Stochastically accelerated perturbative triples correction in coupled cluster calculations

Yann Damour, Alejandro Gallo, Anthony Scemama

Stochastically accelerated perturbative triples correction in coupled cluster calculations

Yann Damour [1], Alejandro Gallo [2] and Anthony Scemama [1, a)]

[1)] *Laboratoire de Chimie et Physique Quantiques (UMR 5626), Université de Toulouse, CNRS, UPS, France*

[2)] *Institute for Theoretical Physics, TU Wien, Wiedner Hauptstraße 8-10/136, 1040 Vienna, Austria*

We introduce a novel algorithm that leverages stochastic sampling techniques to compute the perturbative triples correction in the coupled-cluster (CC) framework. By combining elements of randomness and determinism, our algorithm achieves a favorable balance between accuracy and computational cost. The main advantage of this algorithm is that it allows for the calculation to be stopped at any time, providing an unbiased estimate, with a statistical error that goes to zero as the exact calculation is approached. We provide evidence that our semi-stochastic algorithm achieves substantial computational savings compared to traditional deterministic methods. Specifically, we demonstrate that a precision of 0.5 millihartree can be attained with only 10% of the computational effort required by the full calculation. This work opens up new avenues for efficient and accurate computations, enabling investigations of complex molecular systems that were previously computationally prohibitive.

I. INTRODUCTION

Coupled cluster (CC) theory is an accurate quantum mechanical approach widely used in computational chemistry to describe the electronic structure of atoms and molecules.[1–3] In recent years, CC theories for both ground state and excited states have received considerable attention in the context of material science due to its good balance between accuracy and computational cost.[4,5] CC offers a systematic and rigorous framework for accurate predictions of molecular properties and reactions by accounting for electron correlation effects beyond the mean-field approximation. The CC framework starts with a parameterized wave function, typically referred to as the CC wave function, which is expressed as an exponential series of particle-hole excitation operators acting on a reference state:

$$|\Psi_{\mathrm{CC}}\rangle = e^{\hat{T}} |\Phi\rangle \tag{1}$$

where $|\Phi\rangle$ is the reference determinant, and $\hat{T}$ is the cluster operator representing single, double, triple, and higher excitations on top of the reference wave function.[6–8]

Coupled Cluster with Singles and Doubles (CCSD) includes single and double particle-hole excitations and represents the most commonly used variant of CC theory. CCSD is exact for two-electron systems and includes all terms from third order perturbation theory and beyond. Coupled Cluster with Singles, Doubles, and perturbative Triples (CCSD(T)) incorporates a perturbative correction to the CCSD energy to account for some higher-order correlation effects, and has been termed in the literature as the gold standard of quantum chemistry.[9] CCSD(T) has demonstrated exceptional accuracy and reliability, making it one of the preferred choices for benchmark calculations and highly accurate predictions. It has found successful applications in a diverse range of areas, including spectroscopy,[10–12] reaction kinetics,[13,14] and materials design,[15] and has played a pivotal role in advancing our understanding of complex chemical phenomena.

In the context of CC theory, the perturbative triples correction represents an important contribution to the accuracy of electronic structure calculations.[16] However, the computational cost associated with the calculation of this correction can be prohibitively high, especially for large systems. The inclusion of the perturbative triples in the CCSD(T) method leads to a computational scaling of $\mathcal{O}(N^7)$, where N is proportional to the number of molecular orbitals. This scaling can rapidly become impractical, posing significant challenges in terms of computational resources and time requirements.[17–28]

To address this computational bottleneck, our goal is to develop a novel semi-stochastic algorithm that brings back the computational time to a level smaller or comparable to that of the CCSD method, which has a scaling of $\mathcal{O}(N^6)$, while ensuring well-controlled approximations. Our algorithm strikes a balance between computational efficiency and accuracy, making calculations for larger basis sets more feasible without compromising precision. By incorporating stochastic sampling techniques, our approach provides an alternative avenue for approximating perturbative triples, relieving the computational burden inherent in traditional deterministic methods. This not only reduces the computational time to a more favorable level but also preserves the parallelism capabilities of CC calculations, ensuring efficient utilization of computational resources.

In the following sections, we will provide a brief introduction to the computation of perturbative triples in coupled cluster theory. We will explain the principles of our semi-stochastic algorithm, outlining its key features and advantages. Additionally, we will present implementation details, discussing the technical aspects and considerations involved in the algorithm's practical realization. To demonstrate the effectiveness and applicability of our approach, we finally present illustrative examples

[a)] Electronic mail: scemama@irsamc.ups-tlse.fr

that showcase the algorithm's performance and compare it with the conventional algorithm.

II. THEORETICAL BACKGROUND

The perturbative triples correction,

$$E_{(T)} = \sum_{ijk\,abc} E_{ijk}^{abc}, \tag{2}$$

is a sum of $N_\mathrm{o}^3 \times N_\mathrm{v}^3$ terms, where N_o^3 and N_v^3 denote the number of occupied and virtual molecular orbitals, respectively. For a closed-shell reference with canonical orbitals, each individual term is expressed as[29]

$$E_{ijk}^{abc} = \frac{1}{3}\frac{(4W_{ijk}^{abc} + W_{ijk}^{bca} + W_{ijk}^{cab})(V_{ijk}^{abc} - V_{ijk}^{cba})}{\epsilon_i + \epsilon_j + \epsilon_k - \epsilon_a - \epsilon_b - \epsilon_c}, \tag{3}$$

and depends on the canonical orbital energies ϵ, and on the tensors W and V:

$$W_{ijk}^{abc} = \Pi_{ijk}^{abc}\left(\sum_{d}^{\mathrm{virt}} (bd|ai)t_{kj}^{cd} - \sum_{l}^{\mathrm{occ}} (ck|jl)t_{ab}^{il}\right) \tag{4}$$

$$V_{ijk}^{abc} = W_{ijk}^{abc} + (bj|ck)t_i^a + (ai|ck)t_j^b + (ai|bj)t_k^c \tag{5}$$

where Π_{ijk}^{abc} is a sign-less permutation operator, (t_i^a, t_{ij}^{ab}) are the CCSD amplitudes, $(pq|rs)$ are the two-electron coulomb integrals, and the indices i, j, k, l and a, b, c, d refer to occupied and virtual orbitals, respectively.

The bottleneck of the perturbative triples correction is the computation of the W tensor which requires $\mathcal{O}\big(N_o^3 \times N_v^4\big)$ operations. Fortunately, most of the operations involved in the computation of W can be recast into matrix multiplications,[30] which are among the most efficient operations than can be executed on modern CPUs and accelerators.[31–34]

In the algorithm proposed by Rendell[29], for each given triplet (a, b, c), the sub-tensors W^{abc} and V^{abc} are computed and immediately utilized to calculate their contribution to E^{abc}. Here, we propose a similar approach but introduce a semi-stochastic algorithm to randomly select the triplets (a, b, c), circumventing the need to compute all contributions.

III. SEMI-STOCHASTIC ALGORITHM

A. Stochastic formulation

We propose an algorithm influenced by the semi-stochastic approach introduced in Ref. 35, originally developed for computing the Epstein-Nesbet second-order perturbation correction to the energy.

The perturbative triples correction is expressed as a sum of corrections, each indexed solely by virtual orbitals:

$$E_{(T)} = \sum_{abc} E^{abc}, \text{ where } E^{abc} = \sum_{ijk} E_{ijk}^{abc}. \tag{6}$$

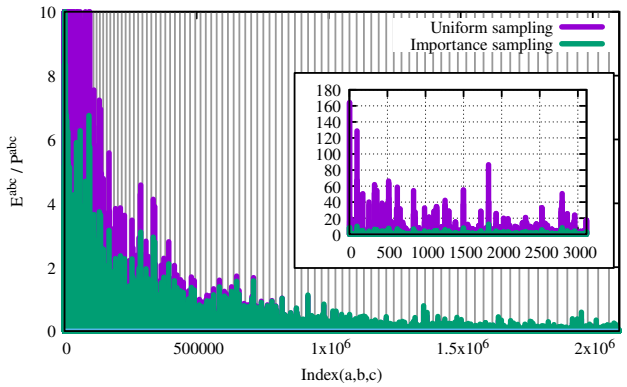

FIG. 1. Ratios $\frac{E^{abc}}{P^{abc}}$ obtained with the data of benzene/cc-pVTZ, using uniform or importance sampling. Every bucket, delimited by vertical bars, contains a number of triplets such that the sum $\sum_{(a,b,c)} P^{abc}$ remains as uniform as possible. The zoomed window corresponds to the first bucket. The fluctuations originating from the discrepancy of the values in the first buckets are considerably reduced by importance sampling.

Monte Carlo sampling is employed by selecting samples E^{abc}. The principal advantage of this formulation is that the number of triplet combinations (a, b, c), given by N_v^3, is sufficiently small to allow for all contributions E^{abc} to be stored in memory. The first time a triplet (a, b, c) is drawn, its corresponding value E^{abc} is computed and then stored. Subsequent drawings of the same triplet retrieve the value from memory. We refer to this technique as *memoization*. Thus, the computational expense of calculating the sample, which scales as $N_\mathrm{o}^3 \times N_\mathrm{v}$, is incurred only once, with all subsequent accesses being made at no cost. Consequently, employing a sufficient number of Monte Carlo samples to ensure that each contribution is selected at least once results in a total computational cost that is only negligibly higher than that of an exact computation.

To reduce the fluctuations of the statistical estimator, we apply importance sampling: the samples are drawn using the probability

$$P^{abc} = \frac{1}{\mathcal{N}} \frac{1}{\max\left(\epsilon_{\min}, \epsilon_a + \epsilon_b + \epsilon_c\right)} \tag{7}$$

where $\mathcal{N}$ normalizes the sum such that $\sum_{abc} P^{abc} = 1$, and $\epsilon_{\min}$ is an arbitrary minimal denominator to ensure that P^{abc} does not diverge. In our calculations, we have set $\epsilon_{\min}$ to 0.2 a.u. The perturbative contribution is then evaluated as an average over M samples

$$E_{(T)} = \left\langle \frac{E^{abc}}{P^{abc}} \right\rangle_{P^{abc}} = \lim_{M\to\infty} \sum_{abc} \frac{n^{abc}}{M} \frac{E^{abc}}{P^{abc}}. \tag{8}$$

where n^{abc} is the number of times the triplet (a, b, c) was drawn with probability P^{abc}.

This approach effectively reduces the statistical error bars by approximately a factor of two for the same computational expense due to two primary reasons: i) the

estimator exhibits reduced fluctuations, ii) triplet combinations with low-energy orbitals are significantly more likely to be selected than others, enhancing the efficiency of memoization (see Fig. 1).

We employ the inverse transform sampling technique to select samples, where an array of pairs $\left(P^{abc},(a,b,c)\right)$ is stored. To further reduce the variance of the samples, this array is sorted in descending order based on P^{abc} and subsequently partitioned into buckets as can be seen diagrammatically in Figure 1. The partitioning into buckets is designed such that the sum $\sum_{(a,b,c)\in B} P^{abc}$ within each bucket B is as uniform as possible across all buckets. As each bucket is equally probable, samples are defined as combinations of triplets, with one triplet drawn from each bucket. Should the values of E^{abc} be skewed, this advanced refinement significantly diminishes the variance.

The total perturbative contribution is computed as the aggregate of contributions from various buckets:

$$E_{(T)} = \sum_B E_B = \sum_B \sum_{(a,b,c)\in B} E^{abc}. \tag{9}$$

Once every triplet within a bucket B has been drawn at least once, the contribution E_B can be determined. At this juncture, there is no longer a necessity to evaluate E_B stochastically, and the buckets can be categorized into deterministic ($\mathcal{D}$) and stochastic ($\mathcal{S}$) groups:

$$E_{(T)} = \sum_{B\in\mathcal{D}} E_B + \frac{1}{|\mathcal{S}|}\sum_{B\in\mathcal{S}} \left\langle \frac{E^B_{abc}}{P^{abc}} \right\rangle_{P^{abc}}. \tag{10}$$

Not all buckets are of equal size (see Figure 1); the number of triplets per bucket increases with the bucket's index. Consequently, the initial buckets transition into the deterministic set first, gradually reducing the stochastic contribution. When every triplet has been drawn, the exact value of $E_{(T)}$ is obtained, devoid of statistical error. To accelerate the completion of the buckets, each Monte Carlo iteration triggers concurrently the computation of the first non-computed triplet. This ensures that after N drawings, the exact contribution from each bucket can be obtained.

The computational time required to generate a random number is negligible compared to the time needed to compute a contribution, E^{abc}. Therefore, it is possible to obtain the exact contribution, characterized by zero statistical error, within a time frame equivalent to that required by a standard deterministic algorithm. This proposed algorithm offers the additional advantage of allowing the calculation to be terminated at any point prior to completion, with a statistical error.

B. Implementation Details

$i_{\min} \leftarrow 1$; $N^{abc}[1,\ldots,N_{\text{triplets}}] \leftarrow [-1,-1,\ldots]$;
$t_0 \leftarrow$ WallClockTime() ;
for $i_{iter} = 1,\ldots,N_{triplets}$ **do**
  /* Deterministic computation */
  while $N^{abc}[i_{min}] > -1$ *and* $i_{min} \le N_{triplets}$ **do**
    $i_{\min} \leftarrow i_{\min} + 1$;
  end
  if $i_{min} \le N_{triplets}$ **then**
    Send OpenMP task {
     $E[i_{\min}] \leftarrow$ Compute($i_{\min}$);
    } ;
  end
  /* Stochastic computation */
  $\eta \leftarrow$ RandomNumber() ;
  for $i_{bucket} = 1,\ldots,N_{buckets}$ **do**
    if $i_{min} \le Last(i_{bucket})$ **then**
      $i_\eta \leftarrow \text{Search}(w_{\text{accu}}, \frac{\eta + i_{\text{bucket}} - 1}{N_{\text{buckets}}}) + 1$;
      if $N^{abc}[i_\eta] = -1$ **then**
        $N^{abc}[i_\eta] \leftarrow 0$;
        Send OpenMP task {
         $E[i_\eta] \leftarrow$ Compute(i_η);
        } ;
      end
      $N^{abc}[i_\eta] \leftarrow N^{abc}[i_\eta] + 1$;
    end
  end
  /* Compute the mean and error every second */
  $t_1 \leftarrow$ WallClockTime() ;
  if $t_1 - t_0 > 1$ *or* $i_{min} \ge N_{triplets}$ **then**
    $i_{\text{bucket}} = 0$;
    while $i_{bucket} < N_{buckets}$ *and* $i_{min} > Last(i_{bucket}+1)$ **do**
      $i_{\text{bucket}} \leftarrow i_{\text{bucket}} + 1$;
    end
    $\mathcal{N} \leftarrow \frac{\sum_{i=\text{First}(i_{\text{bucket}}+1)}^{N_{\text{triplets}}} \max(N^{abc}[i],0)}{1-\sum_{i=1}^{\text{Last}(i_{\text{bucket}})} P[i]}$;
    $E_d \leftarrow \sum_{i=1}^{\text{Last}(i_{\text{bucket}})} E^{abc}[i]$;
    $E_s \leftarrow \frac{1}{\mathcal{N}} \sum_{i=\text{First}(i_{\text{bucket}}+1)}^{N_{\text{triplets}}} \max(N^{abc}[i],0)\, E^{abc}[i]/P[i]$;
    $E_{s^2} \leftarrow \frac{1}{\mathcal{N}} \sum_{i=\text{First}(i_{\text{bucket}}+1)}^{N_{\text{triplets}}} \max(N^{abc}[i],0)\, \left(E^{abc}[i]/P[i]\right)^2$;
    $E \leftarrow E_d + E_s$;
    $\Delta E \leftarrow \sqrt{\left(E_{s^2} - {E_s}^2\right)/(\mathcal{N}-1)}$;
    if $\Delta E < \epsilon$ **then**
      Exit outermost loop ;
    end
  end
end

Algorithm 1: Pseudo-code for the computation of the perturbative triples correction implemented in Quantum Package. $i_{\min}$ denotes the first non-computed triplet, w_{accu} contains the cumulative probability density, Search(A,x) searches for x in array A, First(i) and Last(i) return the first last indices belonging to bucket i.

The algorithm presented in Algorithm 1 was implemented in the QUANTUM PACKAGE software.[36] The stochastic algorithm is implemented using OpenMP tasks, where each task consists in the computation of a single component E^{abc}. The computation of the running average and statistical error is executed every second, for printing or for exiting when the statistical error gets below a given threshold.

The number of samples N^{abc} of each triplet (a, b, c) is initialized to -1, to identify the contributions that have not been already computed. An outer *for loop* runs over the maximum number of iterations, equal by construction to the number of different triplets N_{triplets}.

Within a loop iteration, the index of the first non-computed triplet (a, b, c) is identified, and the task associated with its computation is sent to the task queue. As this triplet has never been drawn, N^{abc} is set to zero. Then, a triplet (a, b, c) is drawn randomly. If the E^{abc} has not been computed (identified by $N^{abc} = -1$), the number of samples is set to zero and the task for the computation of this contribution is enqueued. In any case, N^{abc} is then incremented.

C. Convergence of the statistical error in benzene

In this section we illustrate the convergence of the statistical error of the perturbative triples correction as a function of the computational cost. The benzene molecule serves as our reference system for conducting frozen-core CCSD(T) calculations employing the cc-pVTZ and cc-pVQZ basis sets. Essentially, this involves the correlation of 30 ($N_o = 15$) electrons using either 258 ($N_v = 243$) or 503 ($N_v = 488$) molecular orbitals. The calculations were performed on an AMD EPYC 7513 dual socket server (64 cores in total).

Figure 2 shows the convergence of the CCSD(T) energy as a function of the program execution time using the two basis sets. Notably, the exact CCSD(T) energy always falls within 2σ, affirming the reliability of the statistical error. Figure 3 displays the statistical error as a function of the percentage of computed contributions. Noteworthy in the figure are the curve discontinuities, attributable to readjustments in the separation between the deterministic and stochastic components of the calculation (Eq. (10)). These updates lead to revised estimates and a diminution in statistical error.

Achieving chemical accuracy, defined as $1.6\,\text{m}E_\text{h}$,[37] necessitates less than 1% of the total contributions in both basis sets. Attaining a $0.1\,\text{m}E_\text{h}$ precision level requires computation of 32% and 15% of the contributions for cc-pVTZ and cc-pVQZ, respectively. The more rapid convergence observed with the larger basis set aligns with expectations, as expanding the basis set tends to increase the proportion of minor contributions while maintaining a relatively steady count of significant contributions. This trend underscores the algorithm's enhanced suitability for systems with fewer electrons and extensive basis sets, as opposed to larger electron counts in smaller basis sets.

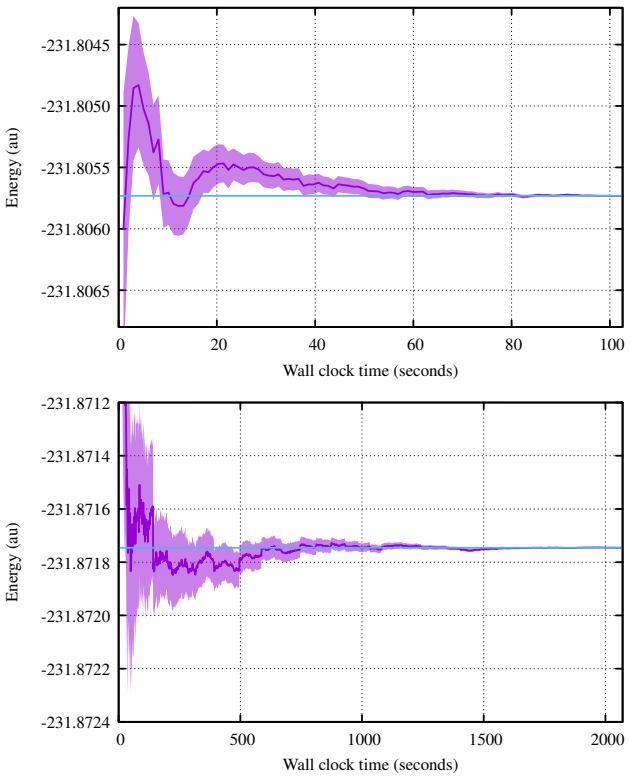

FIG. 2. Energy convergence of benzene plotted against the program execution time, showing comparisons between the cc-pVTZ (upper curve) and cc-pVQZ (lower curve) basis sets. The blue lines indicate the exact CCSD(T) energies.

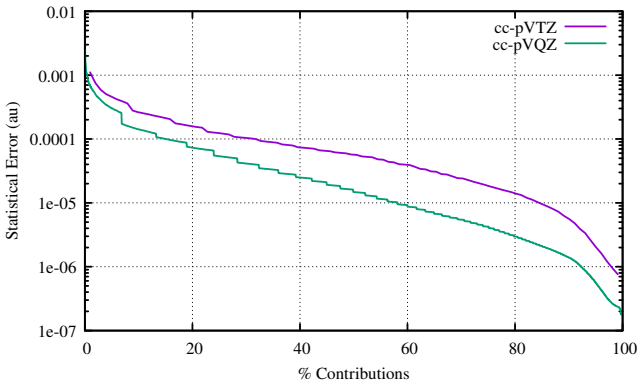

FIG. 3. Convergence of the statistical error of the perturbative triples contribution in benzene as a function of the percentage of computed contributions, for both cc-pVTZ and cc-pVQZ basis sets.

D. Vibrational frequency of copper chloride

Our methodology proves especially advantageous for scenarios requiring the aggregation of numerous CCSD(T) energies, such as neural network training or the exploration of potential energy surfaces. In a recent article, Ceperley *et al* highlight the pivotal role of Quantum Monte Carlo (QMC) in generating data for constructing potential energy surfaces.[38] The study suggests that stochastic noise inherent in QMC can facilitate machine learning model training, demonstrating that models can benefit from numerous, less precise data points. These findings are supported by an analysis of machine learning models, where noise not only helped improve model accuracy but also enabled error estimation in model predictions. Similarly to QMC, our semi-stochastic formulation could take advantage of many points computed with a low accuracy.

In this section, we discuss the application of our novel algorithm within the context of computing vibrational frequencies, specifically through the example of copper chloride (CuCl). A demonstrative application presented here involves the determination of the equilibrium bond length and the computation of the vibrational frequency of CuCl using the CCSD(T)/cc-pVQZ level of theory. The procedure involves determining the CCSD(T) potential energy curve for CuCl, followed by its analytical representation through a Morse potential fitting:

$$E(r) = D_e \left(1 - e^{-a(r-r_e)}\right)^2 + E_0 \tag{11}$$

where $E(r)$ represents the energy at a bond length r, D_e the depth of the potential well, r_e the equilibrium bond length, a the parameter defining the potential well's width, and E_0 the energy at the equilibrium bond length. The vibrational frequency, ν, is derived as follows:

$$\nu = \frac{1}{2\pi c}\sqrt{\frac{2D_e a^2}{\mu}} \tag{12}$$

with μ denoting the reduced mass of the CuCl molecule, and c the speed of light.

The initial step involved the precise calculation of the CCSD(T) energy across various points along the potential curve. We froze the six lowest molecular orbitals, specifically the $1s$ orbital of Cl and the $1s$, $2s$, and $2p$ orbitals of Cu, and correlated 34 electrons within 157 molecular orbitals. The fitted Morse potential revealed a vibrational frequency of $\nu = 414.7\,\mathrm{cm}^{-1}$ and an equilibrium bond length of $r_e = 3.92\,a_0$, aligning remarkably well with experimental values from the NIST database[39] $\nu = 417.6\,\mathrm{cm}^{-1}$ and $r_e = 3.88\,a_0$.

Subsequently, we applied our semi-stochastic algorithm to estimate the perturbative triples correction, utilizing merely 1% of the total contributions. This approach yielded a hundredfold acceleration in computational efficiency, achieving statistical uncertainty within the range of 1.2 to 2.0 mE_h for each data point. The vibrational frequency and equilibrium distance estimated using this data, $\nu = 415.1\,\mathrm{cm}^{-1}$ and $r_e = 3.91\,a_0$, demonstrated comparable precision to the full computational results. Figure 4 illustrates the potential energy surface of CuCl, displaying both the exact CCSD(T) energies and those estimated via the semi-stochastic method.

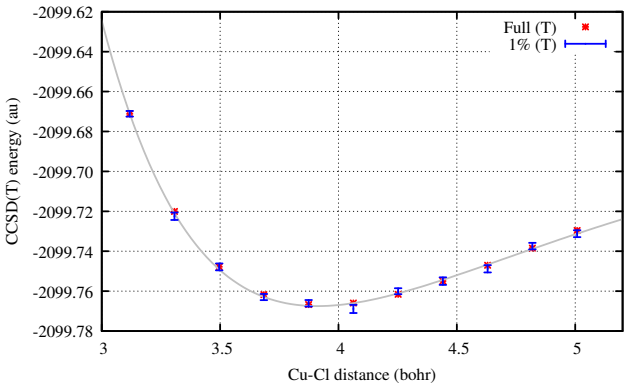

FIG. 4. CCSD(T) energies of CuCl obtained with the exact CCSD(T) algorithm (stars), the stochastic algorithm using only 1% of the contributions (error bars), and the Morse potential fitting the points obtained with the stochastic algorithm.

E. Performance analysis

The primary bottleneck of our proposed algorithm lies in the generation of the sub-tensor W^{abc} for each (a, b, c) triplet, as discussed in Section II. However, we have outlined a strategy to reframe this operation into BLAS matrix multiplications,[30] offering the potential for significantly enhanced efficiency.

We evaluated the efficiency of our implementation using the Likwid[40] performance analysis tool on two distinct x86 platforms: an AMD Epyc 7513 dual-socket server equipped with 64 cores at 2.6 GHz, and an Intel Xeon Gold 6130 dual-socket server with 32 cores at 2.1 GHz. We linked our code with the Intel MKL library for BLAS operations. Additionally, we executed the code on an ARM Q80 server featuring 80 cores at 2.8 GHz, and although performance counters were unavailable, we approximated the Flop/s rate by comparing the total execution time with that measured on the AMD CPU. On the ARM architecture, we utilized the ArmPL library for BLAS operations.

Table I summarizes the performance tests. Peak performance is determined by calculating the maximum achievable Flops/s on the CPU using the formula:

$$P = N_\mathrm{cores} \times N_\mathrm{FMA} \times 2 \times V \times F \tag{13}$$

where F represents the processor frequency, V the number of double precision elements in a vector register,

CPU	N_{cores}	V	F (GHz)	Memory Bandwidth (GB/s)	Peak DP (GFlop/s)	Measured performance (GFlop/s)
EPYC 7513	64	4	2.6	409.6	2 662	1 576
Xeon Gold 6130	32	8	2.1	256.0	2 150	667
ARM Q80	80	2	2.8	204.8	1 792	547

TABLE I. Average performance of the code measured as the number of double precision (DP) floating-point operations per second (Flop/s) on different machines.

N_{FMA} denotes the number of vector fused multiply-accumulate (FMA) units per core (all considered CPUs possess two), and N_{cores} reflects the number of cores. Notably, the Xeon and ARM CPUs both operate at approximately 30% of peak performance, while the AMD Epyc CPU demonstrates twice the efficiency, achieving 60% of the peak.

The relatively modest performance, at around 30% efficiency, is attributed to the small dimensions of the matrices involved. The largest matrix multiplications in the computational task entail a matrix of size ${N_{\text{o}}}^2 \times N_{\text{v}}$ and another of size $N_{\text{v}} \times N_{\text{o}}$ to yield an ${N_{\text{o}}}^2 \times N_{\text{o}}$ matrix. These multiplications exhibit an arithmetic intensity of

$$I = \frac{2\,{N_{\text{o}}}^3\,N_{\text{v}}}{8\left({N_{\text{o}}}^3 + {N_{\text{o}}}^2 N_{\text{v}} + N_{\text{o}} N_{\text{v}}\right)} \tag{14}$$

which can be approximated by $N_{\text{o}}/4$ flops/byte as an upper bound, which is usually relatively low. For instance, in the case of benzene with a triple-zeta basis set, the arithmetic intensity is calculated to be 3.33 flops/byte, falling short of the threshold required to attain peak performance on any of the CPUs. By leveraging memory bandwidth and double precision throughput peak, we determined the critical arithmetic intensity necessary to achieve peak performance. On the Xeon and ARM CPUs, this critical value stands at approximately 8.4 and 8.8 flops/byte, respectively. Meanwhile, the EPYC CPU exhibits a value of 6.5 flops/byte, thanks to its superior memory bandwidth.

F. Parallel efficiency

The parallel speedup performance of the ARM and AMD servers for computations involving the benzene molecule in a triple-zeta basis set is illustrated in Figure 5. The results delineate three distinct performance regimes:

- In the first regime, encompassing up to 24 cores, the performance closely approximates the ideal, with nearly linear speedup.
- The second regime, spanning 24 to 64 cores, shows decent performance, achieving a 40-fold acceleration with 64 cores.
- The third regime begins beyond 64 cores, where parallel efficiency rapidly deteriorates.

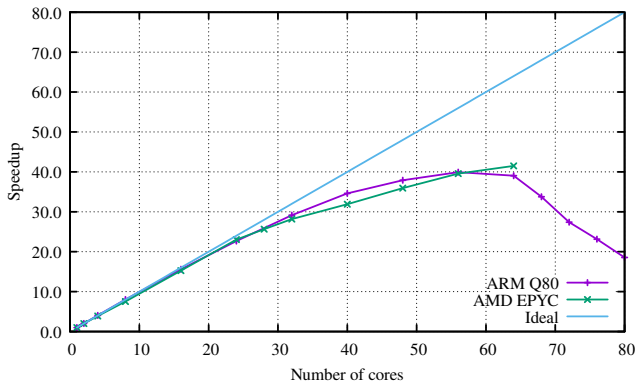

FIG. 5. Parallel speedup obtained with the ARM Q80 and AMD Epyc servers.

This performance behavior can largely be attributed to the arithmetic intensity and the bandwidth characteristics of these servers. On the ARM server, the peak performance is attained at an arithmetic intensity of 8.75 flops/byte. Notably, with fewer cores, the bandwidth per core increases, thereby enhancing efficiency. For the benzene molecule in the triple-zeta basis set, the critical arithmetic intensity is 3.33 flops/byte. This intensity corresponds to a threshold of approximately 30 cores for the ARM server and 32 cores for the AMD server. Beyond these thresholds, particularly after 64 cores on the ARM server, the heavy demand on memory bandwidth results in a rapid decline in speedup.

IV. CONCLUSION

In this work, we introduced a semi-stochastic algorithm for accelerating the computation of the perturbative triples correction in coupled cluster calculations. This novel approach combines deterministic and stochastic methods to optimize both accuracy and computational efficiency. The core of our algorithm is based on selectively calculating contributions labeled by triplets of virtual orbitals leveraging Monte Carlo sampling, and employing memoization to suppress redundant calculations.

Our results demonstrate that the semi-stochastic algorithm substantially reduces the computational effort compared to traditional deterministic methods, achiev-

ing near-exact accuracy with significantly reduced computational resources. Specifically, we have shown that the algorithm can achieve chemical accuracy with a small fraction of the computational effort required by fully deterministic approaches. This efficiency opens up new possibilities for studying larger systems or employing more extensive basis sets that were previously beyond reach due to computational constraints. Additionally, the implementation of this algorithm has proven to be highly parallelizable, demonstrating excellent scalability across different platforms.

An important aspect of our investigation focused on the application of our algorithm to potential energy surface scanning. Our method aligns well with recent findings suggesting the utility of numerous, less precise data points in constructing machine learning models.[38] For instance, we demonstrated that fitting a PES using data points generated with relatively large error bars using our algorithm still resulted in highly accurate values for the vibrational frequency and the equilibrium distance of copper chloride. This capability to produce large datasets with controlled accuracy efficiently will be particularly advantageous for training machine learning models that are robust to variations in input data quality.

Therefore, our semi-stochastic algorithm not only addresses the challenge of computational expense in quantum chemistry calculations but also facilitates the generation of extensive datasets needed for machine learning applications. This method holds significant potential for advancing computational studies in chemistry, particularly in dynamic simulations and large-scale electronic structure investigations. We advocate for continued exploration of this methodology to expand its application to other computationally demanding tasks in quantum chemistry and to explore further integration into machine learning-based chemical research.

ACKNOWLEDGMENTS

The authors kindly acknowledge fruitful discussions with Andreas Grüneis, Andreas Irmler and Tobias Schäfer. This work was supported by the European Centre of Excellence in Exascale Computing TREX — Targeting Real Chemical Accuracy at the Exascale. This project has received funding from the European Union's Horizon 2020 — Research and Innovation program — under grant agreement No. 952165. Y. Damour acknowledges support and funding from the European Research Council (ERC) (Grant Agreement No. 863481). A. Gallo acknowledges support and funding from the European Research Council (ERC) (Grant Agreement No. 101087184). This work was performed using HPC resourced from CALMIP (Toulouse) under allocations p18005 and p22001.

DATA AVAILABILITY STATEMENT

The data used to produce all the plots is available at https://zenodo.org/doi/10.5281/zenodo.11302501.

[1] J. Čížek, J. Chem. Phys. **45**, 4256 (1966).
[2] J. Čížek, in *Advances in Chemical Physics* (John Wiley & Sons, Ltd, Chichester, England, UK, 1969) pp. 35–89.
[3] J. Paldus, in *Methods in Computational Molecular Physics* (Springer, Boston, MA, Boston, MA, USA, 1992) pp. 99–194.
[4] A. Gallo, F. Hummel, A. Irmler, and A. Grüneis, The Journal of Chemical Physics **154**, 064106 (2021).
[5] J. McClain, Q. Sun, G. K.-L. Chan, and T. C. Berkelbach, Journal of Chemical Theory and Computation **13**, 1209 (2017), 1701.04832v1 [cond-mat.mtrl-sci].
[6] T. D. Crawford and H. F. Schaefer, in *Reviews in Computational Chemistry* (John Wiley & Sons, Ltd, Chichester, England, UK, 2000) pp. 33–136.
[7] R. J. Bartlett and M. Musiał, Rev. Mod. Phys. **79**, 291 (2007).
[8] I. Shavitt and R. J. Bartlett, *Many-Body Methods in Chemistry and Physics: MBPT and Coupled-Cluster Theory* (Cambridge University Press, Cambridge, England, UK, 2009).
[9] K. Raghavachari, G. W. Trucks, J. A. Pople, and M. Head-Gordon, Chem. Phys. Lett. **157**, 479 (1989).
[10] M. Villa, M. L. Senent, R. Dominguez-Gomez, O. Alvarez-Bajo, and M. Carvajal, J. Phys. Chem. A **115**, 13573 (2011).
[11] P. D. Watson, H.-w. Yong, K. M. L. Lapere, M. Kettner, A. J. McKinley, and D. A. Wild, Chem. Phys. Lett. **654**, 119 (2016).
[12] P. Vilarrubias, Molecular Physics **118**, e1797915 (2020), https://doi.org/10.1080/00268976.2020.1797915.
[13] M. Döntgen, M.-D. Przybylski-Freund, L. C. Kröger, W. A. Kopp, A. E. Ismail, and K. Leonhard, J. Chem. Theory Comput. **11**, 2517 (2015).
[14] R. Castañeda, C. Iuga, J. R. Álvarez-Idaboy, and A. Vivier-Bunge, J. Mex. Chem. Soc. **56**, 316 (2012).
[15] I. Y. Zhang and A. Grüneis, Front. Mater. **6**, 432749 (2019).
[16] J. F. Stanton, Chem. Phys. Lett. **281**, 130 (1997).
[17] T. Janowski and P. Pulay, J. Chem. Theory Comput. **4**, 1585 (2008).
[18] E. Deumens, V. F. Lotrich, A. Perera, M. J. Ponton, B. A. Sanders, and R. J. Bartlett, WIREs Comput. Mol. Sci. **1**, 895 (2011).
[19] M. Pitoňák, F. Aquilante, P. Hobza, P. Neogrády, J. Noga, and M. Urban, Collect. Czech. Chem. Commun. **76**, 713 (2011).
[20] A. E. I. Deprince and C. D. Sherrill, J. Chem. Theory Comput. **9**, 2687 (2013).
[21] V. M. Anisimov, G. H. Bauer, K. Chadalavada, R. M. Olson, J. W. Glenski, W. T. C. Kramer, E. Aprà, and K. Kowalski, J. Chem. Theory Comput. **10**, 4307 (2014).
[22] C. Peng, J. A. Calvin, and E. F. Valeev, Int. J. Quantum Chem. **119**, e25894 (2019).
[23] T. Shen, Z. Zhu, I. Y. Zhang, and M. Scheffler, J. Chem. Theory Comput. **15**, 4721 (2019).
[24] L. Gyevi-Nagy, M. Kállay, and P. R. Nagy, J. Chem. Theory Comput. **2020**, ,1 (2020).
[25] Z. Wang, M. Guo, and F. Wang, Phys. Chem. Chem. Phys. **22**, 25103 (2020).
[26] D. Datta and M. S. Gordon, J. Chem. Theory Comput. **17**, 4799 (2021).
[27] L. Gyevi-Nagy, M. Kállay, and P. R. Nagy, J. Chem. Theory Comput. **17**, 860 (2021).
[28] A. Jiang, J. M. Turney, and I. Henry F. Schaefer, J. Chem. Theory Comput. **2023**, ,5 (2023).
[29] A. P. Rendell, T. J. Lee, and A. Komornicki, Chem. Phys. Lett. **178**, 462 (1991).
[30] "Formation of the W tensor in quantum package," (2024), https://archive.softwareheritage.org/swh:1:cnt:12a71045f2333584fe7b499f1c70b5ff2dc4989c;origin=https://github.com/QuantumPackage/qp2;visit=swh:1:snp:402c2c2b30ef63cfd75b7a985700bc794ff07859;anchor=swh:

1:rev:0c8845f5f208e1c405a6aa5aba1ceb276ddbdcdf;path=/src/ccsd/ccsd_t_space_orb_abc.irp.f;lines=233-395.
[31] W. Ma, S. Krishnamoorthy, O. Villa, and K. Kowalski, J. Chem. Theory Comput. **7**, 1316 (2011).
[32] A. Haidar, T. Dong, P. Luszczek, S. Tomov, and J. Dongarra, Int. J. High Perform. Comput. Appl. **29**, 193 (2015).
[33] E. Di Napoli, D. Fabregat-Traver, G. Quintana-Ortí, and P. Bientinesi, Appl. Math. Comput. **235**, 454 (2014).
[34] P. Springer and P. Bientinesi, ACM Trans. Math. Software **44**, 1 (2018).
[35] Y. Garniron, A. Scemama, P.-F. Loos, and M. Caffarel, J. Chem. Phys. **147** (2017), 10.1063/1.4992127.
[36] Y. Garniron, T. Applencourt, K. Gasperich, A. Benali, A. Ferté, J. Paquier, B. Pradines, R. Assaraf, P. Reinhardt, J. Toulouse, P. Barbaresco, N. Renon, G. David, J.-P. Malrieu, M. Véril, M. Caffarel, P.-F. Loos, E. Giner, and A. Scemama, J. Chem. Theory Comput. **15**, 3591 (2019), swh:1:dir:6d82ae7ac757c78d7720dd89dfa52d7a453d2f68;origin=https://github.com/QuantumPackage/qp2;visit=swh:1:snp:402c2c2b30ef63cfd75b7a985700bc794ff07859;anchor=swh:1:rev:0c8845f5f208e1c405a6aa5aba1ceb276ddbdcdf;path=/src/ccsd/.
[37] J. A. Pople, Rev. Mod. Phys. **71**, 1267 (1999).
[38] D. M. Ceperley, S. Jensen, Y. Yang, H. Niu, C. Pierleoni, and M. Holzmann, Electron. Struct. **6**, 015011 (2024).
[39] "Diatomic Spectral Database | NIST," (2022), [Online; accessed 28. Mar. 2024].
[40] J. Treibig, G. Hager, and G. Wellein, in *2010 39th International Conference on Parallel Processing Workshops* (IEEE, 2010) pp. 13–16.