introduction of subspace diagonalizations for obtaining physically meaningful and reliable start vectors for the iterative determination of vibrational eigenstates. These concepts led to considerable speed-ups with respect to the overall computation times and thus paved the way for even larger VCI calculations, which allow for highly accurate state determinations. In the second part of this series, even further accelerations are presented originating from two technical refinements concerning mainly the configuration selection process. The first one introduces the concept of configuration classes, which allows to process whole blocks of configurations instead of individual ones. This results in a significantly enhanced screening of the correlation space and can be applied to both, the configuration selection process and the evaluation of the VCI matrix. The second technique exploits state vectors obtained from an expression related to second order vibrational Møller-Plesset perturbation (VMP2) theory, which replace VCI state vectors and thus allow to avoid intermediate diagonalizations in the iterative selection process. The benefits of this approach arise mainly for highly correlated vibrational states–as observed in regions of high-state densities, which require many configurations and thus expensive diagonalizations for their proper representation. In order to be consistent with respect to the first part of this article series, the same set benchmark molecules, namely B_2H_6, C_3H_4, and C_2H_5F, or, more precisely, the same set of potential energy surfaces (PES) has been used, which rely on **n**-mode expansions being truncated after the 4-mode coupling terms. However, we have augmented this set of molecules by the quasi-linear HCCNCS molecule, which shows a Champagne bottle potential with strong quartic contributions. This system, whose potential has also been truncated after the 4-mode coupling terms, shows exceptionally long diagonalization times and thus behaves differently than the other three systems. Details about these have been presented in separate publications, cf. Refs. [7–9] In addition, we also present benchmarks obtained by applying the refinements presented in both parts of this article series in order to demonstrate the overall performance, which can be achieved in calculations aiming at high accuracy.

While the first technique presented in this contribution here can be rigorously applied within any implementation of VCI theory, the second aspect is more specific and requires a VMP2 like selection criterion. We explicitly note here that different approaches for the selection of configurations have been presented in the literature,[10–12] which all show some pros and cons. However, the VMP2 like criterion is applicable to any type of wavefunctions and potentials and is used in several implementations.[13,14]

2 | VCI PROGRAM STRUCTURE - CONFIGURATION SELECTION

In the following, we briefly outline aspects of our VCI implementation being relevant for a general understanding, which is indispensable for revealing the computational bottlenecks. For additional and detailed information concerning our VCI algorithm, we refer to Refs. [15–17].

In order to be able to handle large systems, we employ an iterative configuration selective VCI algorithm based on the commonly used Watson-Hamiltonian.[18] The VCI wavefunction $|\Psi_{\text{VCI}}\rangle$ is given as a linear combination of Hartree products

$$|\Psi_{\text{VCI}}\rangle = c_0\,|\Phi_0\rangle + \sum_S c_s\,|\Phi_S\rangle + \sum_D c_D\,|\Phi_D\rangle + \ldots, \tag{1}$$

that is, the wavefunction is expanded in terms of singly (S), doubly (D), triply (T), $\cdots$ excited configurations $|\Phi\rangle$ regarding the reference $|\Phi_0\rangle$, which corresponds to the vibrational self-consistent field (VSCF) solution. The configurations themselves may be either products of one-dimensional harmonic oscillator functions, or, as used in this work, one-dimensional functions (modals) obtained by solving the VSCF equations. The n_i-th modal in coordinate q_i will be denoted by $\varphi_i^{n_i}$ in the following, that is, a configuration is given as $|\Phi\rangle = |\prod_i \varphi_i^{n_i}\rangle$. Note, that in our implementation of VCI, we exclusively use real basis functions. For further information about handling non-Abelian systems in this workaround see Ref. [17].

Initially a correlation space will be generated being restricted by (a) the number of excitations within a single mode, (b) the maximal number of modes being excited and (c) the total number of quanta within the configuration. In order to subsequently reduce the initial configuration space, we apply the aforementioned VMP2-based selection criterion[19,15]

$$\varepsilon_{AJ}^{(a+1)} = \frac{\left|\sum_{K\in\{a\}} c_{AK}^{(a)} \langle \Phi_K | H' | \Phi_J \rangle\right|^2}{\varepsilon_A^{(a)} - \varepsilon_J} \tag{2}$$

for the purpose of selecting configurations that are supposed to have a non-negligible contribution to the total energy of the state of interest. If $\varepsilon_{AJ}^{(a)}$ is larger than a certain threshold, the tested configuration $|\Phi_J\rangle$ will be included in the configuration space, otherwise not. The VCI wavefunction of a state A in the a - th iteration step is thus given as

$$|\Psi_A^{(a)}\rangle = \sum_{K\in\{a\}} c_{AK}^{(a)}\,|\Phi_K\rangle, \tag{3}$$

with $\{a\}$ being the set of selected configurations until the ath iteration. Within the configuration selection the reference state, cf. Equation (3), is chosen to be the eigenstate of the last iteration step having the largest overlap with the harmonic counterpart.[20,17] This ensures that the configuration space is appropriately chosen to describe the target state. Note that, in the first iteration step, the VSCF reference configuration (in case of molecules belonging to Abelian point groups and canonical normal coordinates) or a wavefunction in a meaningful subspace is used - for details see Ref. [6]. Subsequently, the correlation space is iteratively increased until convergence by using Equation (2). In order to determine the respective eigenvalue of the intermediate VCI matrix, we employ an iterative eigenvalue solver based on the RACE-algorithm,[21] which minimizes the residual norm. This algorithm has been shown to surpass the performance of the commonly used Jacobi-Davidson algorithm. Within the selection iterations, it has to be guaranteed that the correct state is tracked, which is realized by a physically meaningful start vector.[17] Typically, many of the coefficients $c_{AK}^{(a)}$ of the eigenvector in Equation (3) are very small. In order to reduce the computational effort within the configuration selection via Equation (2), which includes a sum over all $K \in \{a\}$, we adjust the eigenvector for insignificant coefficients, that is, configurations belonging to very small coefficients are skipped in the selection process. Note, that the concerning configurations are not removed from the current correlation space, but they are simply not considered in Equation (2).

The procedure described here leads to a VCI algorithm, in which the configuration space is built up iteratively. Each iteration step consists of three parts: (a) Building the VCI matrix within the current correlation space, (b) Diagonalization of this VCI matrix by using an appropriate start vector for the RACE algorithm in order to track the correct state (which is especially important when multi-resonance effects are present), (c) Selecting new configurations based on the VMP2-like criterion (2) using the VCI wavefunction of the eigenstate with the largest overlap integral with respect to the harmonic reference state. Obviously, all three steps mentioned above suffer from large correlation spaces. The reasons are evident: An increasing dimension of configuration spaces leads to higher computational cost for (i) building up the VCI matrix since there are more elements to be evaluated, (ii) diagonalization of the VCI matrix since it grows larger and (iii) selecting configurations since more integrals have to be evaluated and summed up. Therefore it is attractive to reduce (a) the computational cost of individual matrix element evaluations (see part 1 of this article series), (b) the number of matrix elements to be calculated (see Section 3 and 4 and part 1 of this article series), (c) the number of iterations needed (see Section 4 and part 1 of this article series) and (d) the number of diagonalization steps (see below, Section 4).

Energetically separated vibrational states usually require a modest number of iterations within the described VCI procedure. In contrast, states energetically located in a region of high density, as for example CH-stretching modes, tend to slower convergence. Whereas for the first group of states usually less than eight iterations are necessary in order to achieve convergence with respect to the energy eigenvalue of the state of interest, states with high energy may necessitate up to 20 iterations. Therefore, computation times for calculating CH-stretching modes, certain overtones and combination bands in regions of high state density, may be excessive. For this reason, we will focus in particular on such states in our benchmark calculations in order to prove our developments to be very efficient.

3 | CLASSES OF CONFIGURATIONS

3.1 | Method and implementation

There are two computationally demanding parts within the program structure, in which the calculation of matrix elements is needed, namely the set-up of the VCI matrix and the configuration selection step. Both parts suffer from large configuration spaces. Quantitatively, the scaling is the following:

- During the configuration selection, all configurations not included in the current configuration space need to be tested regarding their energy contribution via the VMP2-like energy expression (2). The upper bound of the sum within Equation (2) is given by the number of selected configurations $N_{\text{conf,sel.}}(a)$ in the current iteration step a. The expression itself has to be evaluated $N_{\text{conf,tot.}} - N_{\text{conf,sel.}}(a)$ times with $N_{\text{conf,tot.}}$ being the number of configurations present in the initial configuration space. Obviously, the computational effort for the calculation of a single term (2) dramatically increases with growing $N_{\text{conf,tot.}}$, on the other hand, the sum of elements depends on $N_{\text{conf,sel.}}$.
- In contrast, the number of new VCI matrix elements that have to be evaluated within one iteration step, is approximately given as $\frac{1}{2}\left(N^2_{\text{conf,sel.}}(a+1) - N^2_{\text{conf,sel.}}(a)\right)$, since the matrix is Hermitian. This holds, if the matrix elements from the previous iteration step are kept in memory and need not to be recalculated as in our implementation.

Obviously, in order to reduce the number of elements, the analogs of the Slater-Condon rules, which exploit orthogonality of the basis functions with respect to the order of the operator considered, can be applied to sort out vanishing integrals without explicitly calculating them. Note that, this corresponds to the concept of active and passive terms in vibrational coupled-cluster theory.[22] Nevertheless, a large computational effort remains, especially within the configuration selection process due to the scaling with $N_{\text{conf,tot.}}$. As shown below, by grouping configurations having the same properties regarding the reference configuration and driving the loop structure by these "classes" instead of single configurations significant reductions in CPU time can be achieved.

The VCI wavefunction in a specific iteration step is expanded in terms of configurations according to Equation (3). Technically, within our configuration selective implementation of VCI, we employ two lists, namely the

i. list of (binary coded) configurations in the initial configuration space, which is used in order to determine the energetic contribution of a single configuration via Equation (2), and
ii. the list of configurations already selected via Equation (2) used in order to generate the VCI wavefunction (cf. Equation (3)) and the VCI matrix and to apply the VMP2-like criterion again in the next iteration step.

In order to maximally exploit the analogs of the Slater-Condon rules to gain a smaller number of matrix elements to be evaluated in (a) the matrix set-up and (b) the configuration selection, we presort the configurations in the initial configuration space. Subsequently, we combine certain configurations into blocks, the aforementioned (configuration) classes. The criterion for sorting is the following: Let us consider the set of modes $M_m(|\Phi_I\rangle, |\Phi_J\rangle)$ with $|M_m(|\Phi_I\rangle, |\Phi_J\rangle)| = m$. In these modes, the two configurations $|\Phi_I\rangle$ and $|\Phi_J\rangle$ differ in the respective quantum numbers. All configurations $|\Phi_K\rangle$ sharing the same $M_m(|\Phi_0\rangle, |\Phi_K\rangle)$ with respect to the reference configuration $|\Phi_0\rangle$ are elements of the same class. The reason is obvious: When building the VCI matrix and especially within the configuration selection via Equation (2), the respective integral vanishes if for two configurations $|\Phi_I\rangle$ and $|\Phi_J\rangle$ it holds $|M_m(|\Phi_I\rangle, |\Phi_J\rangle)| > P$, with P being the order of the operator considered, e.g. the potential energy operator or the VAM operator. Thus, sorting the initial configuration space by M_m with respect to $|\Phi_0\rangle$ leads to blocks of configurations with the same properties regarding vanishing integrals. Instead of checking every single configuration for obeying the Slater-Condon rules or even evaluating every single integral, the contribution of whole classes is tested. If a specific class contributes, the respective matrix elements are explicitly calculated, otherwise the whole block is skipped. Obviously, this enables to skip a large number of elements at once. Within the implementation, the loop structure is now driven by classes instead of single configurations. This leads to (a) a significant reduction of if-statements passed and (b) significantly smaller loop lengths.

3.1.1 | Configuration selection

Let K_{tot}, $|K_{\text{tot}}| = N_{\text{conf,tot.}}$ be the set of configurations within the initial configuration space and $K_{\text{sel}}(a)$ the set of configurations that are included in the configuration space in iteration step a, $|K_{\text{sel}}(a)| = N_{\text{conf,sel.}}(a)$. Generally, using the analogs of the Slater-Condon rules, the VMP2-like criterion (2) can be rewritten as

$$\varepsilon_{AJ}^{(a+1)} = \frac{\left|\sum_{I \in K_{\text{sel}}(a) \wedge |M_m(|\Phi_I\rangle, |\Phi_J\rangle)| \leq P}^{N_{\text{conf,sel(a)}}} c_{AI}^{(a)} \langle \Phi_I | H' | \Phi_J \rangle \right|^2}{\varepsilon_A^{(a)} - \varepsilon_J}, \tag{4}$$

with P being the maximum order of the coupling terms present in the PES and $J \in K_{\text{tot}} \backslash K_{\text{sel}}(a)$. Note that H' does not include any VAM contributions. Equation (4) refers to our previous implementation not using the concept of classes introduced here, but application of the Slater-Condon rules to each configuration separately. All reference CPU times shown in the following are based on Equation (4).

We now mathematically introduce (configuration) classes. As a class C_{M_m} we define a set of configurations $|\Phi_I\rangle$ sharing the same $M_m(|\Phi_0\rangle, |\Phi_I\rangle)$:

$$C_{M_m} = \left\{ |\Phi_1^{M_m}\rangle | \Phi_2^{M_m}\rangle \ldots |\Phi_{|C_{M_m}|}^{M_m}\rangle \right\}. \tag{5}$$

With the definition (5), the sets $K_{\text{sel}}(a)$ and $K_{\text{tot}} \backslash K_{\text{sel}}(a)$ can be written as the unions

$$K_{\text{sel}}(a) = \bigcup_{\tau=1}^{N_{C,\text{sel}}(a)} C_{M_m,\tau}^{\text{sel}} \tag{6}$$

and

$$K_{\text{tot}} \backslash K_{\text{sel}}(a) = \bigcup_{\rho=1}^{N_{C,\text{nsel}}(a)} C_{M_m,\rho}^{\text{nsel}}, \tag{7}$$

with $N_{C,\text{sel}(a)}$ and $N_{C,\text{nsel}(a)}$ being the number of classes in the set of the selected configurations in iteration step a and the non-selected ones, respectively. Employing the concept of configuration classes via Equations (6) and (7), we can rewrite Equation (4) as

$$\varepsilon_{AJ}^{(a+1)} = \frac{\left| \sum_{\tau: |M_m(|\Phi_1^\tau\rangle, |\Phi_J\rangle)| \leq P}^{N_{C,\text{sel}}(a)} \sum_{I}^{|C_{M_m,\tau}^{\text{sel}}|} c_{AI}^{(a)} \langle \Phi_I | H' | \Phi_J \rangle \right|^2}{\varepsilon_A^{(a)} - \varepsilon_J} \tag{8}$$

Note, that a specific element $\varepsilon_{AJ}^{(a+1)}$ is only calculated in the case that $|\Phi_J\rangle \in C_{M_m}^{\text{nsel}}$ with $|M_m(|\Phi_1^\tau\rangle | \Phi_1^\rho\rangle)| \leq P|$, that is, $|\Phi_J\rangle$ has to be an element of a contributing class of the non-selected configurations. Technically, we test all the first elements (as all elements of a class have same properties, we choose the first one) of each class via

$$|M_m(|\Phi_1^\tau\rangle | \Phi_1^\rho\rangle)| \leq P \tag{9}$$

with $\tau \in C_{M_m}^{\text{sel}}$ and $\rho \in C_{M_m}^{\text{nsel}}$. Subsequently, we generate a list of the contributing pairs and calculate the elements from these classes.

Thus, regarding efficiency, there are two aspects: First, the summation over the number of configurations in the current configuration space in iteration step a is replaced by the much smaller one over the number of configuration classes. Only in the case that a configuration class renders a non-zero contribution according to the Slater-Condon rules, the matrix element belonging to the concerning class is explicitly evaluated. Second, we screen the classes of the remaining non-selected configurations. Both together, a loop structure consisting of two large loops is broken down to four smaller ones. It is important to notice, that the initial configuration space can be used much more efficiently for screening via the Slater-Condon rules, as the classes are much larger than in the list of selected configurations.

3.1.2 | Set-up of the VCI matrix

For building the VCI matrix, the same technique as described for the configuration selection is applied, but for this case only set (6) is relevant. As described before, in every iteration we solely calculate the missing matrix elements regarding the configurations of the last step. Thus, the list of new configurations is, compared to the number of elements occurring with the configuration selection, relatively small. Therefore, there are less elements in a specific class and the resulting computational saving has to be expected to be significantly smaller.

3.2 | Results

In order to demonstrate the CPU timesavings arising from configuration classes, benchmark calculations for B_2H_6, C_3H_4, and C_2H_5F have been performed. Both, computational savings arising from the matrix set-up and the configuration selection are depicted in Figure 1 for allene and are listed for all systems in Tables 1 (VCI matrix set-up) and 2 (configuration selection).

Since the impact of classes strongly depends on the order of the operator, we performed different sets of tests: Regarding the computational cost for evaluating the VCI matrix, we considered potential energy surfaces up to 3- and 4-mode couplings, whereas VAM terms were entirely neglected or included up to 0D. Note that, within the configuration selection according to Equation (2), VAM terms are always neglected, but are considered within the (intermediate) VCI matrix set-up. As restrictions for the initial configuration spaces, we used a maximum sum of quantum numbers of 15 with at most six modes being simultaneously excited. For allene, a maximum excitation of seven per mode has been utilized, six for B_2H_6 and five for C_2H_5F. As reference, CPU times without using classes have been determined.

As can been seen in Figure 1, the computational savings are much larger for the configuration selection process (last two groups of bars on the rhs) as for the matrix set-up (4 bar groups on the lhs). The red bars depict the reference calculation without classes, the blue ones show the CPU times employing this new concept. Note, that the scaling for the first y-axis is logarithmic. The lines refer to the second y-axis on the right hand side and provide the mean and total saving instead.

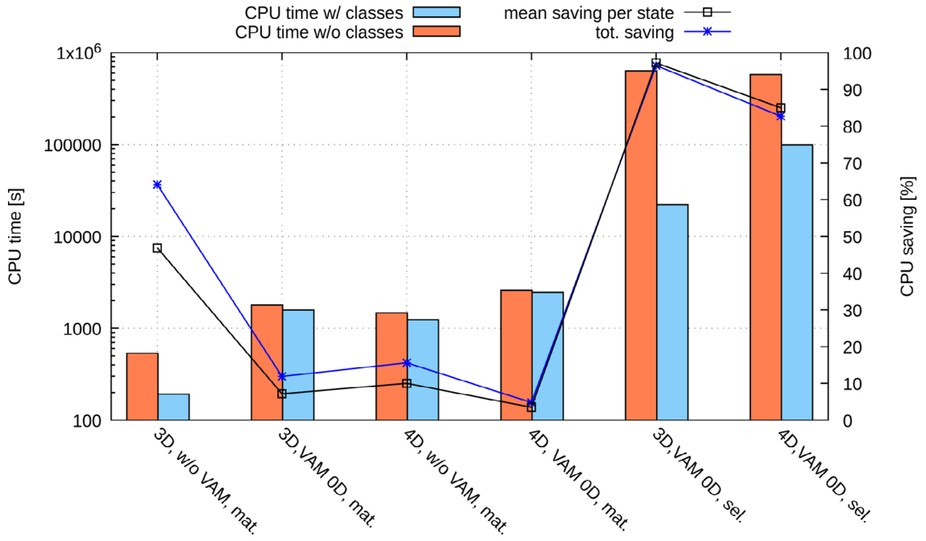

FIGURE 1 Comparison of CPU times for building the VCI matrix (abbreviated as “mat.”) and determining the correlation space using the VMP2-like criterion (abbreviated as “sel.”) in the cases of (A) using the concept of configuration classes (blue bars) and (B) without (red bars). The values given refer to the calculation of the fundamental transitions of C_3H_4, that is, 11 states have been considered. The labels 3D and 4D refer to the order of the multimode expansion of the PES. Note, that the first y-axis giving the CPU time has a logarithmic scaling; the second is non-logarithmic and shows the CPU time savings

TABLE 1 Total CPU times and savings for building the VCI matrix (for all fundamentals) within the calculation depending on the operator included. The potential energy surface has been truncated after 3D or 4D terms, respectively, the VAM operator has been included until 0D terms ($\hat{H}_{\text{VAM(0D)}}$) or has not ($\hat{H}_{\text{w/o VAM}}$)

			tot. CPU [s]	tot. CPU [s]	tot. CPU	Mean CPU saving
Molecule	PES	Operator	(w/o classes)	(w/classes)	Saving (%)	Per state (%)
C_2H_5F	3D	$\hat{H}_{\text{w/o VAM}}$	2237.1	659.1	70.5	46.2
	3D	$\hat{H}_{\text{VAM(0D)}}$	19515.2	17224.0	11.7	4.4
	4D	$\hat{H}_{\text{w/o VAM}}$	6026.2	5376.4	10.8	10.2
	4D	$\hat{H}_{\text{VAM(0D)}}$	27327.8	24198.0	11.5	9.8
B_2H_6	3D	$\hat{H}_{\text{w/o VAM}}$	2028.6	751.0	63.0	46.2
	3D	$\hat{H}_{\text{VAM(0D)}}$	3566.2	2808.2	21.3	14.3
	4D	$\hat{H}_{\text{w/o VAM}}$	1911.2	1710.8	10.5	8.9
	4D	$\hat{H}_{\text{VAM(0D)}}$	3047.0	2822.5	7.4	6.1
C_3H_4	3D	$\hat{H}_{\text{w/o VAM}}$	534.6	191.7	64.1	46.9
	3D	$\hat{H}_{\text{VAM(0D)}}$	1792.8	1580.3	11.9	7.2
	4D	$\hat{H}_{\text{w/o VAM}}$	1471.4	1241.7	15.6	10.0
	4D	$\hat{H}_{\text{VAM(0D)}}$	2592.6	2467.0	4.8	3.5

3.2.1 | Configuration selection

Since the VAM contributions are not considered within Equation (2), the inclusion of the VAM terms show only an indirect effect on the CPU time needed for the configuration selection. Figure 1 clearly illustrates, that merging configurations into classes instead of checking individual configurations leads to a tremendous computational advantage.

Naturally, using a 3D potential energy surface, which includes couplings up to three modes involved, leads to larger savings than employing a 4D PES, because many more integrals will vanish due to the analogs of the Slater-Condon rules. Additionally, driving the configuration selection by classes leads to larger blocks of configurations that can be skipped in advance without evaluating the respective matrix element. Moreover, considering a specific PES, the effect of saving CPU time by using classes grows even larger if the size of (a) the systems and/or (b) the correlation space increases. Both in turn leads to larger blocks that can be neglected and therefore the total CPU time decreases.

We like to emphasize here that the ordering of the configurations in the initial correlation space plays a major role regarding the savings. Expression (4) has to be evaluated for all configurations Φ_J not being included in the current configuration space, which may be a large number (i.e., $N_{\text{conf,tot.}} - N_{\text{conf,sel.}}(a)$). By using the ordering described above, the initial configuration space can be divided into a minimal number of classes, whereas a single class has maximum size. For example, within the calculation of the vibrational ground state of allene, there are 30,926,490 configurations present in the initial configuration space used. These can be divided into 9946 classes, that is, on average a class contains 3108 configurations, while the number of elements per class varies between 6 and 4501. Since many integrals will not contribute within the sum, this will lead to comparatively short lists of contributing blocks. Only if one class contributes, all included configurations will be tested via the VMP2-like criterion, otherwise not. This results in a maximum number of configurations that can be skipped. In summary, driving the configuration selection by classes on average (all three systems investigated, see Table 2) leads to a CPU time saving of about 97% in total for 3D PESs and to about 83 $\%$ for 4D PESs.

3.2.2 | VCI matrix set-up

Regarding the matrix set-up, the largest computational savings can be achieved, once the potential includes up to 3D terms and VAM contributions are neglected within the Watson Hamiltonian, namely 64.1% for allene. This behavior is not surprising, because for a 3-mode operator the number of elements in a single class is much larger than in calculations involving 4-mode operators. Thus, the computational effort for 3-mode operators is much smaller as many more elements can be skipped at once.

The savings in the other three cases (3D potential and 0D VAM terms, 4D potential and no/0D VAM terms) are quite similar to each other, since the total order is four regarding the mode couplings in the underlying operator. The savings scatter between 4.8% and 15.6%. The other systems investigated (see Table 1), show qualitatively the same behavior. On average (all three systems considered), savings are as large as 66% in the first case and about 12% in the three other ones. In any case, the savings are much smaller than for the configuration selection procedure. Essentially, this behavior results from the technical framework within our implementation. The efficiency of our VCI algorithm benefits from a specific ordering of the selected configurations from the previous iteration step, that is, the configuration list is sorted by iterations. In this way, we are able to reuse information and transfer it to the next iteration step in order to avoid recalculation to a great extent. Within the list of already selected configurations, we conserve the original sorting, which renders the most efficient structure regarding classes, only within configurations from the same iteration step. Thus, the list is split into many sections. Since the VCI matrix is generated from already selected configurations, the resulting loop leads to much smaller classes that may be skipped compared to the case of configuration selection. Although the savings are comparatively moderate, for larger systems the benefit from the alternative loop structure (classes vs. no classes) will increase and the savings will grow larger. It is important to notice here, if the VCI implementation is non-iterative and/or the VCI matrix would be generated as a whole from a list ordered in the optimal way, the savings would be significantly larger. In general, the computational savings that can be achieved by using classes increase with a growing number of modes and/or the size of the correlation space.

4 | ELIMINATION OF INTERMEDIATE EIGENPAIR DETERMINATIONS

The computational effort within configuration selective VCI calculations is dominated by the last few iteration steps, when the VCI matrix has already gained a certain size. Consequently, it must be the primary goal to reduce the CPU time in these last steps and thus we will focus in the following on two situations:

- Although the method of prediagonalizing subspaces presented in part 1 of this article series improves the convergence behavior of the iterative VCI implementation by optimizing the start vector for the configuration selection, there are still vibrational states, especially in the regions of high density, showing very slow convergence. For example, for the calculation of the fundamental transitions ν_1 and ν_3 of C_2H_5F (see Table S2 in the supporting information), 24 and 18 iterations within our (former) algorithm are necessary to reach converged energy eigenvalues. Of course, this is an indication for small numerical effects playing a major role for such critical states, since the main physically relevant information is already covered by the first iteration steps. Thus, small uncertainties within the dynamic correlation may sum up during the iteration process leading to a large number of steps.
- Usually, the configuration selection itself dominates the total CPU time of the calculation (see for example Figure 1). In contrast, for computationally very challenging systems as for example HCCNCS, the diagonalizations of the VCI matrix can become the computational bottleneck, although we employ the highly optimized eigenvalue solver based on the RACE algorithm described in Section 2.

TABLE 2 Total CPU times needed for selecting configurations within the iterations via the VMP2-like criterion. In all calculations the Hamiltonian used is $\hat{H}_{\mathrm{VAM(0D)}}$. The values given refer to the sum of CPU times needed in all iterations for the calculation of all fundamental transitions. The savings do not depend on the VAM operator used within the calculation, since it is not considered in Equation (2)

		tot. CPU [s]	tot. CPU [s]	tot. CPU	Mean CPU saving
	PES	(w/o classes)	(w/classes)	Saving (%)	Per state (%)
C_2H_5F	3D	6659572.2	136559.5	98.0	98.1
	4D	4020021.1	652234.4	83.8	84.0
B_2H_6	3D	851357.3	25220.1	97.0	95.7
	4D	263132.0	41849.6	84.1	85.0
C_3H_4	3D	632625.0	22198.1	96.5	97.2
	4D	576226.6	99502.2	82.7	85.0

Both aspects combined may lead to the fact that iterative VCI calculations may slow down unendurably for certain states or even whole systems. In the following, we try to overcome this bottleneck and present a method, which essentially increases the performance of our VCI algorithm for the calculation of states requiring a large number of iterations and/or computationally very demanding diagonalization steps occurring within these.

4.1 | Method and implementation

In order to save CPU time in case of computationally demanding eigenvalue determinations and to further improve the convergence behavior of our algorithm, we modified the criterion for the configuration selection. Note, that the starting point for the following considerations again is Equation (2) used in our original implementation, that is, the prediagonalization of subspaces, presented in part 1 of this series paper, is not considered here. Within second order Rayleigh-Schrödinger perturbation theory (cf. Ref. [23] for example) and the case of small perturbations $|\hat{H}^{'}|/|\hat{H}^{(0)}|=\alpha \ll 1$ described by the operator $\hat{H}^{'}$, while $\Psi_n^{(0)}$ denotes eigenfunctions for the non-perturbed system with the Hamiltonian $\hat{H}^{(0)}$ and $E_n^{(0)}$ the corresponding eigenvalue, the energy correction of second order $E_n^{(2)}$ is given as

$$E_n^{(2)} = \sum_{m \neq n} \frac{|\langle \Psi_m^{(0)} | \hat{H}^{'} | \Psi_n^{(0)} \rangle|^2}{E_n^{(0)} - E_m^{(0)}}, \tag{10}$$

the corresponding wavefunction $\Psi_{1,n}$ is

$$|\Psi_{1,n}\rangle = |\Psi_n^{(0)}\rangle + \sum_{m \neq n} \frac{\langle \Psi_m^{(0)} | \hat{H}^{'} | \Psi_n^{(0)} \rangle}{E_n^{(0)} - E_m^{(0)}} |\Psi_m^{(0)}\rangle. \tag{11}$$

In order to *estimate* the (energy) correction an arbitrary configuration contributes to the total energy of the state $|\Psi_A\rangle$, the basis function $|\Psi_n^{(0)}\rangle$ in Equation (10) is replaced by the wavefunction $|\Psi_A^{(a)}\rangle$ in the iteration step *a* given by Equation (3). This yields the VMP2-like energy expression (2) which is used to decide whether a certain configuration should be included in the correlation space or not. The energy value, cf. Equation (2), itself does not provide a physically meaningful energy correction in the sense of perturbation theory.

In analogy to this, we define a corresponding VMP2-like wavefunction of first order $|\Psi_A^{(a),1}\rangle$ by

$$\begin{aligned} |\Psi_A^{(a+1),1}\rangle &= \sum_{K \in \{a\}} c_{AK}^{(a)} |\Phi_K\rangle + \sum_{J \neq K} \underbrace{\frac{\sum\limits_{K \in \{a\}} c_{AK}^{(a)} \langle \Phi_K | H' | \Phi_J \rangle}{\varepsilon_A^{(a)} - \varepsilon_J}}_{=c_{AK,J}^{(a+1),(1)}} |\Phi_J\rangle \\ &= |\Psi_A^{(a)}\rangle + \sum_{J \neq K} c_{AK,J}^{(a+1),(1)} |\Phi_J\rangle \end{aligned} \tag{12}$$

obtained by one-to-one comparison with the wavefunction (11). We want to emphasize here, that the requirements of perturbation theory formally are not fulfilled for the wavefunction $|\Psi_A^{(a)}\rangle$ (no complete basis set with respect to $\hat{H}^{(0)}$). Thus, neither Equation (2) nor (12) render physically meaningful quantities in the sense of perturbation theory.

The existing configuration selection scheme is now modified as follows:

1. Initially, the configuration selection via criterion (2) is used.
2. If the difference between the energy eigenvalues obtained from two consecutive iteration steps falls below a certain threshold $E_{\text{diff,thres}}$, we replace criterion (2) by

$$\varepsilon_{AJ}^{(a+1),1} = \frac{\left|\left\langle \Psi_A^{(a),1} | H' | \Phi_J \right\rangle\right|^2}{\varepsilon_A^{(a),1} - \varepsilon_J} \tag{13}$$

   with the VMP2-like wavefunction (12) and $\varepsilon_A^{(a),1}$ being the respective energy.
3. We define the correlation space to be converged, if the sum of "energy corrections" (13) is smaller than a certain threshold $E_{\text{corr,thres}}$.
4. Finally, the VCI matrix in the converged configuration space is diagonalized and the state of interest is identified.

It is important to notice, that the criterion (13) is only employed once the major part of physical information is already covered by the current correlation space, that is, static correlation. This is ensured by the threshold regarding the energy difference of two consecutive iteration steps. In the last iterations the main concern is to achieve convergence of the energy eigenvalue by further improving the description of the state of interest by supplementing the correlation space. Consequently, although the VMP2-like wavefunction (12) has not a physical meaning in the sense of perturbation theory, because the respective requirements formally are not fulfilled for the wavefunction $|\Psi_A^{(a)}\rangle$), it can be applied in order to *estimate* the actual VCI wavefunction for determining the correlation space fitting to the state of interest in late iteration steps. As the wavefunction is updated by an VMP2-like procedure, there is no need for any intermediate eigenvalue determinations, which may be costly. The final diagonalization after the very last iteration yields a physical meaningful genuine VCI wavefunction and the energy eigenvalue provided by the selected configuration space.

Note, that the norm of the wavefunction (12) is given as

$$\left\langle \Psi_A^{(a+1),1} | \Psi_A^{(a+1),1} \right\rangle = 1 + \sum_{J \neq K} \left| c_{AK,J'}^{(a+1),(1)} \right|^2, \tag{14}$$

since $\left\langle \Phi_J | \Psi_A^{(a)} \right\rangle = 0$, that is, the wavefunction (12) is formally normalized, because the quadratic terms in $c_{AK,J'}^{(a+1),(1)}$ are of order $\mathcal{O}(\alpha^2)$ with α being the perturbation parameter. Nevertheless, we will renormalize Equation (12) within the VCI iterations. We want to notice here, that the effects of renormalization are usually very small. In order to further reduce the computational cost, one may dispense this step.

As outlined above, many of the coefficients $c_{AK}^{(a)}$ in Equation (3) were found to be very small. For this reason and in order to reduce the computational effort within the configuration selection, in the original algorithm these insignificant coefficients were set to zero. By replacing the genuine VCI wavefunction (3) by the VMP2-like expression (12), this adjustment is not possible any more. Consequently, the computational cost for the configuration selection rises, but is overcompensated by using the proposed method.

4.2 | Results

We benchmarked the method of employing a VMP2-like wavefunction (12) within the configuration selection for all test molecules presented above. In all cases, the PESs have been truncated after the 4-mode coupling terms and 0D VAM contributions have been included within the VCI calculations. In order to restrict the initial configuration space, we used $n_{\mathrm{ex,init.}} = 6$, $n_{\mathrm{max,init.}} = 6$, $n_{\mathrm{sum,init.}} = 15$ for B_2H_6, C_3H_4, and C_2H_5F, for HCCNCS we utilized $n_{\mathrm{ex,init.}} = 5$, $n_{\mathrm{max,init.}} = 5$ and $n_{\mathrm{sum,init.}} = 15$.

CPU timesavings regarding the total computational time required are shown in Table 3. The reference calculation refers to our former VCI implementation using Equation (2) for configuration selection only, the CPU saving refers to the use of the criterion (13) when $E_{\mathrm{diff,thres}} < 2.0$ cm^{-1}. The threshold for convergence has been set to $E_{\mathrm{corr,thres}} = 1.0$ cm^{-1}. In order to obtain the statistical data presented in the table, we calculated the six BH-stretching (CH-stretching) fundamental transitions of B_2H_6 (C_2H_5F), the four CH-stretching fundamental transitions of C_3H_4 as well as its first overtones $2\nu_6$, $2\nu_2$ and $2\nu_7$ and the four highest lying fundamentals of HCCNCS.

As shown in Table 3, on average, the mean absolute energy difference between the results obtained with our former algorithm and the new one employing Equation (13) is 0.3 cm^{-1} and the maximum deviation is 1.1 cm^{-1}. These results show that the final energy eigenvalues obtained by the new method described do not differ significantly from the results generated within the former algorithm. Since deviations of these magnitudes can arise from many error sources within the entire calculation (quality of the potential energy surface, quality of the polynomial fit of the PES, size of the correlation space in VCI, choice of startvector for the diagonalization, thresholds for convergence, $\cdots$) the error must be considered to be small. This behavior had to be expected, because, as mentioned before, the main effect of the late iterations (regarding $E_{\mathrm{diff,thres}}$) is to fine-tune the correlation space whereas all relevant physical information about the state of interest is already covered within the early ones. This refers to the concept of static and dynamic correlation in electronic structure theory. Consequently, the actual energy eigenvalue will not essentially depend on individual configurations, that is, the difference between using Equation (2) or (13) instead will not be very large.

On the other hand, using this technique the total CPU time can be tremendously reduced. The data given in Table 3 show, that using the modified algorithm on average leads to total CPU time savings of 61.9% for the respective states and an average mean CPU time saving of 54.5% per state. Regarding the single systems considered, the mean saving per state varies between 31.5% for B_2H_6 and 73.9% for C_2H_5F, that is, there may be large differences regarding the possible savings, which depend on the behavior of the systems considered within the calculation. In the following, we will discuss different situations exemplified by our benchmarks molecules.

4.2.1 | C_3H_4

A detailed depiction of the computational demands for the vibrational states of allene calculated is shown in Figure 2. The CPU time is given by bars, split up in contributions of the configuration selection (orange), diagonalization of the VCI matrix (red) and the set-up of the VCI matrix (blue). The number of iterations necessary within both of the implementations are given as lines (second y-axis, former algorithm in blue, new one in black). The red numbers provide the final configuration spaces and the black ones the energy difference between the results obtained from both algorithms.

As shown in Figure 2, the replacement of the VCI wavefunction (3) by the VMP2-like wavefunction (12) on average leads to an improvement of the convergence behavior w.r.t. (i) the number of iterations necessary and (ii) the dimension of the final configuration space. On average, 11.0 iterations per state are necessary within our former algorithm to gain convergence of the energy eigenvalue. Our new approach reduces the number of iterations to 7.0 on average; whereas diagonalizations are necessary within 3.9 iterations (see also Tables S1–S4 in the supporting information). Thus, the CPU savings are generated by improving the convergence behavior, because the total CPU time is dominated by the configuration selection while the diagonalization steps amount to a minor contribution. Additionally, the final correlation spaces are usually smaller employing our new

TABLE 3 CPU savings (w.r.t. total computational time) and energy deviations arising from using the VMP2-like vector (12) instead of the VCI eigenvector (3) obtained by diagonalization of the intermediate VCI matrix. The configuration selection has been carried out based on Equation (2) and $E_{\mathrm{diff,thres}} > 2.0$ cm^{-1}. Subsequently, for further augmentation of the correlation space equation (13) has been used ($E_{\mathrm{corr,thres}} = 1.0$ cm^{-1}). For details regarding single states see the supporting information

	B_2H_6	C_2H_5F	C_3H_4	HCCNCS
mean ΔE (cm^{-1}) per state	0.0	0.8	0.2	0.3
max. ΔE (cm^{-1})	0.0	1.1	0.5	0.6
mean CPU saving (%) per state	31.5	73.9	46.6	66.1
tot. CPU saving (%) (all states)	31.1	78.3	55.2	82.8

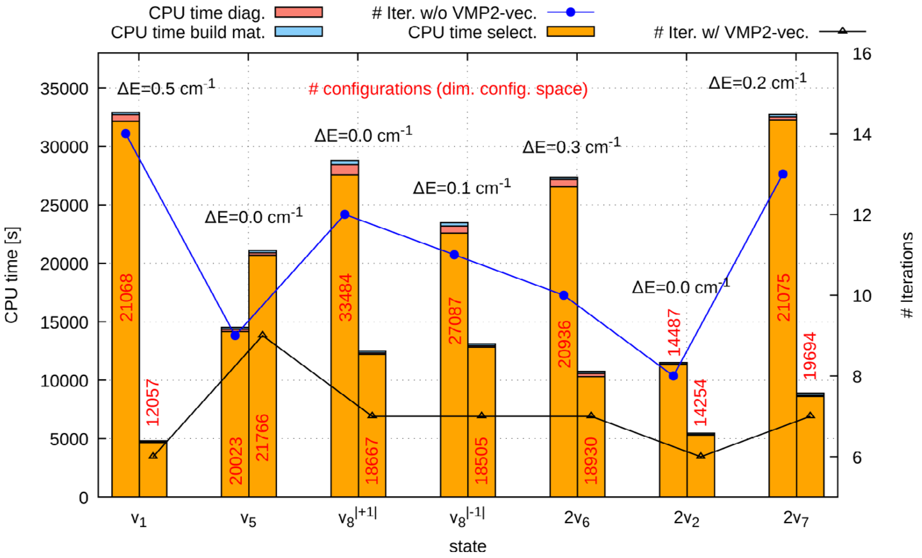

FIGURE 2 CPU times for the calculation of the CH-stretching fundamentals and three overtones of C_3H_4, split up in times for evaluating and diagonalizing the VCI matrix and employing the configuration selection scheme based on Equation (2). For every state shown, the bar on the left hand side depicts the CPU time for a reference calculation using the VCI eigenvector obtained by diagonalization of the VCI matrix in order to apply Equation (2) for selecting configurations. The bar on the right hand side refers to an implementation using a VMP2-like vector of the form (12) instead as long as the thresholds described in the text are reached. The corresponding differences between the energies obtained are given above the bars, the dimensions of the associated correlation spaces are depicted in red. The second y-axis on the right hand side provides the number of iterations required

algorithm. For example, regarding the state ν_1 the configuration space is reduced to almost 50%, whereas the energy difference with respect to the reference value is 0.5 cm^{-1} only.

Obviously, state ν_5 constitutes an exception from this overall tendency. Since the number of iterations stays unchanged in this case and the diagonalization does not dominate the total CPU time, we lose performance by using Equation (13), because negligible configurations cannot be discarded in this formalism. This is in contrast to our former implementation based on Equation (2). All in all, this results in larger CPU times. Note that, none of the other states investigated within our benchmarks suffers from a larger CPU time compared to our former implementation, but show much faster convergence and significant CPU savings can be generated by using the new implementation.

4.2.2 | B_2H_6

Within the calculation of the BH-stretching modes of B_2H_6, on average 7.2 iterations are necessary in order to obtain converged energy eigenvalues using the standard configuration selection algorithm, which is a comparatively small number. The total CPU time is dominated by the configuration selection while the diagonalization steps yield a minor contribution. By using the new algorithm, the number of iterations is reduced to 6.2 on average, while 4.2 iterations contain diagonalizations (see Tables S1–S4 the supporting information). Note that, the iteration steps employing Equation (13) instead suffer for additional cost, since negligible coefficients within the wavefunction (3) may not be eliminated (see above). Therefore, the CPU needed for configuration selection increases, but the effect is overcompensated by the saving of whole (late and thus computationally demanding) iteration steps. In comparison to the other systems investigated, the convergence behavior can be only marginally improved for B_2H_6, which results in a rather small overall CPU time saving of (only) 31.5% per state.

4.2.3 | C_2H_5F

In contrast to B_2H_6, for calculating the CH-stretchings of C_2H_5F, on average 17.3 iterations per state are needed with our former algorithm, while the new requires only 7.7 and 4.5 iterations of these contain VCI matrix diagonalizations (see Tables S1–S4 in the supporting information). These numbers clearly show that the convergence behavior can tremendously be improved by using the configuration selection based on the VMP2-like vector. Additionally, the final configuration spaces for all the states are significantly smaller than before (see Tables S1–S4 in the supporting information). In summary, massive CPU timesavings can be observed, namely 73.9% of the total CPU time per state.

4.2.4 | HCCNCS

Within the conventional algorithm the total CPU time for calculating the states of HCCNCS is dominated by the diagonalization steps

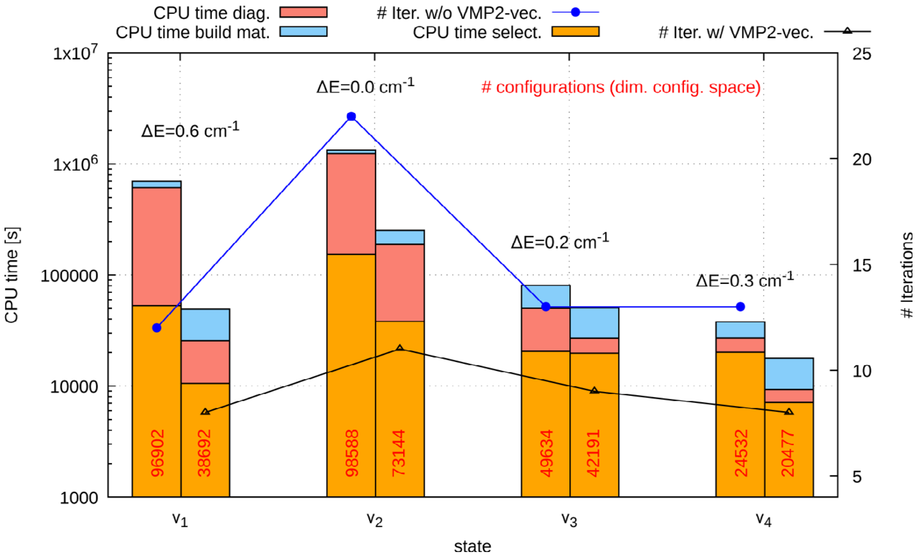

FIGURE 3 CPU times for the calculation of the four highest lying fundamentals of HCCNCS, split up in times for evaluating and diagonalizing the VCI matrix and employing the configuration selection scheme based on Equation (2). Note that, the scaling is logarithmic. For every state shown, the bar on the left hand side depicts the CPU time for a reference calculation using the VCI eigenvector obtained by diagonalization of the VCI matrix in order to apply Equation (2) for selecting configurations. The bar on the right hand side refers to an implementation using a VMP2-like vector of the form (12) instead as long as the thresholds described in the text are reached. The corresponding differences between the energies obtained are given above the bars, the dimensions of the associated correlation spaces are depicted in red. The second y-axis on the right hand side provides the number of iterations required

(see Figure 3, note the logarithmic scale of the first y-axis). Additionally, on average this algorithm needs 15.0 iterations in order to converge. Using the new implementation, the number of iterations can be reduced to 9.0 on average, which constitutes a significant improvement. Besides, there are on average 5.8 iterations containing intermediate diagonalizations of the current VCI matrix, which has a tremendous impact on the overall CPU time. Thus, in the case of HCCNCS, the saving is twofold: On the one hand, the convergence of the energy eigenvalue is increased, on the other hand a large number of computationally demanding eigenvector determinations is omitted. Both aspects, taken together, lead to the tremendous total saving of 82.8% while again the energy differences are negligible.

In summary, our benchmark calculations prove the new algorithm employing a configuration selection based on a VMP2-like wavefunction to be essentially more efficient than using a conventional configuration selection scheme relying on Equation (2). Although our criterion (13) does not include the genuine VCI eigenfunction but an approximation via Equation (12), the resulting energy eigenvalues do not suffer from a loss of accuracy. Regarding CPU time, we partially obtain tremendous savings resulting from an improvement of the convergence behavior (number of iterations, smaller final correlation spaces). On the other hand, the algorithm profits from omitting computationally demanding diagonalizations of the VCI matrix. The loss of performance within the configuration selection caused by additional cost for a larger number of negligible coefficients is clearly overcompensated. Nevertheless, in order to save CPU time using the presented method, the systems/correlation spaces considered must exhibit a certain minimum size, that is, very small systems may be calculated faster by using the conventional method.

5 | OVERALL PERFORMANCE

Here we combine the methods to accelerate iterative VCI calculations based on a configuration selection scheme presented in both parts of this articles series. We modified the following steps within the algorithm: (i) The VAM contributions, wherever necessary, are evaluated by using unrolled equations efficiently using symmetry properties and discarding vanishing integrals in advance. (ii) Prediagonalization of appropriate subspaces is utilized in order to take into account state information beyond the harmonic one right from the beginning of the calculation leading to faster convergence of the energy eigenvalue. (iii) Configuration classes are used to maximally exploit the Slater-Condon type rules within the configuration selection process and the set-up of the VCI matrix. (iv) A modified criterion for configuration selection based on a VMP2-like wavefunction is employed for late iteration steps in order to omit computationally demanding eigenvector determinations and further improve the convergence behavior of the VCI energy.

In order to determine the total improvement with respect to our former implementation obtained by the techniques mentioned, we performed calculations including all aspects. Again, we used B_2H_6, C_3H_4, and C_2H_5F for benchmarking. The potential energy surfaces have been truncated after 4-mode coupling terms; VAM contributions have been included up to 0D terms. The initial configuration spaces has been restricted by $n_{\mathrm{max,init.}} = 6$, $n_{\mathrm{sum,init.}} = 15$ and $n_{\mathrm{ex,init.}} = 6(5)$ for B_2H_6, C_3H_4, (C_2H_5F). In order to define the application of the criterion (13) instead of Equation (2) for configuration selection, we used the thresholds $E_{\mathrm{corr,thres}} = 1.0\ \mathrm{cm}^{-1}$ and $E_{\mathrm{diff,thres}} < 2.0\ \mathrm{cm}^{-1}$. For all systems investigated, the CH (BH)-stretching fundamentals have been evaluated. The respective results can be found in Table 4.

In summary, the mean CPU time saving is 89.8% for B_2H_6, 86.8% for C_3H_4, and 94.7% for C_2H_5F, that is, the performance of the algorithm has been increased tremendously. We will discuss the results exemplified by allene in the following.

In Figure 4, the CPU times and savings for the calculation of the CH-stretching fundamentals of allene are depicted. Note, that the

Molecule	State	Sym.	CPU (old) [s]	CPU (new) [s]	Savings (%)
B_2H_6	ν_1	A_g	28203.1	2406.5	91.5
	ν_2	A_g	46638.3	6157.1	86.8
	ν_5	B_{1g}	24302.8	2338.6	90.4
	ν_{11}	B_{1u}	21107.5	2366.9	88.8
	ν_{13}	B_{2u}	21486.6	1999.8	90.7
	ν_{16}	B_{3u}	16314.8	1583.8	90.3
C_3H_4	ν_1	A_1	116444.1	4890.5	95.8
	ν_5	B_2	63242.9	16243.8	74.3
	$\nu_8^{\lvert+1\rvert}$	E	122646.3	12533.7	89.8
	$\nu_8^{\lvert-1\rvert}$	E	96809.3	12403.3	87.2
C_2H_5F	ν_1	A'	818896.6	53058.9	93.5
	ν_2	A'	2351238.2	63765.3	97.3
	ν_3	A'	1401331.4	18634.0	98.7
	ν_4	A'	1003966.9	100891.5	90.0
	ν_{12}	A"	1801184.2	68445.0	96.2
	ν_{13}	A"	59180.5	4556.9	92.3

TABLE 4 Comparison of CPU times for calculating the BH/CH-stretching modes of B_2H_6, C_3H_4, and C_2H_5F using a conventional configuration-selective VCI algorithm (old) and using all optimizations described in part 1 and part 2 of this article series (new)

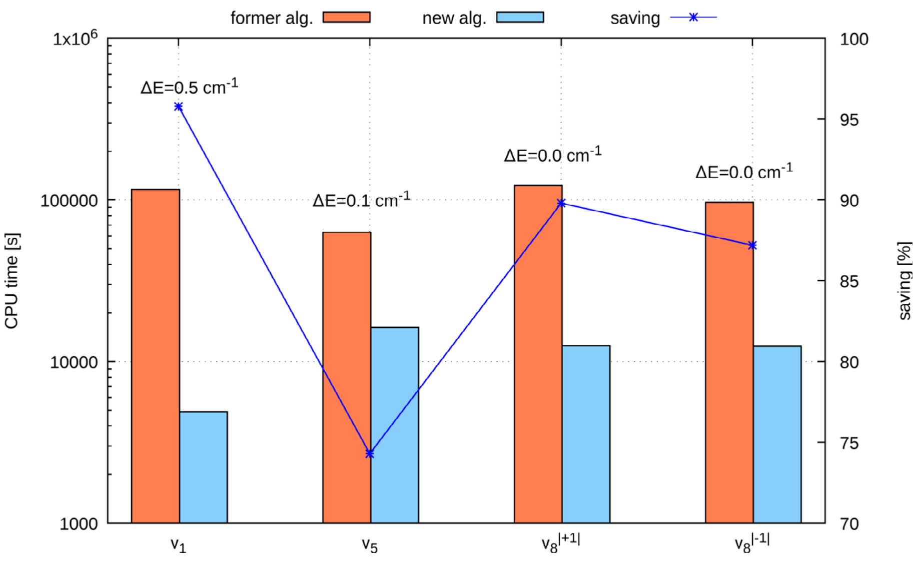

FIGURE 4 Comparison of CPU times for the calculation of the CH-stretching modes of C_3H_4. The implementation labeled "new alg." refers to an implementation using (A) Optimized equations for calculating the VAM contributions, (B) Classes of configurations for rigorously exploiting the slater-Condon rules, (C) Prediagonalization of meaningful subspaces and (D) The technique of replacing diagonalization steps within the configuration selective algorithm by using a VMP2-like wavefunction instead of a VCI eigenvector. The reference CPU times (labeled "former alg.") refer to our former implementation of VCI not using any of the techniques mentioned before. The blue line depicts the computational saving for each state; the black numbers give the respective energy difference between the respective energy eigenvalues. Note that, the scaling regarding the first y-axis is logarithmic

y-axis on the lhs is logarithmic and shows the total CPU time for the calculation of the individual states. It can clearly be seen that our new implementation unifying the optimization techniques presented in both parts of this article series leads to significant CPU timesavings. We explicitly emphasize here, that Figure 4 shows total CPU times and not single steps within the calculation. For allene, the mean CPU time saving is 86.8% with respect to our former implementation. On the other hand, as shown in Figure 4, the results for the energy eigenvalues match with former results, that is, there is almost no loss of accuracy due to the optimizations.

The total CPU time saving is a sum of the following aspects: With respect to our former implementation, we gain a factor of (i) 12.6 within the set-up of the VCI matrix by using optimized equations for the evaluation of the VAM terms (part 1, cf. Ref. [6]), (ii) 1.2 regarding the total CPU time by prediagonalization of appropriate subspaces in order to take information beyond the harmonic one into account from the beginning of the calculation (part 1, cf. Ref. [6]), (iii) 5.8 within the sum of the configuration selection and the matrix set-up by the use of classes of configurations to efficiently exploit the Slater-Condon rules, (iv) 1.9 regarding the total CPU time by using a modified selection scheme for late iteration steps based on a VMP2-like wavefunction further improving the convergence behavior.

Consequently, the optimizations presented in this work lead to a substantial increase of the performance of our VCI algorithm. Since the saving nearly constitutes a whole order of magnitude regarding the necessary computational time, these improvements make larger and more complicated systems than before accessible. Of course, also rovibrational calculations will profit essentially from a faster and more stable VCI implementation.

6 | SUMMARY

Accurate VCI calculations suffer from a significant computational effort increasing with the size of the molecule of interest. Much work has already been devoted to tackle this problem. In this work, we presented four new technical aspects leading to a significant reduction of the computational cost for configuration selective VCI calculations:

i. In part 1 of this article series (cf. Ref. [6]) we provided analytical unrolled equations leading to a fast and efficient evaluation of the vibrational angular momentum terms. The scaling of the computational effort with the number of modes has been reduced by at least one order, some expressions are even independent of the size of the system. Thus, the computational effort has been reduced essentially. Roughly, for the evaluation of zeroth order terms, we gain a factor 10 within the set-up of the VCI matrix with regard to our former, but already optimized implementation.
ii. Also in part 1 (cf. Ref. [6]) we presented an improvement of the convergence behavior within the iteration by defining appropriate subspaces of configurations and prediagonalize them. Combined with our state picking scheme, the obtained eigenvector is used to take into account resonance information from the very first iteration step, which improves the behavior of the algorithm regarding convergence. Approximately, this modification generates a saving of 20 % of the total CPU time.
iii. Within this article here, we introduced so-called classes of configurations in order to maximally exploit the Slater-Condon type rules in a more efficient manner. In particular, the configuration selection can be speeded up substantially, because the number of matrix elements to be evaluated during the iterations is significantly reduced. This leads to a considerably increased efficiency of the algorithm, that is, we roughly gain a factor of 6 within the configuration selection process (which dominates the total CPU time in most cases).
iv. We modified our former configuration selection scheme utilizing the genuine VCI eigenvector by employing a mixed VCI/VMP2-like wavefunction instead. It has been shown that the convergence behavior of the energy eigenvalue of the state of interest is significantly improved by doing so. Additionally, computationally demanding eigenvector determinations can be omitted this way. The overall saving is roughly a factor of 2 regarding total CPU time.

Extensive benchmark calculations have been presented for the time consuming BH/CH-stretching states demonstrating the effects of the single techniques and proving the modifications to significantly increase the performance of the algorithm. It has been shown, that the combination of all four methods results in CPU time saving of about one order of magnitude with respect to the total CPU time. Consequently, the concepts presented allow to apply cs-VCI theory to even larger systems in the future.

ACKNOWLEDGMENTS

Financial support by the Deutsche Forschungsgemeinschaft (project Ra 656/25-1) and the Studienstiftung des Deutschen Volkes is kindly acknowledged. This research was supported in part by the bwHPC initiative and the bwHPC-C5 project provided through associated compute services of the JUSTUS HPC facility at the University of Ulm. bwHPC and bwHPC-C5 are funded by the Ministry of Science, Research and the Arts Baden-Württemberg (MWK).

Open access funding enabled and organized by Projekt DEAL.

DATA AVAILABILITY STATEMENT

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

ORCID

Guntram Rauhut https://orcid.org/0000-0003-0623-3254

REFERENCES

[1] S. Carter, S. J. Culik, J. M. Bowman, *J. Chem. Phys.* **1997**, *107*(24), 10458.

[2] A. R. Sharma, B. J. Braams, S. Carter, B. C. Shepler, J. M. Bowman, *J. Chem. Phys.* **2009**, *130*(17), 174301.

[3] K. Yagi, C. Oyanagi, T. Taketsugu, K. Hirao, *J. Chem. Phys.* **2003**, *118*(4), 1653.

[4] J. M. Bowman, T. Carrington, H. D. Meyer, *Mol. Phys.* **2008**, *106*(16–18), 2145.

[5] C. König, *Int. J. Quantum.* **2021**, *121*(3), e26375.

[6] T. Mathea, G. Rauhut, *J. Comput. Chem.* **2021**. https://doi.org/10.1002/JCC.26762.

[7] B. Ziegler, G. Rauhut, *J. Phys. Chem. A* **2019**, *123*(15), 3367.

[8] D. F. Dinu, B. Ziegler, M. Podewitz, K. R. Liedl, T. Loerting, H. Grothe, G. Rauhut, *J. Mol. Spectrosc.* **2020**, *367*, 111224.

[9] T. Mathea, T. Petrenko, G. Rauhut, *J. Phys. Chem. A* **2021**, *125*(19), 194112.

[10] J. H. Fetherolf, T. C. Berkelbach, *J. Chem. Phys.* **2021**, *154*(7), 074104.

[11] V. Le Bris, M. Odunlami, D. Bégué, I. Baraille, O. Coulaud, *Phys. Chem. Chem. Phys.* **2020**, *22*, 7021.

[12] D. Bégué, N. Gohaud, C. Pouchan, P. Cassam-Chenaï, J. Liévin, *J. Chem. Phys.* **2007**, *127*(16), 164115.

[13] Y. Scribano, D. M. Benoit, *Chem. Phys. Lett.* **2008**, *458*(4), 384.

[14] P. Carbonniere, A. Dargelos, C. Pouchan, *Theor. Chem. Acc.* **2010**, *125*(3–6), 543.

[15] M. Neff, G. Rauhut, *J. Chem. Phys.* **2009**, *131*(12), 124129.

[16] M. Neff, T. Hrenar, D. Oschetzki, G. Rauhut, *J. Chem. Phys.* **2011**, *134*(6), 064105.

[17] T. Mathea, G. Rauhut, *J. Chem. Phys.* **2020**, *152*(19), 194112.

[18] J. K. Watson, *Mol. Phys.* **1968**, *15*(5), 479.

[19] G. Rauhut, *J. Chem. Phys.* **2007**, *127*(18), 184109.

[20] E. Mátyus, C. Fábri, T. Szidarovszky, G. Czakó, W. D. Allen, A. G. Császár, *J. Chem. Phys.* **2010**, *133*(3), 034113.

[21] T. Petrenko, G. Rauhut, *J. Chem. Phys.* **2017**, *146*(12), 124101.

[22] O. Christiansen, *J. Chem. Phys.* **2004**, *120*(5), 2149.

[23] Landau L, Lifshitz E. Quantum Mechanics: *Non-Relativistic Theory*. Course of theoretical physics,Elsevier Science. **1991**.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher's website.

How to cite this article: T. Mathea, T. Petrenko, G. Rauhut, *J. Comput. Chem.* **2022**, *43*(1), 6. https://doi.org/10.1002/jcc.26764