CDFCI: High-Performance Parallel Software for Many-Body Large-Scale Eigenvalue Problems

Yuejia Zhang[1], Zhe Wang[2], Jianfeng Lu[2,3,4], and Yingzhou Li[1,5,6]

[1]School of Mathematical Sciences, Fudan University, Shanghai, China
[2]Department of Mathematics, Duke University, Durham, NC, USA
[3]Department of Physics, Duke University, Durham, NC, USA
[4]Department of Chemistry, Duke University, Durham, NC, USA
[5]Shanghai Key Laboratory for Contemporary Applied Mathematics, Shanghai, China
[6]Key Laboratory of Computational Physical Sciences, Ministry of Education, China

Abstract

CDFCI is a shared-memory parallel numerical program for computing low-lying eigenpairs of large-scale, non-relativistic fermionic Hamiltonians. The software is designed to handle a broad class of many-body quantum models, including both *ab initio* electronic structure Hamiltonians and lattice-based Hamiltonians arising in condensed matter physics. CDFCI combines an efficient coordinate-descent-based selected configuration interaction algorithm with dedicated parallelization strategies, achieving high performance on modern multi-core architectures. Benchmark results on representative quantum chemistry and condensed matter test cases demonstrate that CDFCI attains state-of-the-art accuracy with competitive performance compared to established selected configuration interaction (such as CIPSI or SHCI) and DMRG implementations. The software is open-source, extensively documented, and provides a Python interface for seamless integration with PySCF and other many-body simulation workflows.

Keywords: Full Configuration Interaction, many-body eigenvalue problems, large-scale eigenpair computation, ab initio electronic structure, quantum chemistry algorithms, condensed matter physics models, high-performance computing, shared-memory parallelization

1 Introduction

CDFCI is a software package designed to provide approximate numerical solutions to the fermionic, time-independent, non-relativistic many-body Schrödinger equation. This class of problems underpins computational physics and chemistry, enabling the study of reactivity, spectroscopy, and functional properties of molecular and condensed systems. For most practically relevant problems, the central task is to determine the low-lying eigenvalues and eigenfunctions of the Hamiltonian operator that describes the total energy of the system. It is, however, intrinsically difficult to solve the many-body problem, because neither analytic solutions exist nor are numerical methods tractable, due to the exponential growth of problem size with the number of particles. There are system-level simplifications, including the Hartree–Fock (HF) method and Density Functional Theory (DFT). The former uses a mean-field approximation of particle interactions and reduces the problem to an effective single-particle one, while the latter directly works with the electron density instead of the wavefunction. In both cases, correlation effects are not treated explicitly,

which could lead to inaccurate results for strongly correlated systems. Thus, methods that preserve the many-body wavefunction, also called wavefunction-based methods, remain crucial for treating strong correlation and are the focus of this paper.

One typical wavefunction-based method, known as full configuration interaction (FCI) in quantum chemistry, refers to the process of expanding the many-body wavefunction into a linear combination of many-body basis functions. These many-body basis functions are constructed as antisymmetric tensor products of one-body basis functions, with the aim of enforcing the Pauli exclusion principle. They describe the occupation state of particles and are referred to as configurations throughout this paper. With a given truncated one-body basis set, the FCI discretization procedure reduces the problem to a linear eigenvalue problem whose dimension equals that of the Hilbert space spanned by all possible configurations. Solving such large, sparse and symmetric eigenvalue problems is standard practice in numerical linear algebra, and iterative methods such as the Davidson algorithm [1] have been widely employed to compute the ground state and a few low-lying excitations. Nevertheless, the exponential growth of the Hilbert space soon renders these established techniques impractical for realistic systems. A variety of specialized approaches have therefore been developed, which are commonly divided into categories including selected configuration interaction with perturbation theory, quantum Monte Carlo, density matrix renormalization group and others.

Perturbation theory (PT) inspires the splitting of the Hamiltonian into a solvable reference term and the rest as a perturbation. Correlation effects can be captured at modest additional cost via low-order corrections (e.g., MP2/MPn), although accuracy depends sensitively on the choice of the reference Hamiltonian and the weakness of the perturbation [2]. A classic method of this type in electronic structure calculations is configuration interaction singles-doubles (CISD), where the reference space includes the Hartree–Fock state and all single- and double-excitation states built from it [3]. In the meantime, selected configuration interaction (selected CI) methods emerged, constructing the solution space by adding determinants with the largest contributions iteratively, typically guided by a second-order perturbative estimate (PT2). From the seminal CIPSI approach [4], the selected CI framework has evolved into a broad family of methods, including more sophisticated schemes such as adaptive sampling CI [5, 6], heat-bath CI [7, 8], semistochastic heat-bath CI [9, 10] that incorporate semistochastic PT2 corrections, iterative CI with selection [11], fast randomized iteration method for FCI [12], reinforcement learning CI [13], together with large-scale parallel and GPU-accelerated software implementations [14, 15, 16, 17].

Quantum Monte Carlo (QMC) methods offer another approach for sampling the wavefunction, but suffer from the notorious fermion sign problem [18]. The full configuration interaction quantum Monte Carlo (FCIQMC) method introduced by Booth et al. [19] mitigates this issue by employing a population dynamics algorithm to evolve a set of walkers in the configuration space [20], and further enhancements, such as the initiator method [21] and semi-stochastic projector techniques [22, 23], significantly improve its efficiency and robustness across a wide range of molecular and condensed-phase systems [24, 25]. Density matrix renormalization group (DMRG), on the other hand, reformulates the problem in terms of matrix product states, enabling efficient treatment of one-dimensional or quasi-one-dimensional systems, and has been successfully adapted to *ab initio* quantum chemistry via suitable orbital orderings and active spaces [26, 27, 28, 29]. Efficient implementations of the DMRG method have achieved cutting-edge performance results and nearly ideal parallel scaling on high-performance computing platforms [30, 31, 32]. Finally, many other approximation theories and methods have been developed, such as the FCC reduction method combining full coupled cluster theory (FCC) with FCI solutions [33], and the many-body expansion FCI approaches (MBE-FCI) based on low-order truncations of many-body expansions [34, 35].

This paper introduces the software package CDFCI (short for Coordinate Descent Full Configu-

ration Interaction), which implements a series of methods based on the core idea of transforming the original eigenvalue problem into an equivalent unconstrained optimization problem, and employing a customized coordinate gradient descent framework. Each coordinate corresponds to a configuration, or, more fundamentally, a basis function in Hilbert space. Our method is an iterative approach that progressively expands the variational space — selectively including coordinates with high contributions based on the magnitude of their gradient entries. Compared to the aforementioned selected CI methods, CDFCI selects coordinates based on gradients rather than perturbation estimates. The convergence of the method is theoretically guaranteed under standard assumptions from optimization theory. At the end of all iterations, the lower spectrum of the Hamiltonian is estimated in a threshold-controlled subspace, which can be further corrected when combined with perturbation theory.

The original CDFCI method [36], proposed by Wang, Li, and Lu in 2019, was used to solve for the ground state in electronic structure calculations. With theoretical convergence guarantees [37], the algorithm follows a coordinate descent framework and performs an exact line search. Later, Wang et al. proposed the xCDFCI method [38], which modified the objective function to also target low-lying excited states, while the similar coordinate descent routine was employed. In a series of works surrounding CDFCI, mCDFCI [39] extended the single coordinate descent iteration to multiple coordinates. The improvement of the parallel efficiency owes to increasing workloads per step, which makes shared-memory parallelization more flexible and balanced on multi-core machines. Another related work is OptOrbFCI [40], proposed by Li et al., which allows one-electron orbitals to be further rotated or compressed. The compression of orbitals in the outer loop is combined with the CDFCI iteration in the inner loop, thereby approximating optimal solutions in the variational space within limited memory budget, similar to ideas of complete active space self-consistent field (CASSCF) methods [41].

The rest of the paper is organized as follows. In Section 2, we introduce the formalism of the time-independent Schrödinger equation under discretized basis set for fermions and the reformulation of the eigenvalue problem. The coordinate descent framework and all numerical methods incorporated in the current CDFCI package can be found in Section 3. Following this, Section 4 is dedicated to the practical side, including implementations and software designs, with an illustrative example to demonstrate the basic usage. Finally, Section 5 presents up-to-date numerical results conducted on multi-core machines, showcasing the accuracy and performance of the software, and Section 6 concludes this paper.

2 Problem Formulation

This section presents the theoretical and algorithmic foundations of our approach. We begin by defining the discretized variational space that we consider for the time-independent Schrödinger equation in Section 2.1, and reviewing the formulation of fermionic Hamiltonians in the second quantization formalism in Section 2.2, along with typical examples in electronic structure and lattice models. This provides the necessary physical and mathematical background, before we finally present the problem setup in Section 2.3, including the core idea of reformulating the eigenvalue problem to an optimization problem.

2.1 FCI Variational Space

In quantum mechanics, the Hamiltonian operator $\hat{\boldsymbol{H}}$ represents the total energy of a quantum system, comprising both kinetic and potential contributions. Its spectrum determines many fun-

damental properties of the system, as captured by the time-independent Schrödinger equation,

$$\hat{\boldsymbol{H}}\Psi = E\Psi. \tag{1}$$

Here, Ψ denotes the many-body wavefunction representing an eigenstate of the system with eigenvalue E. Let n be the number of fermions. According to quantum mechanics, $\Psi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) : (\mathbb{R}^3 \times \{\uparrow, \downarrow\})^n \to \mathbb{C}$ is a complex-valued function of the spatial and spin coordinates of the n fermions and Ψ belongs to a complex Hilbert space $\mathcal{H}$. We restrict attention to the antisymmetric subspace of $\mathcal{H}$

$$\mathcal{H}_{\text{sub}} = \bigwedge^{n} L^2(\mathbb{R}^3 \times \{\uparrow, \downarrow\}, \mathbb{C}) \tag{2}$$

which both simplifies the computations and enforces fermionic antisymmetry.

Let $\{\phi_i\}_{i\in\mathbb{N}}$ be a countable orthonormal basis of $L^2(\mathbb{R}^3 \times \{\uparrow, \downarrow\}, \mathbb{C})$. In practical computations, we restrict to a finite subset $\{\phi_1, \ldots, \phi_N\}$. These single-particle basis functions are referred to as spin–orbitals in electronic structure theory and as localized site states in lattice models. Exploiting the isomorphism $L^2(\mathbb{R}^3 \times \{\uparrow, \downarrow\}, \mathbb{C}) \cong L^2(\mathbb{R}^3, \mathbb{C}) \otimes \mathbb{C}^2$, one distinguishes spatial orbitals, which are functions in $L^2(\mathbb{R}^3, \mathbb{C})$ that do not incorporate the spin degree of freedom.

The space $\mathcal{H}_{\text{sub}}$ is conventionally termed the full configuration interaction (FCI) space in electronic structure calculations, with dimension $\binom{N}{n} =: N_{\text{FCI}}$. It is alternatively called the variational space because the associated problem is formulated within a variational framework. A canonical orthonormal basis of $\mathcal{H}_{\text{sub}}$ is furnished by the Slater determinants $\{\Phi_{i_1,\cdots,i_n} = \phi_{i_1} \wedge \cdots \wedge \phi_{i_n} : 1 \leq i_1 < \cdots < i_n \leq N\}$.

2.2 Hamiltonian Representations

Working in Fock space, it is natural to adopt the second quantization formalism, in which the Hamiltonian admits a compact algebraic form: a general fermionic Hamiltonian can be written as a linear combination of ladder operators that obey anti-commutation relations. We consider the Hamiltonian of the following form:

$$\hat{\boldsymbol{H}} = \sum_{p,q}^{N} h_{pq} \hat{a}_p^\dagger \hat{a}_q + \frac{1}{2} \sum_{p,q,r,s}^{N} v_{pqrs} \hat{a}_p^\dagger \hat{a}_q^\dagger \hat{a}_s \hat{a}_r, \tag{3}$$

where $\hat{a}_p^\dagger$ and $\hat{a}_q$ are fermionic creation and annihilation operators that satisfy $\{\hat{a}_p, \hat{a}_q^\dagger\} = \delta_{pq}$, in which $\{\cdot, \cdot\}$ denotes the anticommutator, i.e., $\{\hat{A}, \hat{B}\} = \hat{A}\hat{B} + \hat{B}\hat{A}$. Although the Hamiltonian may, in principle, contain higher-body interactions, in this work we focus exclusively on one- and two-body terms, which is sufficient to describe non-relativistic fermionic Hamiltonians.

Two widely studied examples of this general form (3) illustrate its versatility. The first is the *ab initio* electronic Hamiltonian under the Born–Oppenheimer approximation, where h_{pq} and v_{pqrs} are referred to as one- and two-body integrals respectively,

$$h_{pq} = \int \mathrm{d}\boldsymbol{r}_1 \, \psi_p^\star(\boldsymbol{r}_1) \hat{\boldsymbol{h}}(\boldsymbol{r}_1) \psi_q(\boldsymbol{r}_1), \quad p, q = 1, \ldots, N, \tag{4}$$

$$v_{pqrs} = \int \mathrm{d}\boldsymbol{r}_1 \mathrm{d}\boldsymbol{r}_2 \, \psi_p^\star(\boldsymbol{r}_1) \psi_q^\star(\boldsymbol{r}_2) \hat{\boldsymbol{v}}(\boldsymbol{r}_1, \boldsymbol{r}_2) \psi_r(\boldsymbol{r}_1) \psi_s(\boldsymbol{r}_2). \quad p, q, r, s = 1, \ldots, N. \tag{5}$$

The one-body operator $\hat{\boldsymbol{h}}(\boldsymbol{r}_1)$ and the two-body operator $\hat{\boldsymbol{v}}(\boldsymbol{r}_1, \boldsymbol{r}_2)$ take in the coordinates of one

electron and two electrons respectively:

$$\hat{\boldsymbol{h}}(\boldsymbol{r}_1) = -\frac{1}{2}\nabla^2_{\boldsymbol{r}_1} - \sum_{A=1}^{N_{\text{nuc}}} \frac{Z_A}{\|\boldsymbol{r}_1 - \boldsymbol{r}_A\|}, \tag{6}$$

$$\hat{\boldsymbol{v}}(\boldsymbol{r}_1, \boldsymbol{r}_2) = \frac{1}{\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|}, \tag{7}$$

with N_{nuc} as the number of nuclei in the system, and Z_A as the charge number of the nucleus A.

The second example is the Hubbard Hamiltonian,

$$\hat{\boldsymbol{H}} = -t \sum_{\langle i,j\rangle,\sigma} (\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma} + \text{h.c.}) + U \sum_i \hat{n}_{i\uparrow}\hat{n}_{i\downarrow}, \tag{8}$$

where $\langle i, j\rangle$ denotes neighboring sites i and j on the lattice, and $\sigma \in \{\uparrow, \downarrow\}$ is the spin index. The first term describes the kinetic delocalization of electrons between neighboring sites, and the second term represents onsite repulsion which penalizes double occupancy. Parameters t and U control the strength of hopping and interaction respectively, and their ratio U/t determines whether the system behaves more like a metal or an insulator. The particle number operator $\hat{n}_{i\sigma}$ is defined as $\hat{a}^\dagger_{i\sigma}\hat{a}_{i\sigma}$. Consequently, the onsite repulsion term can be identified as the two-body interactions in the general Hamiltonian form (3).

2.3 Optimization Problem for Eigenvalues

As described in the previous section, the target wavefunction Ψ will be sought in $\mathcal{H}_{\text{sub}}$, spanned by the Slater determinants $\{\Phi_I\}_{I=1}^{N_{\text{FCI}}} = \{\Phi_{i_1,\cdots,i_n} : 1 \le i_1 < \cdots < i_n \le N\}$ of finite one-particle basis functions $\{\phi_1, \ldots, \phi_N\}$. In this work, we restrict the one-particle basis functions to be real-valued, as the systems under consideration generally admit real-valued eigenfunctions. Expressing the wavefunction Ψ as $\Psi = \sum_{I=1}^{N_{\text{FCI}}} c_I \Phi_I$ leads to the matrix representation of the time-independent Schrödinger equation (1) as

$$H\boldsymbol{c} = E\boldsymbol{c} \tag{9}$$

where H is real symmetric and of dimension $N_{\text{FCI}} \times N_{\text{FCI}}$, with each entry

$$H_{I,J} = \begin{cases} \sum_{p\in\text{occ}(\Phi_I)} h_{pp} + \frac{1}{2}\sum_{p,q\in\text{occ}(\Phi_I)}(v_{pqpq} - v_{pqqp}), & \Phi_I = \Phi_J, \\ (\pm)\left[h_{rp} + \sum_{k\in\text{occ}(\Phi_J)}(v_{rkpk} - v_{rkkp})\right], & \Phi_I = \hat{a}^\dagger_r\hat{a}_p\Phi_J, \\ (\pm)(v_{rspq} - v_{rsqp}), & \Phi_I = \hat{a}^\dagger_r\hat{a}^\dagger_s\hat{a}_p\hat{a}_q\Phi_J, \\ 0, & \text{otherwise}, \end{cases} \tag{10}$$

under the assumption that $\hat{\boldsymbol{H}}$ follows the general form (3). Here, $\text{occ}(\Phi_I) = \{i_1, \ldots, i_n\}$ if $\Phi_I = \phi_{i_1} \wedge \cdots \wedge \phi_{i_n}$. The sign $(\pm)$ denotes the phase factor arising from fermionic antisymmetry, determined by the number of orbital permutations required to excite Φ_J to Φ_I. Each entry of the vector $\boldsymbol{c}$ corresponds to the coefficient c_I for basis function Φ_I.

Consider the following unconstrained nonconvex optimization problem, which aims to find the best rank-one symmetric negative definite approximation of the Hamiltonian matrix H,

$$\min_{\boldsymbol{c}\in\mathbb{R}^{N_{\text{FCI}}}} f(\boldsymbol{c}; H) \coloneqq \|H + \boldsymbol{c}\boldsymbol{c}^\top\|_{\text{F}}^2. \tag{11}$$

As analysed in [37], $\pm\sqrt{-E_0}\boldsymbol{v}_0$ are the only two local minimizers of this problem, where $\boldsymbol{v}_0$ is the normalized eigenvector corresponding to the smallest non-degenerate eigenvalue $E_0 < 0$. Thus,

solving the optimization problem (11) reveals the ground-state energy E_0 and the ground-state wavefunction coefficient vector $\boldsymbol{v}_0$.

When targeting not only the ground state Ψ_0, but also several low-lying excited states $\Psi_1, \ldots, \Psi_K$, the respective states are expanded in terms of basis functions

$$\Psi_k = \sum_{I=1}^{N_{\mathrm{FCI}}} C_{I,k} \Phi_I, \quad k = 0, 1, \ldots, K \tag{12}$$

and coefficients $C_{I,k}$ form an orthonormal matrix C of size $N_{\mathrm{FCI}} \times (K+1)$. The optimization problem can subsequently be generalized to a multi-state formulation

$$\min_{C \in \mathbb{R}^{N_{\mathrm{FCI}} \times (K+1)}} f(C; H) \coloneqq \|H + CC^\top\|_{\mathrm{F}}^2, \tag{13}$$

thereby revealing several low-lying eigenpairs simultaneously.

Assume that the smallest eigenvalues of H satisfy $E_0 \leq E_1 \leq \cdots \leq E_K < 0$ and $E_K < E_{(K+1)}$. It was reported in [42] that all local minima are global minima in this problem, and they admit the form $V\sqrt{\Lambda}Q$ where the matrix Λ is a diagonal matrix with diagonal entries of $-E_0, \ldots, -E_K$, the matrix V satisfies $V^\top H V = \Lambda$, and the matrix Q is an arbitrary orthogonal matrix of size $(K+1) \times (K+1)$.

3 Coordinate Descent Methods

This section elaborates on the application of coordinate gradient descent methods to address problem (11) and (13), in order to reveal the ground state as well as a few excited states of the system. As a first-order optimization algorithm, the coordinate gradient descent method is particularly suitable for high-dimensional problems [43]. More importantly, it naturally supports a matrix-free implementation in our problem, since it only requires coordinate-wise access to matrix-vector products and thus avoids explicitly forming or storing the Hamiltonian matrix H, whose dimension is prohibitively large. Coordinate gradient descent proceeds as follows. At each iteration, one coordinate of the optimization variable is updated. The selection strategy may be cyclic, stochastic or based on the Gauss–Southwell rule, which identifies the coordinate with the largest absolute gradient component. The step size is subsequently determined either as a constant, as part of a diminishing sequence or via a line search procedure. After the selected coordinate is updated, this procedure is repeated until convergence.

In the following sections, we describe all the coordinate-descent-based solvers that are currently implemented in the package, including the ground-state solver CDFCI [36] in Section 3.1, the multi-coordinate ground-state solver mCDFCI [39] in Section 3.2, and the excited-state solver xCDFCI [38] in Section 3.3. To address the constraints of limited memory, an additional solver OptOrbFCI [40] has been implemented for the alternating optimization of one-particle basis functions and wavefunction coefficients, and the detailed description is provided in Section 3.4.

3.1 Coordinate-Descent Methods for Ground State

At its core, the CDFCI solver addresses problem (11) through a coordinate descent framework incorporating the Gauss–Southwell rule and an exact line search for step determination. In what follows, we denote $f(\boldsymbol{c})$ as $f(\boldsymbol{c}; H)$ whenever the Hamiltonian H is clear from the context.

At the ℓ-th iteration, the algorithm consists of two steps: (i) selecting the coordinate to be updated, which has the largest absolute gradient value, and (ii) determining the corresponding

step size, which gives the largest descent on the current function value. These two steps can be expressed as the following subproblems

$$\begin{aligned} i^{(\ell+1)} &= \arg\max_i \ |(\nabla f(\boldsymbol{c}^{(\ell)}))_i| \\ &= \arg\max_i \ |4(H\boldsymbol{c}^{(\ell)})_i + 4(\boldsymbol{c}^{(\ell)\top}\boldsymbol{c}^{(\ell)})\boldsymbol{c}_i^{(\ell)}|, \end{aligned} \tag{14}$$

and

$$\begin{aligned} \eta^{(\ell+1)} &= \arg\min_\eta \ f(\boldsymbol{c}^{(\ell)} + \eta \boldsymbol{e}_{i^{(\ell+1)}}) \\ &= \arg\min_\eta \ \eta^4 + 4\boldsymbol{c}^{(\ell)}_{i^{(\ell+1)}}\eta^3 + (2H_{i^{(\ell+1)},i^{(\ell+1)}} + 4(\boldsymbol{c}^{(\ell)}_{i^{(\ell+1)}})^2 + 2\boldsymbol{c}^{(\ell)\top}\boldsymbol{c}^{(\ell)})\eta^2 \\ &\qquad + (4(H\boldsymbol{c}^{(\ell)})_{i^{(\ell+1)}} + 4(\boldsymbol{c}^{(\ell)\top}\boldsymbol{c}^{(\ell)})\boldsymbol{c}^{(\ell)}_{i^{(\ell+1)}})\eta + f(\boldsymbol{c}^{(\ell)}), \end{aligned} \tag{15}$$

where the subscript $\boldsymbol{c}_i$ denotes the i-th coordinate of the vector $\boldsymbol{c}$. Problem (15) can be solved by reducing the minimization of a quartic polynomial to finding the roots of its cubic derivative. The complete derivation is presented in Appendix A. The $i^{(\ell+1)}$-th coordinate of $\boldsymbol{c}$ will be updated subsequently before moving on to the $(\ell+1)$-th iteration.

In both steps (14) and (15), the vector $H\boldsymbol{c}^{(\ell)}$ and the scalar $\boldsymbol{c}^{(\ell)\top}\boldsymbol{c}^{(\ell)}$ are required. To avoid recomputing these quantities, the vector $H\boldsymbol{c}$ and two scalars $\boldsymbol{c}^\top\boldsymbol{c}$, $\boldsymbol{c}^\top H\boldsymbol{c}$ are stored in memory and incrementally updated at the end of each iteration. Let $\boldsymbol{b}$ denote $H\boldsymbol{c}$, μ and α denote $\boldsymbol{c}^\top\boldsymbol{c}$ and $\boldsymbol{c}^\top H\boldsymbol{c}$ respectively. The update of these variables only uses nonzero entries of one column of the Hamiltonian matrix:

$$\begin{aligned} \boldsymbol{c}^{(\ell+1)} &\leftarrow \boldsymbol{c}^{(\ell)} + \eta^{(\ell+1)}\boldsymbol{e}_{i^{(\ell+1)}}, \\ \boldsymbol{b}^{(\ell+1)} &\leftarrow \boldsymbol{b}^{(\ell)} + \eta^{(\ell+1)}H_{:,i^{(\ell+1)}}, \\ \mu^{(\ell+1)} &\leftarrow \mu^{(\ell)} + 2\eta^{(\ell+1)}\boldsymbol{c}^{(\ell)}_{i^{(\ell+1)}} + (\eta^{(\ell+1)})^2, \\ \alpha^{(\ell+1)} &\leftarrow \alpha^{(\ell)} + 2\eta^{(\ell+1)}\boldsymbol{b}^{(\ell)}_{i^{(\ell+1)}} + (\eta^{(\ell+1)})^2 H_{i^{(\ell+1)},i^{(\ell+1)}}. \end{aligned} \tag{16}$$

During the update of $\boldsymbol{b}$, a compression scheme can be employed to restrict the size of the variational space and force the solution vector $\boldsymbol{c}$ to converge in a subspace controlled by a deterministic threshold. The current compression strategy we use is as follows. Given a fixed threshold τ, $\boldsymbol{b}_j^{(\ell+1)}$ will only be updated if $\boldsymbol{c}_j^{(\ell+1)} \neq 0$ or if $|\eta^{(\ell+1)}H_{j,i^{(\ell+1)}}| \geq \tau$. This truncation rule will not affect $\mu^{(\ell+1)}$ and $\alpha^{(\ell+1)}$ and is both effective and cheap.

Two additional considerations should be mentioned here. First, when selecting a descent coordinate for problem (14), an exhaustive search over the full CI space is computationally infeasible. Instead, we restrict our search space to $\mathcal{I}_H(i^{(\ell)})$, where $\mathcal{I}_H(i)$ denotes the index set composed of all indices j such that $H_{i,j} \neq 0$. Second, a modification is introduced in the update of $\boldsymbol{b}$: the $i^{(\ell+1)}$-th entry in $\boldsymbol{b}^{(\ell+1)}$ is explicitly recalculated using the corresponding Hamiltonian entry and coefficients of $\boldsymbol{c}$ that are already computed or retrieved,

$$\boldsymbol{b}^{(\ell+1)}_{i^{(\ell+1)}} = H_{i^{(\ell+1)},:}\boldsymbol{c}^{(\ell+1)} = \sum_{j\in\mathcal{I}_H(i^{(\ell+1)})} H_{j,i^{(\ell+1)}}\boldsymbol{c}_j^{(\ell+1)}. \tag{17}$$

This recalculation improves numerical stability and ensures the correctness when entries of $\boldsymbol{b}$ are truncated by some threshold τ.

The complete algorithm for the CDFCI solver [36] is detailed in Algorithm 1.

Algorithm 1 CDFCI: Single-Coordinate Descent Method for Ground State

Initialize $\boldsymbol{c}^{(0)}$, $\boldsymbol{b}^{(0)} = H\boldsymbol{c}^{(0)}$ and $i^{(0)}$.

for $\ell = 0, 1, 2, \ldots$ until convergence **do**

Select coordinate

$$i^{(\ell+1)} = \underset{i \in \mathcal{I}_H(i^{(\ell)})}{\arg\max} |4\boldsymbol{b}_i^{(\ell)} + 4\mu^{(\ell)}\boldsymbol{c}_i^{(\ell)}|.$$

Compute the step size by finding roots of a cubic polynomial $\frac{\mathrm{d}h(\eta)}{\mathrm{d}\eta}$ (see Appendix A for details):

$$\begin{aligned}
\eta^{(\ell+1)} &= \underset{\eta}{\arg\min}\ h(\eta) \coloneqq f(\boldsymbol{c}^{(\ell)} + \eta \boldsymbol{e}_{i^{(\ell+1)}}) \\
&= \underset{\eta}{\arg\min}\ \eta^4 + 4\boldsymbol{c}_{i^{(\ell+1)}}^{(\ell)}\eta^3 + \left(2H_{i^{(\ell+1)},i^{(\ell+1)}} + 4(\boldsymbol{c}_{i^{(\ell+1)}}^{(\ell)})^2 + 2\mu^{(\ell)}\right)\eta^2 \\
&\qquad + \left(4\boldsymbol{b}_{i^{(\ell+1)}}^{(\ell)} + 4\mu^{(\ell)}\boldsymbol{c}_{i^{(\ell+1)}}^{(\ell)}\right)\eta + f(\boldsymbol{c}^{(\ell)}).
\end{aligned}$$

Update $\boldsymbol{c}^{(\ell+1)}$ and $\boldsymbol{b}^{(\ell+1)}$

$$\begin{aligned}
\boldsymbol{c}^{(\ell+1)} &\leftarrow \boldsymbol{c}^{(\ell)} + \eta^{(\ell+1)}\boldsymbol{e}_{i^{(\ell+1)}}, \\
\boldsymbol{b}^{(\ell+1)} &\leftarrow \boldsymbol{b}^{(\ell)} + \eta^{(\ell+1)}H_{:,i^{(\ell+1)}}.
\end{aligned}$$

Employ compression on $\boldsymbol{b}^{(\ell+1)}$ if desired.

Recalculate $\boldsymbol{b}_{i^{(\ell+1)}}^{(\ell+1)}$

$$\boldsymbol{b}_{i^{(\ell+1)}}^{(\ell+1)} = H_{i^{(\ell+1)},:}\boldsymbol{c}^{(\ell+1)} = \sum_{j \in \mathcal{I}_H(i^{(\ell+1)})} H_{j,i^{(\ell+1)}}\boldsymbol{c}_j^{(\ell+1)}.$$

Update $\mu^{(\ell+1)}$ and $\alpha^{(\ell+1)}$

$$\begin{aligned}
\mu^{(\ell+1)} &\leftarrow \mu^{(\ell)} + 2\eta^{(\ell+1)}\boldsymbol{c}_{i^{(\ell+1)}}^{(\ell)} + (\eta^{(\ell+1)})^2, \\
\alpha^{(\ell+1)} &\leftarrow \alpha^{(\ell)} + 2\eta^{(\ell+1)}\boldsymbol{b}_{i^{(\ell+1)}}^{(\ell)} + (\eta^{(\ell+1)})^2 H_{i^{(\ell+1)},i^{(\ell+1)}}.
\end{aligned}$$

Report energy estimation as the Rayleigh quotient of $\boldsymbol{c}^{(\ell+1)}$: $\alpha^{(\ell+1)}/\mu^{(\ell+1)}$.

end for

3.2 Multi-Coordinate-Descent Methods for Ground State

We sketch an extension of Algorithm 1 where more than one coordinate can be chosen in each iteration. It is reported in [39] that the multi-coordinate update strategy exhibits a similar descent behavior to standard coordinate descent: updating m coordinates in one iteration achieves an energy decrease comparable to performing m sequential single-coordinate updates. The workload per iteration is increased m times, and the coordinate updates can be computed concurrently, making the method more attractive for the parallel processing environment. In this modified coordinate descent framework, the update of $\boldsymbol{c}$ is realized by

$$\boldsymbol{c}^{(\ell+1)} \leftarrow \gamma^{(\ell+1)}\boldsymbol{c}^{(\ell)} + \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{\eta}^{(\ell+1)}, \tag{18}$$

where $\mathcal{I}^{(\ell+1)}$ denotes the index set of coordinates chosen, $|\mathcal{I}^{(\ell+1)}| = m$ and the step size vector $\boldsymbol{\eta}^{(\ell+1)}$ is an extension to the step size scalar η, encoding the step sizes in each direction. The projection matrix $\mathcal{E}_{\mathcal{I}^{(\ell+1)}} \in \mathbb{R}^{N_{\mathrm{FCI}}\times m}$ is defined as $\mathcal{E}_{\mathcal{I}^{(\ell+1)}} = [\boldsymbol{e}_{i_1^{(\ell+1)}}, \ldots, \boldsymbol{e}_{i_m^{(\ell+1)}}]$ with $\mathcal{I}^{(\ell+1)} = \{i_1^{(\ell+1)}, \ldots, i_m^{(\ell+1)}\}$, which, in other words, consists of columns from the identity matrix corresponding to the selected coordinates. Note that a scaling factor $\gamma^{(\ell+1)}$ is inserted into the update formula (18) to enable exact line search in multi-coordinate setting. When $|\mathcal{I}^{(\ell+1)}| = 1$ and $\gamma^{(\ell+1)} = 1$, the original update rule is recovered.

In coordinate selection step, m coordinates with largest absolute gradient values are picked out,

$$\mathcal{I}^{(\ell+1)} = \left\{ i_j^{(\ell+1)}, j = 1, \ldots, m : i_j^{(\ell+1)} = \underset{\substack{i\in\mathcal{I}_H(\mathcal{I}^{(\ell)}) \\ i\neq i_1^{(\ell+1)},\ldots,i_{j-1}^{(\ell+1)}}}{\arg\max} \left|4\boldsymbol{b}_i^{(\ell)} + 4\mu^{(\ell)}\boldsymbol{c}_i^{(\ell)}\right| \right\} \tag{19}$$

where $\mathcal{I}_H(\mathcal{I}^{(\ell)})$ is the union of $\mathcal{I}_H(i)$ for all $i \in \mathcal{I}^{(\ell)}$. Moving forward, the optimal step size vector $\boldsymbol{\eta}^{(\ell+1)}$ and scaling factor $\gamma^{(\ell+1)}$ are determined by minimizing the function value of the next iterate,

$$\begin{aligned} \gamma^{(\ell+1)}, \boldsymbol{\eta}^{(\ell+1)} &= \underset{\gamma\in\mathbb{R},\boldsymbol{\eta}\in\mathbb{R}^m}{\arg\min}\ f(\gamma\boldsymbol{c}^{(\ell)} + \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{\eta}) \\ &= \underset{\gamma\in\mathbb{R},\boldsymbol{\eta}\in\mathbb{R}^m}{\arg\min}\ f\left(\begin{bmatrix}\boldsymbol{c}^{(\ell)} & \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\end{bmatrix}\begin{bmatrix}\gamma \\ \boldsymbol{\eta}\end{bmatrix}\right). \end{aligned} \tag{20}$$

The matrix $\begin{bmatrix}\boldsymbol{c}^{(\ell)} & \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\end{bmatrix}$ admits the factorization

$$\begin{bmatrix}\boldsymbol{c}^{(\ell)} & \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\end{bmatrix} = \begin{bmatrix}\tilde{\boldsymbol{c}}^{(\ell)} & \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\end{bmatrix}\begin{bmatrix}\|\boldsymbol{c}^{(\ell)} - \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}}\| & O \\ \boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}} & I_m\end{bmatrix} \tag{21}$$

with

$$\tilde{\boldsymbol{c}}^{(\ell)} = \begin{cases} \frac{\boldsymbol{c}^{(\ell)} - \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}}}{\|\boldsymbol{c}^{(\ell)} - \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}}\|}, & \text{if } \|\boldsymbol{c}^{(\ell)} - \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}}\| \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

Let $Q^{(\ell+1)} := \begin{bmatrix}\tilde{\boldsymbol{c}}^{(\ell)} & \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\end{bmatrix}$, which has orthogonal columns. Then $\boldsymbol{c}^{(\ell+1)}$ can be expressed as a vector in the span of $Q^{(\ell+1)}$

$$\boldsymbol{c}^{(\ell+1)} = Q^{(\ell+1)}\boldsymbol{z}^{(\ell+1)}, \tag{23}$$

with

$$\boldsymbol{z}^{(\ell+1)} = \begin{bmatrix}\|\boldsymbol{c}^{(\ell)} - \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}}\| & O \\ \boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}} & I_m\end{bmatrix}\begin{bmatrix}\gamma^{(\ell+1)} \\ \boldsymbol{\eta}^{(\ell+1)}\end{bmatrix}. \tag{24}$$

Using the orthogonal invariance property of the Frobenius norm, problem (20) is equivalent to a reduced-size problem of dimensions $(m+1)$:

$$\begin{aligned} \boldsymbol{z}^{(\ell+1)} &= \underset{\boldsymbol{z}\in\mathbb{R}^{m+1}}{\arg\min}\ f\left(Q^{(\ell+1)}\boldsymbol{z}^{(\ell+1)}; H\right) \\ &= \underset{\boldsymbol{z}\in\mathbb{R}^{m+1}}{\arg\min}\ f\left(\boldsymbol{z}^{(\ell+1)}; (Q^{(\ell+1)})^\top H Q^{(\ell+1)}\right) \end{aligned} \tag{25}$$

and can be solved via seeking the smallest eigenpair of $(Q^{(\ell+1)})^\top H Q^{(\ell+1)}$. Optimal values of $\gamma^{(\ell+1)}$ and $\boldsymbol{\eta}^{(\ell+1)}$ are subsequently revealed.

The complete algorithm for multi-coordinate descent method for ground state computation [39] is detailed in Algorithm 2.

Algorithm 2 mCDFCI: Multi-Coordinate Descent Method for Ground State

1: Initialize $\boldsymbol{c}^{(0)}$, $\boldsymbol{b}^{(0)} = H\boldsymbol{c}^{(0)}$ and $\mathcal{I}^{(0)}$.

2: **for** $\ell = 0, 1, 2, \ldots$ until convergence **do**

3: Select coordinates

$$\mathcal{I}^{(\ell+1)} = \left\{ i_j^{(\ell+1)}, j = 1, \ldots, m : i_j^{(\ell+1)} = \underset{\substack{i\in\mathcal{I}_H(\mathcal{I}^{(\ell)}) \\ i\neq i_1^{(\ell+1)},\ldots,i_{j-1}^{(\ell+1)}}}{\arg\max} \left|4\boldsymbol{b}_i^{(\ell)} + 4\mu^{(\ell)}\boldsymbol{c}_i^{(\ell)}\right| \right\}.$$

4: Compute scaling factor $\gamma^{(\ell+1)}$ and step size vector $\boldsymbol{\eta}^{(\ell+1)}$ by solving a $(m+1)$-dimensional eigenvalue problem

$$\gamma^{(\ell+1)}, \boldsymbol{\eta}^{(\ell+1)} = \underset{\gamma\in\mathbb{R},\boldsymbol{\eta}\in\mathbb{R}^m}{\arg\min}\ f\left(\begin{bmatrix} \|\boldsymbol{c}^{(\ell)} - \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}}\| & O \\ \boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}} & I_m \end{bmatrix} \begin{bmatrix}\gamma \\ \boldsymbol{\eta}\end{bmatrix}; (Q^{(\ell+1)})^\top H Q^{(\ell+1)}\right).$$

5: Update $\boldsymbol{c}^{(\ell+1)}$ and $\boldsymbol{b}^{(\ell+1)}$

$$\begin{aligned} \boldsymbol{c}^{(\ell+1)} &\leftarrow \gamma^{(\ell+1)}\boldsymbol{c}^{(\ell)} + \mathcal{E}_{\mathcal{I}^{(\ell+1)}}\boldsymbol{\eta}^{(\ell+1)}, \\ \boldsymbol{b}^{(\ell+1)} &\leftarrow \gamma^{(\ell+1)}\boldsymbol{b}^{(\ell)} + H_{:,\mathcal{I}^{(\ell+1)}}\boldsymbol{\eta}^{(\ell+1)}. \end{aligned}$$

Employ compression on $\boldsymbol{b}$ if desired.

6: Recalculate $\boldsymbol{b}^{(\ell+1)}_{\mathcal{I}^{(\ell+1)}}$

$$\boldsymbol{b}_i^{(\ell+1)} = H_{i,:}\boldsymbol{c}^{(\ell+1)} = \sum_{j\in\mathcal{I}_H(i)} H_{j,i}\boldsymbol{c}_j^{(\ell+1)}, \quad \text{for } i \in \mathcal{I}^{(\ell+1)}.$$

7: Update $\mu^{(\ell+1)}$ and $\alpha^{(\ell+1)}$

$$\begin{aligned} \mu^{(\ell+1)} &\leftarrow (\gamma^{(\ell+1)})^2\mu^{(\ell)} + 2\gamma^{(\ell+1)}(\boldsymbol{\eta}^{(\ell+1)})^\top\boldsymbol{c}^{(\ell)}_{\mathcal{I}^{(\ell+1)}} + (\boldsymbol{\eta}^{(\ell+1)})^\top\boldsymbol{\eta}^{(\ell+1)}, \\ \alpha^{(\ell+1)} &\leftarrow (\gamma^{(\ell+1)})^2\alpha^{(\ell)} + 2\gamma^{(\ell+1)}(\boldsymbol{\eta}^{(\ell+1)})^\top\boldsymbol{b}^{(\ell)}_{\mathcal{I}^{(\ell+1)}} + (\boldsymbol{\eta}^{(\ell+1)})^\top H_{\mathcal{I}^{(\ell+1)},\mathcal{I}^{(\ell+1)}}\boldsymbol{\eta}^{(\ell+1)}. \end{aligned}$$

8: Report energy estimation as the Rayleigh quotient of $\boldsymbol{c}^{(\ell+1)}$: $\alpha^{(\ell+1)}/\mu^{(\ell+1)}$.

9: **end for**

3.3 Coordinate-Descent Methods for Excited States

Consider problem (13) when K low-lying excited states are targeted in addition to the ground state. Since now the optimization variable is matrix $C \in \mathbb{R}^{N_{\text{FCI}} \times (K+1)}$, we introduce the following notations: $B = HC$ as an extension to the $\boldsymbol{b}$ vector, $G = 4B + 4C(C^\top C)$ representing the gradient of f, and two $(K+1) \times (K+1)$ matrices $M = C^\top C$ and $A = C^\top HC$ as extensions to scalars μ and α. At each iteration, one row of C is updated by

$$C^{(\ell+1)} \leftarrow C^{(\ell)} + \eta^{(\ell+1)} \boldsymbol{e}_{i^{(\ell+1)}} G^{(\ell)}_{i^{(\ell+1)},:}. \tag{26}$$

The algorithm identifies the next coordinate and step size by solving two subproblems:

$$i^{(\ell+1)} = \arg\max_i \ \|G^{(\ell)}_{i,:}\|_\infty \tag{27}$$

and

$$\eta^{(\ell+1)} = \arg\min_\eta f(C^{(\ell)} + \eta \boldsymbol{e}_{i^{(\ell+1)}} G^{(\ell)}_{i^{(\ell+1)},:}) \tag{28}$$

which again requires minimizing a fourth-order polynomial with respect to scalar η. Computation details are provided in Appendix A.

The minimizers of (13) only give eigenspaces, but not orthogonal eigenvectors. To retrieve eigenvectors, we need another post-processing step of solving the following general eigenvalue problem of size $(K+1) \times (K+1)$:

$$AU = MU\Gamma \tag{29}$$

for U being eigenvectors and Γ being the eigenvalue matrix.

The complete algorithm of coordinate descent method for excited states [38] is summarized in Algorithm 3.

3.4 Optimal Orbital Selections

In scenarios where the memory budget is limited, it is often beneficial to optimize the one-particle basis functions to achieve a more compact representation of the wavefunction. This approach is particularly useful when the full configuration interaction (FCI) space is too large to be fully explored. Given M one-particle basis functions $\{\psi_1, \ldots, \psi_M\}$, the goal is to find an orthogonal matrix $U \in \mathbb{R}^{M \times N}, U^\top U = I_N$ that transforms the basis functions

$$(\phi_1, \ldots, \phi_N) = (\psi_1, \ldots, \psi_M)U, \tag{30}$$

such that the transformed basis functions $\{\phi_1, \ldots, \phi_N\}$ minimize the ground-state energy within a smaller FCI space.

To solve this problem, we employ a two-level optimization strategy, corresponding to the two levels of variables: the rotation matrix U and the wavefunction coefficients $\boldsymbol{c}$. The outer level that focuses on optimizing the one-particle functions formulates the following optimization problem:

$$\min_{U \in \mathbb{R}^{M \times N}, U^\top U = I_N} E_0(U) \coloneqq \sum_{p',q'=1}^{N} \sum_{p,q=1}^{M} h_{pq} U_{pp'} U_{qq'} {}^1D^{p'}_{q'} + \sum_{p',q',r',s'=1}^{N} \sum_{p,q,r,s=1}^{M} v_{pqrs} U_{pp'} U_{qq'} U_{rr'} U_{ss'} {}^2D^{p'q'}_{r's'} \tag{31}$$

where ${}^1D^{p'}_{q'} = \left\langle \Phi \middle| \hat{\boldsymbol{a}}^\dagger_{p'} \hat{\boldsymbol{a}}_{q'} \middle| \Phi \right\rangle$ and ${}^2D^{p'q'}_{r's'} = \left\langle \Phi \middle| \hat{\boldsymbol{a}}^\dagger_{p'} \hat{\boldsymbol{a}}^\dagger_{q'} \hat{\boldsymbol{a}}_{s'} \hat{\boldsymbol{a}}_{r'} \middle| \Phi \right\rangle$ are the one-body and two-body reduced density matrices (1RDM and 2RDM) associated with the wavefunction Φ, which is represented in the N-particle Hilbert space spanned by the transformed single-particle basis functions

Algorithm 3 xCDFCI: Coordinate Descent Method for Excited States

1: Initialize $C^{(0)}$, $B^{(0)} = HC^{(0)}$ and $i^{(0)}$.

2: **for** $\ell = 0, 1, 2, \ldots$ until convergence **do**

3: Select coordinate

$$i^{(\ell+1)} = \underset{i \in \mathcal{I}_H(i^{(\ell)})}{\arg\max} \|G^{(\ell)}_{i,:}\|_\infty$$

where $G^{(\ell)} = 4B^{(\ell)} + 4C^{(\ell)}M^{(\ell)}$.

4: Compute the step size by solving the roots of a cubic polynomial $\frac{\mathrm{d}h(\eta)}{\mathrm{d}\eta}$ (see Appendix A for details):

$$\eta^{(\ell+1)} = \underset{\eta}{\arg\min}\ h(\eta) \coloneqq f(C^{(\ell)} + \eta \boldsymbol{e}_{i^{(\ell+1)}} G^{(\ell)}_{i^{(\ell+1)},:}).$$

5: Update $C^{(\ell+1)}$ and $B^{(\ell+1)}$

$$C^{(\ell+1)} \leftarrow C^{(\ell)} + \eta^{(\ell+1)} \boldsymbol{e}_{i^{(\ell+1)}} G^{(\ell)}_{i^{(\ell+1)},:},$$
$$B^{(\ell+1)} \leftarrow B^{(\ell)} + \eta^{(\ell+1)} H_{:,i^{(\ell+1)}} G^{(\ell)}_{i^{(\ell+1)},:}.$$

Employ compression on rows of $B^{(\ell+1)}$ if desired.

6: Recalculate $B^{(\ell+1)}_{i^{(\ell+1)}}$

$$B^{(\ell+1)}_{i^{(\ell+1)}} = \sum_{j \in \mathcal{I}_H(i^{(\ell+1)})} H_{j,i^{(\ell+1)}} C^{(\ell+1)}_{j,:}.$$

7: Update $M^{(\ell+1)}$ and $A^{(\ell+1)}$

$$M^{(\ell+1)} \leftarrow M^{(\ell)} + \eta^{(\ell+1)} \left({C^{(\ell)}_{i^{(\ell+1)},:}}^\top G^{(\ell)}_{i^{(\ell+1)},:} + {G^{(\ell)}_{i^{(\ell+1)},:}}^\top C^{(\ell)}_{i^{(\ell+1)},:} \right) + (\eta^{(\ell+1)})^2 \|G^{(\ell)}_{i^{(\ell+1)},:}\|^2,$$
$$A^{(\ell+1)} \leftarrow A^{(\ell)} + \eta^{(\ell+1)} \left({B^{(\ell)}_{i^{(\ell+1)},:}}^\top G^{(\ell)}_{i^{(\ell+1)},:} + {G^{(\ell)}_{i^{(\ell+1)},:}}^\top B^{(\ell)}_{i^{(\ell+1)},:} \right) + (\eta^{(\ell+1)})^2 H_{i^{(\ell+1)},i^{(\ell+1)}} \|G^{(\ell)}_{i^{(\ell+1)},:}\|^2.$$

8: Report energy estimation after solving the general eigenvalue problem of matrix pencil $(A^{(\ell+1)}, M^{(\ell+1)})$.

9: **end for**

$\{\phi_1, \ldots, \phi_N\}$. The objective function is a fourth-order polynomial of U and can be optimized using gradient-based projection methods or other suitable optimization techniques on Stiefel manifolds. In the current implementation, projected gradient descent with alternating Barzilai–Borwein (BB) step size is employed to solve problem (31). On the other hand, the inner level is just solving the FCI problem given a fixed set of one-particle basis functions, and can be efficiently handled by the coordinate descent methods described in the previous sections.

The full algorithm for optimal orbital FCI (OptOrbFCI) [40] is summarized in Algorithm 4.

Algorithm 4 OptOrbFCI: Optimal Orbital Selection with Coordinate Descent FCI

1: Initialize one-particle basis functions $\{\phi_i^{(0)}\}_{i=1}^N$ from a larger set $\{\psi_i\}_{i=1}^M$.
2: **for** $\ell = 0, 1, 2, \ldots$ until convergence **do**
3: Compute one- and two-electron integrals $\{h_{pq}^{(\ell)}\}_{p,q=1}^N$ and $\{v_{pqrs}^{(\ell)}\}_{p,q,r,s=1}^N$ in the current basis $\{\phi_i^{(\ell)}\}_{i=1}^N$.
4: Solve the FCI problem in the current basis using coordinate descent methods to obtain ground-state energy $E_0^{(\ell)}$ and wavefunction coefficients $\boldsymbol{c}^{(\ell)}$.
5: Compute the 1RDM and 2RDM from the ground-state wavefunction.
6: Solve the orthonormal constrained polynomial (31) via projection method with alternating BB step size as

$$U^{(\ell+1)} = \operatorname{orth}\big(U^{(\ell)} - \tau_k \boldsymbol{\nabla}_{U^{(\ell)}} P_4(U^{(\ell)})\big), \tag{32}$$

and obtain $U^{(\ell+1)}$. Here, $\operatorname{orth}\big(X\big)$ denotes the orthogonalization of matrix X.
7: Update one-particle basis functions $(\phi_1^{(\ell+1)}, \ldots, \phi_N^{(\ell+1)}) = (\psi_1, \ldots, \psi_M)U^{(\ell+1)}$.
8: **end for**

4 Overview of the CDFCI Software Package

The CDFCI software package is implemented in modern C++, leveraging advanced features of the C++17 standard and the `Eigen` library [44] for efficient linear algebra operations. The package is designed with a modular architecture, allowing for easy extension and customization of its components. Sections 4.1–4.3 present three major classes that constitute the core of the implementation: the `Hamiltonian` class, the `WaveFunction` class, and the `Solver` class. In particular, parallelization support is embedded in the `WaveFunction` class. We analyze the computational complexity and memory usage in Section 4.3, and conclude with a simple use case demonstrating how to set up and run a CDFCI calculation in Section 4.4.

4.1 The `Hamiltonian` Class

The Hamiltonian class is responsible for representing the fermionic Hamiltonian in the second-quantized form. It provides methods for constructing the Hamiltonian, as well as accessing its matrix elements efficiently.

In the abstract base class `Hamiltonian<N>`, the template parameter `N` refers to the same parameter `N` in the abstract base class `Determinant<N>` for Slater determinants, where `N` represents the number of `size_t` types required to store the determinant in a bit string format. Note that the Hamiltonian matrix is never generated or stored explicitly. Instead, several access methods are provided to retrieve matrix elements or columns on-the-fly:

- `get_entry(`D_1, D_2`)`: returns the matrix element H_{D_1,D_2}, according to the Slater–Condon rules (10).
- `get_diagonal(`D`)`: returns the diagonal element $H_{D,D}$.
- `get_column(`D`)`: returns all nonzero elements in the column corresponding to the determinant D, as an unordered list of $(D', H_{D',D})$ tuples.

This interface is intentionally designed to match the needs of coordinate-descent updates, as the algorithm repeatedly requires column access to H with respect to the current determinant.

Two derived classes of `Hamiltonian<N>` are provided: `HamiltonianMolecule<N>` for molecular Hamiltonians and `HamiltonianLattice<N>` for lattice models. For electronic structure calculations, the Hamiltonian is read from FCIDUMP files, which are a standard format for storing one-body and two-body integrals in quantum chemistry. For lattice models, such as the Hubbard model, the Hamiltonian can be constructed directly from the model parameters.

4.1.1 Molecular Hamiltonians

Class `HamiltonianMolecule<N>` constructs the electronic Hamiltonian from FCIDUMP files, which specify the number of orbitals (`norb`), number of electrons (`nelec`), spin multiplicity (`ms2`), one- and two-electron integrals, and the core energy. The constructor enforces `norb` > 0, `nelec` > 0, `norb` $\geq$ `nelec`, and (`nelec` + `ms2`) mod $2 = 0$. An additional threshold can be set to ignore small integrals.

The definition of one- and two-electron integrals follows (4) and (5). Denote the antisymmetrized two-electron integrals

$$\langle pq \| rs \rangle \;=\; v_{pqrs} - v_{pqsr},$$

with spin–orbital indices. To generate `get_column` efficiently, we precompute three compact structures:

- *Double excitations* (`double_excitation`): for each ordered pair of spin–orbitals $i < j$, store a sorted list of $(a, b, \langle ij \| ab \rangle)$.
- *Single excitations* (`single_excitation`): for each spin–orbital i, store all candidate a together with h_{ia} and the vector $\{\langle ik \| ak \rangle\}_k$.
- *Diagonal cache* (`diagonal`): store vectors $\{h_{ii}\}$ for each spin–orbital i, and $\{\langle ij \| ij \rangle\}$ for each ordered pair $i < j$.

Given a determinant D, `get_column(D)` returns $\{(D, H_{D,D})\} \cup \{(D', H_{D',D})\}$ over all single and double excitations from D. Denote N the number of spin–orbitals and n the number of electrons. The diagonal $H_{D,D}$ is computed in $\mathcal{O}(n^2)$ time using the diagonal cache. Single excitations $(i \to a)$ are generated in $\mathcal{O}(n(N-n))$ time by scanning occupied i and consulting the single-excitation structure. Each matrix element equals $h_{ia} + \sum_k \langle ik \| ak \rangle$ up to the fermionic parity sign. Double excitations $(i, j) \to (a, b)$ are generated in $\mathcal{O}(n^2(N-n)^2)$ time by scanning ordered pairs of occupied (i, j) and consulting the double-excitation structure. Each matrix element equals $\langle ij \| ab \rangle$ up to the fermionic parity sign. To summarize, the time complexity of `get_column(D)` is $\mathcal{O}(n^2 N^2)$ in the worst case.

To facilitate shared-memory parallelism, a class function `get_column_parallel_part(D, tid, nthrds)` is provided to partition the $\binom{n}{2}$ occupied (i, j) orbital pairs evenly among `nthrds` worker threads, with thread ID `tid` in $[0, \texttt{nthrds} - 1]$. More details on the parallelization support are given in Section 4.2.

4.1.2 Lattice Model Hamiltonians

Class `HamiltonianHubbardK<N>` implements a D-dimensional Hubbard model with periodic boundary conditions over an orthogonal lattice specified by integer numbers of sites along each spatial direction (e.g., $[L_x, L_y, L_z]$). In momentum space, the kinetic term becomes

$$-t \sum_{\langle i,j\rangle,\sigma} (\hat{a}^\dagger_{i\sigma}\hat{a}_{j\sigma} + \text{h.c.}) = \sum_{\boldsymbol{k},\sigma} \varepsilon_{\boldsymbol{k}} \hat{a}^\dagger_{\boldsymbol{k}\sigma}\hat{a}_{\boldsymbol{k}\sigma},$$

with

$$\varepsilon_{\boldsymbol{k}} = -2t \sum_{d=1}^{D} \cos\Big(\frac{2\pi k_d}{L_d}\Big), \quad k_d = 0, 1, 2, \ldots, L_d - 1.$$

The interaction term becomes nonlocal,

$$U \sum_i \hat{n}_{i\uparrow}\hat{n}_{i\downarrow} = \frac{U}{L} \sum_{\boldsymbol{k}_1,\boldsymbol{k}_2,\boldsymbol{q}} \hat{a}^\dagger_{\boldsymbol{k}_1,\uparrow}\hat{a}^\dagger_{\boldsymbol{k}_2,\downarrow}\hat{a}_{\boldsymbol{k}_2,\downarrow}\hat{a}_{\boldsymbol{k}_1,\uparrow}$$

with $L = \prod_{d=1}^{D} L_d$.

Therefore, single excitations vanish, and nonzero off-diagonals arise from opposite-spin double excitations $(i,j) \to (a,b)$ obeying momentum conservation $\boldsymbol{k}_i + \boldsymbol{k}_j = \boldsymbol{k}_a + \boldsymbol{k}_b \pmod{L_d}$. Given i, j, a, the partner b is uniquely determined; we generate doubles by scanning a (same spin as i) and computing b. Each nonzero matrix element equals $\frac{U}{L}$ up to the fermionic parity sign.

The class reuses the same column API: `get_column` returns the diagonal plus all momentum-conserving doubles; `get_column_parallel_part` splits the $\binom{n}{2}$ pairs among threads identically to the molecular case.

4.2 The `WaveFunction` Class

In the `WaveFunction` class, the sparse vectors $\boldsymbol{c}$ and $\boldsymbol{b}$ (or rowwise sparse matrices C and B in the multi-state case) are represented by a key-value mapping, where each key denotes a Slater determinant and each value contains the associated coefficients. For each `key_type` D,

$$\texttt{mapped\_type} = \begin{cases} (\boldsymbol{c}_D, \boldsymbol{b}_D), & \text{if } \texttt{NSTATES} = 1, \\ (C_{D,:}, B_{D,:}), & \text{if } \texttt{NSTATES} > 1, \end{cases}$$

where `NSTATES` is a template parameter controlling the number of states.

The class template `WaveFunction<Container, NSTATES>` is agnostic to the storage backend. In the current implementation, we use robin-hood hash map [45] for single-threaded and concurrent cuckoo hash maps [46, 47, 48] for multi-threaded scenarios. In both cases, the implementation performs an explicit overflow check that halts further insertions once the capacity threshold is reached, ensuring that the backend never performs rehashing.

In addition to $\boldsymbol{c}$ and $\boldsymbol{b}$ (or C and B), the class also maintains the vector norm $\mu = \boldsymbol{c}^\top\boldsymbol{c}$ (or $M = C^\top C$), the quadratic form $\alpha = \boldsymbol{c}^\top\boldsymbol{b}$ (or $A = C^\top B$), and the scaling factor γ introduced in Section 3.1-3.3. These variables are accumulated in quadruple precision to suppress the propagation of rounding errors during millions of iterative updates, thereby preserving the numerical stability of the energy estimate.

All the above variables are updated through the function `update_coordinate(det_picked, h, sub_xz)` in each iteration. Here, `det_picked` contains all selected coordinates together with their step sizes; `h` is a pointer to the Hamiltonian object that provides access to the corresponding matrix

columns; and `sub_xz` stores the triplets $\{(j, \boldsymbol{c}_j, \boldsymbol{b}_j)\}_{j \in \mathcal{I}_H(i^{(\ell+1)})}$, corresponding to coordinates that are modified during the update and will be used in the next iteration. The exact update rule depends on the chosen algorithm. The truncation threshold τ, which controls the sparsity of the wavefunction, is provided by the user via the parameter `z_threshold`.

The computational cost of the `update_coordinate` step scales with the number of states `NSTATES`, the average number of nonzeros per column of the Hamiltonian, and the overhead of modifying the underlying data structure (currently a hash map). As this step is often the most time-consuming part of the entire algorithm (see Section 4.3 for a complete breakdown of computational cost), this routine is parallelized when OpenMP support is enabled. Two levels of parallelism are employed: first, for multiple selected coordinates, their respective Hamiltonian columns are processed independently; second, within each column, the Hamiltonian interface `h` splits column construction into a serial (diagonal and singles) part and a parallel (doubles) part. The function `h.get_column_parallel_part` creates parallel tasks that ensure matching of singles and doubles workloads. Each task handles a portion of the column, accumulates its results into a private `sub_xz_parallel`, which is then merged into `sub_xz` in a critical OpenMP section.

Finally, the class provides the method `get_variational_energy()` to compute the Rayleigh quotient or solve the generalized eigenvalue problem for energy estimation. For the ground state (`NSTATES` $= 1$), the Rayleigh quotient is

$$E_0 = \frac{\alpha}{\mu}.$$

For `NSTATES` $= K > 1$, the method forms the generalized eigenproblem $Au = Mu\lambda$. The K smallest eigenvalues are then computed using `GeneralizedSelfAdjointEigenSolver` from the `Eigen` [44] library, and the eigenvalues are reported in ascending order.

4.3 Solvers for Ground and Excited States

We develop a unified solver framework capable of handling both ground and excited states within the same coordinate-descent paradigm. The generic `Solver<H,W>` class accepts user-defined Hamiltonian `H` and wavefunction `W` as template parameters, as well as pluggable `CoordinatePick` and `CoordinateUpdate` strategies.

The `Solver<H,W>::solve` routine performs the following operations in each iteration:

1. pick one or multiple determinants by `CoordinatePick`;
2. compute step size(s) and the possible scaling factor by `CoordinateUpdate`;
3. call `W::update_coordinate` to update $\boldsymbol{c}$ and $\boldsymbol{b}$ (or C and B);
4. call `W::get_variational_energy` to estimate energy and check convergence.

In the last step, the stopping criterion can be user-defined. For the single-coordinate ground-state algorithm, we adopt a damped accumulator

$$d^{(\ell+1)} \leftarrow \theta d^{(\ell)} + (1-\theta)\|\eta^{(\ell+1)}\|_2, \qquad \theta \in [0, 1),$$

with $\theta =$ `stopping_dx_damping_factor` and $t =$ `stopping_dx_threshold`. The run stops when $(1-\theta)d^{(\ell+1)} < t$. The same idea is naturally extended to multi-coordinate or multi-state variants.

The iteration loop employs a configurable `report_interval` to balance monitoring frequency and computational overhead. At each report, diagnostic quantities such as iteration count, variational energies, the damped accumulator, sparsity statistics, and elapsed wall time are recorded.

The solver further supports checkpointing for reliable termination and restart from stored wavefunctions, facilitating long-duration simulations.

The solver variants introduced in Section 3 are implemented as derived classes of the generic template `Solver<H, W>`. The subsequent subsections describe the design of each solver and analyze computational cost per iteration.

4.3.1 Ground-state CDFCI

The `CDFCISolver` realizes the ground-state algorithm by specializing the `CoordinatePick` and `CoordinateUpdate` strategies. For the single-coordinate formulation, it adopts a gradient-based selection with exact line search (`gcd_grad` + `ls`, see Alg. 1). For the multi-coordinate variant, it performs block selection and applies a small-scale eigensolver (`block_gcd_grad` + `eig`, see Alg. 2). Initialization can use either a single reference determinant, for example the Hartree–Fock state obtained from `H::get_hartree_fock`, or user-provided configurations.

Cost summary. Let $N_H = \max_i |\mathcal{I}_H(i)|$ denote the maximum number of nonzero entries in columns of the Hamiltonian matrix. For single-coordinate descent, the determinant selection step involves looping over N_H candidates, resulting in a computation cost of $\mathcal{O}(N_H)$ with a small prefactor. The subsequent line search and update of $\boldsymbol{c}$ incur only negligible $\mathcal{O}(1)$ cost. Updating $\boldsymbol{b}$ requires evaluating $\mathcal{O}(N_H)$ Hamiltonian entries and accessing N_H entries of $\boldsymbol{b}$ and $\boldsymbol{c}$, leading to an overall $\mathcal{O}(N_H)$ complexity. The prefactor here is determined by the per entry evaluation cost of the Hamiltonian, plus the cost of accessing the underlying data structure of the wavefunction. Consequently, the leading order of complexity per iteration is $\mathcal{O}(N_H)$.

For the multi-coordinate descent variant, assume that m coordinates are selected in each iteration. In the coordinate selection step, we maintain a size-m min-heap during a linear scan over all mN_H candidates to track the top m coordinates with the largest gradient magnitudes, combined with hash-based deduplication, reducing the complexity from the naive $\mathcal{O}(m^2 N_H)$ to $\mathcal{O}(m \log mN_H)$. The line search step is independent of N_H and scales with m^3, which is negligible when $N_H \gg m^3$. The computational bottleneck in this case is again updating the vector $\boldsymbol{b}$, which incurs a computation cost of $\mathcal{O}(mN_H)$ per iteration with a much larger prefactor than that of determinant selection. By distributing work across p parallel processes, the cost can be reduced to $\mathcal{O}(\frac{mN_H}{p})$.

4.3.2 Excited-state xCDFCI

The `XCDFCISolver` operates with `NSTATES` $= (K+1) > 1$ and initializes $(K+1)$ starting determinants $\{D_i^{(0)}\}_{i=1}^{K+1}$ by sorting the Hartree–Fock column according to diagonal energies,

$$D_i^{(0)} = \underset{\substack{j \notin \{D_1^{(0)}, \ldots, D_{i-1}^{(0)}\} \\ j \in \mathcal{I}_H(i_{\mathrm{HF}})}}{\arg\min} H_{j,j}, \quad i = 1, \ldots, K+1.$$

The initial coefficient matrix $C^{(0)}$ is constructed as

$$C_{i,j}^{(0)} = \begin{cases} 1, & i = D_j^{(0)}, \\ 0, & \text{otherwise}, \end{cases}$$

and $B^{(0)} = HC^{(0)}$. The subsequent iterations follow the procedure outlined in Alg. 3.

Cost summary. We use the same notation N_H to analyze the computational cost for xCDFCI solver. In the determinant selection step, each iteration loops over N_H candidates and evaluates the norm of the corresponding gradient row. Since this operation involves a multiplication with a $(K+1) \times (K+1)$ matrix, the total cost of this step is $\mathcal{O}(N_H(K+1)^2)$. The line search step is independent of N_H and scales with $(K+1)^2$, which is negligible in practice because typically $N_H \gg (K+1)^2$. Updating the coefficient matrices C and B requires accessing $\mathcal{O}(N_H)$ rows and updating each of the $(K+1)$ states, with an average constant-time cost per hashtable lookup, resulting in a total complexity of $\mathcal{O}(N_H(K+1))$. Therefore, the overall leading-order complexity of one iteration of xCDFCI is bounded by $\mathcal{O}(N_H(K+1)^2)$.

4.4 Example Usage

This section shows minimal, end-to-end examples for molecular Hamiltonians, using the ground-state CDFCI solver. We illustrate the basic input format, compilation, and execution workflow. Examples assume a `C++17` compiler and optionally OpenMP for shared-memory parallelism. More examples and a detailed `README` are provided in the `examples/` directory of the source code.

4.4.1 Input

The input follows the canonical JSON structure consisting of three blocks: `hamiltonian`, `solver`, and optional global settings such as `max_memory`. Below is a simple input configuration for a molecular Hamiltonian:

```
# input.json
{
  "hamiltonian": {
    "type": "molecule",
    "molecule": { "fcidump_path": "h2o_sto3g.FCIDUMP", "threshold": 0.0}
  },
  "solver":{
      "type": "cdfci",
      "cdfci": {
          "num_iterations": 150000,
          "report_interval": 10000,
          "z_threshold": 0,
          "z_threshold_search": false
      }
  },
  "max_memory": 0.05
}
```

For chemical molecules, the solver reads a standard `FCIDUMP` file containing one- and two-electron integrals, and a truncation threshold can be set to neglect small integral values. It then automatically constructs the Hartree–Fock (HF) reference determinant if not provided, and begins coordinate updates from the HF state. Typical parameters under the `solver/cdfci` block include: `num_iterations` and `report_interval`, which control the iteration loop and output frequency; `z_threshold`, which specifies truncation tolerance τ for the vector $\boldsymbol{b}$; and `z_threshold_search`, which can be used to automatically adjust compression to fit available memory. The parameter `max_memory` limits the in-memory wavefunction size (in GB).

4.4.2 Compile and Run

In a Linux environment, the program is compiled using a Makefile and run as a binary.

```
$ make cdfci
$ bin/cdfci input.json        # single-threaded version
$ bin/cdfci_omp input.json    # multi-threaded version
```

Compilation produces several solver binaries (e.g., `cdfci`, `cdfci_omp`, `xcdfci`), located in `bin/`. Execution reads the JSON file, loads the Hamiltonian, and iterates until convergence or the specified iteration limit. When OpenMP is enabled, the code automatically parallelizes Hamiltonian column construction and coordinate updates.

4.4.3 Expected Console Output

The following shows the expected output of the H_2O/STO-3G example, omitting the program header information (e.g., build, machine, input).

```
...
CDFCI calculation
-----------------
Reference determinant occupied spin-orbitals:
0  1  2  3  12  13  16  17
Reference energy: -75.5854987695

Iteration            Energy             dx     |x|_0     |z|_0  |H_i|_0      Time
10000        -75.7160281389     5.8949e-04      9109     58105      321      0.29
20000        -75.7160958597     9.9561e-05     17604     60811      409      0.50
30000        -75.7161047048     1.2322e-04     25284     61354      397      0.71
...
Final FCI Energy:           -75.7161071527957006
```

The central part of the output is a tabulated progress report printed every `report_interval` iterations. Each row lists the iteration number, current variational energy (in Hartree), step size `dx`, the number of nonzero coefficients in the current wavefunction (`|x|_0`) and residual (`|z|_0`), the number of nonzero Hamiltonian entries in selected column(s), and elapsed wall time. The final energy should converge near -75.716107 Hartree for the H_2O/STO-3G example, consistent with reference FCI values.

Additional examples, including OpenMP, multi-coordinate, and excited-state variants, are available in the `examples/` directory.

4.4.4 Python Interface

In addition, the same CDFCI calculations can be called from the Python `cdfci` module in the following way.

```
import cdfci
drv = cdfci.CDFCI("h2o_sto3g.FCIDUMP")
drv.set_num_iterations(150000)
drv.set_report_interval(10000)
```

```
res = drv.run()
print(res.energy)
```

More advanced usage, including additional solver options, input formats, and scripting workflows, can be found in the user manual.

5 Results and Discussion

In this section, we conduct a series of numerical experiments to evaluate the CDFCI software package on accuracy, robustness, and scalability.

The first validation test is the Benzene Blind Test [49], where the goal is to determine the frozen-core ground-state energy of the benzene molecule in a standard correlation-consistent basis set. Several established selected CI methods (SHCI, ASCI, iCI) have been benchmarked in the original paper; we compare our results with theirs. Next, we evaluate a subset of the QUEST database [50, 51, 52], which provides high-accuracy reference excitation energies for a broad range of molecular systems. After that, we turn to lattice Hamiltonians, where we test a 4×4 Hubbard lattice with periodic boundary conditions for different interaction strengths $U/t = 0.5, 4, 10$ and different electron fillings $n = 14, 15, 16$. These results are compared with reference data from exact diagonalization [53]. Finally, we run computations for N_2 in a basis set comprising 220 spin–orbitals, using different numbers of cores and demonstrate the strong parallel scaling for large systems.

All the experiments were conducted on a system equipped with two AMD EPYC 9754 128-core processors and 1.5 TB of memory. The program is compiled using the GNU `g++` compiler version 11.4.0 with the `--O3` optimization flag and its native OpenMP support. The orbitals and integrals are calculated via restricted Hartree–Fock (RHF) in the PSI4 package [54] version 1.8. All energies are reported in Hartree (Ha).

5.1 Accuracy on Representative Benchmarks

5.1.1 Benzene Blind Test

With the cc-pVDZ basis set, the benzene molecule contains 42 electrons and 114 spatial orbitals. After freezing the core electrons, the system still has 30 electrons and 108 spatial orbitals, i.e., 216 spin–orbitals, making it a challenging all-electron calculation. In [49], the authors benchmarked a series of state-of-the-art FCI methods, including SHCI, ASCI, iCI, DMRG, among others. The results show that these methods provide qualitatively similar estimates of the correlation energy (around -863 mHa), which is defined as the difference between the total energy and the Hartree–Fock energy, and usually used to quantify electron correlation effects. Selected CI methods employ a combination of variational estimation and second-order perturbative correction, and extrapolate energies obtained under different thresholds to the FCI limit. In this section, we perform calculations using the CDFCI software package on the same system with optimized orbitals [1], but we focus on the comparison of the variational energies and the number of determinants used.

Table 1 shows the variational correlation energy obtained by CDFCI under different thresholds τ, together with the number of determinants used and the computational time. Clearly, as τ decreases, the correlation energy gradually decreases, while the number of determinants and computational time increase significantly. For the most time-consuming calculation ($\tau = 0.0005$), CDFCI achieves a correlation energy of -806.12 mHa using approximately 2.08×10^7 determinants in about 37.3

[1] Obtained from `https://github.com/seunghoonlee89/SI-benzene-paper-DMRG`.

Table 1: Results of the benzene benchmark test, including the correlation energy, number of Slater determinants included, and computational time, obtained using 64 cores and different thresholds τ.

Threshold τ	**Correlation Energy (mHa)**	**Number of Determinants**	**Time (hours)**
0.01	−716.18	181 253	2.9
0.005	−741.97	533 932	5.8
0.004	−752.84	893 174	6.6
0.002	−774.64	2 870 775	22.0
0.001	−790.45	7 513 377	30.1
0.0005	−806.12	20 832 092	37.3

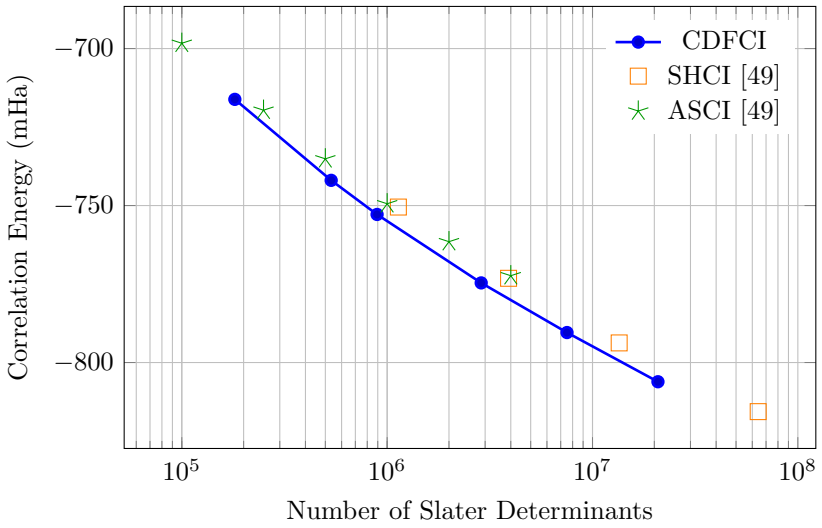

Figure 1: Relationship between the correlation energy and the number of Slater determinants in the variational space in the benzene benchmark test. CDFCI results are compared with SHCI and ASCI reported data [49].

hours, which corresponds to approximately $2.4k$ core-hours, and is comparable to other CI methods reported in [49].

Figure 1 shows an approximately linear relationship between the correlation energy and the number of determinants for CDFCI under different thresholds τ (blue solid line). For comparison, we also include reference data for SHCI and ASCI from [49]. Both SHCI data points (orange squares) and ASCI data points (green stars) indicate that, for a similar number of determinants, the correlation energy obtained by SHCI or ASCI is slightly higher than that of CDFCI. This observation demonstrates the efficiency of CDFCI in capturing important determinants, allowing it to achieve better correlation energy estimates with fewer determinants, thus validating the effectiveness of gradient- or residual-based determinant selection strategies.

Figure 2 further analyzes different aspects of the CDFCI results. The left plot shows the decrease of correlation energy with respect to τ, and exhibits an approximately linear relationship. The middle plot shows the relationship between computational time and the number of determinants, indicating that the growth of computational time approximately follows a power-law trend. The right plot shows the relationship between the number of Slater determinants and τ, where the number of determinants increases approximately following a power-law trend as τ decreases,

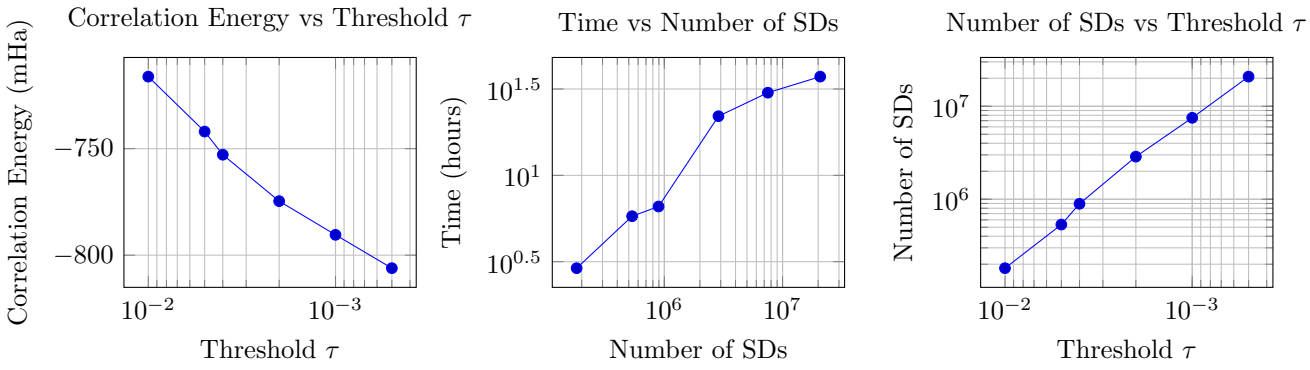

Figure 2: Benzene benchmark test: The left plot shows the effect of threshold τ on the correlation energy, the middle plot shows the effect of the number of Slater determinants (SDs) on computational time, and the right plot shows the effect of threshold τ on the number of SDs.

consistent with the time growth observed in the middle plot.

5.1.2 QUEST Database Subset

To evaluate the performance of xCDFCI in excited-state calculations, we selected three representative molecules from the QUEST dataset [51], covering different types of excitations, including singlet states, triplet states, valence states, and Rydberg states. This benchmark aims to demonstrate that xCDFCI can achieve systematic and reliable accuracy across different excitation types. We report the final vertical excitation energies for each state, as well as the exFCI reference results, obtained via extrapolation from FCI results. The geometries used in the QUEST dataset are optimized at the CC3/aug-cc-pVTZ level. We use the same geometries to perform Hartree–Fock calculations to obtain one- and two-body integrals that are subsequently used in the xCDFCI calculations.

Table 2: Vertical excitation energies of excited states for H_2O/aug-cc-pVTZ in the QUEST dataset, compared with exFCI reference data [51], along with the number of Slater determinants (SDs) used in exFCI. In the xCDFCI calculations, the number of determinants used is 4 935 670.

Excited State	**Vertical Excitation Energy (eV)**	**exFCI Reference (eV)**	**Number of SDs in exFCI**
$^3B_1(n \to 3s)$	7.26	7.24	5 950 423
$^1B_1(n \to 3s)$	7.63	7.62	5 589 200
$^3A_2(n \to 3p)$	9.29	9.24	3 760 373
$^1A_2(n \to 3p)$	9.43	9.41	5 589 200
$^3A_1(n \to 3s)$	9.55	9.54	3 760 373
$^1A_1(n \to 3s)$	9.99	9.99	5 589 200

Water (H_2O), due to its simple structure and important chemical properties, is a classical test system for Rydberg excitation calculations. We compute the ground state and six excited states of water using the aug-cc-pVTZ basis set, including two triplet states and four singlet states, involving excitations from the nonbonding n orbital to Rydberg $3s$ and $3p$ orbitals. To compare with reference data, we use a threshold of $\tau = 0.001$ to control the computational space within the same order of magnitude (approximately 4 935 670 determinants). The xCDFCI program runs in parallel using 8

threads and converges in about one day. Table 2 summarizes the final vertical excitation energies and their errors relative to the reference data. It can be seen that the excitation energies obtained by xCDFCI are very close to the exFCI reference values. Except for the $^3A_2(n \to 3p)$ state, the errors of the other five excited states are within 0.02 eV. Moreover, the excitation energies obtained by xCDFCI are generally lower than the exFCI reference values. Since Rydberg states are more sensitive to the basis set, the correlation effects captured in this system may lead to a systematic lowering of excitation energies, bringing them closer to results obtained with the aug-cc-pVQZ basis set.

Table 3: Vertical excitation energies of excited states for N_2/aug-cc-pVDZ in the QUEST dataset, compared with exFCI reference data [51], along with the number of Slater determinants (SDs) used in exFCI. In the xCDFCI calculations, the number of determinants used is 20 558 436.

Excited State	**Vertical Excitation Energy (eV)**	**exFCI Reference (eV)**	**Number of SDs in exFCI**
$^3\Sigma_u^+(\pi \to \pi^*)$	7.67	7.70	8 139 401
$^3\Pi_g(n \to \pi^*)$	8.08	8.05	2 705 349
$^3\Delta_u(\pi \to \pi^*)$	8.94	8.96	2 705 349
$^1\Pi_g(n \to \pi^*)$	9.44	9.41	2 775 773
$^3\Sigma_u^-(\pi \to \pi^*)$	9.73	9.75	2 705 349
$^1\Sigma_u^-(\pi \to \pi^*)$	10.03	10.05	2 775 773
$^1\Delta_u(\pi \to \pi^*)$	10.41	10.43	2 775 773

Nitrogen (N_2) is a strongly correlated system. We compute its ground state and seven excited states using the aug-cc-pVDZ basis set. Among them, the $^3\Delta_u(\pi \to \pi^*)$ and $^1\Delta_u(\pi \to \pi^*)$ states are doubly degenerate, so in the actual xCDFCI calculations, we compute nine excited states to cover all degeneracies. We use a threshold of $\tau = 0.004$ to control the computational space within a comparable size (approximately 20 558 436 determinants). The xCDFCI program runs in parallel using 8 threads and converges in about 12 hours. Table 3 summarizes the final vertical excitation energies and their errors relative to the reference data. It can be seen that the deviations of xCDFCI excitation energies from exFCI reference values are within 0.03 eV. Unlike the water system, there is no clear systematic bias in the energy differences for this system.

Table 4: Vertical excitation energies of excited states for C_2H_2/aug-cc-pVDZ in the QUEST dataset, compared with exFCI reference data [51], along with the number of Slater determinants (SDs) used in exFCI. In the xCDFCI calculations, the number of determinants used is 27 874 551.

Excited State	**Vertical Excitation Energy (eV)**	**exFCI Reference (eV)**	**Number of SDs in exFCI**
$^3\Sigma_u^+(\pi \to \pi^*)$	5.51	5.50	8 494 075
$^3\Delta_u(\pi \to \pi^*)$	6.46	6.46	4 403 434
$^3\Sigma_u^-(\pi \to \pi^*)$	7.13	7.14	4 403 434
$^1\Sigma_u^-(\pi \to \pi^*)$	7.20	7.20	4 162 848
$^1\Delta_u(\pi \to \pi^*)$	7.51	7.51	4 162 848

Acetylene (C_2H_2) is the smallest conjugated organic molecule with stable low-lying excited states, and is therefore important for studying vertical fluorescence transitions. We compute its five lowest-energy valence excited states in a linear geometry, including two doubly degenerate Δ

states. We use a threshold of $\tau = 0.0002$ to control the computational space within the same order of magnitude (approximately 27 874 551 determinants). The xCDFCI program runs in parallel using 8 threads and converges in about 36 hours. Table 4 summarizes the final vertical excitation energies and their errors relative to the reference data. It can be seen that, although it uses more determinants than exFCI, the excitation energies obtained by xCDFCI agree very well with exFCI reference values, with deviations within 0.01 eV.

5.1.3 2D Hubbard Model

We consider a 4×4 Hubbard lattice with periodic boundary conditions (PBC), with interaction strength $U/t = 4$, and electron fillings $n = 14, 15, 16$. These results demonstrate the capability of CDFCI in handling lattice Hamiltonians, and are compared with reference data from exact diagonalization [53].

The initialization for the Hubbard model is more complicated than that for molecular systems. In molecular calculations, one typically starts from the Hartree–Fock reference state, in which electrons occupy the orbitals with lowest energies. The Hartree–Fock procedure guarantees the presence of an energy gap between the highest occupied molecular orbital (HOMO) and the lowest unoccupied molecular orbital (LUMO). In contrast, for the Hubbard model, localized site states can be degenerate in energy levels, leading to multiple configurations that share the same minimal energy. In the current implementation, we identify all such degenerate lowest-energy configurations and extract the corresponding submatrix of the Hamiltonian. The eigenvector associated with the smallest eigenvalue of this submatrix is then used as the initial state for the CDFCI algorithm. For example, in the 4×4 PBC Hubbard model, 16 site states consist of 5 states with negative energy, 6 with zero energy, and 5 with positive energy. At half filling ($n = 16$) and under the constraint of total spin zero, there are $\binom{6}{3}^2 = 20^2 = 400$ distinct configurations that all attain the same lowest energy. We construct the Hamiltonian submatrix corresponding to these 400 determinants and compute its ground-state eigenvector as the initialization. Numerical experiments show that this strategy yields significantly better performance than alternative initialization approaches for this system.

Figure 3 shows the ground-state energy convergence curves of the two-dimensional Hubbard model with periodic boundary conditions at $U/t = 4$ for different electron fillings $n = 14, 15, 16$. For $n = 14$, CDFCI converges to an energy of -15.74449 Ha within a variational space of approximately 8×10^6 determinants, with a deviation of about 0.1 mHa from the exact diagonalization result -15.74459 Ha. For $n = 15$, CDFCI converges to an energy of -14.66512 Ha within a variational space of approximately 9×10^6 determinants, with a deviation of about 0.1 mHa from the exact diagonalization result -14.66524 Ha. For $n = 16$, CDFCI converges to an energy of -13.62185 Ha with approximately 1×10^7 determinants, which is identical to the exact diagonalization result. From the figure, it can be observed that for $n = 14$ and $n = 15$, the convergence of the energy is relatively slow, whereas for $n = 16$, the convergence is significantly faster. The convergence behavior of the ground-state energy in the Hubbard model is closely related to the electron filling. For $n = 14$ and $n = 15$, the system is in a lightly doped regime with strong electron correlation, leading to slower convergence. For $n = 16$, the system is at half filling with weaker correlation effects, resulting in faster convergence.

Table 5 summarizes the comparison of ground-state energies and computational time for the two-dimensional 4×4 Hubbard model with periodic boundary conditions at half filling ($n = 16$) under different U/t values. Significant differences in computational time can be observed. For weakly correlated systems ($U/t = 0.5$), CDFCI converges to the exact diagonalization energy within about 13 seconds. For moderately correlated systems ($U/t = 4$), CDFCI converges within about

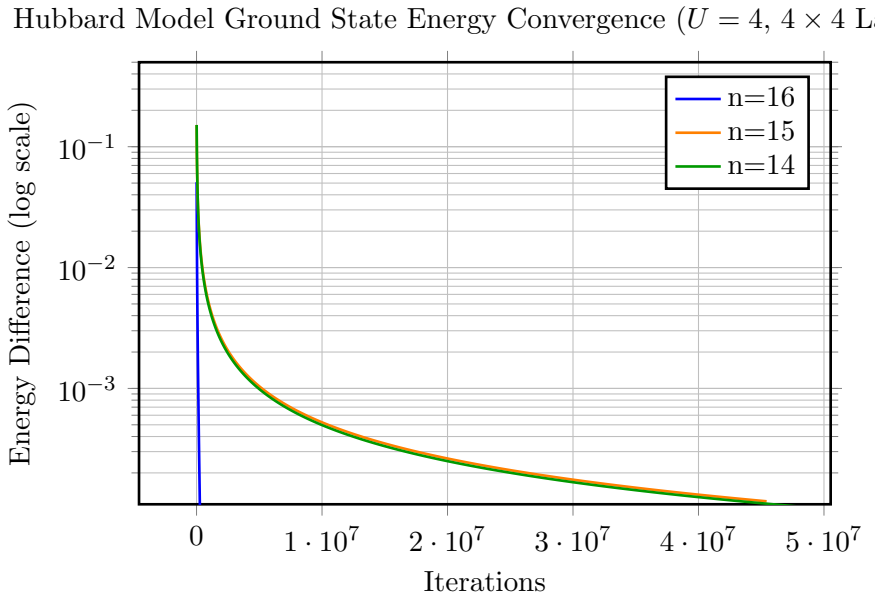

Figure 3: Ground-state energy convergence curves of the two-dimensional Hubbard model with periodic boundary conditions at $U/t = 4$ for different electron fillings. Reference data are from exact diagonalization [53].

Table 5: Ground-state energies and computational time for the two-dimensional 4×4 half-filled Hubbard model with periodic boundary conditions at $U/t = 0.5, 4, 10$. Reference data are from exact diagonalization [53].

U/t	**CDFCI Energy (Ha)**	**Exact Diagonalization Energy (Ha)**	**Time**
0.5	-22.34023	-22.34023	13 seconds
4	-13.62185	-13.62185	9 minutes
10	-7.02682	-7.02900	12 hours

9 minutes. For strongly correlated systems ($U/t = 10$), CDFCI converges to -7.02682 Ha within about 12 hours, with a deviation of approximately 2.2 mHa from the exact diagonalization result -7.02900 Ha. These results indicate that CDFCI performs well across different correlation regimes of the Hubbard model. However, for strongly correlated systems, energy convergence requires longer computational time, suggesting that more efficient optimization algorithms may be needed to accelerate convergence in strongly correlated regimes.

5.2 Scalability

Finally, we evaluate the parallel scalability of CDFCI under different computational resources. We fix the problem size and workload to evaluate the strong scaling performance of CDFCI as computational resources increase. For the N_2/cc-pVQZ system, we run CDFCI ground-state calculations on different numbers of CPU cores, and record the total runtime and speedup for each calculation. We test 8, 16, 32, 64, 128, and 256 CPU cores. The threshold is set to $\tau = 5 \times 10^{-5}$, with 128 coordinates updated in each iteration, and after 400 000 iterations the ground-state energy obtained is -109.46134 Ha. This result differs from the fully converged energy -109.46256 Ha by about 1.3 mHa, which is within chemical accuracy.

Table 6: Runtime and speedup of CDFCI on the N_2/cc-pVQZ system using different numbers of CPU cores. The speedup is defined as $8 \times T_8/T_N$, where T_8 is the runtime using 8 cores and T_N is the runtime using N cores.

Number of CPU Cores	**Runtime (seconds)**	**Speedup**
8	116904.5	8.0
16	54128.3	17.3
32	27514.0	34.0
64	15913.6	58.8
128	12566.7	74.4
256	9375.6	99.8

Table 6 summarizes the runtime and speedup for different numbers of CPU cores. It can be seen that, as the number of CPU cores increases, the runtime decreases significantly, and even superlinear speedup is observed at 16 and 32 cores. However, when the number of CPU cores increases beyond 64, the growth in speedup begins to diminish noticeably.

Figure 4 provides a more intuitive illustration. The top panel shows the decrease in runtime as the number of CPU cores increases, compared with the ideal linear scaling. The bottom panel presents the parallel efficiency, defined as the speedup divided by the number of processors. When the number of CPU cores increases from 8 to 16 and 32, a slight superlinear speedup can be observed. This phenomenon is usually attributed to improved cache utilization and memory hierarchy effects. More specifically, as the workload per thread decreases, the effective working set becomes smaller and fits better into the last-level cache, thereby reducing memory latency and improving effective bandwidth. Since the CDFCI algorithm is mainly memory-bound due to irregular hash-table accesses and Hamiltonian evaluation, improved cache locality can lead to noticeable superlinear scaling at moderate CPU core counts. However, when the number of CPU cores exceeds 64, the speedup gradually saturates. This indicates that memory bandwidth becomes a limiting factor and synchronization overhead increases, which is common in large-scale shared-memory parallelization of sparse algorithms. Overall, CDFCI demonstrates good parallel scalability on the N_2/cc-pVQZ system, especially for moderate numbers of CPU cores, although the performance gain becomes

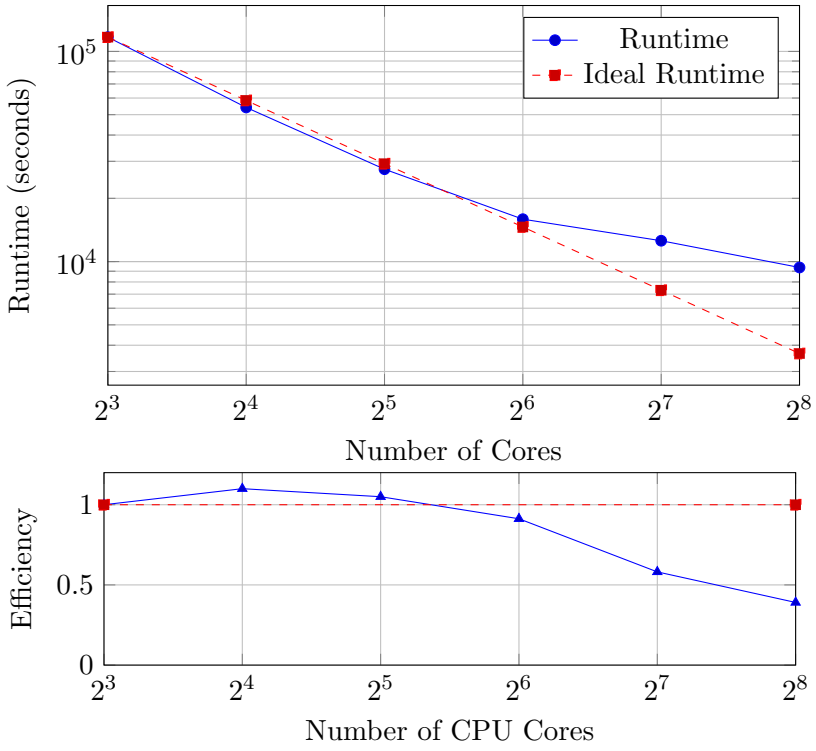

Figure 4: Strong scaling of CDFCI on the N_2/cc-pVQZ system: the top panel shows runtime (log scale) and speedup (including ideal linear scaling), and the bottom panel shows efficiency, defined as speedup divided by the number of processors.

limited in larger-scale parallel computations.

6 Conclusion and Future Work

We have presented CDFCI, an efficient and modular software package for solving the non-relativistic, time-independent fermionic Schrödinger equation within the full configuration interaction framework. CDFCI leverages coordinate descent methods to iteratively refine the wavefunction representation, achieving high accuracy with reduced computational cost. With appropriately defined objective functions, the framework can compute multiple states simultaneously. The software architecture is modular and extensible, facilitating integration of new Hamiltonian types, solvers, and parallel backends. Shared-memory parallelization enables efficient utilization of modern multi-core processors. Benchmark results demonstrate that CDFCI achieves sub-millihartree accuracy on a variety of molecular systems and lattice models.

Future work will focus on incorporating parallel linear algebra libraries to enhance performance, and exploring more advanced perturbation theory techniques to further improve the accuracy of the computed energies. Another direction is to go beyond non-relativistic fermionic Hamiltonians, that is, to extend the framework to bosonic systems and relativistic Hamiltonians. This will broaden the applicability of CDFCI and position it as a general solver across quantum chemistry and condensed matter physics.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 12271109 and 12526211; by the U.S. National Science Foundation (NSF) under Grant Nos. DMS-1454939 and DMS-2012286; by the Shanghai Pilot Program for Basic Research-Fudan University under Grant No. 21TQ1400100 (22TQ017); by the Scientific Research Innovation Capability Support Project for Young Faculty under Grant No. SRICSPYF-ZY2025159; and by the Xuemin Institute of Advanced Studies, Fudan University.

References

[1] Ernest R. Davidson. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. *Journal of Computational Physics*, 17:87–94, 1975.

[2] Chr. Møller and M. S. Plesset. Note on an approximation treatment for many-electron systems. *Physical Review*, 46:618–622, 1934.

[3] J. A. Pople, R. Seeger, and R. Krishnan. Variational Configuration Interaction methods and comparison with perturbation theory. *International Journal of Quantum Chemistry*, S4:149–163, 1977.

[4] B. Huron, J.-P. Malrieu, and P. Rancurel. Iterative perturbation calculations of ground and excited state energies from multiconfigurational zeroth-order wavefunctions. *The Journal of Chemical Physics*, 58:5745–5759, 1973.

[5] Norm M. Tubman, C. Daniel Freeman, Daniel S. Levine, Diptarka Hait, Martin Head-Gordon, and K. Birgitta Whaley. Modern approaches to exact diagonalization and selected Configuration Interaction with the adaptive sampling CI method. *Journal of Chemical Theory and Computation*, 16(4):2139–2159, April 2020.

[6] Kyeong Su Min and Jae Woo Park. Second-order Complete Active Space Perturbation Theory (CASPT2) and N-Electron Valence state Perturbation Theory (NEVPT2) based on Adaptive Sampling Configuration Interaction Self-Consistent Field (ASCI-SCF). *Journal of Chemical Theory and Computation*, 21(11):5425–5436, June 2025.

[7] Adam A. Holmes, Nathan M. Tubman, and C. J. Umrigar. Heat-bath Configuration Interaction: An efficient selected Configuration Interaction algorithm inspired by heat-bath sampling. *Journal of Chemical Theory and Computation*, 12(8):3674–3680, 2016.

[8] Mihkel Ugandi and Michael Roemelt. A configuration-based heatbath-CI for spin-adapted multireference electronic structure calculations with large active spaces. *Journal of Computational Chemistry*, 44(31):2374–2390, 2023. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.27203.

[9] Sandeep Sharma, Adam A. Holmes, Gregory Jeanmairet, Ali Alavi, and C. J. Umrigar. Semistochastic heat-bath Configuration Interaction method: Selected Configuration Interaction with semistochastic perturbation theory. *Journal of Chemical Theory and Computation*, 13(4):1595–1604, 2017.

[10] Xubo Wang and Sandeep Sharma. Relativistic Semistochastic Heat-Bath Configuration Interaction. *Journal of Chemical Theory and Computation*, 19(3):848–855, February 2023.

[11] Ning Zhang, Wenjian Liu, and Mark R. Hoffmann. Iterative Configuration Interaction with selection. *Journal of Chemical Theory and Computation*, 16(4):2296–2316, April 2020. Publisher: American Chemical Society.

[12] Samuel M. Greene, Robert J. Webber, Jonathan Weare, and Timothy C. Berkelbach. Improved fast randomized iteration approach to full Configuration Interaction. *Journal of Chemical Theory and Computation*, 16(9):5572–5585, sep 2020.

[13] Joshua J. Goings, Hang Hu, Chao Yang, and Xiaosong Li. Reinforcement learning Configuration Interaction. *Journal of Chemical Theory and Computation*, 17(9):5482–5491, sep 2021.

[14] David B. Williams-Young, Norm M. Tubman, Carlos Mejuto-Zaera, and Wibe A. de Jong. A parallel, distributed memory implementation of the adaptive sampling Configuration Interaction method. *The Journal of Chemical Physics*, 158(21):214109, June 2023.

[15] Duy-Khoi Dang, Joshua A. Kammeraad, and Paul M. Zimmerman. Advances in parallel heat bath Configuration Interaction. *The Journal of Physical Chemistry A*, 127(1):400–411, January 2023.

[16] Junhao Li, Matthew Otten, Adam A. Holmes, Sandeep Sharma, and C. J. Umrigar. Fast semistochastic heat-bath Configuration Interaction. *The Journal of Chemical Physics*, 149(21):214110, December 2018.

[17] Yann Garniron, Thomas Applencourt, Kevin Gasperich, Anouar Benali, Anthony Ferté, Julien Paquier, Barthélémy Pradines, Roland Assaraf, Peter Reinhardt, Julien Toulouse, Pierrette Barbaresco, Nicolas Renon, Grégoire David, Jean-Paul Malrieu, Mickaël Véril, Michel Caffarel, Pierre-François Loos, Emmanuel Giner, and Anthony Scemama. Quantum Package 2.0: An open-source determinant-driven suite of programs. *Journal of Chemical Theory and Computation*, 15(6):3591–3609, June 2019. arXiv:1902.08154 [physics].

[18] E. Y. Loh, J. E. Gubernatis, R. T. Scalettar, Steven R. White, D. J. Scalapino, and R. L. Sugar. Sign problem in the numerical simulation of many-electron systems. *Physical Review B*, 41:9301–9307, 1990.

[19] George H. Booth, Alex J. W. Thom, and Ali Alavi. Fermion Monte Carlo without fixed nodes: A game of life, death, and annihilation in Slater determinant space. *The Journal of Chemical Physics*, 131(5):054106, 2009.

[20] J. S. Spencer, N. S. Blunt, and W. M. C. Foulkes. The sign problem and population dynamics in the full Configuration Interaction quantum Monte Carlo method. *The Journal of Chemical Physics*, 136:054110, 2012.

[21] Deidre Cleland, George H. Booth, and Ali Alavi. Communications: Survival of the fittest: Accelerating convergence in full Configuration-Interaction quantum Monte Carlo. *The Journal of Chemical Physics*, 132(4):041103, 2010.

[22] F. R. Petruzielo, A. A. Holmes, H. J. Changlani, M. P. Nightingale, and C. J. Umrigar. Semistochastic projector Monte Carlo method. *Physical Review Letters*, 109:230201, 2012.

[23] N. S. Blunt, J. J. Shepherd, D. K. K. Lee, G. H. Booth, and A. Alavi. Semi-stochastic full Configuration Interaction quantum Monte Carlo: Developments and application. *The Journal of Chemical Physics*, 142:184107, 2015.

[24] George H. Booth, Andreas Grüneis, Georg Kresse, and Ali Alavi. Towards an exact description of electronic wavefunctions in real solids. *Nature*, 493(7432):365–370, January 2013.

[25] Kai Guther, Robert J. Anderson, Nick S. Blunt, Nikolay A. Bogdanov, Deidre Cleland, Nike Dattani, Werner Dobrautz, Khaldoon Ghanem, Peter Jeszenszki, Niklas Liebermann, Giovanni Li Manni, Alexander Y. Lozovoi, Hongjun Luo, Dongxia Ma, Florian Merz, Catherine Overy, Markus Rampp, Pradipta Kumar Samanta, Lauretta R. Schwarz, James J. Shepherd, Simon D. Smart, Eugenio Vitale, Oskar Weser, George H. Booth, and Ali Alavi. NECI: N-electron Configuration Interaction with an emphasis on state-of-the-art stochastic methods. *The Journal of Chemical Physics*, 153(3):034107, July 2020.

[26] Steven R. White. Density-matrix algorithms for quantum renormalization groups. *Physical Review B*, 48:10345–10356, 1993.

[27] Steven R. White and Richard L. Martin. Ab initio quantum chemistry using the Density Matrix Renormalization Group. *The Journal of Chemical Physics*, 110:4127–4130, 1999.

[28] Ulrich Schollwöck. The Density-Matrix Renormalization Group in the age of matrix product states. *Annals of Physics*, 326(1):96–192, January 2011.

[29] Sandeep Sharma and Garnet Kin-Lic Chan. Spin-adapted Density Matrix Renormalization Group algorithms for quantum chemistry. *The Journal of Chemical Physics*, 136(12):124121, March 2012.

[30] Huanchen Zhai and Garnet Kin-Lic Chan. Low communication high performance *ab initio* Density Matrix Renormalization Group algorithms. *The Journal of Chemical Physics*, 154(22):224116, June 2021.

[31] Ryan Levy, Edgar Solomonik, and Bryan K Clark. Distributed-memory DMRG via sparse and dense parallel tensor contractions. November 2020.

[32] Andor Menczer, Maarten van Damme, Alan Rask, Lee Huntington, Jeff Hammond, Sotiris S. Xantheas, Martin Ganahl, and Örs Legeza. Parallel implementation of the Density Matrix Renormalization Group method achieving a quarter petaFLOPS performance on a single DGX-H100 GPU node. *Journal of Chemical Theory and Computation*, 20(19):8397–8404, October 2024. Publisher: American Chemical Society.

[33] Enhua Xu, Motoyuki Uejima, and Seiichiro L. Ten-no. Towards near-exact solutions of molecular electronic structure: Full Coupled-Cluster Reduction with a second-order perturbative correction. *The Journal of Physical Chemistry Letters*, 11(22):9775–9780, November 2020. Publisher: American Chemical Society.

[34] Janus J. Eriksen and Jürgen Gauss. Many-body expanded full Configuration Interaction. I. Weakly correlated regime. *Journal of Chemical Theory and Computation*, 14(10):5180–5191, October 2018. Publisher: American Chemical Society.

[35] Janus J. Eriksen and Jürgen Gauss. Many-body expanded full Configuration Interaction. II. Strongly correlated regime. *Journal of Chemical Theory and Computation*, 15(9):4873–4884, September 2019. Publisher: American Chemical Society.

[36] Zhe Wang, Yingzhou Li, and Jianfeng Lu. Coordinate descent full Configuration Interaction. *Journal of Chemical Theory and Computation*, 15(6):3558–3569, 2019.

[37] Yingzhou Li, Jianfeng Lu, and Zhe Wang. Coordinatewise descent methods for leading eigenvalue problem. *SIAM Journal on Scientific Computing*, 41(4):A2681–A2716, 2019.

[38] Zhe Wang, Zhiyuan Zhang, Jianfeng Lu, and Yingzhou Li. Coordinate descent full Configuration Interaction for excited states. *Journal of Chemical Theory and Computation*, 19(21):7731–7739, 2023.

[39] Yuejia Zhang, Weiguo Gao, and Yingzhou Li. Parallel multicoordinate descent methods for full Configuration Interaction. *Journal of Chemical Theory and Computation*, 21(5):2325–2337, 2025.

[40] Yingzhou Li and Jianfeng Lu. Optimal orbital selection for full Configuration Interaction (OptOrbFCI): Pursuing the basis set limit under a budget. *Journal of Chemical Theory and Computation*, 16(10):6207–6221, 2020.

[41] Bjørn O. Roos. The complete active space self-consistent field method and its applications in electronic structure calculations. In K. P. Lawley, editor, *Advances in Chemical Physics: Ab Initio Methods in Quantum Chemistry Part 2*, volume 69, pages 399–445. John Wiley & Sons, New York, 1987.

[42] Weiguo Gao, Yingzhou Li, and Bichen Lu. Triangularized orthogonalization-free method for solving extreme eigenvalue problems. *Journal of Scientific Computing*, 93(3):63, 2022.

[43] Stephen J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151:3–34, 2015.

[44] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. `http://eigen.tuxfamily.org`, 2010.

[45] Martin Leitner-Ankerl. Robin hood hashing. `https://github.com/martinus/robin-hood-hashing`. Accessed: 2025-10-08.

[46] Bin Fan, David G. Andersen, and Michael Kaminsky. MemC3: Compact and concurrent memcache with dumber caching and smarter hashing. In *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 371–384. USENIX Association, 2013.

[47] Xi Wang Li, David G. Andersen, Michael Kaminsky, and Michael J. Freedman. Algorithmic improvements for fast concurrent cuckoo hashing. In *Proceedings of the Ninth European Conference on Computer Systems (EuroSys '14)*, pages 27:1–27:14. ACM, 2014.

[48] Efficient Systems Lab. A high-performance concurrent hash table (libcuckoo). `https://github.com/efficient/libcuckoo`. Accessed: 2025-10-08.

[49] Janus J. Eriksen, Tyler A. Anderson, J. Emiliano Deustua, Khaldoon Ghanem, Diptarka Hait, Mark R. Hoffmann, Seunghoon Lee, Daniel S. Levine, Ilias Magoulas, Jun Shen, Norm M. Tubman, K. Birgitta Whaley, Enhua Xu, Yuan Yao, Ning Zhang, Ali Alavi, Garnet Kin-Lic Chan, Martin Head-Gordon, Wenjian Liu, Piotr Piecuch, Sandeep Sharma, Seiichiro L. Tenno, C. J. Umrigar, and Jürgen Gauss. The Ground State Electronic Energy of Benzene. *The Journal of Physical Chemistry Letters*, 11(20):8922–8929, 2020.

[50] Mickaël Véril, Anthony Scemama, Michel Caffarel, Filippo Lipparini, Mario Boggio-Pasqua, Denis Jacquemin, and Pierre-François Loos. QUESTDB: A database of highly accurate excitation energies for the electronic structure community. *WIREs Computational Molecular Science*, 11(5):e1517, 2021.

[51] Pierre-François Loos, Anthony Scemama, Aymeric Blondel, Yann Garniron, Michel Caffarel, and Denis Jacquemin. A mountaineering strategy to excited states: Highly accurate reference energies and benchmarks. *Journal of Chemical Theory and Computation*, 14(8):4360–4379, 2018.

[52] Pierre-François Loos and collaborators. Quest database of highly accurate excitation energies. `https://github.com/pfloos/QUESTDB`, 2021. Accessed: 2025-10-08.

[53] Elbio Dagotto. Correlated electrons in high-temperature superconductors. *Reviews of Modern Physics*, 66(3):763–840, 1994.

[54] Justin M. Turney, Andrew C. Simmonett, Robert M. Parrish, Edward G. Hohenstein, Francesco A. Evangelista, Justin T. Fermann, Benjamin J. Mintz, Lori A. Burns, Jeremiah J. Wilke, Micah L. Abrams, Nicholas J. Russ, Matthew L. Leininger, Curtis L. Janssen, Edward T. Seidl, Wesley D. Allen, Henry F. Schaefer, Rollin A. King, Edward F. Valeev, C. David Sherrill, and T. Daniel Crawford. Psi4: An open-source *ab initio* electronic structure program. *WIREs Computational Molecular Science*, 2(4):556–565, 2012.

[55] R. W. D. Nickalls. A new approach to solving the cubic: Cardan's solution revealed. *The Mathematical Gazette*, 77(480):354–359, 1993.

A Determining Stepsize in Problem (15) and (28)

We describe in detail how to minimize the quartic polynomial in Problem (15) and (28), thereby determining the optimal step size η.

In Problem (15), we have

$$h(\eta) = f(\boldsymbol{c}^{(\ell)} + \eta \boldsymbol{e}_{i^{(\ell+1)}}) = \eta^4 + v_3^{(\ell)}\eta^3 + v_2^{(\ell)}\eta^2 + v_1^{(\ell)}\eta + v_0^{(\ell)}, \tag{33}$$

with

$$\begin{aligned} v_0^{(\ell)} &= f(\boldsymbol{c}^{(\ell)}), \\ v_1^{(\ell)} &= 4\boldsymbol{b}_{i^{(\ell+1)}}^{(\ell)} + 4\mu^{(\ell)}\boldsymbol{c}_{i^{(\ell+1)}}^{(\ell)}, \\ v_2^{(\ell)} &= 2H_{i^{(\ell+1)},i^{(\ell+1)}} + 4(\boldsymbol{c}_{i^{(\ell+1)}}^{(\ell)})^2 + 2\mu^{(\ell)}, \\ v_3^{(\ell)} &= 4\boldsymbol{c}_{i^{(\ell+1)}}^{(\ell)}. \end{aligned} \tag{34}$$

The stationary condition $\frac{\mathrm{d}h(\eta)}{\mathrm{d}\eta} = 0$ leads to a cubic equation of the form

$$\frac{\mathrm{d}h(\eta)}{\mathrm{d}\eta} = 4\eta^3 + 3v_3^{(\ell)}\eta^2 + 2v_2^{(\ell)}\eta + v_1^{(\ell)} = 0. \tag{35}$$

In Problem (28), we have

$$h(\eta) = f(C^{(\ell)} + \eta \boldsymbol{e}_{i^{(\ell+1)}} G_{i^{(\ell+1)},:}^{(\ell)}) = w_4^{(\ell)}\eta^4 + w_3^{(\ell)}\eta^3 + w_2^{(\ell)}\eta^2 + w_1^{(\ell)}\eta + w_0^{(\ell)}, \tag{36}$$

with

$$
\begin{aligned}
w_0^{(\ell)} &= f(C^{(\ell)}),\\
w_1^{(\ell)} &= 4\left\|G^{(\ell)}_{i^{(\ell+1)},:}\right\|^2,\\
w_2^{(\ell)} &= 2H_{i^{(\ell+1)},i^{(\ell+1)}}\left\|G^{(\ell)}_{i^{(\ell+1)},:}\right\|^2 + 2\left(C^{(\ell)}_{i^{(\ell+1)},:}\left(G^{(\ell)}\right)^\top_{i^{(\ell+1)},:}\right)^2 + 2\left\|C^{(\ell)}_{i^{(\ell+1)},:}\right\|^2\left\|G^{(\ell)}_{i^{(\ell+1)},:}\right\|^2\\
&\quad + 2G^{(\ell)}_{i^{(\ell+1)},:}M^{(\ell)}\left(G^{(\ell)}\right)^\top_{i^{(\ell+1)},:},\\
w_3^{(\ell)} &= 4\left\|G^{(\ell)}_{i^{(\ell+1)},:}\right\|^2 C^{(\ell)}_{i^{(\ell+1)},:}\left(G^{(\ell)}\right)^\top_{i^{(\ell+1)},:},\\
w_4^{(\ell)} &= \left\|G^{(\ell)}_{i^{(\ell+1)},:}\right\|^4.
\end{aligned}
\tag{37}
$$

The stationary condition $\frac{\mathrm{d}h(\eta)}{\mathrm{d}\eta} = 0$ leads to a cubic equation of the form

$$
\frac{\mathrm{d}h(\eta)}{\mathrm{d}\eta} = 4w_4^{(\ell)}\eta^3 + 3w_3^{(\ell)}\eta^2 + 2w_2^{(\ell)}\eta + w_1^{(\ell)} = 0. \tag{38}
$$

To solve the cubic equations (35) and (38), we use Cardano's method [55]. The method first converts the cubic equation into a depressed cubic form $x^3 + px + q = 0$ via the substitution $\eta = x - \frac{b}{3a}$, where a, b, c, and d are the coefficients of the cubic equation. The coefficients p and q are computed as

$$
p = \frac{3ac - b^2}{3a^2}, \quad q = \frac{2b^3 - 9abc + 27a^2d}{27a^3}. \tag{39}
$$

The discriminant $\Delta = \left(\frac{q}{2}\right)^2 + \left(\frac{p}{3}\right)^3$ determines the number of real roots. When $\Delta \geq 0$, the single real root without multiplicity is selected directly. When $\Delta < 0$, the cubic equation has three distinct real roots $x_1 < x_2 < x_3$, and the function exhibits one local maximum and one local minimum between them. The root further away from the middle one minimizes the quartic polynomial.

After obtaining an analytical root, we refine it using Newton's iteration:

$$
x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)},
$$

until the relative change satisfies $|x_{k+1} - x_k|/|x_k| < \varepsilon$. Finally, set $\eta = x_{k+1}$.