High-performance multi-GPU analytic RI-MP2 energy gradients

Ryan Stocks, Elise Palethorpe, and Giuseppe M. J. Barca*

School of Computing, Australian National University, Canberra, ACT 2601, Australia

E-mail: giuseppe.barca@anu.edu.au

Abstract

This article presents a novel algorithm for the calculation of analytic energy gradients from second-order Møller-Plesset perturbation theory within the Resolution-of-the-Identity approximation (RI-MP2) which is designed to achieve high performance on multi-GPU clusters. The algorithm uses GPUs for all major steps of the calculation, including integral generation, formation of all required intermediate tensors, solution of the Z-vector equation and gradient accumulation. The implementation in the EXtreme Scale Electronic Structure System (EXESS) software package includes a tailored, highly efficient, multi-stream scheduling system to hide CPU-GPU data transfer latencies and allows nodes with 8 A100 GPUs to operate at over 80% of theoretical peak floating-point performance. Comparative performance analysis shows a significant reduction in computational time relative to traditional multi-core CPU-based methods, with our approach achieving up to a 95-fold speedup over the single-node performance of established software such as Q-Chem and ORCA. Additionally, we demonstrate that pairing our implementation with the molecular fragmentation framework in EXESS can dratsically lower the computational scaling of RI-MP2 gradient calculations from quintic to sub-quadratic, enabling further substantial savings in runtime while retaining high numerical accuracy in the resulting gradients.

1 Introduction

The calculation of accurate quantum molecular gradients with respect to nuclear displacements stands as one of the most computationally intensive yet fundamental tasks in the gamut of quantum chemistry applications. These gradients are critical for the efficient identification of equilibrium molecular geometries and transition states.[1–3] Furthermore, they serve as a key component in *ab initio* molecular dynamics simulations, where they directly determine the forces acting on atoms.[3]

Second-order Møller-Plesset perturbation theory[4] (MP2) has historically been a prevalent method for obtaining reliable energy derivatives,[5–9] offering a systematic, *ab initio* pathway to molecular gradients that incorporate dynamic correlation effects that extend beyond the Hartree-Fock (HF) mean-field approximation, and including dispersion interactions.[9–11]

The capabilities and limitations of MP2, along with its associated gradients, have been extensively explored and are well-understood. MP2 generally provides accurate equilibrium geometries for large-gap, closed-shell systems,[9,12] though it tends to overestimate the binding in systems where dispersion forces are dominant.[12–19] In contrast, MP2 often falls short in accurately predicting the energetics and geometry of transition metal and open-shell systems when compared to more advanced theoretical methods.[20–23] Additionally, due to its $\mathcal{O}(N^5)$ computational scaling, MP2 becomes less feasible for molecular systems containing more than about 100 atoms, where its computational demand can become impractical.

Thus, with the advent of typically computationally less demanding and sometimes more reliable density functionals[23–29] within the Kohn-Sham density functional theory (KS-DFT) framework,[30] traditional MP2 has partly lost its appeal as an economic post-HF approach to describe electronic correlation effects.

Yet, the significance of MP2 theory as a foundational component of more accurate quantum chemical methods remains undiminished.

A first approach to improve the performance of MP2 is to self-consistently optimize its reference molecular orbitals (MOs) with respect to an MP2 correlation functional,[31–34] or even by simply using DFT orbitals.[34] Orbital Optimized MP2 (OOMP2) yields improved energetics for spin-unrestricted reference wave functions, albeit not without pitfalls.[23,33]

Another common amelioration is to suitably scale the same-spin (SS) and opposite-spin (OS) components of the MP2 energy.[21,31,35–39] Spin-Component-Scaled MP2 (SCS-MP2) provides major improvements in accuracy and robustness over traditional MP2 for closed-shell systems. Furthermore, its orbital optimized cousin, SCS-OOMP2,[31,39,40] dramatically reduces the MP2 errors for spin-contaminated reference wave functions, which are typically associated with radicals and transitions states.

A third approach is to incorporate in the MP2 energy expression a semi-empirical regularization term that prevents divergence of correlation energy at zero gap.[41–45] Regularized MP2 methods, especially κ-MP2 and κ-OOMP2,[33,46] have been reported to significantly improve upon standard MP2 for noncovalent interactions and radical systems.[46,47]

An additional, very successful extension to MP2 theory is to combine hybrid meta-GGA (generalized gradient approximation) exchange and correlation with correlation from second-order Görling–Levy perturbation theory[48] (GLPT2)[1] to obtain the so-called double-hybrid (DH) density functionals.[28,49–52] The spectrum of double-hybrid functionals has been extensively benchmarked over the past decade,[53–60] and DH-DFT is arguably the most accurate tool in the KS-DFT arsenal.

While these methods hold considerable promise, their practical application to large molecules is hindered by the steep $\mathcal{O}(N^5)$ computational scaling of the underlying MP2 calculations. Consequently, there has been tremendous research effort over recent decades on devising faster and more efficient algorithms and software for the evaluation of the MP2 energy[40,61–81] and gradients.[5,8,21,36,82–105]

In particular, the adoption of the Resolution-of-the-Identity (RI) approximation to yield the RI-MP2 method,[61,63,106] has gained prominence as a an effective technique to accelerate these calculations by reducing the computational pre-factor of the quintic MP2 algorithm while introducing negligible error.

Additional efforts were systematically undertaken to reduce the steep computational scaling of MP2 energies and gradients, thereby enabling their application to larger molecular systems. Therefore, numerous lower-order scaling algorithms were developed, offering accurate approximations for both the SS- and OS-MP2 energy components at a substantially reduced computational expense.[62,64,66–73,107] These methods primarily reduce the scaling order with system size by leveraging the local nature of electronic correlation employing strategies such as orbital localization,[62,64,67,70] atomic-level truncation and exploitation of sparsity in matrix elements,[66,69,70,72,73,75,107] or molecular fragmentation.[68,71,76,77,80,81]

Another, less explored pathway to significantly accelerate these calculations is by redesigning the underpinning algorithms to harness the massively parallel nature of modern computing hardware. A major paradigm shift in this hardware has occurred over the last decade with the widespread adoption of heterogeneous architectures. Accelerators such as graphical processing units (GPUs) now often provide the vast majority ($> 95\%$) of the computational power in the fastest machines.[108,109] Due to their fundamental architectural differences with CPUs, exploiting the computational capabilities of GPUs is challenging and requires a fundamental redesign of the complex algorithms underpinning quantum chemical methods.[108]

While significant research effort has been devoted to designing quantum chemistry software targeting the GPU architecture,[110–131] there is a dearth of efficient GPU algorithms for analytic gradients at the MP2 level of theory.

The RI-MP2 energy calculation was one of the first post-HF quantum chemistry algorithms to be accelerated using GPUs.[132] RI-MP2 was particularly identified for early adoption as the primary bottleneck can mostly be reduced to a series of matrix multiplications, for which the GPU hardware offers significant performance superiority compared to CPUs. There have since been several high-performance GPU accelerated RI-MP2 implementations in various software packages,[77,133] including by some of the present authors.[80,81] Our implementation, which achieves linear scaling with system size through usage of molecular fragmentation, enabled us to perform RI-MP2 energy calculations using the cc-pVDZ/cc-pVDZ-RIFIT basis sets on over 145,000 atoms in about 40 minutes, using approximately 27,000 GPUs on the Summit supercomputer at the Oak Ridge National Laboratory.[81]

While numerous efficient CPU based MP2 gradient algorithms and implementations have been developed, in the literature to date, there have only been two attempts to use GPUs to accelerate the MP2 or RI-MP2 gradients.

The first implementation was incorporated into Q-Chem 4.0.[134] The Q-Chem GPU code used a single GPU and only to speed up the matrix multiplications within the RI-MP2 gradient algorithm. In our tests this GPU implementation performed significantly worse than the present CPU implementation in Q-Chem 6.0. Hence, the Q-Chem single-node CPU performance will be used as the reference benchmark in performance testing.

The second implementation used Tensor Hyper-Contraction (THC) and was integrated within the Terachem GPU software package.[101] However, the THC-MP2 gradient implementation is not available in the production version of Terachem and hence for detailed comparisons.

In this article, we present a novel high-performance algorithm and software implementation for the calculation of RI-MP2 energy gradients on multiple GPUs, which was integrated in the EXtreme-scale Electronic Structure System (EXESS).[109]

Specifically, key algorithmic contributions of this work include:

- The development of efficient algorithms to perform the four major stages of the RI-MP2 gradient calculation process — the contraction of B^P_{ia}, the formation of the MP2 Lagrangians, solving the Z-vector equation, and accumulating the integral derivatives — all on GPUs.
- A multi-stream scheduling system to hide mem-

[1] Effectively MP2 using KS orbitals.

ory transfer latencies whilst utilising symmetry in the B_{ia}^{P} contraction with dynamic load balancing across multiple GPUs.

Furthermore, by leveraging the massively parallel molecular fragmentation framework in EXESS, we demonstrate that when using a Many Body Expansion truncated at the three-body level (MBE3) the resulting MBE3/RI-MP2 calculations can achieve sub-quadratic scaling with negligible loss of accuracy compared to the quintic, fragmentation-less approach.

The remainder of this article is organized as follows. Section 2 introduces the notation conventions used throughout the rest of this article. Sections 3 and 4 detail the analytic RI-MP2 gradient formulation and the challenges associated with an efficient multi-GPU implementation respectively. Section 5 discusses the novel algorithms and implementation while Section 6 presents the performance and timing results including comparison with the existing CPU based codes in Q-Chem 6.0 and ORCA.

2 Notation

Henceforth we adopt the following notation conventions. Basis functions and orbitals are identified by the index symbol as follows

- i, j, k — doubly occupied molecular orbitals (MOs).
- a, b, c — virtual orbitals.
- p, q, r, s — all molecular orbitals.
- $\mu, \nu, \lambda, \sigma$ — primary atomic orbital basis functions $\phi_\nu(\boldsymbol{r})$.
- P, Q, R — auxiliary atomic orbital (AO) basis functions.

Sums involving these indices are over the full range of the index unless otherwise specified. We follow a restricted closed-shell formalism, and denote the total number of atomic orbital basis functions N, the number of doubly occupied molecular orbitals N_{occ}, the number of virtual molecular orbitals N_{virt}, the number of auxiliary basis functions N_{aux}, and the number of atoms N_A. Note that the frozen-core approximation is not used. Thus, for conciseness we neglect basis set linear dependence effects and assume $N = N_{occ} + N_{virt}$.

Chemist's notation is adopted for the electron repulsion integrals (ERIs)

$$(\mu\nu|\lambda\sigma) = \int \phi_\mu^*(\boldsymbol{r}_1)\phi_\nu(\boldsymbol{r}_1)\frac{1}{|\boldsymbol{r}_1 - \boldsymbol{r}_2|}\phi_\lambda^*(\boldsymbol{r}_2)\phi_\sigma(\boldsymbol{r}_2)\, d\boldsymbol{r}_1 d\boldsymbol{r}_2, \quad (1)$$

and likewise $(P|Q)$ and $(\mu\nu|P)$ for the two-center (2C) and three-center (3C) ERIs, respectively.

3 RI-MP2 gradient formulation

The restricted closed-shell MP2 energy correction [135] is defined as follows

$$E_{MP2} = \sum_{ij}^{N_{occ}} \sum_{ab}^{N_{virt}} \frac{(ia|jb)[2(ia|jb) - (ib|ja)]}{\epsilon_i + \epsilon_j - \epsilon_a - \epsilon_b}. \quad (2)$$

Using the Resolution of the Identity (RI) approximation, [136] each of the four-center ERIs in Eq. 2 is approximated as a contraction of two and three-center integrals

$$(ia|jb) \approx (ia|jb)_{RI} = \sum_{P,Q} (ia|P) J_{PQ}^{-1} (jb|Q), \quad (3)$$

where $J_{PQ} = (P|Q)$ is the auxiliary basis Coulomb inner-product tensor and J_{PQ}^{-1} is its inverse.

Since J_{PQ} is positive-definite, it is possible to calculate the inverse square root $J_{PQ}^{-\frac{1}{2}}$ such that

$$(ia|jb)_{RI} = \sum_{P} B_{ia}^{P} B_{jb}^{P}, \quad (4)$$

where

$$B_{ia}^{P} = \sum_{Q} (ia|P) J_{PQ}^{-\frac{1}{2}}. \quad (5)$$

An analytic gradient for the MP2 correction energy was first presented in 1979 by Pople *et al.* [5] One of the significant challenges posed by the evaluation of the MP2 gradient is the evaluation of the derivatives of the repulsion integrals in the MO basis. This requires computing not only the derivatives of the four-center ERIs in the atomic orbital basis, but also the derivatives of the HF molecular orbital coefficients $C_{p\mu}$. In turn, the latter task involves solving the Coupled Perturbed Hartree-Fock (CPHF) [137] equations for every gradient dimension ($3N_A$ in the case of a geometry optimization) which becomes a significant bottleneck. To eliminate these repeated calculations, in 1984 Handy and Schaefer proposed the Z-vector method. [82] This allows the CPHF equations to be solved once to form the so-called Z-vector, which can then be re-used for each of the gradient dimensions. In the case of MP2 first-order gradients, the Z-vector method calculates the occupied-virtual block of the relaxed MP2 density matrix D_{ai}. [138–140]

The first derivative of the restricted closed-shell RI-MP2 electronic energy with respect to a perturbation ξ obtained by direct differentiation of Eq. 2 under the Z-vector formalism is [83,92,97,106]

$$E_{RI-MP2}^{\xi} = 4 \sum_{\mu\nu P} \Gamma_{\mu\nu}^{P} (P|\mu\nu)^{\xi} - 2 \sum_{PQ} \gamma_{PQ} (P|Q)^{\xi} \quad (6)$$

$$+ 2 \sum_{\mu\nu} \left\{ D_{\mu\nu} F_{\mu\nu}^{\xi} - W_{\mu\nu} S_{\mu\nu}^{\xi} \right\}, \quad (7)$$

where the superscript ξ denotes the first derivative of the quantity with respect to a perturbation ξ. The in-

termediate values required for the computation are

$$\Gamma^P_{ia} = \sum_{bjQ} (2X^{ab}_{ij} - X^{ba}_{ij}) B^Q_{jb} J^{-\frac{1}{2}}_{PQ}, \tag{8}$$

which is back transformed to the AO basis to form $\Gamma^P_{\mu\nu}$,

$$X^{ab}_{ij} = \frac{(ia|jb)_{RI}}{\epsilon_i + \epsilon_j - \epsilon_a - \epsilon_b}, \tag{9}$$

and

$$\gamma_{PQ} = \sum_{iaR} \Gamma^P_{ia} B^R_{ia} J^{-\frac{1}{2}}_{QR}. \tag{10}$$

The remaining intermediates are the first derivative of the Fock matrix integrals

$$F^\xi_{uv} = h^\xi_{\mu\nu} + \sum_{\lambda\sigma} \left[2(\mu\nu|\lambda\sigma)^\xi - (\mu\sigma|\lambda\nu)^\xi \right] D^{SCF}_{\lambda\sigma}, \tag{11}$$

the first derivative of the overlap matrix in the atomic orbital basis $S^\xi_{\mu\nu}$, and the energy weighted density matrix

$$W_{pq} = \frac{1}{2} \left\{ (\epsilon_p + \epsilon_q) D_{pq} \oplus \sum_{qr} A_{ipqr} D_{qr} \oplus L'_{iq} \oplus L''_{aq} \right\}, \tag{12}$$

where

$$L'_{iq} = 2 \sum_{aP} \Gamma^P_{ia} (P|qa), \tag{13}$$

$$L''_{aq} = 2 \sum_{iP} \Gamma^P_{ia} (P|iq), \tag{14}$$

and

$$A_{pqrs} = 4(pq|rs) - (ps|rq) - (pr|sq), \tag{15}$$

which is transformed to the atomic orbital basis in the usual manner.

The symbol $\oplus$ is used to denote the sum of intermediate tensors that may not fill the full domain of the resulting tensor. Note that indexing for the second term in Eq. 12 contributes only to the occupied rows of W_{pq}. There is some disagreement in literature regarding this indexing,[84,88,92,97] although it is required to match gradients calculated by finite-differences.[106]

Finally, the MP2 relaxed density matrix D is defined in blocks such that

$$D_{pq} = D_{ij} \oplus D_{ab} \oplus D_{ai}, \tag{16}$$

where the occupied-occupied and virtual-virtual blocks are

$$D_{ij} = \sum_{abk} (2X^{ab}_{ik} - X^{ab}_{ki}) X^{ab}_{jk}, \tag{17}$$

$$D_{ab} = \sum_{ijc} (2X^{ac}_{ij} - X^{ca}_{ij}) X^{bc}_{ij}, \tag{18}$$

and the occupied-virtual component is obtained by solving the Z-vector equation

$$L''_{ai} - L'_{ia} - \sum_{pq} A_{aipq} \overline{D}_{pq} = (\epsilon_a - \epsilon_i) D_{ai} + \sum_{bj} A_{aibj} D_{bj}, \tag{19}$$

where $\overline{D} = D_{ij} \oplus D_{ab}$.

For a more detailed account on this formalism, the reader can refer to Weigend and Häser[106] along with the full derivation of the MP2 gradient in Aikens *et al.*[89]

The implementation presented in this article primarily considers perturbations with respect to atomic positions which are important for geometry optimization and molecular dynamics. However, the code and formulation above can be trivially adapted to electronic field perturbations as the basis function integral derivatives (*e.g.*, $(P|\mu\nu)^\xi$ and S^ξ_{pq}) will be zero due to the basis set not being perturbed and only the core-Hamiltonian derivative $h^\xi_{\mu\nu}$ would be significant.

The RI approximation is not applied to the A_{pqrs} tensor or Fock derivative to remain analytic as this approximation was not used for the HF steps of the energy calculation.

4 Multi-GPU algorithmic challenges

The theory of MP2 gradients is well known and formulated as described in Section 3. The computation of RI-MP2 gradients is thus primarily an algorithmic and implementation challenge. There are several factors that make formulating an efficient, high-performance RI-MP2 gradient challenging, particularly when adapting to a GPU architecture. A summary of some of the most significant challenges follows:

- **Limited storage space.** There are several large intermediates required when calculating the RI-MP2 gradients, with both B^P_{ia} and Γ^P_{ia} having a space complexity of $\mathcal{O}(N_{occ} N_{virt} N_{aux})$ which can be in the hundreds of GB for the larger molecules tested in Section 6. However, each GPU has limited storage (16–80 GB on modern machines). Therefore, these tensors cannot be stored entirely on each GPU for large systems and must be distributed between multiple GPUs or stored on the host.

- **Managing high latencies.** There are significant overheads and latencies involved when moving data and invoking kernels between the host CPU and the GPU. This challenge is exacerbated by the limited integrated memory available on GPUs relative to host RAM, forcing continuous data transfer. The host to device transfer bandwidth is significantly lower than the computational throughput. Therefore, time spent transferring data between the CPU and GPU can be a significant bottleneck. Ideally, this transfer

can be overlapped with additional GPU computation, though this presents its own implementation and algorithmic challenges to minimize data dependencies.

- **Efficient parallelization.** All modern high-performance systems rely heavily on parallel computation to speed up expensive operations. This requires efficient methods for distributing and balancing work amongst many parallel processes. For example, when calculating the four-center integrals $(\mu\nu|\lambda\sigma)$, basis functions with different angular momenta have enormously varied workloads. There are also many data dependencies that arise when multiple processes are working on a single component of the computation which need to be resolved with minimal synchronization overhead or data duplication. This is particularly relevant to GPU implementations where sufficient parallelism is needed to keep several thousand GPU cores fully occupied.

- **Utilizing symmetries.** Redundant work can be avoided by utilizing symmetries in a variety of the intermediate calculations. For example, the ERIs have 8-fold symmetry $(\mu\nu|\lambda\sigma) = (\mu\nu|\sigma\lambda) = (\nu\mu|\lambda\sigma) = (\nu\mu|\sigma\lambda) = (\lambda\sigma|\mu\nu) = (\lambda\sigma|\nu\mu) = (\sigma\lambda|\mu\nu) = (\sigma\lambda|\nu\mu)$. There are additional symmetries in the density matrices $(D_{pq} = D_{qp})$ that can also be utilized to reduce computational effort.

The algorithm presented in Section 5 aims to address these challenges to obtain a high-performance and scalable RI-MP2 gradient implementation for GPU clusters.

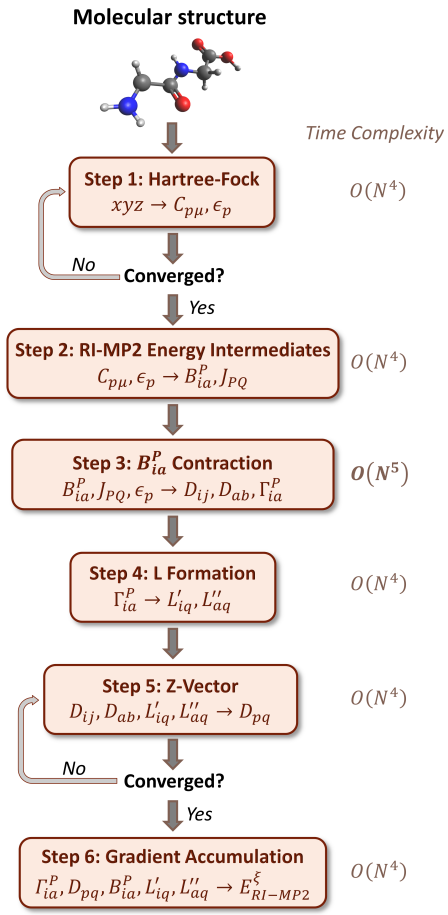

Figure 1: Overview of the 6 algorithmic steps for computing RI-MP2 energy gradients. Required inputs and outputs for each step are labelled along with their time complexities.

5 Algorithm and implementation

The calculation of the RI-MP2 gradient can be split into 6 consecutive steps as shown in Figure 1. First, an HF calculation is performed to find the MO coefficients and eigenvalues. Second, the first few steps of an RI-MP2 energy calculation are used to obtain the B^P_{ia} and J_{PQ} intermediates. Third, these integral intermediates are used to form the two blocks of the density matrix D_{ij} and D_{ab} along with Γ^P_{ia}. Fourth, Γ^P_{ia} is used to calculate the MP2 Lagrangians L' and L''. Fifth, the Z-vector equation (19) is solved iteratively to find the full MP2 relaxed density matrix. Finally, all the intermediate values are combined with integral derivatives to calculate the full analytic gradient.

The present work builds on the existing RI-MP2 energy algorithm in EXESS[141] detailed in a previous publication.[80] The MO coefficients and eigenvalues are taken from the HF stage and a similar approach is used to calculate the B^P_{ia} and J_{PQ} intermediates. The following four subsections will detail our novel multi-GPU algorithms for each of the final four stages of the gradient calculation. This will be followed by a brief discussion of some implementation details in Section 5.5.

The EXESS software utilizes distributed MPI parallelism with one MPI process per GPU along with an additional coordinating process for dynamic work distribution.[141] The majority of the intermediate values are stored in a MPI shared memory window to avoid duplication across ranks and reduce communication costs.[80]

5.1 Step 3: Contraction of B^P_{ia}

The first step unique to the RI-MP2 gradient calculation is computing the intermediate values D_{ij}, D_{ab} and Γ^P_{ia} through contraction of B^P_{ia}. This is the only quintic scaling step making it the primary bottleneck for sufficiently large molecular systems.

The cubic tensors B^P_{ia} and Γ^P_{ia} are allocated in pinned CPU memory for fast asynchronous GPU transfers and shared between all processes using an MPI shared memory window. The GPU memory footprint is reduced to $\mathcal{O}(N^2)$ by copying the aforementioned tensors from the CPU to the GPU memory in slices. The D_{ij} and D_{ab} data structures are duplicated across each GPU and reduced across GPUs once at the end to minimize synchronization.

1 **for** $i' \in batch\ i$ **do**
2 **for** $j' \in batch\ j \geq i'$ **do**
3 Copy $B^P_{i'a}, B^P_{j'b}$ slices to GPU
4 **forall** a, b **do**
5 $(i'a|j'b)_{RI} = \sum_P B^P_{i'a} B^P_{j'b}$
6 $X^{ab}_{i'j'} = \frac{(i'a|j'b)_{RI}}{\epsilon_{i'} + \epsilon_{j'} - \epsilon_a - \epsilon_b}$
7 **end**
8 **forall** a, P **do**
9 $\Gamma'^P_{i'a} = \sum_{bj'} (2X^{ab}_{i'j'} - X^{ba}_{i'j'}) B^P_{j'b}$
10 **end**
11 **if** $i' \neq j'$ **then**
12 **forall** b, P **do**
13 $\Gamma'^P_{j'b} = \sum_{aj'} (2X^{ba}_{i'j'} - X^{ab}_{i'j'}) B^P_{i'a}$
14 **end**
15 **forall** b, P **do**
16 $\Gamma^P_{j'b}$ += $\sum_Q \Gamma'^Q_{j'b} J^L_{PQ}$
17 **end**
18 **forall** a, b **do**
19 D_{ab} += $\sum_{i'j'c} (2X^{ca}_{i'j'} - X^{ac}_{i'j'}) X^{cb}_{i'j'}$
20 **end**
21 **end**
22 **forall** a, P **do**
23 $\Gamma^P_{i'a}$ += $\sum_Q \Gamma'^Q_{i'a} J^L_{PQ}$
24 **end**
25 **forall** a, b **do**
26 D_{ab} += $\sum_{i'j'c} (2X^{ac}_{i'j'} - X^{ca}_{i'j'}) X^{bc}_{i'j'}$
27 **end**
28 **end**
29 **end**
30 **for** $a' \in batch\ a$ **do**
31 **for** $b' \in batch\ b \geq a'$ **do**
32 Copy $B^P_{ia'}, B^P_{jb'}$ slices to GPU
33 **forall** i, j **do**
34 $(ia'|jb')_{RI} = \sum_P B^P_{ia'} B^P_{jb'}$
35 $X^{a'b'}_{ij} = \frac{(ia'|jb')_{RI}}{\epsilon_i + \epsilon_j - \epsilon_{a'} - \epsilon_{b'}}$
36 **end**
37 **forall** i, j **do**
38 D_{ij} += $\sum_{a'b'k} (2X^{a'b'}_{ki} - X^{a'b'}_{ik}) X^{a'b'}_{jk}$
39 **end**
40 **if** $a' \neq b'$ **then**
41 **forall** i, j **do**
42 D_{ij} += $\sum_{a'b'k} (2X^{a'b'}_{ik} - X^{a'b'}_{ki}) X^{a'b'}_{kj}$
43 **end**
44 **end**
45 **end**
46 **end**
47 Copy and reduce D_{ij}, D_{ab} on CPU

Algorithm 1: B^P_{ia} contraction computing D_{ij}, D_{ab} and Γ^P_{ia} intermediates

The full multi-GPU algorithm for the B^P_{ia} contraction steps is shown in Algorithm 1. The algorithm is split into two parts. The first part (lines 1–29) computes the D_{ab} and Γ^P_{ia} intermediates, whilst the second part (lines 30–45) computes the D_{ij} intermediate.

At each iteration of each loop, two slices of the B^P_{ia} matrix are copied to the GPU and used to form the integrals $(ia|jb)_{RI}$ (lines 5 and 34) and the energy weighted integrals X^{ab}_{ij} (lines 6 and 35). The intermediate X^{jb}_{ia} is then used to form D_{ab} in the first loop (line 26) and D_{ij} in the second (line 38). Despite the two blocks of the density matrix D both depending on X^{ab}_{ij}, they must be computed in separate loops as they require very different access patterns. When forming D_{ab}, the sum $\sum_c X^{ca}_{ij} X^{bc}_{ij}$ must be calculated over all virtual orbitals c. To perform this efficiently, X^{ab}_{ij} must be stored for all virtual orbitals a and b simultaneously. However, to calculate D_{ij}, X^{ab}_{ij} must be stored for all occupied orbitals i and j. This is infeasible in a single loop as X^{ab}_{ij} is too large to store in full. Thus, the computation of D_{ab} must be batched over the occupied orbitals i, j as is done in the first loop. Likewise, the computation of D_{ij} is batched over virtual orbitals a, b in the second loop.

Following the two loops, the components of D_{ij} and D_{ab} computed on each GPU are copied back to the host and added together to provide the full occupied-occupied and virtual-virtual blocks of the MP2 density matrix respectively. The computation of Γ'^P_{ia} (lines 9 and 13) followed by Γ^P_{ia} (lines 16 and 23) is inserted into the first loop as it can re-use the batched values of X used for computing D_{ab}.

We exploit the $(ia|jb) = (jb|ia)$ symmetry by only using batches of orbitals where $i \leq j$ and where $a \leq b$ in the first and second loops respectively. This halves the memory bandwidth required for copying the B^P_{ia} slices to the GPU and halves the number of elements of X^{ab}_{ij} that must be computed. This makes the reduction of Γ^P_{ia} more challenging as multiple processes are accumulating the same region of the tensor. This requires synchronization as described in Subsection 5.1.2

The most significant modification relative to the original CPU based approach of Weigend and Häser[106] is that instead of storing X^{ab}_{ij} between the D_{ab} and D_{ij} computation loops, these values are instead recomputed. This both reduces storage requirements and helps overcome memory bandwidth latencies on the GPU.

Ishikawa and Kuwata have previously presented an approach that reduces the memory consumption by avoiding storing the full Γ^P_{ia} tensor.[97] This was achieved by additionally batching over the auxiliary basis functions P when computing Γ^P_{ia}. However, this requires repeating some calculations and reduces the dimensions of the matrix multiplications. Hence, this approach was not utilised in the GPU implementation as the host memory consumption was not found to significantly limit the sizes of systems that could be considered.

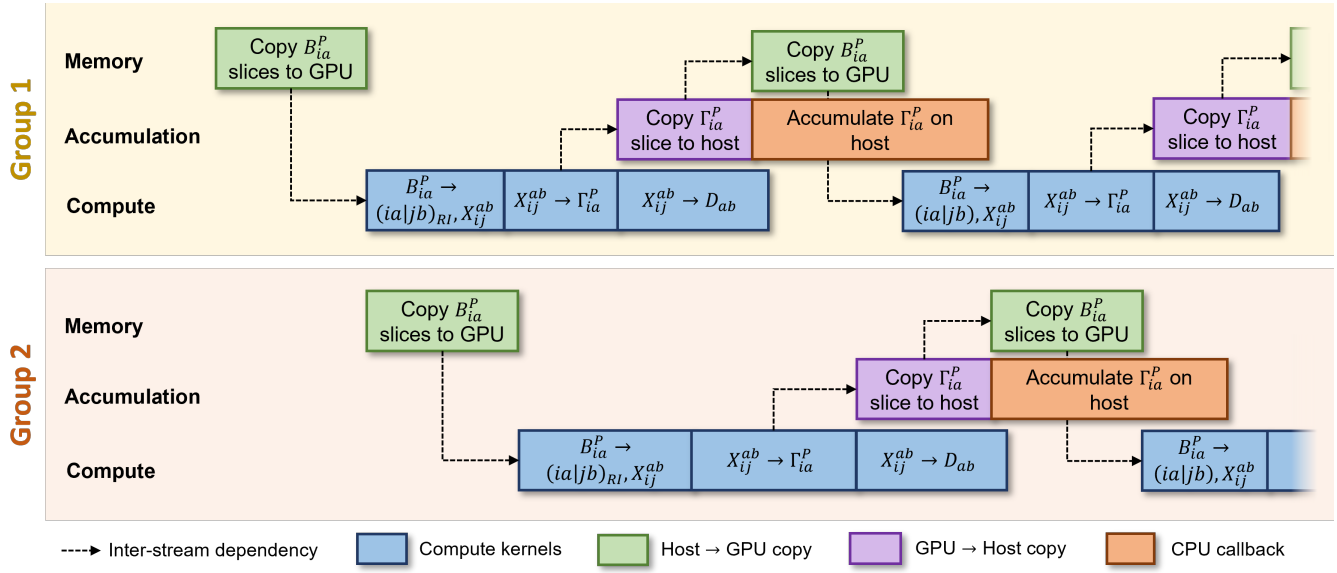

Figure 2: Illustration of the first few iterations of the multi-stream scheduling implementation of the Γ_{ia}^P and D_{ab} formation in Algorithm 1 for a single GPU. Note that the diagram is not to scale and the calculation and accumulation of the j slice of Γ_{jb}^P is not shown for clarity.

5.1.1 Hiding memory transfer latency

To maintain peak computational throughput, it is essential to overlap memory copies with GPU computations. This can be achieved using asynchronous kernel calls in multiple CUDA streams. Whilst one stream is copying matrix slices from the host to the device or vice-versa, the other stream can be performing useful computation. In this implementation, we utilise two groups with three streams each. Each group is split into a computation stream, a memory copy stream and an accumulation stream. The memory stream manages the copying of B_{ia}^P from the host to the GPU, the accumulation stream manages the accumulation of Γ_{ia}^P, and all remaining computation is performed in the computation stream. This allows overlapping of the host Γ_{ia}^P accumulation with useful computation whilst simultaneously copying the next B_{ia}^P slices for the next iteration. An example schedule is shown in Figure 2.

CUDA events are used for synchronization between streams — for example the compute stream must wait on the B_{ia}^P copies before beginning computation. In addition, B_{ia}^P memory slices are re-used to hold Γ_{ia}^P, hence the next iteration's B_{ia}^P copies must wait until Γ_{ia}^P has been copied to the host for accumulation.

5.1.2 Mutual exclusion for Γ_{ia}^P

Since the elements of Γ_{ia}^P are dependent on $(ia|jb)_{RI}$ for all indexes of j, synchronization is required to reduce these values without race conditions when batching over i and j (as these tensors are too large to be duplicated across GPUs). As Γ_{ia}^P is stored in host shared memory, this accumulation is performed asynchronously on the host in a CUDA host callback function.

Initially these accumulates were implemented using the built in MPI function within the shared memory window, however this became a significant bottleneck (likely due to excessive synchronization despite low contention). Instead, the accumulation synchronization was performed manually using inter-process atomics. One C++ atomic int was created in the shared memory window per occupied index i. This was then used as a lock for mutual exclusion of updates to the corresponding slice of the tensor Γ_{ia}^P. When performing an accumulation, a process will loop through each of the i indices in it's allocated slice, obtain the lock, perform the memory increment and then release the lock for that index before moving onto the next index. This reduced the synchronization overhead sufficiently that the accumulation could be fully overlapped with computation.

5.1.3 Load balancing

One of the significant challenges with any multi-GPU algorithm is load balancing. This requires computational work to be evenly distributed between the GPUs. In this case, the occupied and virtual batch sizes must be sufficiently small that the matrix slices can be distributed between the GPUs and hide any size-discrepancies between slices. However, it is also important that each of the computational operations is sufficiently big relative to the memory copies to fully hide the transfer latency. For example, the data transfer from CPU to GPU in a DGX A100 node is limited by the PCIe Gen 4 bus bandwidth which is approximately 25 GB/s. Compared to 19.5 TFLOP/s of double precision compute performance on a NVIDIA A100, this requires a computational intensity of at least 780 FLOPs per byte of memory transferred from the host to device. Compute kernels must also have enough threads to fill all the compute cores on the GPU. There is hence a

vital compromise between minimizing load imbalances between GPUs whilst maintaining peak throughput on each GPU. It was empirically found that a maximum batch size of $\frac{N_{occ}}{2\times\#GPUs}$ is a good compromise.

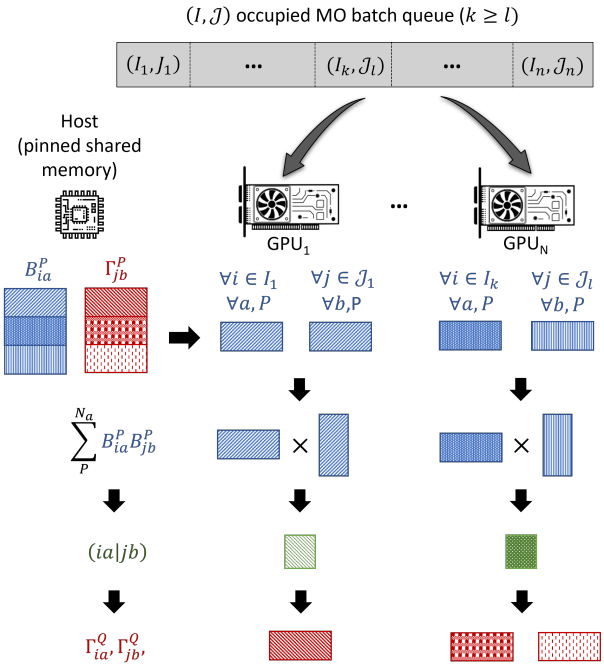

Figure 3: Dynamic load distribution scheme for scheduling work across GPUs.

To overcome load imbalances due to differently sized workloads, a dynamic load distribution scheme was used where the coordinating process maintains a list of work and distributes it to the worker processes as depicted in Figure 3. This minimizes time spent idle waiting for other processes to finish their work. To avoid synchronization costs, this distribution is performed asynchronously with respect to the computation stream. This allows the compute stream to begin working on the next iteration as soon as it has finished the previous without waiting on the memory copy.

5.2 Step 4: L formation

A very similar approach to that used for the D_{ij} reduction in Algorithm 1 is employed to compute the MP2 Lagrangians L'_{ip} and L''_{ap} (Equations 13 and 14) as shown in Algorithm 2. Here, the 3C integrals are re-computed as $(pq|P)$ is required for all orbitals p and q. This keeps the memory footprint to $\mathcal{O}(N_{occ}N_{virt}N_{aux})$ rather than $\mathcal{O}(N^2 N_{aux})$ if all 3C integrals were stored. A dynamic load distribution scheme is of increased importance here as the integral computation incurs vastly different computational effort for basis functions with different angular momenta.[142]

for $P' \in batch\ P$ **do**
  forall μ, ν **do**
    Compute $(\mu\nu|P')$
  end
  Transform $(\mu\nu|P') \rightarrow (pq|P')$
  Copy $\Gamma^{P'}_{ia}$ slice to GPU
  forall i, q **do**
    L'_{iq} += $2\sum_{aP'} \Gamma^{P'}_{ia}(aq|P')$
  end
  forall a, q **do**
    L''_{aq} += $2\sum_{iP'} \Gamma^{P'}_{ia}(iq|P')$
  end
end
Copy and reduce L'_{iq}, L''_{aq} on CPU

Algorithm 2: Computing L'_{ip} and L''_{ap} intermediates

5.3 Step 5: Z-vector equation

Once the intermediates D_{ij}, D_{ab}, L'_{iq}, and L''_{aq} have been calculated, the Z-vector equation (Eq. 19) must be solved to obtain the occupied-virtual block of the MP2 relaxed density matrix. The Z-vector equation is a linear equation requiring self-consistency and is thus solved iteratively until convergence. A single iteration to calculate a new D_{ai} is performed as follows

$$D_{ai} = \frac{L_{ai} - \sum_{bj} A_{aibj} D_{bj}}{\epsilon_a - \epsilon_i}, \tag{20}$$

where

$$L_{ai} = \left(L''_{ai} - L'_{ia} - \sum_{pq} A_{aipq} \overline{D}_{pq} \right) \tag{21}$$

is the MP2 Lagrangian which is constant throughout iterations of Eq. 20. There are several challenges presented by this formulation. The primary algorithmic challenge of this formulation is that the four-center integrals A_{iajb} (defined in Eq. 15) are too large to store in memory and must be computed on the fly for each iteration. This sum can be performed directly in the atomic orbital basis instead. In this manner, $\sum_{bj} A_{aibj} D_{bj}$ is reduced to $T_{\mu\nu} = \sum_{\lambda\sigma} A_{\mu\nu\lambda\sigma} D_{\lambda\sigma}$. This approach is summarized in Algorithm 3.

The sum

$$T_{\mu\nu} = \sum_{\lambda\sigma} A_{\mu\nu\lambda\sigma} D_{\lambda\sigma} \tag{22}$$

is computationally identical to the Fock build process

$$F_{\mu\nu} = \sum_{\lambda\sigma} [2(\mu\nu|\lambda\sigma) - (\mu\sigma|\lambda\nu)] D_{\lambda\sigma}. \tag{23}$$

Thus, the existing multi-GPU Fock-build implementation described in previous work[79,80] was used directly for this component. This utilizes the 8-fold symmetry of the four-center integrals to avoid redundant calculations and digests the values immediately into the Fock

Transform $D_{ij} \oplus D_{ab} \rightarrow \overline{D}_{\mu\nu}$
forall μ, ν **do**
  $M_{\mu\nu} = \sum_{\lambda\sigma} A_{\mu\nu\lambda\sigma} \overline{D}_{\lambda\sigma}$
end
Transform $M_{\mu\nu} \rightarrow M_{ai}$
forall a, i **do**
  $L_{ai} = L''_{ai} - L'_{ia} - M_{ai}$
end
Guess $D_{ai} = 0$
while $\neg$ *(D_{ai} converged)* **do**
  Transform $D_{ai} \rightarrow D_{\mu\nu}$
  forall μ, ν **do**
    $T_{\mu\nu} = \sum_{\lambda\sigma} A_{\mu\nu\lambda\sigma} D_{\lambda\sigma}$
  end
  Transform $T_{\mu\nu} \rightarrow T_{ai}$
  forall a, i **do**
    $D_{ai} = \frac{L_{ai} - T_{ai}}{\epsilon_a - \epsilon_i}$
  end
end

Algorithm 3: Solving the Z-vector equation.

matrix.

The convergence of the Z-vector equation is accelerated using Direct Inversion of the Iterative Subspace (DIIS).[143,144] This extrapolates a guess at each iteration using a linear combination of the previous guesses. The single iteration change in the density matrix D_{ai} was used as the error metric for the extrapolation.

5.4 Step 6: Gradient accumulation

Following the calculation of required intermediates (L'_{ip}, L''_{ap}, Γ^P_{ia}, and $D_{\mu\nu}$), the final step is accumulating the gradient as given in Eq. 6. The accumulation of each term is shown in Algorithm 4. The evaluation of the atomic orbital integral derivatives (*e.g.*, $(\mu\nu|\lambda\sigma)^\xi$) is performed using the approach outlined by Head-Gordon and Pople.[145] A comprehensive analysis of the integral derivative evaluation on GPU clusters will be released in a future publication.

For the first term calculating 3C integral derivatives, the accumulation is batched over auxiliary orbitals P. This significantly reduces the memory consumption by allowing Γ to be stored only in the molecular orbital basis. The second term involving 2C integral derivatives requires similarly computing γ_{PQ} by batching over occupied orbitals.

The third and fourth terms involving the Fock and overlap derivatives are transformed into the atomic orbital basis as this allows the gradient accumulation without storing any of the integral values. Symmetry utilization allows the derivatives of the four-center integrals $(\mu\nu|\lambda\sigma)^\xi$ to be calculated with respect to the first basis function only, *i.e.* $(\frac{d\mu}{d\xi}\nu|\lambda\sigma)$.

At first, it appears that the computational cost of calculating all of the integral derivatives (for example $(P|Q)^\xi$) would scale with the number of atoms multi-

Term 1: 3C derivatives $\sum_{\mu\nu P} \Gamma^P_{\mu\nu} (P|\mu\nu)^\xi$

for $P' \in$ *batch* P **do**
  Transform $\Gamma^{P'}_{ia} \rightarrow \Gamma^{P'}_{\mu\nu}$
  $E^\xi_{RI-MP2} \mathrel{+}= 4 \sum_{\mu \leq \nu P} \Gamma^P_{\mu\nu} (P|\mu\nu)^\xi$
end

Term 2: 2C derivatives $\sum_{PQ} \gamma_{PQ} J^\xi_{PQ}$

for $i' \in$ *batch* i **do**
  $\gamma'_{PR} \mathrel{+}= \sum_{i'a} \Gamma^P_{i'a} B^R_{i'a}$
end
$\gamma_{PQ} = \sum_R \gamma'_{PR} J^L_{RQ}$
$E^\xi_{RI-MP2} \mathrel{-}= 2 \sum_{PQ} \gamma_{PQ} J^\xi_{PQ}$

Term 3: Fock derivative $\sum_{\mu\nu} D_{\mu\nu} F^\xi_{\mu\nu}$

$E^\xi_{RI-MP2} \mathrel{+}= 2 \sum_{\mu \leq \nu} D_{\mu\nu} h^\xi_{\mu\nu}$
for *Batch* $\mu\nu\lambda\sigma$ **do**
  $E^\xi_{RI-MP2} \mathrel{+}= 2 \sum_{\mu\nu\lambda\sigma} (2 D_{\mu\nu} D^{SCF}_{\lambda\sigma} - D_{\mu\sigma} D^{SCF}_{\nu\lambda}) (\mu\nu|\lambda\sigma)^\xi$
end

Term 4: Overlap derivatives $\sum_{\mu\nu} W_{\mu\nu} S^\xi_{\mu\nu}$

$Y_{\mu\nu} = \sum_{\lambda\sigma} A_{\mu\nu\lambda\sigma} D_{\lambda\sigma}$
Transform $Y_{\mu\nu} \rightarrow Y_{ip}$
$W_{pq} = \frac{1}{2} \left\{ (\epsilon_p + \epsilon_q) D_{pq} \oplus Y_{ip} \oplus L'_{iq} \oplus L''_{aq} \right\}$
Transform $W_{pq} \rightarrow W_{\mu\nu}$
$E^\xi_{RI-MP2} \mathrel{-}= 2 \sum_{\mu\nu} W_{\mu\nu} S^\xi_{\mu\nu}$

Algorithm 4: Gradient accumulation

plied by the number of integrals. However, the derivative $(P|Q)^\xi$ with respect to a nuclear position perturbation is only non-zero if one of the basis functions P or Q is centered at the corresponding nuclei. For each pair of basis functions $\{P, Q\}$, the derivative is non-zero for at most 6 elements (3 directions for each of the 2 atom centers). This applies for the three and four-center integral derivatives as well making the computational cost of the integral derivatives a constant factor higher than the cost of the integrals themselves.

5.5 Resource acquisition

Through performance testing and profiling, it was discovered that resource acquisition of BLAS handles, pinned memory allocation and GPU memory allocation added significant non-parallelizable overhead to EXESS calculations. The EXESS code was hence refactored to avoid repeat allocations of memory resources by utilising memory pools. At the beginning of a calculation, 80% of the available GPU memory is allocated and then all subsequent GPU allocations are made within this pre-allocated memory pool. This memory pool uses a custom stack allocation scheme which ensures there is no memory fragmentation and allows extremely fast allocation and de-allocation regardless of object size. A similar approach is used for host pinned memory, however this is more expensive to allocate so the ex-

act pinned memory requirements are predicted and allocated up front, then re-allocated in the same manner. This allows the memory to be allocated and pinned once and re-used for multiple calculations or fragments in a fragmented calculation.

6 Results

In this Section, we present results from the new RI-MP2 gradient GPU implementation. In Subsection 6.1 we demonstrate the scaling of each component with respect to system size followed by the floating-point performance in Subsection 6.2. The parallel strong scaling efficiency is then presented in Subsection 6.3 along with a performance comparison against the Q-Chem and ORCA software packages in Subsection 6.4. Finally, an analysis of the performance benefits of a MBE3 fragmentation is provided in Subsection 6.5.

The GPU timing results were obtained by running the code on a NVIDIA DGX A100 node with 2 × 64-core AMD EPYC 7742 2.25 GHz CPUs, 2 TB of DDR4 memory and 8 NVIDIA Ampere A100 GPUs (released Q2 2020), each with 80 GB of High-Bandwidth Memory. All results across both GPU and CPU codes were obtained using double floating-point precision.

The performance tests were run on poly-glycine chains of varying lengths using the standard cc-pVDZ primary and cc-pVDZ auxiliary basis sets with a cartesian orbital representation. Each glycine in the chain adds 30 electrons across 7 atoms C_2H_3NO with H and OH end caps. We will denote a poly-glycine chain of length n as $\text{gly}_\text{n} = H(C_2H_3NO)_nOH$. With the cc-pVDZ primary basis and cc-pVDZ-RIFIT auxiliary basis, this has $75n + 25$ primary basis functions and $309n + 96$ auxiliary basis functions. This has a maximum of d orbital angular momentum in the primary basis and f angular momentum in the auxiliary basis. Currently, EXESS only supports four-center repulsion integral derivatives up to d functions, however there is ongoing work to enable support for higher angular momentum basis sets such as cc-pVTZ. There is also ongoing work to support density-fit Hartree-Fock, in which case the analytic derivatives would no longer require four-center integrals.

The correctness of the implementation was verified against finite-differences and cross checked against the RI-MP2 gradient implementations in Q-Chem[134] and ORCA.[146]

6.1 Scaling

To investigate the bottlenecks in the current implementation, the execution time of each component of the RI-MP2 gradient calculation was measured for a variety of system sizes. The execution time is broken into 5 components; the base HF time, the B_{ia}^P contraction time (Algorithm 1), the L formation time (Algorithm 2), the Z-vector solution (Algorithm 3), and the final gradient accumulation (Algorithm 4). This breakdown for poly-glycine chains of length 5 to 45 is shown in Figure 4. Note that the second step computing J_{PQ} and B_{ia}^P has been incorporated into the B_{ia}^P timing as it formed a negligible portion of the overall time.

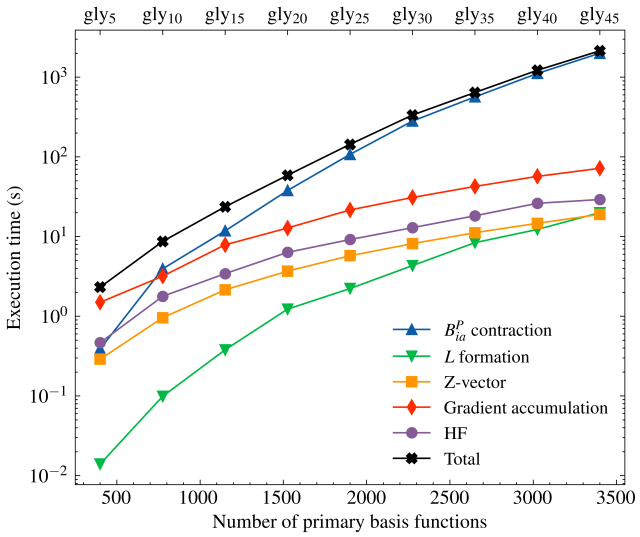

Figure 4: Computational time for each of the 5 components of the RI-MP2 gradient calculation. Timings for chains of 5 to 45 glycine amino acids are compared on 8 A100 GPUs.

As expected, most of the execution time is spent in the quintic scaling contraction of B_{ia}^P to form D_{ij}, D_{ab} and Γ_{ia}^P (Algorithm 1), particularly for larger systems. The rest of the time is split between the Z-vector calculation and the final gradient accumulation, which have comparable cost to the HF energy calculation regardless of system size.

6.2 FLOP performance

To determine the hardware utilization of the current implementation, its floating-point performance is compared to the theoretical peak performance of the A100 GPUs. A single A100 GPU can perform up to 19.5 double precision TFLOP/s when using the tensor cores. This defines a maximum theoretical performance of 156 TFLOP/s when using 8 GPUs.

The floating-point performance and percentage of peak for each of the poly-glycine chains is presented in Figure 5.

More than half the peak floating-point throughput is achieved for all systems larger than gly_{20} and greater than 80% of peak performance for gly_{45}.

The reduced performance for smaller systems is primarily due to the increased proportion of time spent in the Z-vector and gradient accumulation components as seen in Figure 4. These components have much lower floating-point throughput as the integral computations are mostly limited by memory bandwidth. In addition, for smaller systems, the contraction of B_{ia}^P does not fully

Table 1: Timings of the multi-GPU EXESS RI-MP2 gradient implementation on 8 A100 GPUs for various sized water clusters using the cc-pVDZ and def2-SVP basis sets (with the respective cc-pVDZ-RIFIT and def2-SVP-RIFIT auxiliary basis sets). The timings for Q-Chem 6 on 104 core Sapphire Rapid node are presented for comparison. OOM indicates Out Of Memory due to the reduced host memory capacity of the Sapphire Rapid node

Software	**Basis set**	**Time (s)**									
		$(H_2O)_{10}$	$(H_2O)_{20}$	$(H_2O)_{30}$	$(H_2O)_{40}$	$(H_2O)_{50}$	$(H_2O)_{60}$	$(H_2O)_{70}$	$(H_2O)_{80}$	$(H_2O)_{90}$	$(H_2O)_{100}$
EXESS	cc-pVDZ	1.58	4.48	10.6	19.9	36.7	62.6	107	183	285	462
	def2-SVP	2.11	4.46	8.15	16.9	31.5	54.5	88.5	161	275	404
Q-Chem	cc-pVDZ	7.60	41.5	183	573	2312	5012	11602	19183	26090	OOM
	def2-SVP	7.85	42.1	125	503	1174	4769	9343	18000	26875	OOM

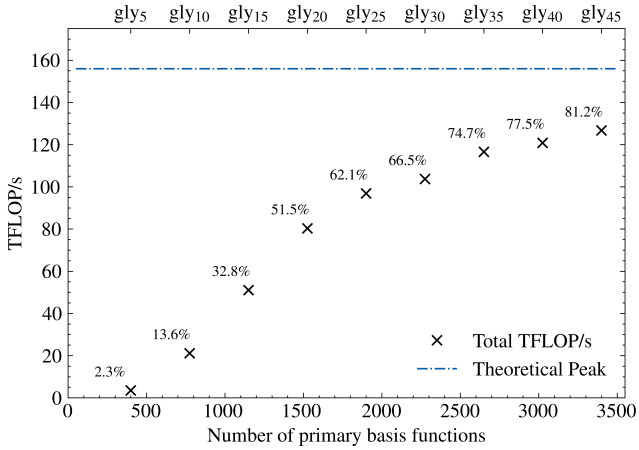

Figure 5: Floating point performance of RI-MP2 gradient implementation on 8 A100 GPUs. The data labels represent the percentage of the theoretical peak.

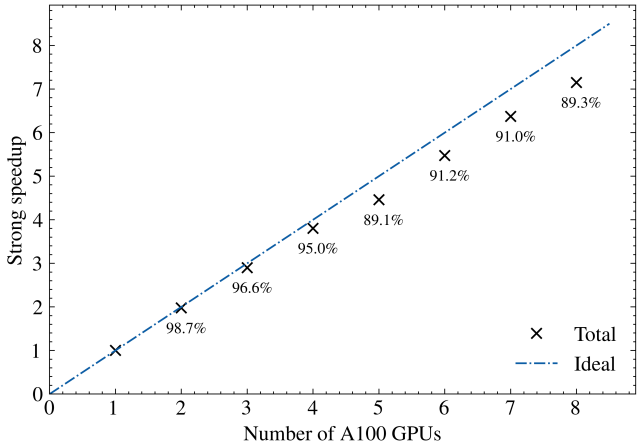

Figure 6: Strong scaling of multi-GPU RI-MP2 gradient implementation with respect to the number of GPUs for a gly_{20} chain (148 atoms, 1525 primary and 6276 auxiliary basis functions). The labels denote the total parallel speedup efficiency as a proportion of the ideal speedup relative to 1 GPU.

utilize the GPU since the ratio of computations to memory transfers is much lower. It is likely that for these smaller systems the overhead of transferring memory to the GPU is limiting the floating-point performance. For larger systems, the contraction of B^P_{ia} can fully utilize the GPU and hence the floating-point performance is much closer to the theoretical peak. There is also decreased parallel efficiency and load imbalances when using 8 GPUs, as load distribution is more challenging with smaller workloads. For example, the calculation of the RI-MP2 gradients for gly_5 is faster on 1 GPU than 8.

6.3 Multi-GPU strong scaling

To explore the parallel efficiency of the code, the overall strong scaling speedup of the RI-MP2 gradient was measured on increasing numbers of GPUs. This is shown in Figure 6. A very good speedup is observed on 8 GPUs obtaining greater than 89% total parallel efficiency for gly_{20}. This indicates efficient load distribution across the 8 GPUs even for a mid-size system.

The drop in efficiency is primarily caused by load imbalances due to insufficient work in the dynamic distribution pool which improves significantly for larger systems (as seen with the corresponding improvement in floating-point performance in Section 6.2)

6.4 Timings and speedups over existing software

The current implementation was compared to the existing CPU-based RI-MP2 gradient code in ORCA as well as the Q-Chem 6 software package. This was run on both a CPU node with 2 × 24-core Intel Xeon Platinum 8274 (Cascade Lake) 3.2 GHz CPUs (released Q2 2019) with 192 GB DDR4 RAM as well as a next generation node with 2 × 52-core Intel Xeon Platinum 8470Q (Sapphire Rapid) 2.1 GHz CPUs (released Q1 2023) and 512 GB DDR4 RAM. The timing comparison is shown in Figure 7. We compare the full single-node to single-node performance as this is the primary limiting factor when fragmentation calculation are used in EXESS, as a single fragment is assigned to a single node.

Relative to the current fastest CPU implementation (Q-Chem 6 on 104 Sapphire Rapid cores), a 95× speedup is observed for gly_{20}. The speedup would likely

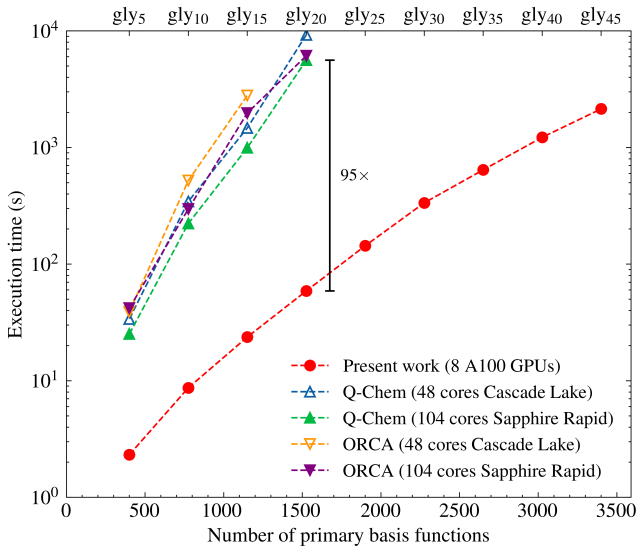

Figure 7: Comparison of the multi-GPU RI-MP2 gradient implementation with the single node CPU and GPU performance of Q-Chem.

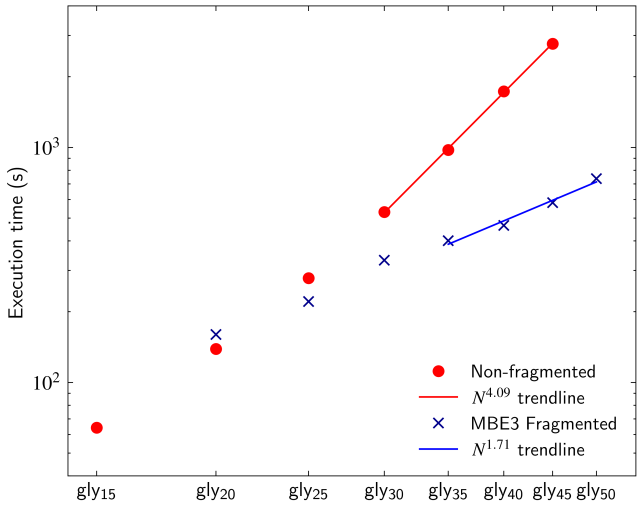

Figure 8: Comparison of MBE3 fragmented and non-fragmented RI-MP2 gradient performance.

be improved even further for gly_{25} onwards given the improvements in floating-point performance observed above for these systems. The gly_{45} calculation demonstrated here is the largest single fragment MP2 level gradient calculation reported in literature, reducing a calculation that would take several days on a CPU node down to just 36 minutes. Additional timings for water clusters with 10 to 100 water molecules are presented in Table 1.

Another important comparison is the power efficiency as the primary cost of running modern high-performance computing infrastructure is the electricity consumption. The Intel Xeon Platinum 8470Q CPUs have a power draw of approximately 350 W whilst each A100 GPU has a maximum power draw of 400 W. An 8 GPU node will therefore consume approximately 5× more power than the dual CPU nodes used in the comparisons above. Thus, the 95× speedup in execution time corresponds to a 19× improvement in power efficiency. This still represents a remarkable decrease in cost allowing a greater array of systems to now be optimized with correlated wavefunction level accuracy.

6.5 Fragmentation

A successful technique to reduce the computational cost of large electronic structure calculations is to divide the system into fragments rather than considering the system as a whole. The simplest fragmentation approach is the Many Body Expansion (MBE) truncated to a finite order.

The EXESS codebase is built on top of an efficient massively parallel, distributed memory molecular fragmentation framework. This enabled us to perform a comparison of the full execution time of non-fragmented (traditional) calculations presented in previous subsec-

Table 2: Comparison of the MBE3 fragmented and non-fragmented RI-MP2 gradient accuracy for poly-glycine chains.

gly_n	Absolute error (Hartree/Bohr)		
	Mean	Max	RMSD
20	4.87E-07	2.54E-05	2.05E-06
25	9.76E-07	3.96E-05	3.56E-06
30	1.44E-06	4.95E-05	5.12E-06
35	1.83E-06	6.36E-05	6.59E-06
40	2.19E-06	7.78E-05	7.92E-06
45	3.15E-06	8.84E-05	9.67E-06

tions with MBE3 calculations for glycine chains, as shown in Figure 8. Note that these timings are slightly longer than those presented above as the full execution time is included, accounting for the additional walltime of the startup overhead and the HF energy components.

Linear poly-glycine chains were chosen to highlight the reduced scaling and analyse the accuracy relative to a non-fragmented system. For the MBE3 calculations, the molecules were fragmented into gly_5 sub-chains with H caps added on broken single bonds. Conservatively, all fragments that contributed more than 10^{-6} Hartrees to the total energy were considered. This includes all dimers with centre of mass within 120 $\AA$ and all trimers with adjacent glycine units. Asymptotically, this will reduce the theoretical scaling of the RI-MP2 calculations from $O(N^5)$ to $O(N)$.

As shown in Figure 8, the MBE3 performance is superior to the non-fragmented algorithm for all systems larger than gly_{25} (≈ 1700 basis functions). Specifically, the MBE3/RI-MP2 algorithm achieves a sub-quadratic computational scaling with system size for the larger poly-glycine molecules in our benchmark, yielding a speedup of 4.75× for the gly_{45} system over the corresponding non-fragmented calculation.

There is remarkably little loss in accuracy due to fragmentation with a maximum single-atom absolute error of 8.84×10^{-5} Hartree/Bohr relative to the non-fragmented gradient. This is tighter than the convergence threshold of 10^{-4} used by most geometry optimizers. An additional advantage of the fragmentation based approach is the additional parallelism that can be exploited by computing fragments in parallel across many nodes. This further reduces the time to solution for larger systems.

7 Conclusions

In this work, a novel high-performance multi-GPU RI-MP2 gradient algorithm and implementation were presented. The algorithm was integrated in the EXESS software package, where it can leverage its massively parallel molecular fragmentation framework.

Without exploiting fragmentation, the algorithm maintains the standard quintic scaling but has a reduced CPU memory requirement of $\mathcal{O}(NN_{occ}N_{virt})$ and flexible GPU memory footprint of just $\mathcal{O}(N^2)$ by calculating many intermediates in batches or on the fly. The implementation incorporates highly efficient multi-stream scheduling to hide GPU-CPU data transfer latencies and allow 8 A100 GPUs to operate at over 80% of theoretical peak floating-point performance. The code exhibits near-ideal strong parallel scaling demonstrating efficient resource usage and parallelization. Without fragmentation, substantial speedups were observed relative to the Q-Chem and ORCA software packages with 95× single-node speedup leading to a 19× improvement in power efficiency for larger inputs.

Furthermore, by utilizing molecular fragmentation including up to three-body terms (MBE3) a reduction in the computational scaling of the algorithm from quintic to sub-quadratic was observed. This enables a significant reduction in time to solution while retaining high numerical accuracy in the resulting gradients. For instance, in our benchmark tests with large molecular systems such as gly_{45} the MBE3/RI-MP2 algorithm yields a speedup of 4.75× over the corresponding unfragmented approach.

8 Supporting Information

The supporting information contains the xyz structures of the glycine chains and water clusters used for the performance benchmarking.

9 Acknowledgements

The authors would like to acknowledge many helpful discussions with Dr. Andrew Gilbert who also assisted with the Q-Chem calculations and compiling a custom version of the Q-Chem GPU implementation. GMJB thanks the Pawsey Centre for Extreme Scale Readiness (PaCER) for both funding and computing resources. GMBJ thanks the National Computational Merit Allocation Scheme (NCMAS) and the Australian National University Merit Allocation Scheme (ANUMAS) for their respective computational resources grants on the Gadi supercomputer at National Computational Infrastructure and on the Setonix supercomputer at the Pawsey Supercomputing Centre.

References

(1) Schaefer, H. F.; Yamaguchi, Y. A New dimension to quantum chemistry: Theoretical methods for the analytic evaluation of first, second, and third derivatives of the molecular electronic energy with respect to nuclear coordinates. *J. Mol. Struct. : THEOCHEM* **1986**, *135*, 369–390.

(2) Schlegel, H. B. Geometry optimization. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2011**, *1*, 790–809.

(3) Pulay, P. Analytical derivatives, forces, force constants, molecular geometries, and related response properties in electronic structure theory. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2014**, *4*, 169–181.

(4) Møller, C.; Plesset, M. S. Note on an Approximation Treatment for Many-Electron Systems. *Phys. Rev.* **1934**, *46*, 618–622.

(5) Pople, J. A.; Krishnan, R.; Schlegel, H. B.; Binkley, J. S. Derivative studies in hartree-fock and

møller-plesset theories. *Int. J. Quantum Chem.* **1979**, *16*, 225–241.

(6) Handy, N. C.; Gaw, J. F.; Simandiras, E. D. Accurate ab initio prediction of molecular geometries and spectroscopic constants, using SCF and MP2 energy derivatives. *J. Chem. Soc., Faraday Trans. 2* **1987**, *83*, 1577–1593.

(7) Jørgensen, P.; Helgaker, T. Møller–Plesset energy derivatives. *J. Chem. Phys.* **1988**, *89*, 1560–1570.

(8) Frisch, M. J.; Head-Gordon, M.; Pople, J. A. Semi-direct algorithms for the MP2 energy and gradient. *Chem. Phys. Lett.* **1990**, *166*, 281–289.

(9) Helgaker, T.; Gauss, J.; Jørgensen, P.; Olsen, J. The prediction of molecular equilibrium structures by the standard electronic wave functions. *J. Chem. Phys.* **1997**, *106*, 6430–6440.

(10) Tanja Van Mourik, A. K. W.; Dunning, T. H. Benchmark calculations with correlated molecular wavefunctions. XIII. Potential energy curves for He2, Ne2 and Ar2 using correlation consistent basis sets through augmented sextuple zeta. *Mol. Phys.* **1999**, *96*, 529–547.

(11) Riley, K. E.; Hobza, P. Assessment of the MP2 Method, along with Several Basis Sets, for the Computation of Interaction Energies of Biologically Relevant Hydrogen Bonded and Dispersion Bound Complexes. *J. Phys. Chem. A* **2007**, *111*, 8257–8263, PMID: 17649987.

(12) Witte, J.; Goldey, M.; Neaton, J. B.; Head-Gordon, M. Beyond Energies: Geometries of Nonbonded Molecular Complexes as Metrics for Assessing Electronic Structure Approaches. *J. Chem. Theory Comput.* **2015**, *11*, 1481–1492, PMID: 26574359.

(13) Raghavachari, K.; Trucks, G. W.; Pople, J. A.; Head-Gordon, M. A fifth-order perturbation comparison of electron correlation theories. *Chem. Phys. Lett.* **1989**, *157*, 479–483.

(14) Sinnokrot, M. O.; Sherrill, C. D. Highly Accurate Coupled Cluster Potential Energy Curves for the Benzene Dimer: Sandwich, T-Shaped, and Parallel-Displaced Configurations. *J. Phys. Chem. A* **2004**, *108*, 10200–10207.

(15) Jurečka, P.; Šponer, J.; Černý, J.; Hobza, P. Benchmark database of accurate (MP2 and CCSD(T) complete basis set limit) interaction energies of small model complexes, DNA base pairs, and amino acid pairs. *Phys. Chem. Chem. Phys.* **2006**, *8*, 1985–1993.

(16) Sherrill, C. D.; Takatani, T.; Hohenstein, E. G. An Assessment of Theoretical Methods for Nonbonded Interactions: Comparison to Complete Basis Set Limit Coupled-Cluster Potential Energy Curves for the Benzene Dimer, the Methane Dimer, Benzene-Methane, and Benzene-H2S. *J. Phys. Chem. A* **2009**, *113*, 10146–10159, PMID: 19689152.

(17) Tkatchenko, A.; DiStasio, J., Robert A.; Head-Gordon, M.; Scheffler, M. Dispersion-corrected Møller–Plesset second-order perturbation theory. *J. Chem. Phys.* **2009**, *131*, 094106.

(18) Tomasz Janowski, A. R. F.; Pulay, P. Accurate correlated calculation of the intermolecular potential surface in the coronene dimer. *Mol. Phys.* **2010**, *108*, 249–257.

(19) Hobza, P. Calculations on Noncovalent Interactions and Databases of Benchmark Interaction Energies. *Acc. Chem. Res.* **2012**, *45*, 663–672, PMID: 22225511.

(20) Byrd, E. F. C.; Sherrill, C. D.; Head-Gordon, M. The Theoretical Prediction of Molecular Radical Species: a Systematic Study of Equilibrium Geometries and Harmonic Vibrational Frequencies. *J. Phys. Chem. A* **2001**, *105*, 9736–9747.

(21) Lochan, R. C.; Shao, Y.; Head-Gordon, M. Quartic-Scaling Analytical Energy Gradient of Scaled Opposite-Spin Second-Order Møller-Plesset Perturbation Theory. *J. Chem. Theory Comput.* **2007**, *3*, 988–1003, PMID: 26627418.

(22) Tentscher, P. R.; Arey, J. S. Geometries and Vibrational Frequencies of Small Radicals: Performance of Coupled Cluster and More Approximate Methods. *J. Chem. Theory Comput.* **2012**, *8*, 2165–2179, PMID: 26593847.

(23) Loipersberger, M.; Bertels, L. W.; Lee, J.; Head-Gordon, M. Exploring the Limits of Second- and Third-Order Møller–Plesset Perturbation Theories for Noncovalent Interactions: Revisiting MP2.5 and Assessing the Importance of Regularization and Reference Orbitals. *J. Chem. Theory Comput.* **2021**, *17*, 5582–5599, PMID: 34382394.

(24) Grimme, S.; Antony, J.; Ehrlich, S.; Krieg, H. A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu. *J. Chem. Phys.* **2010**, *132*, 154104.

(25) Grimme, S. Density functional theory with London dispersion corrections. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2011**, *1*, 211–228.

(26) Vydrov, O. A.; Van Voorhis, T. Nonlocal van der Waals density functional: The simpler the better. *J. Chem. Phys.* **2010**, *133*, 244103.

(27) Ehrlich, S.; Moellmann, J.; Grimme, S. Dispersion-Corrected Density Functional Theory for Aromatic Interactions in Complex Systems. *Acc. Chem. Res.* **2013**, *46*, 916–926, PMID: 22702344.

(28) Goerigk, L.; Grimme, S. Double-hybrid density functionals. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2014**, *4*, 576–600.

(29) Mardirossian, N.; Head-Gordon, M. Thirty years of density functional theory in computational chemistry: an overview and extensive assessment of 200 density functionals. *Mol. Phys.* **2017**, *115*, 2315–2372.

(30) Kohn, W.; Sham, L. J. Self-consistent equations including exchange and correlation effects. *Phys. Rev.* **1965**, *140*, A1133.

(31) Neese, F.; Schwabe, T.; Kossmann, S.; Schirmer, B.; Grimme, S. Assessment of Orbital-Optimized, Spin-Component Scaled Second-Order Many-Body Perturbation Theory for Thermochemistry and Kinetics. *J. Chem. Theory Comput.* **2009**, *5*, 3060–3073, PMID: 26609985.

(32) Bozkaya, U.; Sherrill, C. D. Orbital-optimized MP2.5 and its analytic gradients: Approaching CCSD(T) quality for noncovalent interactions. *J. Chem. Phys.* **2014**, *141*, 204105.

(33) Lee, J.; Head-Gordon, M. Regularized Orbital-Optimized Second-Order Møller–Plesset Perturbation Theory: A Reliable Fifth-Order-Scaling Electron Correlation Model with Orbital Energy Dependent Regularizers. *J. Chem. Theory Comput.* **2018**, *14*, 5203–5219, PMID: 30130398.

(34) Rettig, A.; Hait, D.; Bertels, L. W.; Head-Gordon, M. Third-Order Møller–Plesset Theory Made More Useful? The Role of Density Functional Theory Orbitals. *J. Chem. Theory Comput.* **2020**, *16*, 7473–7489, PMID: 33161713.

(35) Grimme, S. Improved second-order Møller–Plesset perturbation theory by separate scaling of parallel- and antiparallel-spin pair correlation energies. *J. Chem. Phys.* **2003**, *118*, 9095–9102.

(36) Gerenkamp, M.; Grimme, S. Spin-component scaled second-order Møller–Plesset perturbation theory for the calculation of molecular geometries and harmonic vibrational frequencies. *Chem. Phys. Lett.* **2004**, *392*, 229–235.

(37) Jung, Y.; Lochan, R. C.; Dutoi, A. D.; Head-Gordon, M. Scaled opposite-spin second order Møller–Plesset correlation energy: An economical electronic structure method. *J. Chem. Phys.* **2004**, *121*, 9793–9802.

(38) Lochan, R. C.; Jung, Y.; Head-Gordon, M. Scaled Opposite Spin Second Order Møller-Plesset Theory with Improved Physical Description of Long-Range Dispersion Interactions. *J. Phys. Chem. A* **2005**, *109*, 7598–7605, PMID: 16834130.

(39) Grimme, S.; Goerigk, L.; Fink, R. F. Spin-component-scaled electron correlation methods. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2012**, *2*, 886–906.

(40) JR., R. A. D.; Head-Gordon, M. Optimized spin-component scaled second-order Møller-Plesset perturbation theory for intermolecular interaction energies. *Mol. Phys.* **2007**, *105*, 1073–1083.

(41) Assfeld, X.; Almlöf, J. E.; Truhlar, D. G. Degeneracy-corrected perturbation theory for electronic structure calculations. *Chem. Phys. Lett.* **1995**, *241*, 438–444.

(42) Stück, D.; Head-Gordon, M. Regularized orbital-optimized second-order perturbation theory. *J. Chem. Phys.* **2013**, *139*, 244109.

(43) Ohnishi, Y.-y.; Ishimura, K.; Ten-no, S. Interaction Energy of Large Molecules from Restrained Denominator MP2-F12. *J. Chem. Theory Comput.* **2014**, *10*, 4857–4861, PMID: 26584372.

(44) Evangelista, F. A. A driven similarity renormalization group approach to quantum many-body problems. *J. Chem. Phys.* **2014**, *141*, 054109.

(45) Rostam M. Razban, D. S.; Head-Gordon, M. Addressing first derivative discontinuities in orbital-optimised opposite-spin scaled second-order perturbation theory with regularisation. *Mol. Phys.* **2017**, *115*, 2102–2109.

(46) Shee, J.; Loipersberger, M.; Rettig, A.; Lee, J.; Head-Gordon, M. Regularized Second-Order Møller–Plesset Theory: A More Accurate Alternative to Conventional MP2 for Noncovalent Interactions and Transition Metal Thermochemistry for the Same Computational Cost. *J. Phys. Chem. Lett.* **2021**, *12*, 12084–12097, PMID: 34910484.

(47) Rettig, A.; Shee, J.; Lee, J.; Head-Gordon, M. Revisiting the Orbital Energy-Dependent Regularization of Orbital-Optimized Second-Order Møller–Plesset Theory. *J. Chem. Theory Comput.* **2022**, *18*, 5382–5392, PMID: 36050889.

(48) Görling, A.; Levy, M. Correlation-energy functional and its high-density limit obtained from a coupling-constant perturbation expansion. *Phys. Rev. B* **1993**, *47*, 13105–13113.

(49) Zhao, Y.; Lynch, B. J.; Truhlar, D. G. Doubly Hybrid Meta DFT: New Multi-Coefficient Correlation and Density Functional Methods for Thermochemistry and Thermochemical Kinetics. *J. Phys. Chem. A* **2004**, *108*, 4786–4791.

(50) Grimme, S. Semiempirical hybrid density functional with perturbative second-order correlation. *J. Chem. Phys.* **2006**, *124*, 034108.

(51) Tarnopolsky, A.; Karton, A.; Sertchook, R.; Vuzman, D.; Martin, J. M. L. Double-Hybrid Functionals for Thermochemical Kinetics. *J. Phys. Chem. A* **2008**, *112*, 3–8, PMID: 18081266.

(52) Chai, J.-D.; Head-Gordon, M. Long-range corrected double-hybrid density functionals. *J. Chem. Phys.* **2009**, *131*, 174105.

(53) Karton, A.; Tarnopolsky, A.; Lamère, J.-F.; Schatz, G. C.; Martin, J. M. L. Highly Accurate First-Principles Benchmark Data Sets for the Parametrization and Validation of Density Functional and Other Approximate Methods. Derivation of a Robust, Generally Applicable, Double-Hybrid Functional for Thermochemistry and Thermochemical Kinetics. *J. Phys. Chem. A* **2008**, *112*, 12868–12886, PMID: 18714947.

(54) Zhang, I. Y.; Xu, X. Doubly hybrid density functional for accurate description of thermochemistry, thermochemical kinetics and nonbonded interactions. *Int. Rev. Phys. Chem.* **2011**, *30*, 115–160.

(55) Ji, H.; Shao, Y.; Goddard, W. A.; Jung, Y. Analytic Derivatives of Quartic-Scaling Doubly Hybrid XYGJ-OS Functional: Theory, Implementation, and Benchmark Comparison with M06-2X and MP2 Geometries for Nonbonded Complexes. *J. Chem. Theory Comput.* **2013**, *9*, 1971–1976, PMID: 23671408.

(56) Goerigk, L.; Hansen, A.; Bauer, C.; Ehrlich, S.; Najibi, A.; Grimme, S. A look at the density functional theory zoo with the advanced GMTKN55 database for general main group thermochemistry, kinetics and noncovalent interactions. *Phys. Chem. Chem. Phys.* **2017**, *19*, 32184–32215.

(57) Mardirossian, N.; Head-Gordon, M. Survival of the most transferable at the top of Jacob's ladder: Defining and testing the ωB97M(2) double hybrid density functional. *J. Chem. Phys.* **2018**, *148*, 241736.

(58) Santra, G.; Cho, M.; Martin, J. M. L. Exploring Avenues beyond Revised DSD Functionals: I. Range Separation, with xDSD as a Special Case. *J. Phys. Chem. A* **2021**, *125*, 4614–4627, PMID: 34009986.

(59) Santra, G.; Martin, J. M. L. Do Double-Hybrid Functionals Benefit from Regularization in the PT2 Term? Observations from an Extensive Benchmark. *J. Phys. Chem. Lett.* **2022**, *13*, 3499–3506, PMID: 35417181.

(60) Wappett, D. A.; Goerigk, L. Benchmarking Density Functional Theory Methods for Metalloenzyme Reactions: The Introduction of the MME55 Set. *J. Chem. Theory Comput.* **2023**, *19*, 8365–8383, PMID: 37943578.

(61) Feyereisen, M.; Fitzgerald, G.; Komornicki, A. Use of approximate integrals in ab initio theory. An application in MP2 energy calculations. *Chem. Phys. Lett.* **1993**, *208*, 359–363.

(62) Saebo, S.; Pulay, P. Local Treatment of Electron Correlation. *Annu. Rev. Phys. Chem.* **1993**, *44*, 213–236.

(63) Weigend, F.; Häser, M.; Patzelt, H.; Ahlrichs, R. RI-MP2: optimized auxiliary basis sets and demonstration of efficiency. *Chem. Phys. Lett.* **1998**, *294*, 143–152.

(64) Schütz, M.; Hetzer, G.; Werner, H.-J. Low-order scaling local electron correlation methods. I. Linear scaling local MP2. *J. Chem. Phys.* **1999**, *111*, 5691–5705.

(65) Ayala, P. Y.; Scuseria, G. E. Linear scaling second-order Moller–Plesset theory in the atomic orbital basis for large molecular systems. *J. Chem. Phys.* **1999**, *110*, 3660–3671.

(66) Lee, M. S.; Maslen, P. E.; Head-Gordon, M. Closely approximating second-order Mo/ller–Plesset perturbation theory with a local triatomics in molecules model. *J. Chem. Phys.* **2000**, *112*, 3592–3601.

(67) Werner, H.-J.; Manby, F. R.; Knowles, P. J. Fast linear scaling second-order Møller-Plesset perturbation theory (MP2) using local and density fitting approximations. *J. Chem. Phys.* **2003**, *118*, 8149–8160.

(68) Mochizuki, Y.; Koikegami, S.; Nakano, T.; Amari, S.; Kitaura, K. Large scale MP2 calculations with fragment molecular orbital scheme. *Chem. Phys. Lett.* **2004**, *396*, 473–479.

(69) Doser, B.; Lambrecht, D. S.; Kussmann, J.; Ochsenfeld, C. Linear-scaling atomic orbital-based second-order Møller–Plesset perturbation theory by rigorous integral screening criteria. *J. Chem. Phys.* **2009**, *130*, 064107.

(70) Pinski, P.; Riplinger, C.; Valeev, E. F.; Neese, F. Sparse maps—A systematic infrastructure for reduced-scaling electronic structure methods. I. An efficient and simple linear scaling local MP2 method that uses an intermediate basis of pair natural orbitals. *J. Chem. Phys.* **2015**, *143*, 034108.

(71) Baudin, P.; Ettenhuber, P.; Reine, S.; Kristensen, K.; Kjærgaard, T. Efficient linear-scaling second-order Møller-Plesset perturbation theory: The divide–expand–consolidate RI-MP2 model. *J. Chem. Phys.* **2016**, *144*, 054102.

(72) Pavošević, F.; Pinski, P.; Riplinger, C.; Neese, F.; Valeev, E. F. SparseMaps—A systematic infrastructure for reduced-scaling electronic structure methods. IV. Linear-scaling second-order explicitly correlated energy with pair natural orbitals. *J. Chem. Phys.* **2016**, *144*, 144109.

(73) Song, C.; Martínez, T. J. Atomic orbital-based SOS-MP2 with tensor hypercontraction. I. GPU-based tensor construction and exploiting sparsity. *J. Chem. Phys.* **2016**, *144*, 174111.

(74) Katouda, M.; Naruse, A.; Hirano, Y.; Nakajima, T. Massively parallel algorithm and implementation of RI-MP2 energy calculation for peta-scale many-core supercomputers. *J. Comput. Chem.* **2016**, *37*, 2623–2633.

(75) Song, C.; Martínez, T. J. Atomic orbital-based SOS-MP2 with tensor hypercontraction. II. Local tensor hypercontraction. *J. Chem. Phys.* **2017**, *146*, 034104.

(76) Kjærgaard, T.; Baudin, P.; Bykov, D.; Eriksen, J. J.; Ettenhuber, P.; Kristensen, K.; Larkin, J.; Liakh, D.; Pawlowski, F.; Vose, A.; Wang, Y. M.; Jørgensen, P. Massively parallel and linear-scaling algorithm for second-order Møller–Plesset perturbation theory applied to the study of supramolecular wires. *Comput. Phys. Commun.* **2017**, *212*, 152–160.

(77) Bykov, D.; Kjaergaard, T. The GPU-enabled divide-expand-consolidate RI-MP2 method (DEC-RI-MP2). *J. Comput. Chem.* **2017**, *38*, 228–237.

(78) Pham, B. Q.; Gordon, M. S. Hybrid Distributed/Shared Memory Model for the RI-MP2 Method in the Fragment Molecular Orbital Framework. *J. Chem. Theory Comput.* **2019**, *15*, 5252–5258, PMID: 31509402.

(79) Barca, G. M. J.; McKenzie, S. C.; Bloomfield, N. J.; Gilbert, A. T. B.; Gill, P. M. W. Q-MP2-OS: Møller–Plesset Correlation Energy by Quadrature. *J. Chem. Theory Comput.* **2020**, *16*, 1568–1577, PMID: 31972086.

(80) Barca, G. M. J.; Vallejo, J. L. G.; Poole, D. L.; Alkan, M.; Stocks, R.; Rendell, A. P.; Gordon, M. S. Enabling Large-Scale Correlated Electronic Structure Calculations: Scaling the RI-MP2 Method on Summit. Int. Conf. High Perform. Comput. Netw. Storage Anal. New York, NY, USA, 2021.

(81) Barca, G. M. J.; Snowdon, C.; Vallejo, J. L. G.; Kazemian, F.; Rendell, A. P.; Gordon, M. S. Scaling Correlated Fragment Molecular Orbital Calculations on Summit. Int. Conf. High Perform. Comput. Netw. Storage Anal. 2022; pp 1–14.

(82) Handy, N. C.; Schaefer, H. F. On the evaluation of analytic energy derivatives for correlated wave functions. *J. Chem. Phys.* **1984**, *81*, 5031–5033.

(83) Handy, N. C.; Amos, R. D.; Gaw, J. F.; Rice, J. E.; Simandiras, E. D. The elimination of singularities in derivative calculations. *Chem. Phys. Lett.* **1985**, *120*, 151–158.

(84) Frisch, M. J.; Head-Gordon, M.; Pople, J. A. A direct MP2 gradient method. *Chem. Phys. Lett.* **1990**, *166*, 275–280.

(85) Haase, F.; Ahlrichs, R. Semidirect MP2 gradient evaluation on workstation computers: The MP-GRAD program. *J. Comput. Chem.* **1993**, *14*, 907–912.

(86) Graham D. Fletcher, P. S., Alistair P. Rendell A parallel second-order Møller-Plesset gradient. *Mol. Phys.* **1997**, *91*, 431–438.

(87) El Azhary, A.; Rauhut, G.; Pulay, P.; Werner, H.-J. Analytical energy gradients for local second-order Mo/ller–Plesset perturbation theory. *J. Chem. Phys.* **1998**, *108*, 5185–5193.

(88) Head-Gordon, M. An improved semidirect MP2 gradient method. *Mol. Phys.* **1999**, *96*, 673–679.

(89) Aikens, C. M.; Webb, S. P.; Bell, R. L.; Fletcher, G. D.; Schmidt, M. W.; Gordon, M. S. A derivation of the frozen-orbital unrestricted open-shell and restricted closed-shell second-order perturbation theory analytic gradient expressions. *Theor. Chem. Acc.* **2003**, *110*, 233–253.

(90) Saebø, S.; Baker, J.; Wolinski, K.; Pulay, P. An efficient atomic orbital based second-order Møller–Plesset gradient program. *J. Chem. Phys.* **2004**, *120*, 11423–11431.

(91) Rhee, Y. M.; DiStasio Jr, R. A.; Lochan, R. C.; Head-Gordon, M. Analytical gradient of restricted second-order Møller–Plesset correlation energy with the resolution of the identity approximation, applied to the TCNE dimer anion complex. *Chem. Phys. Lett.* **2006**, *426*, 197–203.

(92) Distasio, R. A.; Steele, R. P.; Rhee, Y. M.; Shao, Y.; Head-Gordon, M. An improved algorithm for analytical gradient evaluation in resolution-of-the-identity second-order Møller-Plesset perturbation theory: Application to alanine tetrapeptide conformational analysis. *J. Comput. Chem.* **2007**, *28*, 839–856.

(93) Schweizer, S.; Doser, B.; Ochsenfeld, C. An atomic orbital-based reformulation of energy gradients in second-order Møller–Plesset perturbation theory. *J. Chem. Phys.* **2008**, *128*.

(94) Kossmann, S.; Neese, F. Efficient structure optimization with second-order many-body perturbation theory: The RIJCOSX-MP2 method. *J. Chem. Theory Comput.* **2010**, *6*, 2325–2338.

(95) Nagata, T.; Fedorov, D. G.; Ishimura, K.; Kitaura, K. Analytic energy gradient for second-order Møller-Plesset perturbation theory based on the fragment molecular orbital method. *J. Chem. Phys.* **2011**, *135*, 044110.

(96) Mochizuki, Y.; Nakano, T.; Komeiji, Y.; Yamashita, K.; Okiyama, Y.; Yoshikawa, H.; Yamataka, H. Fragment molecular orbital-based molecular dynamics (FMO-MD) method with MP2 gradient. *Chem. Phys. Lett.* **2011**, *504*, 95–99.

(97) Ishikawa, T.; Kuwata, K. RI-MP2 gradient calculation of large molecules using the fragment molecular orbital method. *J. Phys. Chem. Lett.* **2012**, *3*, 375–379.

(98) Bozkaya, U.; Sherrill, C. D. Analytic energy gradients for the orbital-optimized second-order Møller–Plesset perturbation theory. *J. Chem. Phys.* **2013**, *138*.

(99) Bykov, D.; Kristensen, K.; Kjærgaard, T. The molecular gradient using the divide-expand-consolidate resolution of the identity second-order Møller-Plesset perturbation theory: The DEC-RI-MP2 gradient. *J. Chem. Phys.* **2016**, *145*.

(100) Frank, M. S.; Schmitz, G.; Hättig, C. The PNO–MP2 gradient and its application to molecular geometry optimisations. *Mol. Phys.* **2017**, *115*, 343–356.

(101) Song, C.; Martínez, T. J. Analytical gradients for tensor hyper-contracted MP2 and SOS-MP2 on graphical processing units. *J. Chem. Phys.* **2017**, *147*.

(102) Pinski, P.; Neese, F. Communication: Exact analytical derivatives for the domain-based local pair natural orbital MP2 method (DLPNO-MP2). *J. Chem. Phys.* **2018**, *148*, 031101.

(103) Ni, Z.; Wang, Y.; Li, W.; Pulay, P.; Li, S. Analytical Energy Gradients for the Cluster-in-Molecule MP2 Method and Its Application to Geometry Optimizations of Large Systems. *J. Chem. Theory Comput.* **2019**, *15*, 3623–3634.

(104) Pinski, P.; Neese, F. Analytical gradient for the domain-based local pair natural orbital second order Møller-Plesset perturbation theory method (DLPNO-MP2). *J. Chem. Phys.* **2019**, *150*.

(105) Pham, B. Q.; Gordon, M. S. Development of the FMO/RI-MP2 fully analytic gradient using a hybrid-distributed/shared memory programming model. *J. Chem. Theory Comput.* **2020**, *16*, 1039–1054.

(106) Weigend, F.; Häser, M. RI-MP2: First derivatives and global consistency. *Theor. Chem. Acc.* **1997**, *97*, 331–340.

(107) DiStasio, R. A.; Jung, Y.; Head-Gordon, M. A resolution-of-the-identity implementation of the local triatomics-in-molecules model for second-order Møller- Plesset perturbation theory with application to alanine tetrapeptide conformational energies. *J. Chem. Theory Comput.* **2005**, *1*, 862–876.

(108) Gordon, M. S.; Barca, G. M.; Leang, S. S.; Poole, D.; Rendell, A. P.; Galvez Vallejo, J. L.; Westheimer, B. Novel Computer Architectures and Quantum Chemistry. *J. Phys. Chem. A* **2020**, *124*, 4557–4582.

(109) Galvez Vallejo, J. L.; Snowdon, C.; Stocks, R.; Kazemian, F.; Yan Yu, F. C.; Seidl, C.; Seeger, Z.; Alkan, M.; Poole, D.; Westheimer, B. M.; others Toward an extreme-scale electronic structure system. *J. Chem. Phys.* **2023**, *159*.

(110) Yasuda, K. Two-electron integral evaluation on the graphics processor unit. *J. Comput. Chem.* **2008**, *29*, 334–342.

(111) Ufimtsev, I. S.; Martínez, T. J. Quantum chemistry on graphical processing units. 1. strategies for two-electron integral evaluation. *J. Chem. Theory Comput.* **2008**, *4*, 222–231.

(112) Ufimtsev, I. S.; Martinez, T. J. Quantum chemistry on graphical processing units. 2. direct self-consistent-field implementation. *J. Chem. Theory Comput.* **2009**, *5*, 1004–1015.

(113) Ufimtsev, I. S.; Martinez, T. J. Quantum chemistry on graphical processing units. 3. Analytical energy gradients, geometry optimization, and first principles molecular dynamics. *J. Chem. Theory Comput.* **2009**, *5*, 2619–2628.

(114) Götz, A. W.; Wölfle, T.; Walker, R. C. *Annu. Rep. Comput. Chem.*; Elsevier, 2010; Vol. 6; pp 21–35.

(115) Luehr, N.; Ufimtsev, I. S.; Martínez, T. J. Dynamic precision for electron repulsion integral evaluation on graphical processing units (GPUs). *J. Chem. Theory Comput.* **2011**, *7*, 949–954.

(116) Asadchev, A.; Gordon, M. S. New multithreaded hybrid CPU/GPU approach to hartree-fock. *J. Chem. Theory Comput.* **2012**, *8*, 4166–4176.

(117) Titov, A. V.; Ufimtsev, I. S.; Luehr, N.; Martinez, T. J. Generating efficient quantum chemistry codes for novel architectures. *J. Chem. Theory Comput.* **2013**, *9*, 213–221.

(118) Miao, Y.; Merz, K. M. Acceleration of electron repulsion integral evaluation on graphics processing units via use of recurrence relations. *J. Chem. Theory Comput.* **2013**, *9*, 965–976.

(119) Kussmann, J.; Ochsenfeld, C. Preselective screening for linear-scaling exact exchange-gradient calculations for graphics processing units and general strong-scaling massively parallel calculations. *J. Chem. Theory Comput.* **2015**, *11*, 918–922.

(120) Miao, Y.; Merz, K. M. Acceleration of high angular momentum electron repulsion integrals and integral derivatives on graphics processing units. *J. Chem. Theory Comput.* **2015**, *11*, 1449–1462.

(121) Tornai, G. J.; Ladjánszki, I.; Rák, Á.; Kis, G.; Cserey, G. Calculation of quantum chemical two-electron integrals by applying compiler technology on GPU. *J. Chem. Theory Comput.* **2019**, *15*, 5319–5331.

(122) Kussmann, J.; Ochsenfeld, C. Hybrid CPU/GPU integral engine for strong-scaling ab initio methods. *J. Chem. Theory Comput.* **2017**, *13*, 3153–3159.

(123) Barca, G. M.; Poole, D. L.; Vallejo, J. L. G.; Alkan, M.; Bertoni, C.; Rendell, A. P.; Gordon, M. S. Scaling the hartree-fock matrix build on summit. Int. Conf. High Perform. Comput. Netw. Storage Anal. 2020; pp 1–14.

(124) Barca, G. M.; Galvez-Vallejo, J. L.; Poole, D. L.; Rendell, A. P.; Gordon, M. S. High-Performance, Graphics Processing Unit-Accelerated Fock Build Algorithm. *J. Chem. Theory Comput.* **2020**, *16*, 7232–7238.

(125) Barca, G. M.; Alkan, M.; Galvez-Vallejo, J. L.; Poole, D. L.; Rendell, A. P.; Gordon, M. S. Faster Self-Consistent Field (SCF) Calculations on GPU Clusters. *J. Chem. Theory Comput.* **2021**, *17*, 7486–7503.

(126) Cruzeiro, V. W. D.; Manathunga, M.; Merz Jr, K. M.; Götz, A. W. Open-source multi-GPU-accelerated QM/MM simulations with AMBER and QUICK. *J. Chem. Inf. Model.* **2021**, *61*, 2109–2115.

(127) Seritan, S.; Bannwarth, C.; Fales, B. S.; Hohenstein, E. G.; Isborn, C. M.; Kokkila-Schumacher, S. I.; Li, X.; Liu, F.; Luehr, N.; Snyder Jr, J. W.; others TeraChem: A graphical processing unit-accelerated electronic structure package for large-scale ab initio molecular dynamics. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2021**, *11*, e1494.

(128) Pederson, R.; Kozlowski, J.; Song, R.; Beall, J.; Ganahl, M.; Hauru, M.; Lewis, A. G.; Yao, Y.; Mallick, S. B.; Blum, V.; others Large scale quantum chemistry with tensor processing units. *J. Chem. Theory Comput.* **2022**, *19*, 25–32.

(129) Dang, D.-K.; Wilson, L. W.; Zimmerman, P. M. The numerical evaluation of Slater integrals on graphics processing units. *J. Comput. Chem.* **2022**, *43*, 1680–1689.

(130) Manathunga, M.; Aktulga, H. M.; Götz, A. W.; Merz Jr, K. M. Quantum mechanics/molecular mechanics simulations on NVIDIA and AMD graphics processing units. *J. Chem. Inf. Model.* **2023**, *63*, 711–717.

(131) Galvez Vallejo, J. L.; Barca, G. M.; Gordon, M. S. High-performance GPU-accelerated evaluation of electron repulsion integrals. *Mol. Phys.* **2023**, *121*, e2112987.

(132) Vogt, L.; Olivares-Amaya, R.; Kermes, S.; Shao, Y.; Amador-Bedolla, C.; Aspuru-Guzik, A. Accelerating Resolution-of-the-Identity Second-Order Møller-Plesset Quantum Chemistry Calculations with Graphical Processing Units. *J. Phys. Chem. A* **2008**, *112*, 2049–2057.

(133) Kwack, J. H.; Bertoni, C.; Pham, B.; Larkin, J. Performance of the RI-MP2 Fortran Kernel of GAMESS on GPUs via Directive-Based Offloading with Math Libraries. *Lect. Notes Comput. Sci.* **2020**, *12017 LNCS*, 91–113.

(134) Shao, Y.; Gan, Z.; Epifanovsky, E.; Gilbert, A. T.; Wormit, M.; Kussmann, J.; Lange, A. W.; Behn, A.; Deng, J.; Feng, X.; Ghosh, D.; Goldey, M.; Horn, P. R.; Jacobson, L. D.; Kaliman, I.; Khaliullin, R. Z.; Kus, T.; Landau, A.; Liu, J.; Proynov, E. I.; Rhee, Y. M.; Richard, R. M.; Rohrdanz, M. A.; Steele, R. P.; Sundstrom, E. J.; Woodcock, H. L.; Zimmerman, P. M.; Zuev, D.; Albrecht, B.; Alguire, E.; Austin, B.; Beran, G. J.; Bernard, Y. A.; Berquist, E.; Brandhorst, K.; Bravaya, K. B.; Brown, S. T.; Casanova, D.; Chang, C. M.; Chen, Y.; Chien, S. H.; Closser, K. D.; Crittenden, D. L.; Diedenhofen, M.; Distasio, R. A.; Do, H.; Dutoi, A. D.; Edgar, R. G.; Fatehi, S.; Fusti-Molnar, L.; Ghysels, A.; Golubeva-Zadorozhnaya, A.; Gomes, J.; Hanson-Heine, M. W.; Harbach, P. H.; Hauser, A. W.; Hohenstein, E. G.; Holden, Z. C.; Jagau, T. C.; Ji, H.; Kaduk, B.; Khistyaev, K.; Kim, J.; Kim, J.; King, R. A.; Klunzinger, P.; Kosenkov, D.; Kowalczyk, T.; Krauter, C. M.; Lao, K. U.; Laurent, A. D.; Lawler, K. V.; Levchenko, S. V.; Lin, C. Y.; Liu, F.; Livshits, E.; Lochan, R. C.; Luenser, A.; Manohar, P.; Manzer, S. F.; Mao, S. P.; Mardirossian, N.; Marenich, A. V.; Maurer, S. A.; Mayhall, N. J.; Neuscamman, E.; Oana, C. M.; Olivares-Amaya, R.; Oneill, D. P.; Parkhill, J. A.; Perrine, T. M.; Peverati, R.; Prociuk, A.; Rehn, D. R.; Rosta, E.; Russ, N. J.; Sharada, S. M.; Sharma, S.; Small, D. W.; Sodt, A.; Stein, T.; Stück, D.; Su, Y. C.; Thom, A. J.; Tsuchimochi, T.; Vanovschi, V.; Vogt, L.; Vydrov, O.; Wang, T.; Watson, M. A.; Wenzel, J.; White, A.; Williams, C. F.; Yang, J.; Yeganeh, S.; Yost, S. R.; You, Z. Q.; Zhang, I. Y.; Zhang, X.; Zhao, Y.; Brooks, B. R.; Chan, G. K.; Chipman, D. M.; Cramer, C. J.; Goddard, W. A.; Gordon, M. S.; Hehre, W. J.; Klamt, A.; Schaefer, H. F.; Schmidt, M. W.; Sherrill, C. D.; Truhlar, D. G.; Warshel, A.; Xu, X.; Aspuru-Guzik, A.; Baer, R.; Bell, A. T.; Besley, N. A.; Chai, J. D.; Dreuw, A.; Dunietz, B. D.; Furlani, T. R.; Gwaltney, S. R.; Hsu, C. P.; Jung, Y.; Kong, J.; Lambrecht, D. S.; Liang, W.; Ochsenfeld, C.; Rassolov, V. A.; Slipchenko, L. V.; Subotnik, J. E.; Van Voorhis, T.; Herbert, J. M.; Krylov, A. I.; Gill, P. M.; Head-Gordon, M. Advances in molecular quantum chemistry contained in the Q-Chem 4 program package. *Mol. Phys.* **2015**, *113*, 184–215.

(135) Møller, C.; Plesset, M. S. Note on an approximation treatment for many-electron systems. *Phys. Rev.* **1934**, *46*, 618–622.

(136) Vahtras, O.; Almlöf, J.; Feyereisen, M. W. Integral approximations for LCAO-SCF calculations. *Chem. Phys. Lett.* **1993**, *213*, 514–518.

(137) Stevens, R. M.; Pitzee, R. M.; Lipscomb, W. N. Perturbed hartree-fock calculations. I. Magnetic susceptibility and shielding in the LiH molecule. *J. Chem. Phys.* **1963**, *38*, 550–560.

(138) Salter, E. A.; Trucks, G. W.; Fitzgerald, G.; Bartlett, R. J. Theory and application of MBPT(3) gradients: The density approach. *Chem. Phys. Lett.* **1987**, *141*, 61–70.

(139) Trucks, G. W.; Salter, E. A.; Sosa, C.; Bartlett, R. J. Theory and implementation of the MBPT density matrix. An application to one-electron properties. *Chem. Phys. Lett.* **1988**, *147*, 359–366.

(140) Amos, R. D. Corrections to molecular one-electron properties using møller-plesset perturbation theory. *Chem. Phys. Lett.* **1980**, *73*, 602–606.

(141) Galvez Vallejo, J. L.; Snowdon, C.; Stocks, R.; Kazemian, F.; Yan Yu, F. C.; Seidl, C.; Seeger, Z.; Alkan, M.; Poole, D.; Westheimer, B. M.; Basha, M.; De La Pierre, M.; Rendell, A.; Izgorodina, E. I.; Gordon, M. S.; Barca, G. M. Toward an extreme-scale electronic structure system. *J. Chem. Phys.* **2023**, *159*, 44112.

(142) Barca, G. M.; Galvez Vallejo, J. L.; Poole, D. L.; Alkan, M.; Stocks, R.; Rendell, A. P.; Gordon, M. S. Enabling Large-Scale Correlated Electronic Structure Calculations: Scaling the RI-MP2 Method on Summit. Int. Conf. High Perform. Comput. Netw. Storage Anal. New York, NY, USA, 2021; pp 1–15.

(143) Pulay, P. Convergence acceleration of iterative sequences. the case of scf iteration. *Chem. Phys. Lett.* **1980**, *73*, 393–398.

(144) Sherrill, C. D. Some Comments on Accelerating Convergence of Iterative Sequences Using Direct Inversion of the Iterative Subspace (DIIS) The Mathematics of DIIS. *Lecture Notes - School of Chemistry and Biochemistry, Georgia Institute of Technology* **1998**, 1–5.

(145) Head-Gordon, M.; Pople, J. A. A method for two-electron Gaussian integral and integral derivative evaluation using recurrence relations. *J. Chem. Phys.* **1988**, *89*, 5777–5786.

(146) Neese, F.; Wiley, J. The ORCA program system. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2012**, *2*, 73–78.