High Performance Strategies for the Recent MRSF-TDDFT in GAMESS

Konstantin Komarov,[1] Vladimir Mironov,[2] Seunghoon Lee,[3] Buu Q. Pham,[4] Mark S. Gordon,[4] and Cheol Ho Choi[2]

[1)]*Center for Quantum Dynamics, Pohang University of Science and Technology, Pohang 37673, South Korea*

[2)]*Department of Chemistry, Kyungpook National University, Daegu 41566, South Korea*

[3)]*Division of Chemistry and Chemical Engineering, California Institute of Technology, Pasadena, California 91125, USA*

[4)]*Department of Chemistry and Ames Laboratory, Iowa State University, IA, 50011, USA*

(*Electronic mail: cchoi@knu.ac.kr)

(*Electronic mail: mgordon@iastate.edu)

(Dated: 20 April 2023)

Multiple ERI (Electron Repulsion Integral) tensor contractions (METC) with several matrices are ubiquitous in quantum chemistry. In response theories, the contraction operation, rather than ERI computations, can be the major bottleneck, as its computational demands are proportional to the multiplicatively combined contributions of the number of excited states and the kernel pre-factors. This paper presents several high-performance strategies for METC. Optimal approaches involve either the data layout reformations of interim density and Fock matrices, the introduction of intermediate ERI quartet buffer, and loop reordering optimization for a higher cache hit rate. The combined strategies remarkably improve the performance of the MRSF (mixed reference spin flip)-TDDFT (time-dependent density functional theory) by nearly 300%. The results of this study are not limited to the MRSF-TDDFT method and can be applied to other METC scenarios.

I. INTRODUCTION

Proper and efficient descriptions of electronic structures, especially the excited states have become increasingly important. These methods are essential in studies of photochemical and photophysical processes, such as light energy harvesting photovoltaics[1] for clean energy and efficient photoluminescence by thermally activated delayed fluorescence (TADF)[2] for high energy efficient devices, which are prominent in addressing the global energy crisis. A widely used methodology for studying molecular excited states is the spin-conserving linear-response time-dependent density functional theory (LR-TDDFT).[3–5] It is based on the time-dependent Kohn–Sham (TD–KS) equation with the linear-response formalism using a singlet ground state as a reference state. The LR-TDDFT assumes that the non-interacting system can be represented by a single Slater determinant wave function, which is referred to as an idempotency condition in terms of the density matrix.[6] Although the low-lying excitation energies of molecules are described by the conventional LR-TDDFT with sufficiently high accuracy, there are well-known failures of this methodology, e.g., in describing the energy of long-range charge transfer excitations,[7–11] excited states with substantial double excitation character,[12–14] excited states of molecules undergoing bond breaking,[15,16] and real and avoided conical intersections (CI) between the ground (S_0) and first excited (S_1) states of molecules (CI_{10}).[17–20] In addition, it was recently shown that TDDFT also fails to describe internal conversions (IC) between excited states (CI_{21}) of s-*trans*-butadiene and s-*trans*-hexatriene,[21] which is due to the missing doubly excited configurations, the unavoidable consequence of linear response formulations.

Some of these drawbacks, in particular, the incorrect description of the CI_{10} and the poor description of multi-reference electronic states, can be corrected by the spin-flip (SF)-TDDFT,[22–24] which employs an open-shell high-spin triplet. However, SF-TDDFT suffers from considerable spin contamination due to the missing responses that are needed for proper spin adaptations.[25] This has been discussed in detail by Mato and Gordon who developed a solution for Hartreee-Fock and configuration interaction wave function methods.[26,27] A fundamental solution for the problem is to expand the response space of SF-TDDFT such that it can include the missing ones. However, unlike wave function theories, a considerable challenge remains with respect to TDDFT when going beyond the adiabatic approximation to account for more than single excitations.[28] The introduction of the tensor equation-of-motion (TEOM) formalism yields the spin-adopted (SA) SF-TDDFT, which is also free of spin contamination by expanding electron configurations.[29]

However, the matrix elements of TEOM are evaluated using the Wigner-Eckart theorem, which is not satisfied by the approximate density functionals. Hence, the SA-SF-DFT formalism was derived by introducing *a posteriori* DFT correction into the equations. In addition, due to the complexity of TEOM, the analytic energy gradient for the SA-SF-DFT has yet to be derived.

Instead of introducing higher excitations from a single reference, two references ($M_s = +1$ and $M_s = -1$ of ROHF) were adopted as an alternative way of expanding the response space.[30] To recover the *idempotency* condition of its corresponding density[6], a *hypothetical* single reference by a novel spinor-like transformation has been introduced, finally yielding a mixed reference (MR) spin-flip (SF)-TDDFT (MRSF for brevity).[30,31] Consequently, MRSF produces twice as many responses as SF-TDDFT. The MRSF-TDDFT approach not only eliminates the problematic spin-contamination pitfalls of SF-TDDFT[24] but also introduces additional multi-reference features to TDDFT. Like SF-TDDFT, the MRSF method also considers the singlet ground state and other excited states on an equal footing, which not only eliminates the general topological problem[32] of CI_{10} by TDDFT but also facilitates the description of open-shell ground singlet states such as diradicals.[33] Furthermore, the one-electron spin-flip excitation from the mixed triplet reference includes the HOMO-to-LUMO *doubly* excited configuration, which is the main ingredient to properly account for the conical intersection CI_{21} between bright $1^1B_u^+$ and dark $2^1A_g^-$ states of s-*trans*-butadiene and s-*trans*-hexatriene.[21] In short, MRSF gives a balanced treatment of dynamic and nondynamic electron correlations for both ground and excited states, with the convenience of single determinant orbital optimization. These advantages allow the MRSF to reproduce the results of the more expensive multi-reference *ab initio* wave function theories.[21] In a series of studies,[32–42] it has been demonstrated that the MRSF approach can also yield accurate nonadiabatic coupling matrix elements (NACMEs),[34,36] enabling reliable nonadiabatic molecular dynamics (NAMD) simulations,[41,43–48] a topologically correct description of conical intersections,[32,37,40], accurate values of singlet–triplet gaps[33,35] and accurate spin-orbit couplings (SOC).[49] This method has also been successfully used in designing high-performance optoelectronic materials.[50–52]

However, the merits of MRSF-TDDFT come at the cost of increased computational demands. The second reference ($M_s = -1$ of ROHF) inevitably introduces additional contractions between electron repulsion integrals (ERIs) and electronic state vectors. In the special case of response theory, such a multiple ERI tensor contraction (METC) is a result of avoiding the expensive $\mathscr{O}(N^5)$ AO to MO integral transformations. The number of METC operations depends not only on the specific response theories but also on the number of excited states. Optimization of METC will

dramatically expand the applicability domain of the MRSF. In fact, METC plays an important role in quantum chemistry such as solving of the complex linear response equation[53], spectrum calculations with correlated wave functions as well as integral transformations driven by Fock matrices.[54] Therefore, in this paper, first, the existing METC algorithms will be analyzed and profiled. Second, new strategies will be proposed and implemented. Finally, benchmark calculations for natural molecular systems using high-performance computing hardware will be performed and examined.

II. BACKGROUND

A. Orbital Hessian Matrix Elements and Spin-Pairing Coupling

In the linear-response TD-DFT, the excitation energies can be computed by solving the linear response non-Hermitian eigenproblem known as Casida equation[5]. It is traditionally written in the matrix form:

$$\begin{bmatrix} A & B \\ B & A \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix}. \tag{1}$$

Here, A and B are the matrix blocks of orbital Hessian corresponding to the excitation and de-excitation, and X and Y comprise elements of the transition density matrix. The same approach applied to the RHF method is also called random-phase approximation (RPA). The equation can be simplified by neglecting the de-excitation terms. In the so-called Tamm-Dancoff approximation (TDA) the B matrix is assumed to be zero and the Casida equations (1) become[55]:

$$\mathbf{AX} = \omega\mathbf{X}. \tag{2}$$

The resulting equation is Hermitian and has twice as small dimension. On the basis of TDA approximation methods like SF and MRSF have been developed. As described in the introductions, MRSF takes a mixed reference to expand its response space. In the MRSF method, the orbital Hessian matrix **A** has the following form:

$$A_{pq,rs} = U_{pq}\left[\delta_{pr}F^{\mathrm{p}}_{qs} - \delta_{qs}F^{\mathrm{h}}_{pr} - c_H(pr|sq)\right]U_{rs} + A'_{pq,rs}, \tag{3}$$

where c_H is the mixing coefficient of the exact Hartree-Fock exchange for the hybrid functionals. **U** is the unitary transformation[30], which is used to get response states of the required multiplicity

in the MRSF method. F^p, F^h are Fock-like matrices, which are represented by

$$F^{\mathrm{p}}_{pq} = h_{pq} + \sum_i [2(pq|ii) - c_H(pi|iq)] + \sum_x (pq|xx), \tag{4a}$$

$$F^{\mathrm{h}}_{pq} = h_{pq} + \sum_i [2(pq|ii) - c_H(pi|iq)] + \sum_x [(pq|xx) - c_H(px|xq)], \tag{4b}$$

where the h_{pq} and $(pq|rs)$ are one- and two-electron integrals in the MO basis, respectively. The spatial orbital indices i and x are for doubly and singly occupied orbitals of the mixed reference state, and p,q,r are for general orbitals. Because no coupling occurs between the responses originating from the two references of the mixed-reference reduced density matrix, *a posteriori* coupling was introduced in the previous study[31] as:

$$A'_{pq,rs} = c_H \left\langle \Psi^{\mathrm{M}_S=+1}_{p\alpha q\beta} \right| \hat{H} \left| \Psi^{\mathrm{M}_S=-1}_{r\beta s\alpha} \right\rangle, \tag{5}$$

where $\left\langle \Psi^{\mathrm{M}_S=+1}_{p\alpha q\beta} \right|$ and $\left| \Psi^{\mathrm{M}_S=-1}_{r\beta s\alpha} \right\rangle$ are the bra and ket vectors for configurations originating from the $\mathrm{M}_S = +1$ and $\mathrm{M}_S = -1$ components of the mixed reference state, respectively.

B. The main computational bottleneck of the MRSF method

The matrix dimension of Casida equation eq. (2) is $(N_{\mathrm{occ}} \cdot N_{\mathrm{vir}})$, where N_{occ} and N_{vir} denote number of occupied and virtual orbitals respectively. Direct solution of the equation using convenient dense linear algebra methods will require $\mathscr{O}((N_{\mathrm{occ}} \cdot N_{\mathrm{vir}})^3) \sim \mathscr{O}(N^6)$ floating point operations and becomes impractical for large molecules, which is even one-order higher than the expensive AO-to-MO integral transformations. Here N denotes the number of basis functions and is roughly proportional to the size of the chemical system.

Fortunately, a complete set of eigenvalues is not needed except for the first few ones (electronic states). Therefore, more effective Krylov subspace methods such as Davidson's iterative process[56,57] can be used, which only require several calculations of the orbital Hessian matrix (**A**) products on trial excitation vectors (**X**), which can be computed in a direct way without explicit storage of the whole orbital Hessian. Computing each of these products only takes $\mathscr{O}(N^4)$ floating point operations. Nevertheless, this step is still the most computationally intensive task in Davidson's method.

In the Davidson's method, the equation (2) is projected onto the smaller subspace; the resulting reduced eigenvalue problem can be solved using convenient methods. The solution is iteratively improved by expanding the reduced problem space with residual vectors computed using previous

approximation to the true eigenvectors. The number of these residual vectors is equal to the number of electronic states in the TDA-based methods and is twice that number for the RPA. Thus, on every iteration of Davidson's method in MRSF one needs to compute n_{state} products of the orbital Hessian matrix $\mathbf{A}$ with trial vectors.

Computing the $\mathbf{AX}$ products in the MRSF method is achieved by the following three steps:

1. Transformation of the $\mathbf{X}$ vector into AO basis

$$\mathbf{X}^{\text{AO}} = \mathbf{C}(\mathbf{U} \circ \mathbf{X}^{\text{MO}})\mathbf{C}^{\mathsf{T}} \tag{6a}$$

2. Contraction of ERIs and the $\mathbf{X}$ vector in AO basis

$$\mathbf{G}^{\text{AO}}_{\kappa\lambda} = (\mathbf{A}\mathbf{X}^{\text{AO}})_{\kappa\lambda} = \sum_{\mu\nu} (\kappa\mu|\nu\lambda) X_{\mu\nu} \tag{6b}$$

3. *Back*-transformation of $\mathbf{AX}$ to MO basis

$$\mathbf{G}^{\text{MO}} = \mathbf{U} \circ (\mathbf{C}^{\mathsf{T}}\mathbf{G}^{\text{AO}}\mathbf{C}). \tag{6c}$$

Here, $\mathbf{C}$ is a matrix of MO coefficients in AO basis, $\mathbf{G}$ is a two-electron contribution to the $\mathbf{AX}$ product, AO and MO superscripts denote matrix/vector representation in AO and MO basis respectively. Steps 1 and 3 involve $\mathscr{O}(N^3)$ matrix multiplications with $\mathscr{O}(N^2)$ memory requirements, while step 2 requires calculations with the four-dimensional ERI tensor of $\mathscr{O}(N^4)$ computational complexity. Overall, this algorithm scales as $\mathscr{O}(N^4)$ in theory but in practice better scaling (up to $\mathscr{O}(N^2)$ for very large systems) can be achieved by applying integral screening techniques. In the current study, we use Schwarz inequality to screen small ERI values without their explicit evaluation.[58] A generic pseudo code representing above processes of obtaining ERI values and contractions is shown in Algorithm 1. It is generally assumed that the former (line 2) is more expensive than the latter (lines 4 to 6), i.e. $T_{\text{ERI}} \gg T_{\text{C}}$, in which T_{ERI} and T_{C} are the accumulative wall-clock times for the ERI evaluation and the ERI contraction with vector X, respectively. This is because the ERI computations require either conducting sophisticated calculations over quartet shell loops or reading pre-computed values from inefficient disk-based storage, while contractions take only a few floating-point operations entirely within fast system memory. Therefore, a typical algorithm for ERI tensor contraction is organized in such a way that minimizes the number of ERI evaluations. For example, in the Algorithm 1, the outermost loop (line 1) runs over the ERI

Algorithm 1 Two-electron tensor contraction

for each symmetry unique integral $(\mu\nu|\kappa\lambda)$ **do**

▷ Get two-electron integral value

  $value \leftarrow$ GET_ERI(μ, ν, κ, λ)

  for each excitation amplitude vector **X do**

▷ Compute contribution of $(\mu\nu|\kappa\lambda)$ to $\mathbf{G} \equiv \mathbf{AX}$:

    $G[\mu,\nu] \leftarrow X[\kappa,\lambda] \cdot value$ ▷ Coulomb-like contribution

    $G[\mu,\kappa] \leftarrow X[\nu,\lambda] \cdot value$ ▷ Exchange-like contribution

    ...

  end for

end for

quartets, which are either computed on-the-fly or read from the disk/memory. Therefore, each ERI quartet is computed only once for the whole duration of contractions.

The empirical rule $T_{\mathrm{ERI}} \gg T_{\mathrm{C}}$ works well when only one or two of such contractions is required as in the cases of the Hartree-Fock and KS-DFT methods. However, T_{C} grows linearly with the number of contractions, while T_{ERI} is constant. When multiple contractions are needed, T_{C} may become comparable to or even exceed T_{ERI}. In the TDDFT method and its variants, the number of contractions is usually proportional to the number of excited states (n_{state}) and the number of contraction kernels per each state (n_{kernel}) as shown in eq. (7).

$$T_{\mathrm{C}} = \bar{T}_{\mathrm{C}} \times n_{\mathrm{state}} \times n_{\mathrm{kernel}}, \tag{7}$$

where $\bar{T}_{\mathrm{C}}$ is the contraction time per state and kernel. The kernel here represents the independent ERI tensor contraction term required to assemble the product of the orbital Hessian matrix to the trial vector. The numbers of kernels (n_{kernel}) for SF-TDDFT, closed-shell TDDFT, and open-shell TDDFT are 1, 2, and 4 respectively. The extra spin-pairing coupling in eq. (5) introduces several additional contraction kernels, resulting in $n_{\mathrm{kernel}} = 11$ for the MRSF-TDDFT method. As a result, T_{C} can significantly contribute to the time of MRSF-TDDFT calculation, especially when multiple excited states are considered.

III. GENERIC AND ALTERNATIVE ALGORITHMS

As discussed in section II, the METC of the MRSF can be the main bottleneck, which needs to be alleviated especially for its practical use in computationally demanding tasks such as nonadiabatic molecular dynamics (NAMD) simulations.[41] In this section we introduce alternative algorithms addressing the METC performance.

A. Disk-based vs Direct ERI Evaluation

First, the high-performance scheme of ERI evaluation is discussed as it can be one of the main computational bottlenecks. Historically, ERIs were computed and saved to disk files for the SCF (Self-Consistent-Field) process. However, disk-based methods suffer from high I/O overhead when increasing the size of molecular systems to be computed. The "in-core" memory approach saves ERI to computer fast memory. However, it is only applicable to small molecular systems. The direct SCF method[59] was developed to eliminate $\mathcal{O}(N^4)$ storage requirements by computing needed ERIs on-the-fly. Although direct methods are more computationally demanding, the growth of computing power and the development of various strategies[58,60] such as density-driven ERI screening and incremental Fock building technique, have made the direct SCF method the most efficient approach.

In the past decade, the capacity and performance of computer memory and disks have been improved significantly, making the storage-based approach attractive again. Non-volatile memory express (NVMe) drives are especially promising for ERI storage as they provide a very high bandwidth, which is just several times slower than system memory. In the current study, both direct and storage-based ERI computations shall be comparatively benchmarked.

B. ERI calculation in GAMESS

The GAMESS[61] quantum chemistry program, which has both direct and storage-based ERI methods, is adopted for the current study. ERIs in GAMESS are evaluated over quadruple shell loops which are stored in an intermediate four-dimensional (4D) array named `GHONDO`. This 4D array is then used either by a contraction subroutine or a subroutine that saves ERIs on a storage device. Both of these subroutines share a similar unpacking process of the data stored in `GHONDO` array, i.e., traversing the array, dropping insignificant ERIs, and computing basis function indices

Algorithm 2 (Integrals–Fock, IF) Direct ERI tensor contraction: integrals, then amplitudes

1: **for each** symmetry unique shell quartet (i, j, k, l) **do**

▷ Compute two-electron integrals:

2: GHONDO ← COMPUTE_ERI_BLOCK(*i, j, k, l*)

3: **for each** ERI **in** GHONDO **do**

4: Compute indices $\mu, \nu, \kappa, \lambda$

5: **for each** excitation amplitude vector **X do**

6: Compute contribution of ERI to **G**...

7: **end for**

8: **end for**

9: **end for**

of the remaining ERIs. In direct methods, these indices and ERIs are sufficient for the next step, e.g., contracting with the density matrix. In the storage-based approach, one must save indices and ERI values into two linear arrays. In terms of data size, each ERI entry needs 16 bytes of storage, i.e., 8 bytes for the double-precision ERI floating point value, and 8 bytes for the four half-precision (2 bytes each) integer indices.

The quartet computation and the `GHONDO` array unpack in the direct approach introduce additional loop levels shown in Algorithm 2. The first loop (line 1) runs over unique quartets. The second loop (line 3) runs over individual ERI of the quartet stored in the `GHONDO` array.

C. Strategies addressing METC problem

1. Direct integral calculation

The order of loops in the Algorithm 2, i.e. upper loop over the `GHONDO` elements and bottom loop over the contractions, is optimal for work with `GHONDO` array, i.e., individual ERIs and their corresponding indices are loaded and retrieved only once. However, such loop organization makes the data access to **X** and **G** elements inefficient. This is because when data points in memory are loaded to CPU registers, adjacent elements are also loaded into cache lines. For the next round of computing, if the requested elements are already in the cache, it is not loaded again. This is

called a cache hit, which results in highly efficient data access. If the requested data points are not found in the cache, existing data points in the cache must be expunged before needed elements are loaded from memory. This is an expensive cache miss, which will impair the computational performance. This is especially true when the size of **X** and **G** is much bigger than that of CPU cache. As the matrix dimension of Casida equation eq. (2) is $N_{\text{occ}} \cdot N_{\text{vir}}$, the **X** and **G** matrices have the same large dimension. In Algorithm 2. For each element of `GHONDO`, the inner loop (line 5) will need to scan through entire big arrays **X** and **G**, whose elements are scattered in a large area of memory. This likely results in a very low cache hit rate.

In comparison with **X** and **G**, the `GHONDO` array is much smaller. For instance, with normal basis sets up to d-function, the size of `GHONDO` is $6^4 = 1296$ double precision numbers, which is ≈ 10 KB. In many modern hardware sizes of CPU caches exceed this amount. For example, in the current work, even the smallest CPU L1 cache of 32KB can fit an entire `GHONDO` array. Therefore, to increase the cache utilization, an alternative algorithm of the integral-direct ERI tensor contraction (Algorithm 3) is devised. In Algorithm 3, for each element of array **X** in the outer loop (line 3), the inner loop (line 4) will scan through all `GHONDO` elements and compute contribution to array **G**. Since `GHONDO` is relatively small, it is likely that a large part or the entire array will be already present in CPU cache, yielding a high cache hit rate. This improvement in Algorithm 3 comes at the cost of recomputing ERI indices multiple times. However, this is a

Algorithm 3 (Fock–Integrals, FI) Direct ERI tensor contraction: amplitudes, then integrals

```
1: for each symmetry unique shell quartet (i, j, k, l) do
   ▷ Compute two-electron integrals:
2:     GHONDO ←COMPUTE_ERI_BLOCK(i, j, k, l)
3:     for each excitation amplitude vector X do
4:         for each ERI in GHONDO do
5:             Compute indices μ, ν, κ, λ
6:             Compute contribution of ERI to G...
7:         end for
8:     end for
9: end for
```

Algorithm 4 (Buffer, B) Direct ERI tensor contraction: amplitudes, then integrals, buffered

for each symmetry unique shell quartet (i, j, k, l) **do**
  GHONDO ←COMPUTE_ERI_BLOCK(*i, j, k, l*) ▷ Compute two-electron integrals
  BUFFER_ADD(*buffer,* GHONDO*, i, j, k, l*) ▷ Append ERIs and corresponding indices to the buffer
  if BUFFER_IS_FULL(*buffer*) **then**
    BUFFER_FLUSH(*buffer*, $\mathbf{X}, \mathbf{G}$) ▷ Do contractions, clean the buffer
  end if
end for
BUFFER_FLUSH(*buffer*, $\mathbf{X}, \mathbf{G}$) ▷ Process remaining integrals in the buffer

procedure BUFFER_ADD(*buffer,* GHONDO*, i, j, k, l*) ▷ Move ERIs from sparse storage to dense one
  for each ERI **in** GHONDO **do**
    Compute indices $\mu, \nu, \kappa, \lambda$
    Append ERI and $\mu, \nu, \kappa, \lambda$ to *buffer*
  end for
end procedure

procedure BUFFER_FLUSH(*buffer*, $\mathbf{X}, \mathbf{G}$) ▷ Compute contractions, then reset the *buffer*
  for each excitation amplitude vector **X do**
    for each ERI **in** *buffer* **do**
      Compute contribution of ERI to **G**...
    end for
  end for
  BUFFER_RESET(*buffer*)
end procedure

reasonable trade-off because index computation is a simple operation with negligible overhead. Algorithm 3 is dubbed Fock–Integrals (FI), i.e., loop over Fock-like matrices (states) and over Integrals. Algorithm 2 originally implemented in GAMESS will be referred to as Integrals–Fock (IF) (i.e., loop over Integrals and over Fock-like matrices), which are summarized in Table I.

TABLE I: The matrix of contraction algorithms used in this work: Integrals–Fock (IF) (Algorithm 2) utilizes loop over Integrals and over Fock (states); Integrals–Fock&Transpose (IF&T) uses the data transposition; Fock–Integrals (FI) (Algorithm 3) is loop over Fock and over Integrals. In addition to contraction algorithms are used linear ERI buffer (B) option (Algorithm 4), provides a compact ERI storage with sequential memory access.

		Memory access		
		Integrals–Fock (IF)	IF&Transpose (IF&T)	Fock–Integrals (FI)
Buffer	No buffer	IF	IF&T	FI
	Buffered, 16-bit (2 bytes) integers (B2)	-	IF&T-B2	FI-B2
	Buffered, 64-bit (8 bytes) integers (B8)	-	-	FI-B8

Algorithm 3 can be further improved by pre-processing the `GHONDO` array. Since it is defined as a fixed array with a large array size to reserve all possible ERI blocks, the `GHONDO` array is quite sparse in reality, making memory access less efficient. To overcome this, a dense linear ERI buffer is introduced in Algorithm 4, which provides a compact ERI storage with sequential memory access. Yet another advantage of the buffering is that unpacking `GHONDO` array and **G** and **X** array updates can be split into two separate steps. The buffer technique is denoted Buffered (B) in Table I. The ERI buffer is a pair of temporary arrays. One of them is used to store ERI values, while the other is used to store corresponding AO indices. In order to reduce the size of the latter array, we can take advantage of the fact that in typical simulations, the maximum values of the associated indices (which are equal to the size of the AO basis set) usually do not exceed several thousand. This means that we can store these indices using signed 16-bit integers, which can hold values up to $2^{15} - 1 = 32767$. In our study, we compared the performance of algorithms that use 16-bit and 64-bit indices, which take up 2 bytes or 8 bytes of memory, respectively. These algorithms are referred to as B2 and B8 in Table I. ERI values were always stored using double precision floating point numbers which take $N_{\text{bytes}}^{\text{ERI}} = 8$ bytes. The size of each ERI entry is then $N_{\text{bytes}}^{\text{entry}} = N_{\text{bytes}}^{\text{ERI}} + 4 \times N_{\text{bytes}}^{\text{index}}$, which is equal to 16 and 40 bytes for algorithms with 2-byte and 8-byte indices respectively.

Finally, an alternative to the loop interchange approach is a reorganization of the **G** and **X** data. The order of the loops in the IF algorithm implies that elements of matrices **G** and **X** with the same indices are always accessed together. We can make elements of the matrices with the same

indices reside as close as possible with each other in memory by a data transpose: $X[\mu\nu, vec] \leftrightarrow X[vec, \mu\nu]$, and $G[\mu\nu, vec] \leftrightarrow G[vec, \mu\nu]$. Note, that the loop interchange is not beneficial in this case and the algorithm becomes similar to Algorithm 2. Thus, this approach will be further referred to as IF&T (Integrals–Fock&Transpose), where T denotes the data transposition.

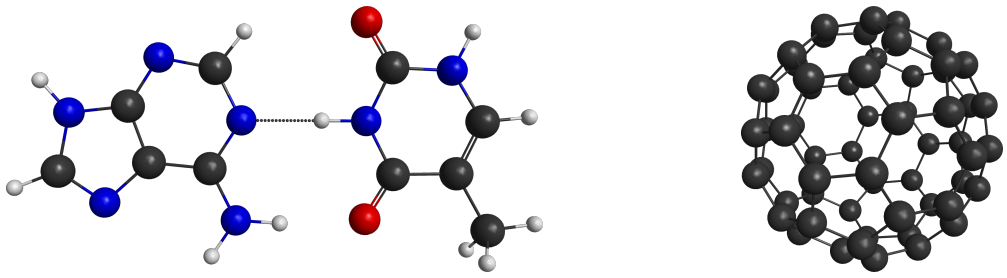
The data transposition affects performance in the following way. CPU fetches data from memory to caches in small contiguous memory blocks, which are called cache lines. The typical cache line size for modern Intel CPUs is 64 bytes or 8 double precision numbers. By arranging the elements of **X** and **G** arrays in a specific way, we can reduce the number of memory access operations and thereby improve performance. Specifically, if we place elements with the same AO indices next to each other in memory, the CPU can load multiple elements of these arrays in a single memory access operation. That will significantly increase cache usage efficiency. In contrast, the original (with non-transposed data) IF algorithm requires separate memory access for every element of these matrices.

2. Storage-based integral calculation

In order to be efficient, disk-based methods should take into account the detailed specifications of the storage hardware. It is clear that the latency of disk access is orders of magnitude higher than that of system memory. Furthermore, hard disk drives and to a lesser extent solid-state drives prefer linear access patterns when large chunks of data are loaded/stored simultaneously. It is therefore beneficial to reduce disk I/O or combine multiple data loads/stores into a bigger transaction. Operating systems and applied software implement various kinds of memory caches for disk I/O. As discussed in Section III B, each ERI data point in GAMESS is stored on disk in a 16-byte entry, which contains one 8-byte integral floating point and four 2-byte AO index integers. Entries on the same record are read/written from/to disk at once. The number of entries per record is controlled by users. Like the batched integral calculations in direct methods, the I/O-based methods also introduce an additional loop level, giving rise to several variants of the algorithm, denoted by IF, FI, and IF&T (see Table I). The memory access patterns will be analogous to the direct methods in Section III C. By design, all storage-based methods use dense buffers. Therefore, there are no special buffered versions (i.e., to reduce the sparsity of the `GHONDO` array).

IV. COMPUTATIONAL AND HARDWARE DETAILS

The benchmarks reported in this paper were performed on dual-socket Dell PowerEdge R630 nodes with dual Intel® Xeon® E5-2697v3 CPUs. These CPUs have 32KB L1 instruction and data caches per core, 256 KB L2 cache per core, and 35 MB L3 cache shared among all cores on the same socket. The hyperthreading feature was disabled during benchmarks. Each node has 64 GB of system memory. Seagate Constellation.2™ SAS drives were utilized for HDD benchmarks. Two SK Hynix P31 Gold NVMe drives stripped on RAID-0 were utilized for SSD test. All nodes were running Rocky Linux 8.6 with 4.18 kernel. Core frequencies were controlled by `intel_pstate` driver with "performance" governor. Intel OneAPI HPC Toolkit (v. 2022.0.2) was used to compile the modified GAMESS.



(a) (b)

FIG. 1: The structure of adenine-thymine pair (a) and C_{60} molecule (b) used in this study. Hydrogen, carbon, nitrogen, and oxygen atoms are denoted as white, gray, blue, and red spheres respectively.

Two medium-size molecules of adenine-thymine pair and fullerene (C_{60}) were adopted for the benchmarks (see, Figure 1 and Table II). Multi-state energy calculations were performed by MRSF-TDDFT with BH&HLYP functional in combination with different basis sets of 6-31G, 6-31G(d,p), 6-311G(d,p) and cc-pVTZ.

In this work, both elapsed wall clock time (T_{Wall}), and CPU time (T_{CPU}) were measured in benchmarks. CPU time refers to the time during which the CPU cores were active doing useful work, excluding wait and I/O time. Elapsed wall clock time is calculated as the time difference between the end and the start of the calculation. Each calculation was performed at least three times and the smallest value was chosen as the benchmark value.

The contraction time (T_{C}) was estimated as the difference between the total elapsed time of

TABLE II: Model chemical systems used in benchmarks. N_{BF} is the number of basis functions.

	Adenine-thymine pair				C_{60} fullerene		
Basis set	6-31G	6-31G(d,p)	6-311G(d,p)	cc-pVTZ	6-31G	6-31G(d,p)	6-311G(d,p)
Basis function types	S, P	S, P, D	S, P, D	S, P, D, F	S, P	S, P, D	S, P, D
N_{BF}	193	340	427	830	540	900	1140
X array size, MB	0.28	0.88	1.4	5.3	2.2	6.2	9.9
`GHONDO` size, KB	2.0	10.1	10.1	78.1	2.0	10.1	10.1
Non-zero ERIs	0.5×10^8	2.9×10^8	6.4×10^8	81.7×10^8	3.1×10^9	14.2×10^9	44.2×10^9
ERI storage, GB	0.7	4.4	9.8	130.8	46.3	211.9	707.4

the ERI calculation and contraction, and the ERI calculation time. To measure the ERI calculation time, benchmarks were run on the modified source code with the contraction step manually disabled.

In the storage-based approach, ERIs are computed and stored prior to the first SCF step. They are then read at every SCF step and every iteration of Davidson's algorithm. In this study, ERI computing and ERI read time were measured separately. Due to high I/O overhead, the ERI computation time is measured as the CPU time of the corresponding step. To measure ERI reading time from storage (T_R), the contraction step was manually disabled in the source code, and the T_R was obtained as the difference between elapsed wall clock time and CPU time. The time for contraction step (T_C) in storage-based methods was measured by subtracting T_R from the elapsed time of the contraction step, measured using the non-modified code.

It should be mentioned that modern operating systems aggressively cache I/O data access using fast system memory, which can distort the benchmark results. To address this issue, the operating system I/O caches were manually flushed after all ERI calculations and before entering Davidson's algorithm subroutine to eliminate this effect. The time for OS cache flushing (T_{Flush}) was measured and added to the total time of corresponding step.

V. RESULTS AND DISCUSSION

In this section, time to compute ERIs, time to read ERIs from storage (T_{ERI}), and time to compute contraction (T_C) will be discussed. Benchmark results are presented in Tables III to IV.

TABLE III: The time (in seconds) of a single kernel contraction ($\bar{T}_C$) at the first iteration of Davidson's method for adenine-thymine pair and C_{60} benchmarks. $\bar{T}_C$ is $T_C/(n_{\text{state}} \cdot n_{\text{kernel}})$, where $n_{\text{state}} = 5$ and $n_{\text{kernel}} = 11$. The contraction schemes used include combinations of Integrals–Fock (IF), IF&Transpose (IF&T), and Fock–Integrals (FI) algorithms (see Section III C) with different buffer implementations: no buffer, B2 (2-byte integers for AO indices), and B8 (8-byte indices). All calculations were performed on a single CPU core.

	IF	FI	IF&T	FI-B8	FI-B2	IF&T-B2
Adenine-thymine pair						
6-31G	0.84	0.36	0.35	0.33	0.31	0.34
6-31G(d,p)	5.43	2.19	2.13	2.08	1.98	2.13
6-311G(d,p)	12.25	5.28	5.10	4.86	4.57	5.13
cc-pVTZ	161.53	65.78	66.46	66.80	60.98	67.27
C_{60} fullerene						
6-31G	61.97	24.17	24.82	24.97	23.38	25.04
6-31G(d,p)	274.43	117.59	112.54	121.04	112.49	114.18
6-311G(d,p)	655.52	265.48	267.27	286.54	265.46	270.15

In direct methods, the time required to obtain ERI values (T_{ERI}) for contraction is determined solely by the ERI computation time. In contrast, storage-based methods do not compute ERIs, but instead read them from the storage. Therefore, in storage-based methods, T_{ERI} corresponds to T_{R}.

We identified the proper size of the buffer for buffered versions of considered algorithms. A smaller buffer can fit into most CPU caches but suffers from higher overhead from frequent calls to `buffer_flush` subroutine (Algorithm 4, line 15). We checked several buffer sizes in the range of 10^3 to 10^7, and the optimal value was found to be 10^6, which corresponds to the 16 MB for B2 algorithm and 40 MB for B8 algorithm.

A. Comparative Performances of the Contraction Algorithms

The contraction schemes can be obtained by combining various memory access (Algorithm 2 Integrals–Fock (IF), IF&Transpose (IF&T) and Algorithm 3 Fock–Integrals (FI)) algorithms with buffering Algorithm 4 (none buffer, buffered with 2-byte integers (B2), and 8-byte integers (B8)),

as summarized in Table I. The resulting timings for each contraction scheme using a single CPU core are presented in Table III, where the average time of a single kernel contraction ($\bar{T}_C$) is reported. According to eq. (7), we compute it as $\bar{T}_C = T_C/(n_{\text{state}} \times n_{\text{kernel}}) = T_C/(5 \times 11)$ for 5 excited states and 11 MRSF kernels used in the benchmark.

In our benchmarks, algorithms with optimized memory access (FI, IF&T, and their buffered versions) outperformed the IF algorithm. Remarkably, a change of the loop order (FI algorithm) results in around 230–260% performance improvement compared to the IF algorithm for both benchmark molecules. Likewise, a change of the data layout (IF&T algorithm) gives 240–255% performance improvement.

Buffered access to integrals can further improve the contraction performance under IF, IF&T and FI algorithms. However, the improvement varies depending on the underlying contraction algorithm used. For example, one can achieve an additional 9-40% performance improvement by buffering on FI (i.e. FI-B2), making it the fastest contraction method. We also briefly remind, as was noted in Section III C, that storing ERI indices with 2-byte integers (denoted as B2 in Table I) instead of 8-byte also positively affects the performance. The reason is the 2.5x reduction of storage size per ERI entry from $4 \times 8 + 8 = 40$ bytes to $4 \times 2 + 8 = 16$ bytes, which greatly reduces the data bandwidth. On the other hand, the real impact of buffering on IF&T performance is negligible. The IF&T algorithm perhaps already has good memory bandwidth utilization, making additional performance improvements by optimized access to ERI data negligible.

Overall, the fastest algorithms to compute ERI tensor contractions are the FI algorithm with a buffer based on 2-byte integers (FI-B2) and IF&Transpose (IF&T). We performed an additional study on these two algorithms to investigate how the performance improvement depends on the number of excited states, separating them into only even or odd sets of states, as shown on the Figure 2. Here we used the performance of the original IF method at a given number of excited states as a baseline. The observed speedup of the FI-B2 algorithm (see a) and c) panels on the Figure 2) is around 2x up to the four states, which trends to $\approx$3x at a larger number of states. Most of the performance improvement observed already at $\sim$6 states. Interestingly, for $N_{\text{states}} = 16$ (as for 12 but to a lesser extent) we observe significant performance drop in several benchmarks: AT pair with 6-31G(d,p) basis set and C_{60} with 6-31G, 6-31G(d,p) and 6-311G(d,p) basis sets. This observation can be attributed to certain details of CPU cache hardware implementation, specifically, cache associativity[62]. In so-called "fully-associative" caches an entry from the system memory can be stored in any location of a cache. However, most CPU cache designs do not allow it. In this

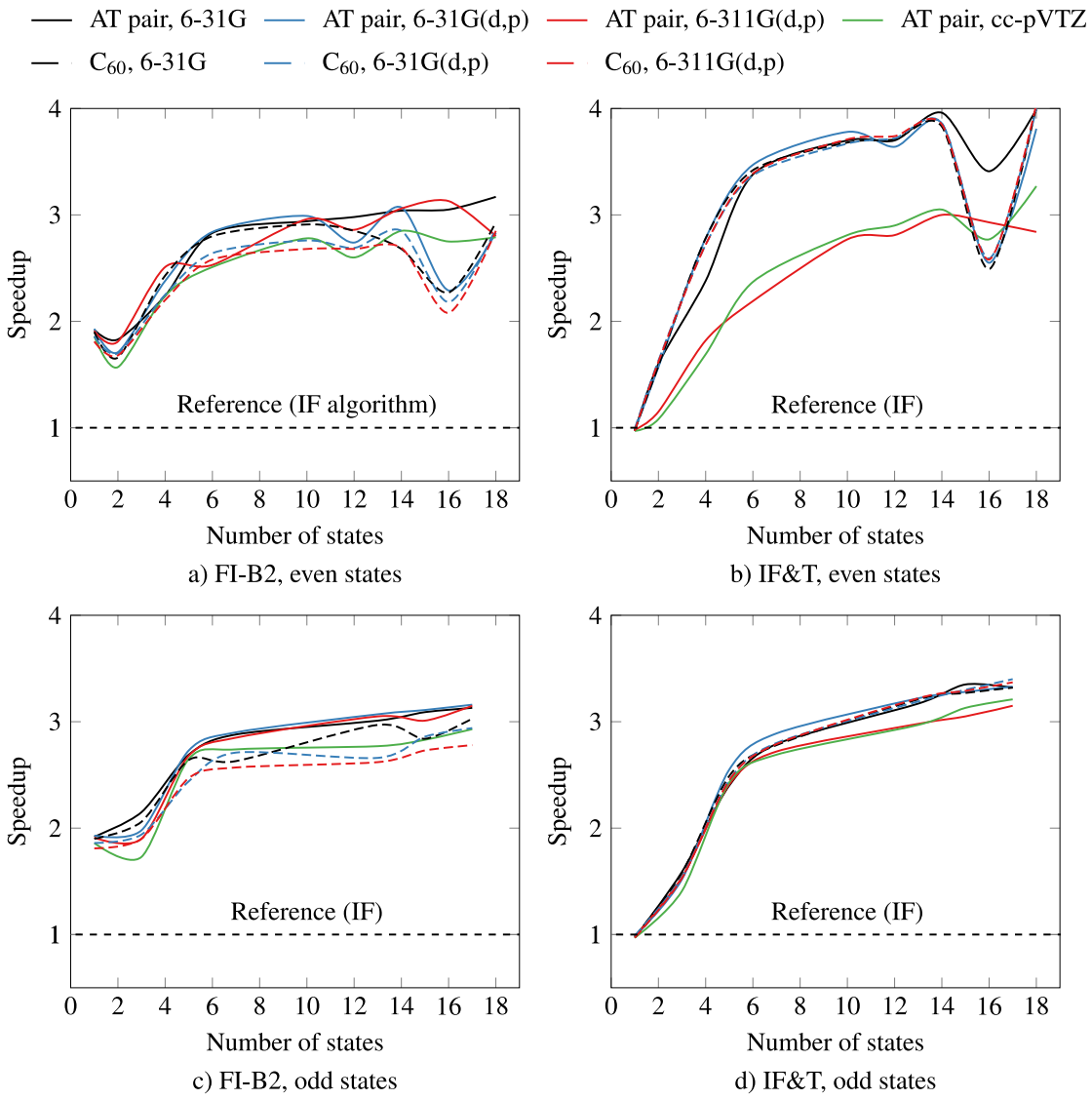


FIG. 2: The a) and c) panels show the speedup of the contraction of the 2-byte integer buffered Fock-Integrals algorithm (FI-B2), while the b) and d) panels show the speedup of the contraction of the Integrals-Fock&Transpose algorithm (IF&T), both algorithms compared to the original Integrals-Fock (IF) algorithm implementation and plotted as a function of a number of states, where a) and b) utilize only even states, but c) and d) – odd states. The solid curves correspond to AT benchmarks, while the dashed curves correspond to C_{60} fullerene, where black, blue, red and green colors denote the used 6-31G, 6-31G(d,p), 6-311G(d,p) and cc-pVTZ basis sets, respectively. See Table I and Table II for details. All benchmarks were performed on a single CPU core using direct MRSF-TD-DFT method.

case, an entry can only be stored in a few cache locations determined by the lowest few bits of its

memory address. The cache is called "n-way associative" if the number of such locations equals to n. For example, Intel CPUs used in this work have 8-way associative caches. If all these cache slots are occupied, adding a new data entry with the same lowest bits in its memory address results in the eviction of data from one of these slots, even if other cache entries are much more favorable for eviction. This leads to a highly increased cache miss rate if the data is accessed with strides equal to some power of two, which is not too small. It became an issue in some of our benchmarks for 16 states and even number of basis functions, where leading dimensions of the matrices appear to be a multiple of 2^9 bytes.

The observed performance of the IF&T algorithm varies on whether the number of excited states is even (the b) panel on the Figure 2) or odd (the d) panel on the Figure 2). In both cases, the performance grows with increasing of the number of states. However, for the odd number of states, the resulting performance is $\sim 25\%$ lower than expected based on the analysis of benchmark results with the even number of states. We attribute this observation to better data alignment with the cache line boundary in the set of benchmarks corresponding to an even number of states. This behavior is not observed for the FI-B2 algorithm because the non-ERI data is accessed non-contiguously, and data alignment is less critical. An exception is the AT benchmark with 6-311G(d,p) and cc-pVTZ basis sets (see b) panel on the Figure 2), where the benchmarks on the even number of states run slower than expected.

In addition, the parallel efficiency (PE) of the above-mentioned algorithms has been evaluated (Figure 3). By parallel efficiency, we mean the following quantity

$$\mathrm{PE}(N_{\mathrm{CPU}}) = \frac{1}{N_{\mathrm{CPU}}} \cdot \frac{T(1\ \mathrm{CPU})}{T(N_{\mathrm{CPU}})} \cdot 100\%, \tag{8}$$

where $T(N_{\mathrm{CPU}})$ is a benchmark run time when using N_{CPU} (number of CPU cores). Both FI-B2 and IF&T contraction algorithms exhibit good PE on a single node. However, the former algorithm shows slightly better PE at a larger number of states. It suggests that FI-B2 algorithm utilizes shared hardware resources, like the memory bandwidth, more efficiently than the IF&T algorithm. (Results of PE with up to 76 CPUs are presented in Supporting Information.)

B. HDD vs. SSD Performance on Storage-Based ERI

The average elapsed times of ERI calculation in storage-based ($T_{\mathrm{ERI}}^{\mathrm{S}}$) and integral-direct method ($T_{\mathrm{ERI}}^{\mathrm{D}}$), as well as ERI reading time using different types of storage: HDD ($T_{\mathrm{R}}^{\mathrm{HDD}}$), SSD ($T_{\mathrm{R}}^{\mathrm{SSD}}$)

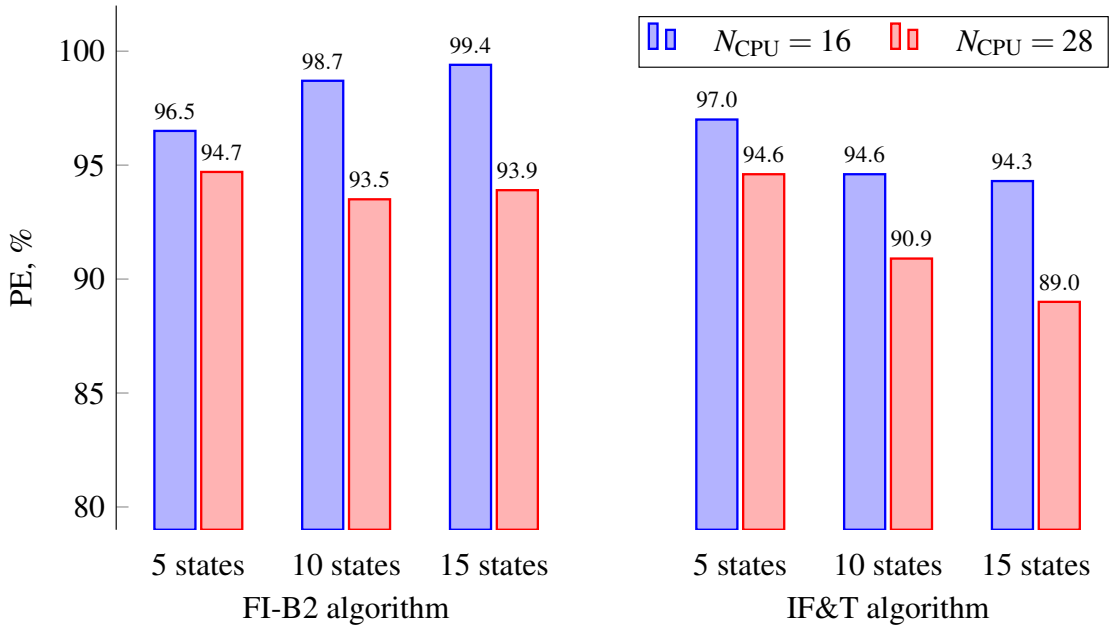


FIG. 3: Parallel efficiency (PE) of the direct 2-byte integer buffered Fock–Integrals (FI-B2) and Integrals–Fock&Transpose (IF&T) algorithms depending on the number of CPU cores (N_{CPU}); higher values are better. Benchmark: MRSF calculation with 5, 10 and 15 states with C_{60} fullerene with 6-31G(d,p) basis set ($N_{BF} = 900$).

and DRAM memory (T_R^{MEM}), are summarized in Table IV. Note, that the integral-direct approach allows to utilize additional density-based screening, resulting in much smaller number of ERIs evaluated. As a result, the corresponding time (T_{ERI}^{D}) is around 2x smaller than T_{ERI}^{S}. In general, ERI calculation step has good parallel efficiency both in integral-direct and storage-based approaches.

The ratio between time to recompute ERIs and time to read them from storage in serial is presented in Figure 4. In all benchmarks, the T_{ERI}^{S}/T_R of HDD is below one, indicating that it is always faster to recompute ERIs than use HDD to store from. In the case of SSD, the value T_{ERI}^{S}/T_R varies from 7 to 15, exhibiting a drastic improvement of I/O performance of SSD over HDD. Interestingly, the observed difference between SSD and DRAM is not very big, and is only about 2x. This difference indicates, that the serial SSD-based calculation is bound by memory latency, as the ERI data are consumed in two steps: first, the data are loaded to system memory from storage, and then, second, taken from the system memory to compute contractions. In contrast, in the in-core method, all ERIs already reside in the system memory and only one step (taking data from memory) is needed.

TABLE IV: Time of ERI calculation (T_{ERI}) and I/O (T_{R}) in seconds. For ERI calculation step, storage-based and integral-direct approaches are denoted by S and D superscripts respectively. The number of ERIs evaluated in the direct approach is less than in storage-based approach due to the density-based screening applied. For ERI read step, hard disk drive, solid-state drive, and DRAM media are denoted by HDD, SSD, and MEM superscripts respectively. Timings are presented for serial and parallel (16 cores, in parentheses) runs.

	ERI calculation		ERI reading		
	Storage ($T_{\mathrm{ERI}}^{\mathrm{S}}$)	Direct ($T_{\mathrm{ERI}}^{\mathrm{D}}$)	HDD ($T_{\mathrm{R}}^{\mathrm{HDD}}$)	SSD ($T_{\mathrm{R}}^{\mathrm{SSD}}$)	DRAM ($T_{\mathrm{R}}^{\mathrm{MEM}}$)
Adenine-thymine pair					
6-31G	3.7 (0.2)	2.2 (0.2)	5.6 (6.3)	0.3 (0.1)	0.2 (0.0)
6-31G(d,p)	20.1 (1.3)	11.5 (0.7)	41.4 (64.4)	1.8 (0.7)	1.2 (0.1)
6-311G(d,p)	40.8 (2.6)	21.3 (1.4)	104.3 (146.2)	4.2 (1.6)	2.7 (0.2)
C_{60} fullerene					
6-31G	173.8 (11.5)	104.4 (6.6)	491.5 (636.1)	21.3 (8.1)	13.3 (0.8)
6-31G(d,p)	816.4 (53.7)	429.1 (26.9)	2875.3 (3239.2)	111.4 (33.6)	N/A

Next, the parallel efficiency of the different storage media has been investigated. The performance of HDD does not scale with an increasing number of parallel processes, and the resulting ERI reading times are even worse than those for serial timings. We believe that the possible reason for this is data fragmentation. In parallel benchmark, the data is written to and read from the storage from distant non-contiguous regions of the disk. With rotational drives, this leads to pauses when repositioning the magnetic head during the read operation for each parallel process, slowing down the I/O operations. In contrast to HDD, SSD media has much better random access performance. As a result, the ERI reading time from SSD using 16 CPU cores is improved ≈ 3 times comparing to the serial run. The best, but the most expensive option is to use DRAM as a storage. Reading ERIs from system memory is the fastest approach among others both in serial and in parallel benchmarks.

An interesting observation is, that in serial run, the ERI read time from SSD is much less then a time needed to recompute all ERIs again, i.e. $T_{\mathrm{ERI}}^{\mathrm{S}} > T_{\mathrm{ERI}}^{\mathrm{D}} \gg T_{\mathrm{R}}^{\mathrm{SSD}}$ (1 CPU). The performance scaling of SSD storage used in this work is not ideal, but even with 16 CPU cores, the storage-based

ERI reading from an SSD is faster than the storage-based ERI computations. However, direct ERI calculation becomes preferable over ERI reading from SSD at a larger number of CPU cores due to advanced ERI screening together with linear parallel scaling of the ERI calculation step. In fact, the parallel performance of the SSD-based ERI calculation is no longer limited by the memory latency as in serial run, but by the bandwidth of the storage hardware[63]. To outperform the direct ERI calculation in parallel, one needs to increase the maximum bandwidth of the storage, which can be achieved, for example, by aggregating more SSD drives in a RAID array. Alternatively, SSD drive can be still effectively used as an ERI storage on the nodes with a small number of CPU cores.

C. ERI vs. Contraction Overheads

With the help of eq. (7) and the benchmark results from Table III and Table IV, the relative computational overhead of ERI calculation and contraction can be estimated. It depends on the quantum chemistry method (TD-DFT, SF-TD-DFT, MRSF-TD-DFT), contraction algorithm (IF, FI, IF&T, FI-B2, etc., see Table I), possible ERI storage type, the number of excited states (n_{state}), and the basis set. The estimated $T_{\text{C}}/T_{\text{ERI}}^{\text{D}}$ ratio is presented in Figure 5 for the case of MRSF-TDDFT as a function of n_{state}. In all algorithms the ratio scales nearly linearly with the number of states. The IF and FI-B2 algorithms show the worst and the best performance respectively in terms of the ratio. The other algorithms have performance similar to the FI-B2 algorithm. Nonetheless, not the ERI calculation, but the contraction is the major computational bottleneck with $n_{\text{state}} > 2$.

VI. CONCLUSION

Simultaneous multiple ERI tensor contractions (METC) with several density-like matrices are ubiquitous computational tasks in quantum chemistry such as multi-state TD-DFT calculations and analytical vibrational frequencies in HF or DFT methods. However, we found that as the number of contractions increases, it is not the ERI calculation that becomes the computational bottleneck, but rather the METC itself. This is an important consideration for optimizing performance. While using fast storage to save ERI values can be helpful for Hartree-Fock and KS-DFT methods, it does not provide significant advantages for METC problems, as the overhead of ERI calculation is relatively small, particularly in the case of MRSF-TDDFT.

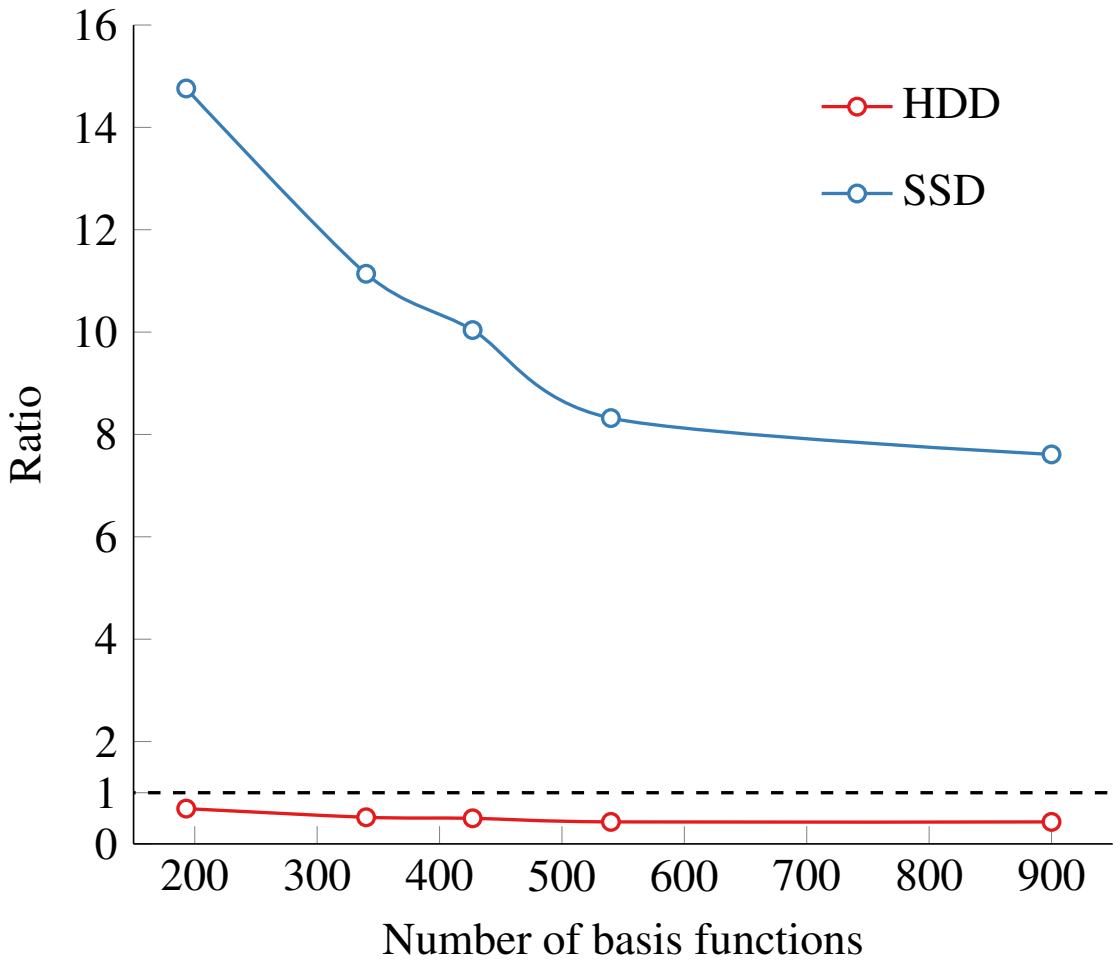


FIG. 4: The ratio between the time of computing and reading all elements of ERI tensor using HDD and SSD media ($T^{\mathrm{S}}_{\mathrm{ERI}}/T_{\mathrm{R}}$) as a function of the number of basis functions (N_{BF}). Benchmarks: storage-based MRSF-TDDFT calculation, adenine-thymine pair with 6-31G ($N_{\mathrm{BF}} = 193$), 6-31G(d,p) ($N_{\mathrm{BF}} = 340$), 6-311G(d,p) ($N_{\mathrm{BF}} = 427$), and C_{60} fullerene with 6-31G ($N_{\mathrm{BF}} = 540$), 6-31G(d,p) ($N_{\mathrm{BF}} = 900$).

In the current paper, several high-performance strategies for efficient contractions were proposed and implemented for the MRSF-TDDFT method. Optimal approaches involve either the data layout reformation of the interim densities and the Fock matrices, or ERI buffering with loop order optimization for data access locality. Remarkably, these techniques improve the performance of the MRSF-TDDFT method by 300% as compared to the original generic algorithm. The parallel efficiencies are fairly close to 100% with 16 cores regardless of the contraction algorithm. The newly implemented METC algorithms can be also applied for similar cases such as close-shell TD-DFT, open-shell TD-DFT, or other methods. As the identified METC frameworks are mostly

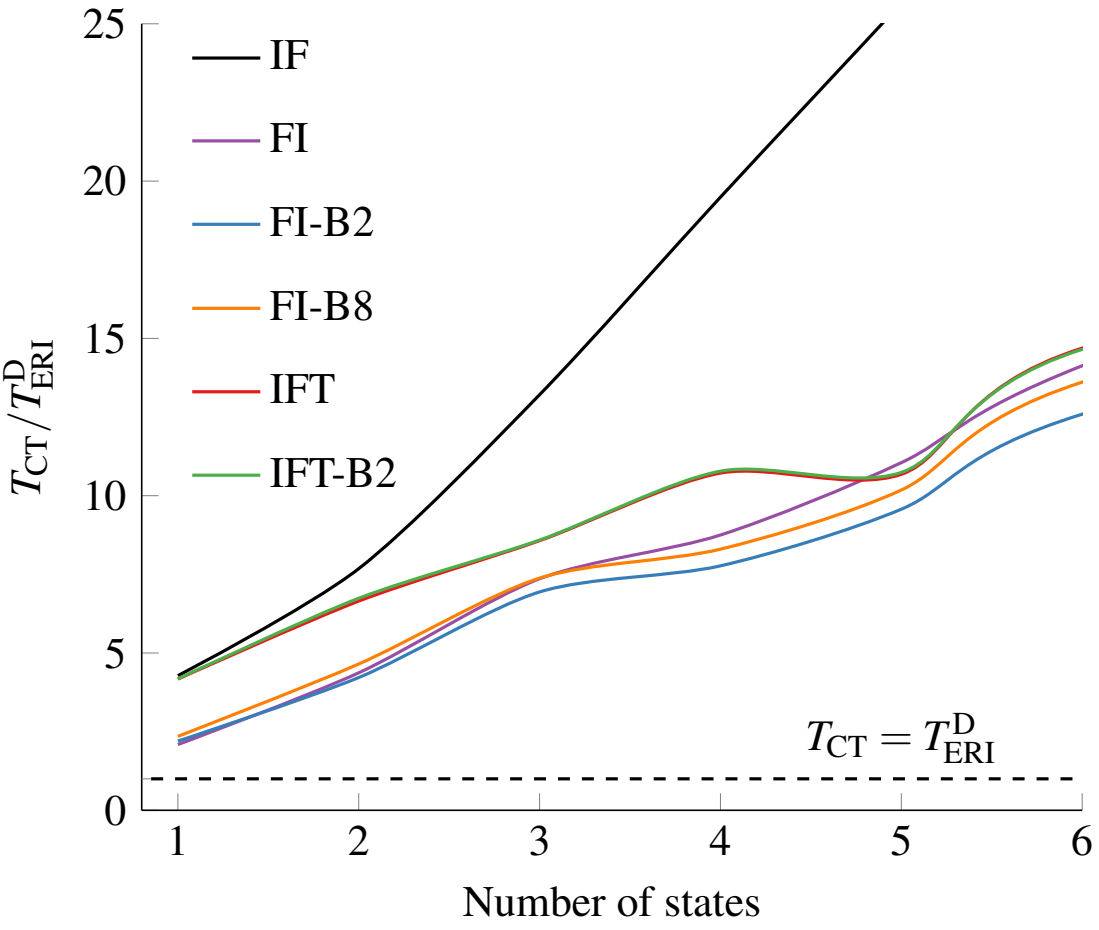


FIG. 5: The ratio between the time to compute ERI contractions and direct ERI values calculation (T_{CT}/T^{D}_{ERI}) using different contraction methods. See Table I for details. Benchmark description: direct MRSF-TDDFT calculation on a single CPU core, adenine-thymine pair, 6-311G(d,p) basis set.

composed of matrix multiplicative operations, it would be ideal for GPU acceleration, which shall be explored in the future.

SUPPLEMENTARY MATERIAL

A separate supplementary file containing additional parallel benchmarks, detailed timings, Cartesian coordinates, and synthetic benchmarks with pseudo-code is provided.

ACKNOWLEDGMENTS

This work was supported by the NRF funded by the Ministry of Science and ICT (2020R1A2C2008246 and 2020R1A5A1019141) Work by S.L. was funded by the US Department of Energy, Office of Science, via Award DE-SC0019374. BQP and MSG acknowledge the support of the US Department of Energy GAMESS Exascale Computing Project, to the Ames Laboratory, Project Number: 17-SC-20-SC. The Ames National Laboratory is operated by Iowa State University under Contract No. DE-AC02-07CH11338.

AVAILABILITY OF DATA

The data that supports the findings of this study are available within the article and its supplementary material.

REFERENCES

[1]G. J. Hedley, A. Ruseckas, and I. D. Samuel, "Light harvesting for organic photovoltaics," Chemical Reviews **117**, 796–837 (2017).

[2]A. Endo, M. Ogasawara, A. Takahashi, D. Yokoyama, Y. Kato, and C. Adachi, "Thermally activated delayed fluorescence from Sn^{4+}–porphyrin complexes and their application to organic light emitting diodes—A novel mechanism for electroluminescence," Advanced Materials **21**, 4802–4806 (2009).

[3]E. Runge and E. K. Gross, "Density-functional theory for time-dependent systems," Physical review letters **52**, 997 (1984).

[4]M. E. Casida and D. Chong, "Recent advances in density functional methods," Computational Chemistry: Reviews of Current Trends (1995).

[5]M. E. Casida and M. Huix-Rotllant, "Progress in time-dependent density-functional theory," Annual review of physical chemistry **63**, 287–323 (2012).

[6]R. McWeeny and B. Sutcliffe, *Methods of Molecular Quantum Mechanics* (Academic Press, 1992).

[7]A. Dreuw, J. L. Weisman, and M. Head-Gordon, "Long-range charge-transfer excited states in time-dependent density functional theory require non-local exchange," The Journal of chemical physics **119**, 2943–2946 (2003).

[8]A. Dreuw and M. Head-Gordon, "Single-reference ab initio methods for the calculation of excited states of large molecules," Chemical reviews **105**, 4009–4037 (2005).

[9]P. Dev, S. Agrawal, and N. J. English, "Determining the appropriate exchange-correlation functional for time-dependent density functional theory studies of charge-transfer excitations in organic dyes," The Journal of Chemical Physics **136**, 224301 (2012).

[10]N. T. Maitra, "Undoing static correlation: Long-range charge transfer in time-dependent density-functional theory," The Journal of chemical physics **122**, 234104 (2005).

[11]E. Baerends, O. Gritsenko, and R. Van Meer, "The Kohn–Sham gap, the fundamental gap and the optical gap: the physical meaning of occupied and virtual Kohn–Sham orbital energies," Physical Chemistry Chemical Physics **15**, 16408–16425 (2013).

[12]R. J. Cave, F. Zhang, N. T. Maitra, and K. Burke, "A dressed TDDFT treatment of the 2^1A_g states of butadiene and hexatriene," Chemical Physics Letters **389**, 39–42 (2004).

[13]J. Neugebauer, E. J. Baerends, and M. Nooijen, "Vibronic coupling and double excitations in linear response time-dependent density functional calculations: Dipole-allowed states of N_2," The Journal of chemical physics **121**, 6155–6166 (2004).

[14]N. T. Maitra, F. Zhang, R. J. Cave, and K. Burke, "Double excitations within time-dependent density functional theory linear response," The Journal of Chemical Physics **120**, 5932–5937 (2004).

[15]F. Aryasetiawan, O. Gunnarsson, and A. Rubio, "Excitation energies from time-dependent density-functional formalism for small systems," EPL (Europhysics Letters) **57**, 683 (2002).

[16]M. Filatov, "Ensemble DFT approach to excited states of strongly correlated molecular systems," Density-functional methods for excited states , 97–124 (2015).

[17]B. G. Levine, C. Ko, J. Quenneville, and T. J. Martínez, "Conical intersections and double excitations in time-dependent density functional theory," Molecular Physics **104**, 1039–1051 (2006).

[18]M. Huix-Rotllant, M. Filatov, S. Gozem, I. Schapiro, M. Olivucci, and N. Ferré, "Assessment of density functional theory for describing the correlation effects on the ground and excited state potential energy surfaces of a retinal chromophore model," Journal of chemical theory and computation **9**, 3917–3932 (2013).

[19]S. Gozem, F. Melaccio, A. Valentini, M. Filatov, M. Huix-Rotllant, N. Ferré, L. M. Frutos, C. Angeli, A. I. Krylov, A. A. Granovsky, *et al.*, "Shape of multireference, equation-of-motion coupled-cluster, and density functional theory potential energy surfaces at a conical intersection," Journal of chemical theory and computation **10**, 3074–3084 (2014).

[20]N. Ferré, M. Filatov, M. Huix-Rotllant, and C. Adamo, *Density-functional methods for excited states*, Vol. 368 (Springer, 2016).

[21]W. Park, J. Shen, S. Lee, P. Piecuch, M. Filatov, and C. H. Choi, "Internal Conversion between Bright ($1^1B_u^+$) and Dark ($2^1A_g^-$) States in s-trans-Butadiene and s-trans-Hexatriene," J. Phys. Chem. Lett. **12**, 9720–9729 (2021).

[22]Z. Li and W. Liu, "Theoretical and numerical assessments of spin-flip time-dependent density functional theory," The Journal of chemical physics **136**, 024107 (2012).

[23]F. Wang and T. Ziegler, "Time-dependent density functional theory based on a noncollinear formulation of the exchange-correlation potential," The Journal of chemical physics **121**, 12191–12196 (2004).

[24]Y. Shao, M. Head-Gordon, and A. I. Krylov, "The spin–flip approach within time-dependent density functional theory: Theory and applications to diradicals," The Journal of chemical physics **118**, 4807–4818 (2003).

[25]J. S. Sears, C. D. Sherrill, and A. I. Krylov, "A spin-complete version of the spin-flip approach to bond breaking: What is the impact of obtaining spin eigenfunctions?" The Journal of chemical physics **118**, 9084–9094 (2003).

[26]J. Mato and M. S. Gordon, "A general spin-complete spin-flip configuration interaction method," Physical Chemistry Chemical Physics **20**, 2615–2626 (2018).

[27]J. Mato and M. S. Gordon, "Analytic non-adiabatic couplings for the spin-flip ormas method," Physical Chemistry Chemical Physics **22**, 1475–1484 (2020).

[28]M. E. Casida, "Propagator corrections to adiabatic time-dependent density-functional theory linear response theory," The Journal of chemical physics **122**, 054111 (2005).

[29]X. Zhang and J. M. Herbert, "Spin-flip, tensor equation-of-motion configuration interaction with a density-functional correction: A spin-complete method for exploring excited-state potential energy surfaces," The Journal of Chemical Physics **143**, 234107 (2015), https://doi.org/10.1063/1.4937571.

[30]S. Lee, M. Filatov, S. Lee, and C. H. Choi, "Eliminating spin-contamination of spin-flip time dependent density functional theory within linear response formalism by the use of zeroth-order

mixed-reference (MR) reduced density matrix," J. Chem. Phys. **149**, 104101 (2018).

[31]S. Lee, E. E. Kim, H. Nakata, S. Lee, and C. H. Choi, "Efficient implementations of analytic energy gradient for mixed-reference spin-flip time-dependent density functional theory (MRSF-TDDFT)," J. Chem. Phys. **150**, 184111 (2019).

[32]S. Lee, S. Shostak, M. Filatov, and C. H. Choi, "Conical intersections in organic molecules: Benchmarking mixed-reference spin-flip time-dependent dft (MRSF-TD-DFT) vs spin-flip TD-DFT," J. Phys. Chem. A **123**, 6455–6462 (2019).

[33]Y. Horbatenko, S. Sadiq, S. Lee, M. Filatov, and C. H. Choi, "Mixed-reference spin-flip time-dependent density functional theory (MRSF-TDDFT) as a simple yet accurate method for diradicals and diradicaloids," J. Chem. Theory Comput. **17**, 848–859 (2021).

[34]S. Lee, E. Kim, S. Lee, and C. H. Choi, "Fast overlap evaluations for nonadiabatic molecular dynamics simulations: Applications to SF-TDDFT and TDDFT," J. Chem. Theory Comput. **15**, 882 (2019).

[35]Y. Horbatenko, S. Lee, M. Filatov, and C. H. Choi, "Performance analysis and optimization of mixed-reference spin-flip time-dependent density functional theory (MRSF-TDDFT) for vertical excitation energies and singlet–triplet energy gaps," J. Phys. Chem. A **123**, 7991 (2019).

[36]S. Lee, Y. Horbatenko, M. Filatov, and C. H. Choi, "Fast and accurate computation of nonadiabatic coupling matrix elements using the truncated Leibniz formula and mixed-reference spin-flip time-dependent density functional theory," J. Phys. Chem. Lett. **12**, 4722–4728 (2021).

[37]Y. S. Baek, S. Lee, M. Filatov, and C. H. Choi, "Optimization of three state conical intersections by adaptive penalty function algorithm in connection with the mixed-reference spin-flip time-dependent density functional theory method (MRSF-TDDFT)," J. Phys. Chem. A **125**, 1994–2006 (2021).

[38]V. Pomogaev, S. Lee, S. Shaik, M. Filatov, and C. H. Choi, "Exploring Dyson's orbitals and their electron binding energies for conceptualizing excited states from response methodology," J. Phys. Chem. Lett. **12**, 9963–9972 (2021).

[39]Y. Horbatenko, S. Lee, M. Filatov, and C. H. Choi, "How beneficial is the explicit account of doubly-excited configurations in linear response theory?" J. Chem. Theory Comput. **17**, 975–984 (2021).

[40]H. Kim, W. Park, Y. Kim, M. Filatov, C. H. Choi, and D. Lee, "Relief of excited-state antiaromaticity enables the smallest red emitter," Nat. Commun. **12**, 1–9 (2021).

[41]W. Park, S. Lee, M. Huix-Rotllant, M. Filatov, and C. H. Choi, "Impact of the dynamic electron correlation on the unusually long excited-state lifetime of thymine," J. Phys. Chem. Lett. **12**, 4339–4346 (2021).

[42]S. Lee, W. Park, H. Nakata, M. Filatov, and C. H. Choi, "Recent advances in ensemble density functional theory and linear response theory for strong correlation," Bull. Korean Chem. Soc. **43**, 17–34 (2022).

[43]W. Park, J. Shen, S. Lee, P. Piecuch, T. Joo, M. Filatov, and C. H. Choi, "Dual fluorescence of octatetraene hints at a novel type of singlet-to-singlet thermally activated delayed fluorescence process," J. Phys. Chem. C (2022).

[44]W. Park, M. Filatov, S. Sadiq, I. Gerasimov, S. Lee, T. Joo, and C. H. Choi, "A plausible mechanism of uracil photohydration involves an unusual intermediate," J. Phys. Chem. Lett. **13**, 7072–7080 (2022).

[45]M. Huix-Rotllant, K. Schwinn, V. Pomogaev, M. Farmani, N. Ferré, S. Lee, and C. H. Choi, "Photochemistry of thymine in solution and dna revealed by an electrostatic embedding qm/mm combined with mixed-reference spin-flip tddft," Journal of Chemical Theory and Computation (2022).

[46]S. Lee, W. Park, H. Nakata, M. Filatov, and C. H. Choi, "Mixed-reference spin-flip time-dependent density functional theory (mrsf-tddft) as a method of choice for nonadiabatic molecular dynamics," in *Time-Dependent Density Functional Theory* (Jenny Stanford Publishing, 2023) pp. 101–139.

[47]S. Shostak, W. Park, J. Oh, J. Kim, S. Lee, H. Nam, M. Filatov, D. Kim, and C. H. Choi, "Ultrafast excited state aromatization in dihydroazulene," Journal of the American Chemical Society (2023).

[48]S. Sadiq, W. Park, V. Mironov, S. Lee, M. Filatov, and C. H. Choi, "Prototropically controlled dynamics of cytosine photodecay," The Journal of Physical Chemistry Letters **14**, 791–797 (2023).

[49]K. Komarov, W. Park, S. Lee, T. Zeng, and C. H. Choi, "Accurate spin–orbit coupling by relativistic mixed-reference spin-flip-tddft," Journal of Chemical Theory and Computation (2023).

[50]A. Japahuge, S. Lee, C. H. Choi, and T. Zeng, "Design of singlet fission chromophores with cyclic (alkyl)(amino) carbene building blocks," J. Chem. Phys. **150**, 234306 (2019).

[51]E. Pradhan, S. Lee, C. H. Choi, and T. Zeng, "Diboron-and diaza-doped anthracenes and phenanthrenes: their electronic structures for being singlet fission chromophores," J. Phys.

Chem. A **124**, 8159–8172 (2020).

[52]D. James, E. Pradhan, S. Lee, C. H. Choi, and T. Zeng, "Dicarbonyl anthracenes and phenanthrenes as singlet fission chromophores," Can. J. Chem. **99**, 1–10 (2022).

[53]Z. Rinkevicius, X. Li, O. Vahtras, K. Ahmadzadeh, M. Brand, M. Ringholm, N. H. List, M. Scheurer, M. Scott, A. Dreuw, *et al.*, "Veloxchem: A python-driven density-functional theory program for spectroscopy simulations in high-performance computing environments," Wiley Interdisciplinary Reviews: Computational Molecular Science **10**, e1457 (2020).

[54]D. R. Rehn, Z. Rinkevicius, M. F. Herbst, X. Li, M. Scheurer, M. Brand, A. L. Dempwolff, I. E. Brumboiu, T. Fransson, A. Dreuw, *et al.*, "Gator: A python-driven program for spectroscopy simulations using correlated wave functions," Wiley Interdisciplinary Reviews: Computational Molecular Science **11**, e1528 (2021).

[55]S. Hirata and M. Head-Gordon, "Time-dependent density functional theory within the tamm–dancoff approximation," Chemical Physics Letters **314**, 291–299 (1999).

[56]J. Olsen, H. J. A. Jensen, and P. Jørgensen, "Solution of the large matrix equations which occur in response theory," Journal of Computational Physics **74**, 265–282 (1988).

[57]R. E. Stratmann, G. E. Scuseria, and M. J. Frisch, "An efficient implementation of time-dependent density-functional theory for the calculation of excitation energies of large molecules," The Journal of chemical physics **109**, 8218–8224 (1998).

[58]M. Häser and R. Ahlrichs, "Improvements on the direct SCF method," Journal of Computational Chemistry **10**, 104–111 (1989).

[59]J. Almlöf and P. R. Taylors, "Computational aspects of direct SCF and MCSCF methods," in *Advanced Theories and Computational Approaches to the Electronic Structure of Molecules* (Springer, 1984) pp. 107–125.

[60]J. Almlöf, K. Faegri, and K. Korsell, "Principles for a direct SCF approach to LCAO-MO ab-initio calculations," Journal of Computational Chemistry **3**, 385–399 (1982).

[61]G. M. J. Barca, C. Bertoni, L. Carrington, D. Datta, N. De Silva, J. E. Deustua, D. G. Fedorov, J. R. Gour, A. O. Gunina, E. Guidez, T. Harville, S. Irle, J. Ivanic, K. Kowalski, S. S. Leang, H. Li, W. Li, J. J. Lutz, I. Magoulas, J. Mato, V. Mironov, H. Nakata, B. Q. Pham, P. Piecuch, D. Poole, S. R. Pruitt, A. P. Rendell, L. B. Roskop, K. Ruedenberg, T. Sattasathuchana, M. W. Schmidt, J. Shen, L. Slipchenko, M. Sosonkina, V. Sundriyal, A. Tiwari, J. L. Galvez Vallejo, B. Westheimer, M. Wloch, P. Xu, F. Zahariev, and M. S. Gordon, "Recent developments in the general atomic and molecular electronic structure system," The Journal of Chemical Physics

152, 154102 (2020).

[62]C. Hong, W. Bao, A. Cohen, S. Krishnamoorthy, L.-N. Pouchet, F. Rastello, J. Ramanujam, and P. Sadayappan, "Effective padding of multidimensional arrays to avoid cache conflict misses," ACM SIGPLAN Notices **51**, 129–144 (2016).

[63]V. Mironov, A. Moskovsky, and Y. Alexeev, "Power measurements of Hartree-Fock algorithms using different storage devices," in *2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (2017) pp. 1004–1011.