THE UNIVERSITY OF CHICAGO

GPU-ACCELERATED MULTIREFERENCE METHODS FOR POLYNUCLEAR TRANSITION METAL COMPLEXES

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF CHEMISTRY

BY
VALAY AGARAWAL

CHICAGO, ILLINOIS
AUGUST 2026

ORCID iD: https://orcid.org/0000-0003-2989-8013

ACKNOWLEDGMENTS

Five years is a long time. I have met and learned so much from so many people and I apologize to the people I miss mentioning here.

I'd like begin by thanking my advisor Prof. Laura Gagliardi for her constant mentorship and endless patience through out my graduate studies. Her scientific rigour has helped me become a better researcher and communicator. I am deeply grateful to Prof. Matthew R. Hermes at University of Chicago, and Dr. Christopher J. Knight at the Argonne National Lab, who, though not formally my co-advisors, have been nothing less, and have mentored me closely and were instrumental to my growth. Prof. Hermes taught me valuable concepts of multireference chemistry and Dr. Knight trained me from the ground up in GPU programming. I further extend my gratitude to Prof. Cong Liu and Dr. Rishu Khurana, whose deep expertise in real-world scientific challenges kept me connected to the broader chemical sciences and made our collaboration truly rewarding.

It's been an absolute privilige to work with many collaborators in the Gagliardi group including Abhishek, Joanna and Bhavnesh. I have also benefited greatly from discussions with other group members, Ruhee, Adam, and Andrea, who offered fresh perspectives and alternative approaches to problem-solving. Beyond the research, the entire Gagliardi group has given me some of my most cherished memories.

I'd like to thank my committee members, Prof. David Mazziotti and Prof. Dipti Jasrasaria and Dr. Christopher Knight for valuable feedback on my research.

I am grateful to the supportive staff in the chemistry department. I want to especially shout out Vera for her support with navigating the challenging terrain of

graduate school.

This research would not have been possible without the dedicated efforts of the staff at the Research and Computing Center at University of Chicago and at the Argonne Leadership Computational Facility at Argonne National Laboratory.

I am grateful to Rachel, Andrew, Chris, Elaine, Thea, Wolf, Matthew and many other graduate workers across the University. You all inspire me and made me feel like part of something bigger and tangible - working to make graduate school more accessible for the next generation.

And finally I am immensely grateful to my family whose constant support and encouragement made this degree possible.

"[that] materials are made of atoms is a fundamental limitations. And it's not that far away. ... We're pushing up against some fairly fundamental limits, so one of these days we're going to have to stop making things smaller."

- Gordon Moore (co-founder of Intel), 2005

40th anniversary of Moore's Law

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Computational modelling of molecules with multiple transition metals having strongly correlated electrons is a ever-present and difficult challenge in the field of theoretical chemistry. Conventional multireference wave function methods have been used successfully for strongly correlated systems but the high cost associated with them limits their usage for polynuclear transition metals. Practically, a numerical efficent method using modern computing architecture is needed to reduce the time-to-solution.

This dissertation focuses on development and application of GPU-accelerated chemically-guided fragmentation based multireference methods. The central exploration is with the localized active space self-consistent field (LASSCF) method. The method models locally correlated centers in spatially separated regions of a molecule. While the method scales better than the conventional complete active space (CAS), it can give qualitatively incorrect wave function in strongly correlated fragments. The LAS-state interaction method adds back correlation explicitly. This work scales the method to highly challenging systems.

To make the method numerically efficient, we have leveraged the modern GPU-accelerated computing architecture. We have ensured the usage of portable programming practices to retain performance across various hardware and software ecosystems. With this, we have studied the $[Fe_4S_4(SMe)_4]^{2-}$ cluster and tried to construct simple models for the low-lying electronic states.

This work expands our abilities to model challenging polynuclear transition metal complexes and obtain qualitatively correct wave functions with high numerical speed. As a byproduct of this work, we also have access to order of magnitude faster con-

ventional methods like HF, DFT and CASSCF.

CHAPTER 1

INTRODUCTION

1.1 Motivation: The challenge of electronic structure modelling

Computational modelling and prediction of the properties of molecules and materials can enhance our fundamental understanding of the intricate chemistry and provide efficiency in terms of reduced cost or acceleration of discovery, or both[3, 4, 5, 6]. A significant challenge for exact computational modelling is the calculation of all-electron interactions. The interaction energies, in exact terms, while easy to write, are exponentially costly to calculate with system size and can scale faster than computing resources. This challenge has motivated scientists to build models with various degrees of approximation. Computational modelling requires the scientist to make a trade-off between cost and the severity of the approximation. The level of approximation is often based on the nature of the system under study, and more specifically, the property or phenomenon of interest. With the development of computing architecture of central processing units, and in the last decade, the explosion of capabilities of graphics processing units, we have an opportunity to model increasingly complex problems, or conversely, to build model with increasing details.[7, 8, 9, 10] Most newer supercomputers have a mixed architecture of CPUs and GPUs, and derive the bulk of their computing power from GPUs.[11] To efficiently use these computing architectures, our methods must be able to utilize GPUs for performing costly bottlenecks to achieve numerical performance and reduce the time-to-solution.

On one hand, methods like Hartree-Fock (HF)[12] and Kohn-Sham density functional theory (DFT)[13, 14] offer simple and computationally inexpensive ways to model electron correlations. On the other hand, wave function methods such as perturbation theory (PT), coupled cluster (CC)[15, 16, 17] and multiconfiguration self-consistent field (MCSCF)[18] methods offer flexible models of electron correlations but with increased costs.

MCSCF methods, and most the commonly known variant, complete active space self-consistent field (CASSCF)[18], can incorporate chemical knowledge of the system or phenomenon to reduce the complexity of the calculations. Instead of modeling all electron correlations for the entire system, CASSCF allows scientists to limit such calculations only to a chemically interesting part of the system, such as d orbitals and electrons for transition metals, π orbitals and electrons for extended conjugation systems or orbitals and electrons involved in bonding for chemical reactions. The chemically interesting part is called the active space. This approach allows for a sea change in the modeling of correlation in complex systems but has two shortcomings. First, since the electron correlation is only calculated inside the active space, the correlations from outside the active space, however weak that may be, are missed. Second, CASSCF still calculates all-electron correlation in the active space and cost remains exponential with the size of the active space. This limits the applicability of the method to relatively small systems.[19]

For the first shortcoming, common post-MCSCF approaches include perturbation theory[20, 21, 22, 23] and pair-density functional theory[24]. Generally, CASSCF is used to provide a qualitatively accurate wave function in the active space with

post-MCSCF methods then providing quantitative accuracy. This approach has been used in single-molecule magnets[25, 26, 27], excitations in DNAs[28, 29, 30], and studying conjugated inorganic molecules.[31, 32] The second shortcoming of exponential scaling has seen a number of approaches, which are discussed in later chapters.[33] In this thesis, we focus on a fragmentation-based method called localized active space self-consistent field (LASSCF).[34]

We start with the theoretical background of multireference methods in Sec. 1.2. We then discuss some background ideas around leveraging GPUs for accelerating calculations in Sec. 1.3. The rest of the thesis discusses LASSCF, it's shortcomings, and LAS state interaction (LASSI) method to recover qualitatively correct wave function in Sec 2, our efforts to accelerate LASSCF and LASSI with GPUs in Sec. 3 and the application of the entire method stack on a challenging $[Fe_4S_4(SMe)_4]^{2-}$ cluster.

1.2 Theoretical background

The electronic Hamiltonian ($\hat{H}$) with N electrons over M atoms with $Z_i, i \in \{1, 2, ..., M\}$ can be written in atomic units as

$$\hat{H} = \sum_{I<J}^{M} \frac{Z_I Z_J}{R_{IJ}} - \sum_{i}^{N} \frac{\nabla_i^2}{2} - \sum_{i}^{N} \sum_{I}^{M} \frac{Z_I}{r_{Ii}} + \sum_{i<j}^{N} \frac{1}{r_{ij}} \tag{1.1}$$

where the terms correspond to the nuclear-nuclear repulsion, electron kinetic energies, electron-nuclear attraction, and electron-electron repulsion. This Hamiltonian ignores nuclear motion (Born-Oppenheimer approximation) and relativistic

effects.[35, 36] The electron-electron repulsion term makes analytical solution impossible for all non-hydrogenic systems,[37] and is usually solved by a self-consistent iterative loop. We first choose a basis of spin-orbitals $\psi_p(\mathbf{x})$, where $\mathbf{x}$ is a combined representation of space and spin coordinates. In second quantized form, the Hamiltonian is transformed into

$$\hat{H} = V_{NN} + \sum_{pq} h_{pq} \hat{c}_p^\dagger \hat{c}_q + \frac{1}{2} \sum_{pqrs} \langle pq|rs\rangle \, \hat{c}_p^\dagger \hat{c}_q^\dagger \hat{c}_r \hat{c}_s \tag{1.2}$$

where $\hat{c}_p^\dagger$ is the creation operator for spin-orbital ψ_p, and $\hat{c}_p$ is the annihilation operator. These operators obey the fermionic anticommutation rules and respect the Pauli exclusion principle. $V_{NN} = \sum_{I<J}^{M} \frac{Z_I Z_J}{R_{IJ}}$ is a constant for a given set of nuclear coordinates. h_{pq} is the one-electron integral written as $\langle \psi_p | \, \hat{h} \, | \psi_q \rangle$, defined as

$$h_{pq} = \int \psi_p^*(\mathbf{x}) \left(-\frac{1}{2}\nabla^2 - \sum_I \frac{Z_I}{r_{Ii}} \right) \psi_q(\mathbf{x}) \, d\mathbf{x} \tag{1.3}$$

This integrates the one-electron terms: kinetic energy and nuclear-electron attraction. The $\langle pq|rs\rangle$ is the antisymmetrized two-electron integral ($\langle pq|rs\rangle - \langle pq|sr\rangle$), defined as

$$\langle pq|rs\rangle = \iint \psi_p^*(\mathbf{x}_1) \psi_q^*(\mathbf{x}_2) \frac{1}{r_{12}} \psi_r(\mathbf{x}_1) \psi_s(\mathbf{x}_2) \, d\mathbf{x}_1 d\mathbf{x}_2 \tag{1.4}$$

which integrates the electron-electron repulsion. This term is also called the electron repulsion integral (ERI).

Given a basis set, the wave function can be approximated as a weighted sum of all possible electronic configurations. Each configuration is represented as a deter-

minant, since it makes the wave function an antisymmetric product of spin-orbitals respecting the Pauli exclusion principle. The determinant can be presented as an occupation number vector $|\mathbf{n}\rangle = |n_1 n_2 ... n_O\rangle$, where each n_i is either 0 or 1. In other words, each spin-orbital can be filled or empty. All arrangements of electrons, i.e., all ways to choose N electrons from O orbitals, combine to form the full Hilbert space.

As an example, consider an H_2 molecule, where each H atom has one electron in two spin-orbitals $(1s_\alpha, 1s_\beta)$, and the molecular spin orbitals are arranged as

$$\left|1s_\alpha(1), 1s_\beta(1), 1s_\alpha(2), 1s_\beta(2)\right\rangle \tag{1.5}$$

All possible singlet ($S = 0$) configurations are $|1100\rangle, |1001\rangle, |0110\rangle, |0011\rangle$. The exact wave function in that basis is represented as

$$\Psi = \sum_{\mathbf{n}} c_{\mathbf{n}} |\mathbf{n}\rangle \tag{1.6}$$

where $c_{\mathbf{n}}$ is the coefficient of each determinant. Each coefficient can be calculated by the variational principle, which states that the energy of any trial wave function is bounded below by the true ground state energy. In other words,

$$E[\Psi_{\text{trial}}] = \frac{\langle \Psi_{\text{trial}}| \hat{H} |\Psi_{\text{trial}}\rangle}{\langle \Psi_{\text{trial}}|\Psi_{\text{trial}}|\Psi_{\text{trial}}|\Psi_{\text{trial}}\rangle} \geq E_0 \tag{1.7}$$

where the equality holds only when $|\Psi_{\text{trial}}\rangle$ is equal to the true ground state wave function. When all configurations are considered, the wave function is termed a full configuration interaction (FCI) wave function.

Even though algebraically simple, the number of configurations increases combinatorially with the size of the system and quickly hits the limits of computational power.[19] However, for most systems, an overwhelming majority of these configurations contribute a negligible amount to the wave function.[38] Based on the approximations used to ignore configurations, several methods in quantum chemistry can be derived.

A simple and arguably severe approximation is to consider only one configuration, where all electron interactions are estimated as an electron moving in a sea of other electrons, or more precisely, in a mean field of all the other electrons. This is called the Hartree–Fock approximation,[36] which begins by representing the N-electron wave function as a single Slater determinant. The optimal orbitals are obtained by solving the self-consistent field equations $\hat{f}\psi_p = \epsilon_p \psi_p$, where the Fock operator is defined as $\hat{f}(1) = \hat{h}(1) + \sum_j^N [\hat{J}_j(1) - \hat{K}_j(1)]$, with $\hat{J}_j$ as the Coulomb operator and $\hat{K}_j$ as the exchange operator. They are defined as

$$\hat{J}_j(1)\,\psi_i(1) = \left[\int \frac{|\psi_j(\mathbf{r}_2)|^2}{|\mathbf{r}_1 - \mathbf{r}_2|}\,d\mathbf{r}_2\right]\psi_i(\mathbf{r}_1) \tag{1.8}$$

$$\hat{K}_j(1)\,\psi_i(1) = \left[\int \frac{\psi_j^*(\mathbf{r}_2)\,\psi_i(\mathbf{r}_2)}{|\mathbf{r}_1 - \mathbf{r}_2|}\,d\mathbf{r}_2\right]\psi_j(\mathbf{r}_1) \tag{1.9}$$

This mean-field treatment captures the bulk of the electronic energy but entirely neglects electron correlation. The correlation energy is thus defined as the difference $E_{\text{corr}} = E_0 - E_{\text{HF}}$ between the true non-relativistic ground state energy and the HF energy.

A significant portion of this correlation energy is static correlation, which arises when the wave function cannot be described by a single determinant because several configurations have comparable weights. This is common in bond breaking, transition metal complexes, diradicals, or systems with near-degenerate orbitals. In such cases, the wave function must be expanded as $\Psi = \sum_i c_i |\Phi_i\rangle$ with multiple significant coefficients c_i, and the error in the Hartree–Fock energy due to these near-degeneracy effects cannot be fixed by perturbative corrections starting from a single reference determinant.

The remaining contribution is dynamic correlation, which originates from the instantaneous Coulomb repulsion between electrons, causing them to avoid one another at short distances. Unlike static correlation, dynamic correlation can be reasonably captured from a single reference determinant, for example by Møller–Plesset perturbation theory, where the second-order energy correction is given by $E^{(2)}_{\text{dyn}} = \sum_{i<j,a<b} \frac{|\langle ij\|ab\rangle|^2}{\epsilon_i+\epsilon_j-\epsilon_a-\epsilon_b}$, with i, j indexing occupied orbitals, a, b virtual orbitals, and $\langle ij\|ab\rangle$ the antisymmetrized two-electron integral.

To handle static correlation properly, one turns to multiconfigurational methods such as CASSCF (complete active space self-consistent field). In CASSCF, one selects a set of active electrons and active orbitals, and includes all configurations generated by distributing the active electrons among the active orbitals in every possible way, yielding

$$\Psi_{\text{CASSCF}} = \sum_I c_I |\Phi_I\rangle \tag{1.10}$$

where the sum runs over all determinants in the active space. The inactive orbitals remain doubly occupied, the external orbitals stay empty, and both the coefficients

c_I and the molecular orbitals themselves are optimized variationally. This provides a qualitatively correct zeroth-order wave function for strongly correlated systems, but recovers very little dynamic correlation because the active space is typically too small to capture short-range electron repulsion.

To add dynamic correlation on top of a multiconfigurational reference, one can use complete active space perturbation theory (CASPT2), which is a second-order perturbation theory applied to a CASSCF wave function. The CASSCF state defines the zeroth-order Hamiltonian $\hat{H}_0$, and the fluctuation potential $\hat{V} = \hat{H} - \hat{H}_0$ acts as the perturbation. The CASPT2 energy correction is $E^{(2)} = \sum_{k \neq 0} \frac{|\langle \Psi_k | \hat{V} | \Psi_{\mathrm{CASSCF}} \rangle|^2}{E_0^{(0)} - E_k^{(0)}}$, where the sum runs over all excited configurations (internal, semi-internal, and external) not already included in the CASSCF reference. While widely used for electronically excited states, bond dissociation, and transition metal chemistry, CASPT2 can suffer from intruder state problems, where near-degeneracies between the reference and excited configurations cause the denominators to become very small.

An alternative approach is MC-PDFT (multiconfigurational pair-density functional theory), which replaces the perturbative correction with a density functional. The MC-PDFT energy is written as $E_{\text{MC-PDFT}} = E_{\mathrm{CASSCF}} + E_{\mathrm{ot}}[\rho, \Pi]$, where $\rho(\mathbf{r})$ is the electron density and $\Pi(\mathbf{r})$ is the on-top pair density, defined as $\Pi(\mathbf{r}) = \langle \Psi_{\mathrm{CASSCF}} | \sum_{i<j} \delta(\mathbf{r}_i - \mathbf{r}) \delta(\mathbf{r}_j - \mathbf{r}) | \Psi_{\mathrm{CASSCF}} \rangle$. The correction functional E_{ot} is taken from translated Kohn–Sham functionals but evaluated using the pair density rather than the kinetic energy density. This method is largely free from intruder state problems and spin-contamination issues, scales more favorably than CASPT2, and maintains good accuracy for ground and excited states where static correlation is

significant.

1.3 GPU accleration basics

Apart from theoretical scaling, the software implementation of a method must also be numerically efficient. The architecture of graphics processing units (GPUs) consists of a large number of low-powered processors, which excel at executing many simple tasks simultaneously. Consequently, GPUs are widely used in gaming, video editing, and, more recently, running artificial intelligence models. Their ability to distribute vectorized workloads across many processors makes them highly suitable for matrix multiplications and other vectorizable computations.

For example, consider a CPU with 64 cores and a GPU with 1,000 cores. Suppose a task requires an operation on 10,000 elements, where each operation takes one core one cycle, and cycles on both CPU and GPU take the same amount of time. If the workload is perfectly parallelizeable without overhead, the CPU would require approximately 157 cycles (10,000 / 64), whereas the GPU would need 10 cycles (10,000 / 1,000), leading to a 15x speedup. Since a bulk of computationally intensive steps in electronic structure theory involve operations over several indices, GPUs are naturally well-suited for applications in the field.

Many modern supercomputers derive over 80% of their theoretical peak computing capacity from GPUs.[11] Therefore, leveraging GPUs for quantum chemistry is essential for scientists to utilize these machines efficiently. Significant effort in GPU programming has focused on creating direct substitutes for math library calls (`numpy`[39] to `cupy`[40] for NVIDIA GPUs), enabling calculations to run on GPUs

instead of CPUs with minimal software development cost. However, such direct substitutions can become less effective if not used carefully, often due to data transfer overhead. To demonstrate this further, a high-end CPU server may host upwards of 1 TB of RAM (Random Access Memory) using DDR5 memory (Double Data Rate 5). An NVIDIA A100 GPU, for instance, only has with 40 GB HBM2 (High Bandwidth Memory second generation) or 80 GB HBM2e memory (HBM2 enhanced). The large difference between the memory sizes typically leads to a code structure where most data initially resides in CPU memory, and only the subset being operated on is transferred to the GPU, with results transferred back. This physical data movement is constrained by the speed of the Peripheral Component Interconnect Express (PCIe) bus. For PCIe 5.0, a 16-lane (x16) connection achieves a maximum effective data rate of approximately 63 GB/s in a single direction. By comparison, an A100 VRAM (video random access memory) can be accessed at speeds over 1 TB/s with lower latency (PCIe latency is 1-10 microseconds; VRAM is 100-500 nanoseconds). Hence, even if the GPUs perform a calculation quickly, the movement of input data from CPU to GPU and the copying of results from GPU to CPU may have significant time requirements. A simple drop-in replacement may not yield performance gains if the data transfer cost dominates.

From a software development perspective, two approaches can be adopted: accelerating only the computational bottlenecks of a program, or making the entire code GPU-resident. Each approach has trade-offs. While a fully GPU-resident code is likely more performant due to minimal data transfers, program development can be more time-consuming than the bottleneck-only counterpart. It would also be reason-

able to make a fully GPU-resident code if the memory requirement of the program is less than the GPU memory. Finally, if the program contains several computationally intensive tasks that involve large data transfers, it could be more efficient to develop a GPU-resident program. Conversely, bottleneck-only acceleration is more attractive if the memory requirements are high or there are only a few computational bottlenecks that dominate the cost of a calculation.

Several GPU vendors — NVIDIA, AMD, and Intel — each have different programming models and device-optimized math libraries. NVIDIA uses CUDA with the cuBLAS library, AMD uses HIP with the rocBLAS library, and Intel uses SYCL/oneAPI with the oneMKL library. Code written in one library does not readily run on another vendor's GPU without translation, which reduces code usability and increases development costs. Translation tools such as HIPify (CUDA to HIP) and SYCLomatic (CUDA to SYCL) are available to assist migration. Abstracting common math library and programming model functions can help in maintenance.

In summary, while GPUs offer immense computational power for the linear algebra-dominated calculations central to electronic structure theory, effective utilization requires careful attention to both hardware architecture and software implementation. The high bandwidth of GPU memory and the large number of cores can enable substantial speed-ups for vectorizable tasks, but the comparatively slower data transfer between CPU and GPU can impose a significant performance penalty. Consequently, simple direct substitutions of math libraries often yield diminishing returns unless data movement is avoided, minimized or hidden. Fully GPU-resident codes avoid most of this overhead but demand greater development effort. Given

these trade-offs, and recognizing that modern supercomputers derive the majority of their peak performance from GPUs, we have developed a codebase for performing LAS-based methods with GPUs. We focus on identifying and accelerating computational bottlenecks in quantum chemistry workflows and moving them to the GPU, while managing data transfers explicitly to achieve efficient, vendor-portable performance. This approach of low-lying computational bottlenecks has also yielded GPU-accelerated HF, DFT, and CASSCF methods as dividend. We discuss the specific implementation, optimization strategies, and performance benchmarks of the GPU-accelerated code developed in Chapter 3.

CHAPTER 2

AUTOMATIC STATE INTERACTION WITH LARGE LOCALIZED ACTIVE SPACES FOR MULTIMETALLIC SYSTEMS

2.1 Introduction

While complete active space self-consistent field (CASSCF) theory provides a rigorous framework for treating static correlation, its exponential scaling with respect to active space size limits applications to systems with approximately 20 electrons in 20 orbitals.[19] This limitation is particularly severe for multimetallic complexes where strong correlation manifests on multiple spatially separated centers.[41, 42]

Several approaches in the literature have been explored to address this scaling. One set of approaches pursues automatic reduction in configuration space functions (CSFs) or Slater determinants (SDs) to control the cost. These include selected configuration interaction (SCI)[43, 44, 45, 46, 47, 48] and density matrix renormalization group (DMRG)[49]. The second class of approaches involves reducing the size of Hilbert space through chemical intuition by partitioning a large chemically relevant active space into smaller subpsaces or fragments. Some well-known methods are restricted active space SCF (RASSCF)[50, 51], generalized active space SCF (GASSCF)[52, 53, 54] and cluster mean field (cMF)[55] or localized active space self-consistent field (LASSCF)[34, 56]. In this work, we adopt the LASSCF approach, which addresses this challenge through a fragment-based factorization of the molecular wave function. The fundamental ansatz expresses the total wave function as an

antisymmetrized product of localized fragment wave functions:

$$|\Psi_{\mathrm{LAS}}\rangle = \bigwedge_{K=1}^{N_{\mathrm{frag}}} \left|\Psi_{A_K}\right\rangle \bigwedge |\Psi_D\rangle \tag{2.1}$$

where $\left|\Psi_{A_K}\right\rangle$ is the wave function of fragment K active space and $|\Psi_D\rangle$ is the remaining single determinant. Each fragment active space function is characterized by the number of electrons (N_K), number of orbitals (M_K), and overall spin (S_K).

Within each fragment, the wave function is expanded as a CAS ansatz, which means that the cost of the method increases exponentially with the size of the fragment active space. However, since LAS splits the total active space into a number of fragments, and each fragment's active space is treated independently, the cost grows polynomially with the number of fragments. This is effective in cases where the inter-fragment correlation is weak and can be modeled sufficiently by a mean-field approximation.[34] However, this factorization introduces a severe approximation: inter-fragment correlation effects are only included through the mean-field interaction between fragments.

In this chapter, we first discuss the theory of LASSCF, and two subsequent approaches to correct LASSCF, namely LAS state interaction (LASSI) and its improvement LASSI singles (LASSIS). For LASSI and LASSIS, we discuss the theoretical approximations and the results of the methods on some challenging polynuclear transition metal complexes.

2.2 LAS Self-Consistent Field Theory (LASSCF)

The LASSCF energy is written as the expectation value of the full molecular Hamiltonian with respect to the antisymmetrized product wave function. For a system partitioned into N_{frag} fragments, the energy is given by:

$$E_{\text{LAS}} = \frac{\langle \Psi_{\text{LAS}} | \hat{H} | \Psi_{\text{LAS}} \rangle}{\langle \Psi_{\text{LAS}} | \Psi_{\text{LAS}} \rangle} \tag{2.2}$$

LASSCF optimizes the fragment wave function $\left|\Psi_{A_k}\right\rangle$ by minimizing the fragment energy written as

$$E_K = \left\langle \Phi_{E_K} \right| \wedge \left\langle \Psi_{A_K} \right| \hat{H}_{E_K} \left| \Psi_{A_K} \right\rangle \wedge \left| \Phi_{E_K} \right\rangle \tag{2.3}$$

where Φ_{E_K} is the determinant containing the fragment embedding space inactive orbitals. The embedding space Hamiltonian $\hat{H}_{E_K}$ is

$$\begin{aligned} \hat{H}_{E_K} = (h_{e_{K_1}}^{e_{K_2}} + \{v^{jk}\}_{e_{K_1}}^{e_{K_2}} - \{v_\sigma^{\text{self}}\}_{e_{K_1}}^{e_{K_2}}) \hat{c}_{e_{K_1}}^\dagger \hat{c}_{e_{K_2}} \\ + \frac{1}{2} g_{e_{K_2} e_{K_4}}^{e_{K_1} e_{K_3}} \hat{c}_{e_{K_1 \sigma}}^\dagger \hat{c}_{e_{K_3 \tau}}^\dagger \hat{c}_{e_{K_4 \tau}} \hat{c}_{e_{K_2 \sigma}} \end{aligned} \tag{2.4}$$

where

$$\{v^{jk}\}_{\mu_1}^{\mu_2} = g_{\mu_2 \mu_4}^{\mu_1 \mu_3} D_{\mu_4}^{\mu_3} - g_{\mu_4 \mu_2}^{\mu_1 \mu_3} \{\gamma_\sigma\}_{\mu_4}^{\mu_3} \tag{2.5}$$

$$\{v^{(\text{self})}\}_{e_{K_1}}^{e_{K_2}} = g_{e_{K_2} e_{K_4}}^{e_{K_1} e_{K_3}} D_{e_{K_4}}^{e_{K_3}} - g_{e_{K_4} e_{K_2}}^{e_{K_1} e_{K_3}} \{\gamma_\sigma\}_{e_{K_4}}^{e_{K_3}} \tag{2.6}$$

are the effective embedding potential and effective self potential respectively. Here

e_{K_i} denotes the i^{th} embedding orbital of the K^{th} fragment. D is the one-body reduced density matrix, γ_σ is the spin separated two-body density matrix, and μ_i is the i^{th} atomic orbital. Each LASSCF fragment is solved as a CASSCF problem with the given effective Hamiltonian and the active space wave function. The CASSCF optimization condition requires that the derivative of the fragment energy with respect to the CI vector and derivative of the energy of any orbital with respect to change in any other orbital is zero. This can be written as

$$\{E^{(1)}\}_{\mathrm{CI}} = \{E^{(1)}\}_{a_K}^{e_K} = 0 \tag{2.7}$$

Once all fragment CASSCFs are solved, the active space orbitals of all fragments are not necessarily orthogonal to each other since they are optimized independently and must be reorthogonalized. The active orbitals and CI vectors are then frozen and the whole molecule's inactive orbitals are optimized. Additionally, the energy is minimized with respect to rotation between two active subspaces (K, L) and their CI vectors. This corresponds to solving the equation

$$\{E^{(1)}\}_{\mathrm{CI}} = \{E^{(1)}\}_{a_L}^{a_K} = \{E^{(1)}\}_{d}^{v} = 0 \tag{2.8}$$

Eq. 3.1 is satisfied through a second order algorithm that optimizes

$$\mathbf{E}^{(1)} + \mathbf{E}^{(2)}\mathbf{x} = \vec{0}, \tag{2.9}$$

where $\mathbf{x}$ is a unitary generator containing all orbital and CI transformation variables,

$\mathbf{E}^{(2)}$ is the second-order derivative.

2.3 Localized Active Space State Interaction (LASSI)

LASSCF captures intra-fragment correlation exactly within each fragment's active space, but inter-fragment correlation is treated only at a mean-field level. This approximation is effective when the fragments are well separated and can be represented faithfully with independent centers. When that is not the case, LASSCF can yield a qualitatively incorrect wave function due to missing correlation. The localized active space state interaction (LASSI) method reintroduces these omitted correlations by constructing and diagonalizing the full molecular Hamiltonian in a basis of LAS states.[41, 42] Considering only the active-space part of the wave function (i.e., treating the doubly-occupied inactive electrons as part of the frozen core), the LAS wave function can be rewritten as

$$|\mathrm{LAS}\rangle = \bigwedge_{K}^{\text{fragments}} |0_K\rangle_{\mathcal{P}} \,, \tag{2.10}$$

This is the antisymmetrized product of general many-electron fragment wave functions $|0_K\rangle_{\mathcal{P}}$, defined in a particular Hilbert space labeled $\mathcal{P}$, about which see below. The fragment wave functions are the eigenstates of the projected Hamiltonian,

$$\hat{H}_{\mathcal{P}K} \equiv \hat{\mathcal{P}}_K^\dagger \hat{H} \hat{\mathcal{P}}_K, \tag{2.11}$$

$$\hat{\mathcal{P}}_K = \bigotimes_{L \neq K}^{\text{fragments}} |0_L\rangle_{\mathcal{P}}\,, \tag{2.12}$$

$$0 = \left(\hat{H}_{\mathcal{P}K} - E_{n\mathcal{P}K}\right) |n_K\rangle_{\mathcal{P}}\,, \tag{2.13}$$

where $n \geq 0$ is an integer indexing the excitation number of the particular fragment. Each fragment Hamiltonian, $\hat{H}_{\mathcal{P}K}$, describes the Kth fragment in the mean field of all other fragments in their respective ground states. The operation in Eq. (2.11) can be understood as a projection (i.e., $|0_L\rangle \langle 0_L|$ for all $L \neq K$) followed by a trace over all fragments other than K. The LAS states in total are then indexed as

$$|\vec{n}\rangle_{\mathcal{P}} \equiv \bigwedge_{K}^{\text{fragments}} |n_K\rangle_{\mathcal{P}} \tag{2.14}$$

where $\vec{n} = \{n_1, n_2, n_3, \ldots\}$ is the "excitation number vector" (ENV) of nonnegative integers that index the various eigenstates of the $\hat{H}_{\mathcal{P}K}$ operators.

The Hilbert space indexed by $\mathcal{P}$ in which the roots of the projected Hamiltonians are sought (henceforth the 'rootspace') is defined by the number of electrons in each fragment, the spin quantum numbers of each fragment, and the point group irreducible representation of each fragment wave function. In other words, $\mathcal{P}$ is a compound index representing the space table below, in which different rows represent different fragments and different columns represent different types of quantum

number:

$$\mathcal{P} = \begin{bmatrix} N_\uparrow^{(1)} & N_\downarrow^{(1)} & S^{(1)} & \Gamma^{(1)} \\ N_\uparrow^{(2)} & N_\downarrow^{(2)} & S^{(2)} & \Gamma^{(2)} \\ N_\uparrow^{(3)} & N_\downarrow^{(3)} & S^{(3)} & \Gamma^{(3)} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}, \tag{2.15}$$

with $N_\uparrow$ for spin-up electrons, $N_\downarrow$ for spin-down electrons, S for spin magnitude, and Γ for point-group irrep. These fictitious, fragment-indexed quantum numbers are not conserved by the exact, correlated many-electron wave function of the whole system, but they are (in practice) conserved by the approximate $|\mathrm{LAS}\rangle$ wave function. Much like a single determinant, a single LAS state is generally not an eigenstate of $\hat{S}^2$ unless at most one fragment has a nonzero $S^{(K)}$.

In addition to breaking spin symmetry, a single LAS state omits all entanglement between fragments; fragments interact exclusively *via* a mean-field potential. This approximation is obviously quantitatively inaccurate, and can also be qualitatively inaccurate in certain cases.[56] However, interaction and entanglement between fragments, as well as global $\hat{S}^2$ symmetry, can be restored by constructing a linear combination of multiple states of the LAS type, which we refer to as LASSI.[41] The LASSI wave functions are linear combinations of the LAS states $|\vec{n}\rangle_{\mathcal{P}}$:

$$|\mathrm{LASSI}_i\rangle \equiv \sum_{\mathcal{P}}^{\text{rootspaces}} \sum_{\vec{n}}^{\text{ENVs}} |\vec{n}\rangle_{\mathcal{P}} \, C^{(i)}_{\mathcal{P}\vec{n}} \tag{2.16}$$

where $C^{(i)}_{\mathcal{P}\vec{n}}$ is an eigenvector coefficient of the Hamiltonian projected into the model space of LAS states:

$$\left\langle \mathrm{LASSI}_i|\hat{H}|\vec{n}\right|\left.\mathrm{LASSI}_i|\hat{H}|\vec{n}\right\rangle_{\mathcal{P}} = E^{(i)}_{\mathrm{LASSI}} C^{(i)}_{\mathcal{P}\vec{n}}. \tag{2.17}$$

Note that the different fragment basis functions, $|n_K\rangle_{\mathcal{P}}$ for different $\mathcal{P}$, are eigenstates of different fragment operators, $\hat{H}_{\mathcal{P}K}$, although they all share the same set of molecular orbitals.

2.3.1 Analyzing LASSI Results

In Sec. 2.4 below, we draw a distinction between significant model states and "deadwood" model states by analyzing the LASSI eigenvector, $C^{(i)}_{\mathcal{P}\vec{n}}$, which can be decomposed as

$$C^{(i)}_{\mathcal{P}\vec{n}} = \sqrt{w^{(i)}_{\mathcal{P}}}\, u^{(i),\mathcal{P}}_{\vec{n}}, \tag{2.18}$$

with the normalized factors

$$\sum_{\mathcal{P}}^{\text{rootspaces}} w^{(i)}_{\mathcal{P}} = 1 \;\forall\; i \tag{2.19}$$

$$\sum_{\vec{n}}^{\text{ENVs}} (u^{(i),\mathcal{P}}_{\vec{n}}) * u^{(i),\mathcal{P}}_{\vec{n}} = 1 \;\forall\; i, \mathcal{P}. \tag{2.20}$$

The $u^{(i),\mathcal{P}}_{\vec{n}}$ is the normalized vector of CI in a given rootspace $\mathcal{P}$, and weights $(w^{(i)}_{\mathcal{P}})$ is the inverse of the normalizing coefficient of the given $u^{(i),\mathcal{P}}_{\vec{n}}$. Weights close to zero indicate that an entire rootspace is deadwood in the ith LASSI eigenstate. For rootspaces with nonzero weights, the ENV coefficients $(u^{(i),\mathcal{P}}_{\vec{n}})$ can be used to

compute the single-fragment density matrix in the excitation number basis,

$$d^{(i),K\mathcal{P}}_{m,m'} \equiv \sum_{\vec{n}\in\{n_K=m\}}^{\text{ENVs}} \sum_{\vec{n}'\in\{n_K=m'\}}^{\text{ENVs}} \prod_{L\neq K}^{\text{fragments}} \delta_{n_L n'_L} \times (u^{(i),\mathcal{P}}_{\vec{n}})^* u^{(i),\mathcal{P}}_{\vec{n}'}. \tag{2.21}$$

This is equivalent to tracing out the environment for all other fragments after projecting the density matrix in a given $\mathcal{P}$. In the application considered below, we examine spin-state energy gaps as obtained from energy differences of different LASSI eigenstates, and for this purpose, the weights and density matrices are averaged over the lowest-energy $i_{\max}+1$ LASSI eigenstates,

$$w_{\mathcal{P}} \equiv \frac{1}{i_{\max}+1} \sum_{i=0}^{i_{\max}} w^{(i)}_{\mathcal{P}}, \tag{2.22}$$

$$d^{K\mathcal{P}}_{m,m'} \equiv \frac{1}{i_{\max}+1} \sum_{i=0}^{i_{\max}} d^{(i),K\mathcal{P}}_{m,m'}, \tag{2.23}$$

where in the spin-ladder applications below, $i_{\max}+1$ is the number of states in the spin ladder.

From the (state-averaged) single-fragment density matrix, we extract the average excitation number of the Kth fragment in the $\mathcal{P}$th rootspace,

$$\bar{n}_{K\mathcal{P}} = \sum_{m=0} m \times d^{K\mathcal{P}}_{m,m}, \tag{2.24}$$

as well as the von Neumann entropy,

$$s_{K\mathcal{P}} = -\sum_m \left(\rho_m^{K\mathcal{P}} \log \rho_m^{K\mathcal{P}}\right), \tag{2.25}$$

where $\{\rho_m^{K\mathcal{P}}\}$ are the eigenvalues of the matrix $d^{K\mathcal{P}}$. If $\bar{n}_{K\mathcal{P}}$ and $s_{K\mathcal{P}}$ are close to zero, then excited states of Kth fragment in the $\mathcal{P}$th rootspace are deadwood.

2.3.2 LASSI[r, q]

If the LASSI model space consists of all possible rootspaces described by all possible space tables [Eq. (2.15)] along with all possible excitation number vectors, then LASSI reproduces CASCI, with the same exponential-scaling computational cost. However, the conjecture underlying LAS methods in general (that electrons interact more strongly within fragments than between fragments) suggests that a small approximate model space could reproduce CASCI qualitatively with orders of magnitude fewer model states than the number of determinants or CSFs in the CAS. In order for this to be practical, such an approximate model space must be generable semi-automatically. As mentioned above, when LASSI was introduced in Ref. 41, model states were constructed by the user entering all relevant space tables by hand; additionally, only the ground-state ENV ($\vec{0} = \{0_1, 0_2, 0_3, \ldots\}$) was considered in any rootspace. Here we explore model space construction more systematically.

First, we stipulate that all model spaces should be chosen to restore spin symmetry: LASSI wave functions should be $\hat{S}^2$ eigenvalues. This is accomplished by

permuting

$$M^{(K)} \equiv \frac{1}{2}\left(N_{\uparrow}^{(K)} - N_{\downarrow}^{(K)}\right) \tag{2.26}$$

for each fragment in all possible ways consistent with the fixed whole-molecule total number of spin up and spin down electrons, as well as the constraint $M^{(K)} \leq |S^{(K)}|$, and including all such permutations in any model space. Omitting point-group symmetry for the current work, the space table [Eq. (2.15)] can now be simplified to

$$\mathcal{P} = \begin{bmatrix} N^{(1)} & S^{(1)} \\ N^{(2)} & S^{(2)} \\ N^{(3)} & S^{(3)} \\ \vdots & \vdots \end{bmatrix}, \tag{2.27}$$

with the understanding that implicitly, all possible orientations of nonzero fragment spins are included in the model space.

Second, we construct simplified space tables [Eq. (2.27)] by systematically moving one electron from any one fragment to any other fragment up to r times in all possible ways, where r is a non-negative integer chosen by the user. For each increment or decrement of $N^{(K)}$, the corresponding local spin $S^{(K)}$ can (in general) either increase or decrease by $\frac{1}{2}$. A schematic for this is presented in Table 1.

Third, we stipulate a maximum number of fragment roots q, obtain up to the qth eigenstate of each fragment Hamiltonian in each rootspace $\hat{H}_{\mathcal{P}K}$ (or all eigenstates if there are fewer than q), and build all possible ENVs from this product space of these sets.

We use the formulation "LASSI[r, q]" to refer to LASSI calculations whose model

spaces are constructed in this way. Note that the CASCI limit is achieved as both parameters approach infinity:

$$\lim_{r\to\infty}\lim_{q\to\infty} \text{LASSI}[r,q] \equiv \text{CASCI}. \tag{2.28}$$

2.3.3 Efficient Selection with LASSI[r, q_{CT}]

We will show evidence in Sec. 2.4 from the results of LASSI[r,q] calculations that it is profitable to use different number of eigenstates for different fragments in different rootspaces, because whole categories of model states can be shown to be "deadwood" by examining the LASSI eigenvectors. In particular, we will show that to obtain qualitative agreement with CASCI, it is often not necessary to include "local excitations" (i.e., $n_K > 0$), *except* where a particular fragment has a different number of electrons than in the reference LAS wave function.

For instance, for a molecule with three fragments, if the reference ($r = 0$) space table is written as Eq. (2.27), so that a single $r = 1$ space ($\mathcal{Q}$) table might be

$$\mathcal{Q} = \begin{bmatrix} N^{(1)}+1 & S^{(1)}+\frac{1}{2} \\ N^{(2)}-1 & S^{(2)}+\frac{1}{2} \\ N^{(3)} & S^{(3)} \end{bmatrix}, \tag{2.29}$$

then only in the first and second fragments of this rootspace do local excited states, $n_K > 0$, contribute to low-energy LASSI eigenstates.

We will therefore define "LASSI[r, q_{CT}]" as an efficient approximate alternative to LASSI[r, q], where "CT" stands for "charge transfer" and indicates that the model

space includes up to the qth local eigenstate *only* if the number of electrons in the given fragment in the given rootspace differs from that in the reference rootspace; otherwise, only the ground eigenstate of $\hat{H}_{\mathcal{P}K}$ is included. Such calculations are indicated in the text by the Roman subscript CT on the integer q value; for instance "LASSI$[1, 5_{\mathrm{CT}}]$."

2.4 Performance of LASSI

This methodology was applied on two systems, an aluminium di-iron complex and a tri-iron complex. The $AlFe_2$ complex has one Al $^{3+}$, one Fe^{2+} and a Fe^{3+}, while the Fe_3 complex has one Fe^{2+} and two Fe^{3+} centers. For $AlFe_2$, we consider two active spaces, (11e,10o) consisting of Fe $3d$, and (11e,20o) consisting of Fe $3d$ and $4d$ orbitals with all $3d$ electrons.

This system consists of one d^6 Fe^{2+}center, one d^5 Fe^{3+} center and a d^0 Al^{3+} center. Since Al^{3+} is closed shell, we do not consider any active space on it. We explore two different active spaces for this system as shown in Fig. 3.6, namely (11e,10o) with all $3d$ orbitals of both Fe atoms and (11e,20o) with all $3d$ and $4d$ orbitals of both Fe atoms. In these different parameters, we investigate the dependence of the J coupling according to the Yamaguchi formula[57]

$$J = -\frac{E_{HS} - E_{LS}}{\langle S^2 | S^2 \rangle_{HS} - \langle S^2 | S^2 \rangle_{LS}} \tag{2.30}$$

for the sake of brevity. The full spin ladder is reported in the SI table II and III.

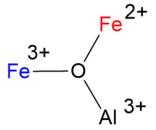
Rootspace depiction		
	$r = 0$	
(a)	↑↓ ↑ ↑ ↑ ↑ | ↓ ↓ ↓ ↓ ↓	$\mathcal{P} = \begin{bmatrix} N^{(1)} & S^{(1)} \\ N^{(2)} & S^{(2)} \end{bmatrix}$
(b)	↑↓ ↑ ↑ ↑ ↓ | ↓ ↓ ↓ ↓ ↑	
(c)	↑↓ ↑ ↑ ↓ ↓ | ↓ ↓ ↓ ↑ ↑	
(d)	↑↓ ↑ ↓ ↓ ↓ | ↓ ↓ ↑ ↑ ↑	
(e)	↑↓ ↓ ↓ ↓ ↓ | ↓ ↑ ↑ ↑ ↑	
	$r = 1$	
(f)	↑ ↑ ↑ ↑ ↓ | ↓ ↓ ↓ ↓ ↑↓	$\begin{bmatrix} N^{(1)} - 1 & S^{(1)} \pm 0.5 \\ N^{(2)} + 1 & S^{(2)} - 0.5 \end{bmatrix}$
(g)	↑↓ ↑ ↑ ↑ _ | ↓ ↓ ↓ ↓ ↑↓	
(h)	↑↓ ↑↓ ↑ ↑ ↑ | ↓ ↓ ↓ ↓ _	$\begin{bmatrix} N^{(1)} + 1 & S^{(1)} - 0.5 \\ N^{(2)} - 1 & S^{(2)} - 0.5 \end{bmatrix}$
	$r = 2$	
(i)	↑↓ ↑ ↑ ↑ ↓ | _ ↓ ↓ ↓ ↑↓	$\begin{bmatrix} N^{(1)} & S^{(1)} \\ N^{(2)} & S^{(2)} - 1 \end{bmatrix}$
(j)	↑↓ ↑↓ ↑ ↑ _ | _ ↓ ↓ ↓ ↑↓	$\begin{bmatrix} N^{(1)} & S^{(1)} - 1 \\ N^{(2)} & S^{(2)} - 1 \end{bmatrix}$
(k)	↑↓ ↑↓ ↑↓ ↑ ↑ | ↓ ↓ ↓ _ _	$\begin{bmatrix} N^{(1)} + 2 & S^{(1)} - 1 \\ N^{(2)} - 2 & S^{(2)} - 1 \end{bmatrix}$

Table 2.1: Examples of the rootspaces included in a LASSI calculation for a LASSCF reference with the red Fe^{2+} (6e,5o) atom and blue Fe^{3+} (5e,5o) atoms, depicted in terms of representative single determinants, the corresponding simplified space table [Eq. 2.27], and the r value at which they appear for $AlFe_2$ systems. (a)–(e) are the various cases of the local spin polarizations, $2M^{(K)} \equiv N_{\uparrow}^{(K)} - N_{\downarrow}^{(K)}$. (f)–(h) are examples of $r = 1$ rootspaces generated from (b) and are not exhaustive of all possible spin polarizations. Rootspace (f) adds 0.5 to $S^{(1)}$, and (g) decreases $S^{(1)}$ by 0.5 since S corresponds to the number of singly occupied molecular orbitals divided by 2. (i)–(k) are examples of $r = 2$ rootspaces. Specifically, (i), (j) and (k) rootspaces are generated by moving electron from the blue atom to the red atom from (f), (g) and (h) rootspaces respectively. $r = 2$ rootspaces are not exhaustive.

2.4.1 Aluminium Di Iron System

In Fig. 2.1, we compare the performance of LASSI with its theoretical limit LAS-CASCI, under two variations. We first vary r and q for the (11e,10o) active space. LASSI[0,1] includes no charge transfer rootspaces and only ground state eigenfunctions of the projected Hamiltonian in each fragment (equivalent to (a)–(e) rootspaces in Table 2.1). This generates only enough model states to ensure that the LASSI eigenstates are eigenfunctions of the $\hat{S}^2$ operator and wrongly predicts ferromagnetic interaction ($J > 0$) as compared to LAS-CASCI due to the complete omission of kinetic exchange and superexchange. As r and q increase, the agreement between LASSI and LAS-CASCI improves. The correct negative sign of J is observed for $r > 0$; $q > 4$. At LASSI[1,5] (Fig. 2.1: green curve with upside down triangles), we observe agreement within CASCI to within 1 cm^{-1}, from a wave function built from 250 model states. At LASSI[2,10] (1780 states) (purple curve with triangles), the difference is less than 0.1 cm^{-1}, at the price of requiring about 7 times as many model states compared to LASSI[1,5]. No performance improvement was observed for LASSI[3,q] series. LASSI[2,10] wave function still has another 30 times fewer model states than the 52920 single determinants in the CASCI limit. All corresponding absolute energies are in SI table II and III. For comparison, CASSCF predicts a J coupling of -7.3 cm^{-1}, as opposed to the LAS-CASCI limit of -3.8 cm^{-1}, which suggests that any inaccuracy in the LASSI[1,5] result is more likely attributable to the LASSCF orbitals than to the LASSI approximation.

The second variation explored is the active space dependence of J. For all three active space choices, agreement of J to within 3 cm^{-1} was achieved with vastly fewer

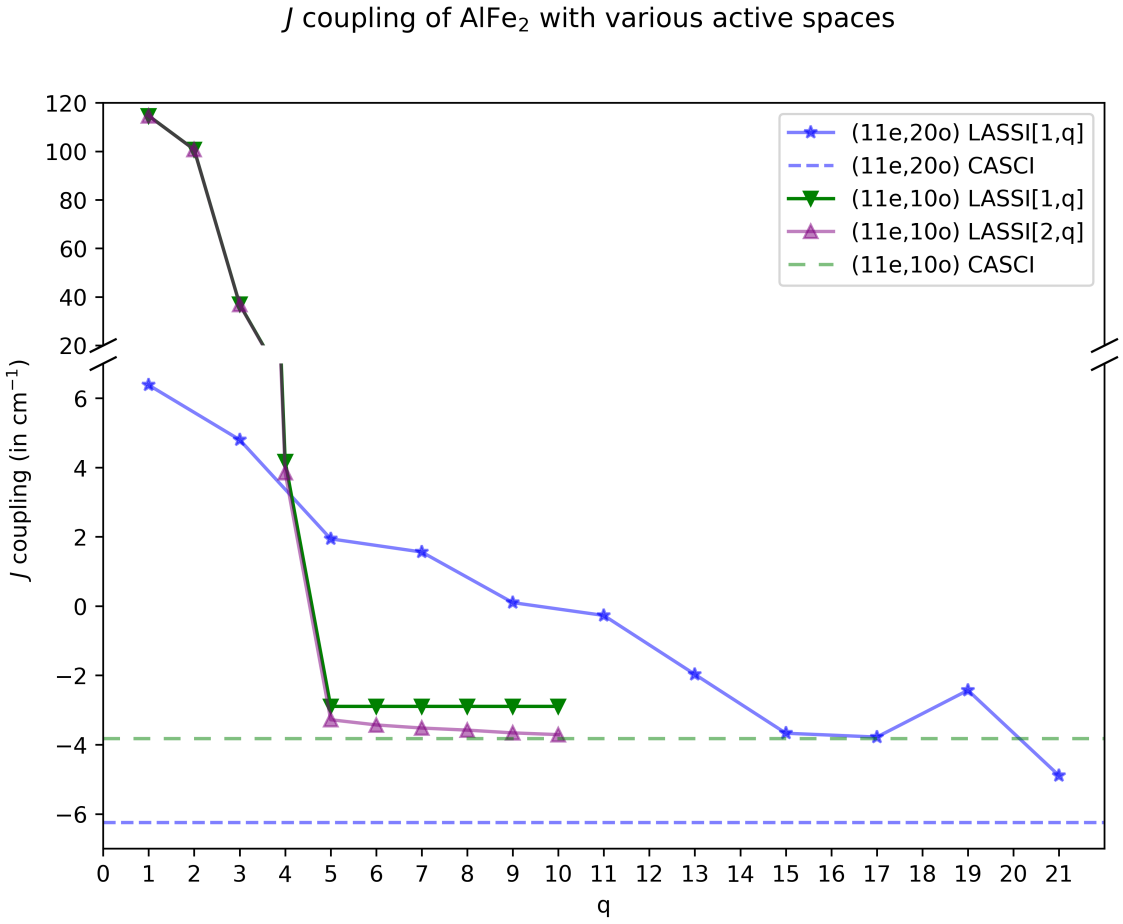


Figure 2.1: LASSI[1,q] and CASCI predicted J coupling for an $AlFe_2$ system with active spaces of (11e,10), and (11e,20o). For (11e,10o) active space, $J_{\text{LASSI}[0,q]} = +116.69 cm^{-1}\ \forall\ q = \{1, 10\}$ and $J_{\text{LASSI}[2,q]} \approx J_{\text{LASSI}[3,q]}\ \forall\ q = \{1, 10\}$.The LAS-CASCI result for(11e,10o) is -3.82 cm^{-1} and for (11e,20o) is -6.245 cm^{-1}.

model states than the number of single determinants in the corresponding CASCI: 7425 model states for the LASSI[1,15](11e,20o) calculation, as compared to $\approx 6\times10^8$ determinants in the CASCI(11e,20o) wave function. Clearly, the convergence with respect to q is less rapid in the larger active space. However, we note that the von Neumann entropies [Eq. (2.25)] for the LASSI[1,15](11e,20o) ($\max(s_{K\mathcal{P}}) = 0.95$) results are similar to the LASSI[1,5](11e,10o) ($\max(s_{K\mathcal{P}}) = 0.95$) results. (See supporting information for a complete list.) This means that the entanglement between the two fragments does not increase in the larger active space, suggesting that the higher required q in the larger active space is due rather to the less-accurate modeling of each fragment individually by the LASSI model states constructed from the eigenstates of $\hat{H}_{\mathcal{P}K}$.

2.4.2 Tri Iron System

This system consists of one d^6 Fe^{2+} center and two d^5 Fe^{3+} centers. We explore the (16e,15o) active space of $3d$ orbitals as well as the (16e,30o) active space including $3d$ and $4d$ orbitals. This system is more challenging for MR methods due to the size of active space, and as a reminder, DFT predicts that all Fe centers are identical[58]. We calculate $S = 0, 1, ..., 7$ spin states for each active space for all methods.

Table 2.2 shows the LASSI results for the (16e,15o) active space against the LAS-CASCI limit. The absolute values are reported in SI table XI. LASSI[0,1] incorrectly predicts a high-spin ground state, while the LAS-CASCI predicts a triply-degenerate low-spin ground state (i.e., singlet, triplet, and quintet). The LASSI[1,5] (6910 model states) results differ from the LAS-CASCI (4.1×10^7 determinants) re-

Spin Ladder (in kJ/mol)

Spin	CASPT2†	CASSCF†	LAS-CASCI	LASSI[1,5]	LASSI[1,5$_{\mathrm{CT}}$]
$S = 7$	22.8	6.6	5.0	4.9	4.9
$S = 6$	21.0	4.8	3.5	3.4	3.4
$S = 5$	13.1	3.1	2.2	2.2	2.2
$S = 4$	6.2	1.6	1.2	1.2	1.2
$S = 3$	-	0.5	0.5	0.4	0.4
$S = 2$	-	0.9	-0.1	-0.1	-0.1
$S = 1$	-	0.0	-0.1	-0.1	-0.1
$S = 0$	0.0	0.0	0.0	0.0	0.0
States	n/a	$\approx 4.1 \times 10^7$		6910	3014

Table 2.2: Spin ladder of Fe_3 system with (16e,15o) active space. $S = 0$ state is set as a reference. Results of LAS-CASCI and LASSI[1,5] compared with previous CASSCF and CASPT2 results from Ref. 58 with the same (16e,15o) active space. Empty CASPT2 rows are indicative of results not available in Ref. 58. Absolute and relative energies LASSI[1,5] and LASSI[1,5$_{\mathrm{CT}}$] differ by less than 0.01kJ/mol. LASSI[1,5$_{\mathrm{CT}}$] has 3014 states. †: Results from Ref. 58 use relativistic all-electron ANO-RCC triple-ζ basis sets for all O and Fe atoms, and double-ζ for C and H atoms.

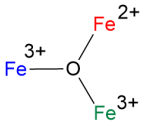
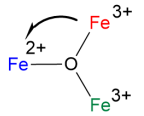
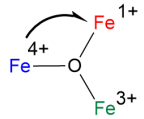
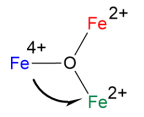
Charge	ΔS	$\bar{n}_{\mathcal{P}K}$
	$\begin{bmatrix} S^{(1)} \\ S^{(2)} \\ S^{(3)} \end{bmatrix} = \begin{bmatrix} 2 \\ 2.5 \\ 2.5 \end{bmatrix}$	$\begin{bmatrix} 0.00 \\ 0.00 \\ 0.00 \end{bmatrix}$
	$\begin{bmatrix} -0.5 \\ -0.5 \\ 0 \end{bmatrix}^*$	$\begin{bmatrix} 1.04 \\ 0.91 \\ 0.00 \end{bmatrix}$
	$\begin{bmatrix} -0.5 \\ -0.5 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 1.18 \\ 2.02 \\ 0.00 \end{bmatrix}$
	$\begin{bmatrix} 0 \\ -0.5 \\ -0.5 \end{bmatrix}$	$\begin{bmatrix} 0.00 \\ 1.19 \\ 1.40 \end{bmatrix}$

Figure 2.2: Summary table of LASSI[1,5](16e,15o) results for the Fe_3 system in different rootspaces. The first column denotes a pictorial representation of the rootspace, in which the charge on each iron is that after the electron is transferred in the direction of the arrow. The change in spin on each iron in the rootspace is shown in the 2nd column. The third column shows the average excitation number [$\bar{n}_{K\mathcal{P}}$, Eq. (2.24)] on each fragment in the given rootspace and it's all possible spin polarizations.

sults only by 0.1 kJ/mol while using four orders of magnitude fewer states. LASSI[1,5] and LASSI[1,5_{CT}] (3014 model states) differ by less than 0.01kJ/mol. Additionally, CASPT2[59] predicts an energy difference between the $S = 0$ and $S = 7$ states of 23 kJ/mol, indicating the importance of dynamical correlation.

However, the LASSI[1,q] approach appears to create significant deadwood in these low-energy wave functions. We find that 48 of the 182 rootspaces generated at the $r = 1$ level have zero average weight [$w_{\mathcal{P}}$, Eq. (2.22)]; these correspond to rootspaces where the charge transfer $Fe^{2+} \rightarrow Fe^{3+}$ results in increased S on Fe^{2+} center. Additionally, we find that the average excitation number [$\bar{n}_{K\mathcal{P}}$, Eq. (2.24)] is zero if the corresponding fragment does not participate in a charge transfer compared

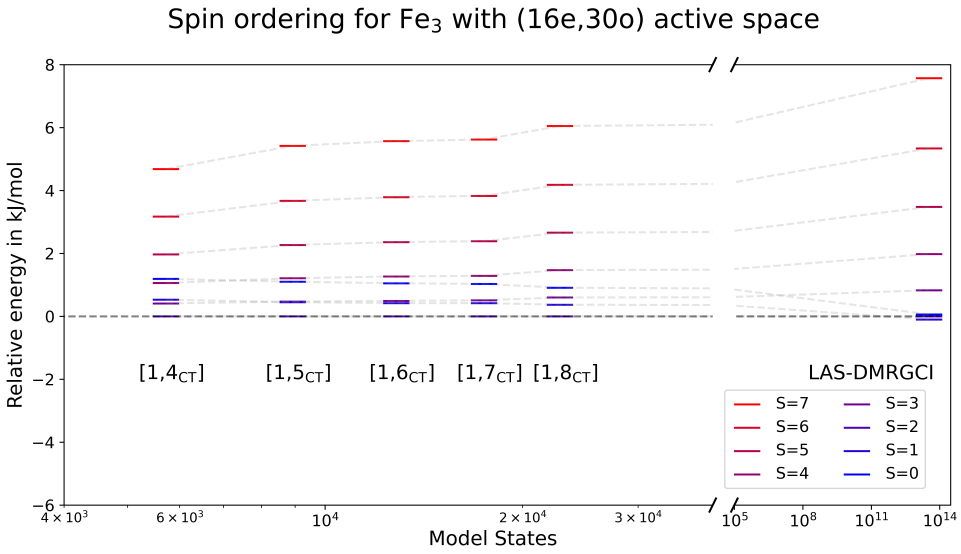


Figure 2.3: Spin ladder for Fe_3 system with (16e,30o) active space. [1,q_{CT}] are the corresponding LASSI calculations. The state $S = 2$ is set as a reference. DMRGCI is placed on the abscissa at the number of single determinants in the corresponding hypothetical LAS-CASCI calculation. LASSI[1,8_{CT}] recovers LAS-DMRGCI spin ladder qualitatively.

to the reference state (Figure 2.2). In other words, for any $r = 1$ rootspace (cf. Table 2.1), a "spectator" fragment does not need local excitations. This observation motivates the development of LASSI[r,q_{CT}], as described in Sec. 2.3.3. The result for LASSI[1,5_{CT}] is quantitatively the same as LASSI[1,5], but with fewer than half the model states of LASSI[1,5].

We now discuss results with (16e,30o) active space presented in Fig. 2.3. Since one cannot perform CASSCF or CASCI in this active space, we have used DMRG [60] to calculate the reference value starting from optimized orbitals of LASSCF(16e,30o). DMRG was performed with a bond dimension $M = 500$. We show DMRGCI results in Fig. 2.3 on the abscissa at 34 trillion states corresponding to the size of a (16e,30o)

CAS calculations. We see that the qualitative features of the DMRG(16e,30o) are generally reproduced at the LASSI[1,8_{CT}] level (22808 model states), except that LASSI predicts a quintet ground state and DMRG predicts a degenerate singlet, triplet, and quintet.

2.5 Automatising LASSI

While the method of LASSI[r, q] can be useful, the method still suffers from a large number of states being included in the model Hamiltonian. Furthermore, the method also requires a user input of the variables r, q that need to be selected based on previous experiences of using the method. We distilled the optimal choices of model space construction into an automatised recipe. The model space now consists of

- The LASSCF reference state
- All fragment statelets that are a single spin away from the LASSCF reference. In other words, state with total spin $S = S_{ref} \pm 1$ of the reference are taken in the model space.
- All single-charge hop states that move at most one electron from one fragment to another, while keeping other fragments at the reference.
- All possible spin polarizations of each of the statelets to ensure that the LASSI wave function is spin pure.

With the recipe, and further dressing of Hamiltonian discussed in detail in a follow-up work[61], we were able to produce the LASSI singles (LASSIS) method.

The method was shown to have fast convergence and qualitatively accuracy compared to exact diagonalization (or DMRGCI when exact diagonalization is not possible).

CHAPTER 3

ENABLING MULTIREFERENCE CALCULATIONS ON MULTIMETALLIC SYSTEMS WITH GRAPHIC PROCESSING UNITS

3.1 Introduction

Efforts on GPU-accelerated quantum chemistry codes for large systems have mainly focused on integral generation[62, 63] and single-reference methods, primarily HF[64], DFT[65] and its relativistic counterpart[66], MP2[67, 64, 68] and CC[69, 70, 71, 72]. GPU-accelerated HF, DFT and MP2 have been applied to large systems. Specifically, HF calculations have been performed on organic molecules, water clusters and glycine clusters [73, 74, 75, 76], DFT calculations have been performed on organic molecules[65], model systems[77], solids[78], and gold clusters for relativistic DFT[66], MP2 calculations have been performed on glycine [68] and water clusters[79].

Among multiconfigurational methods, CASSCF[80, 81, 82], DMRG[83, 84], non-orthogonal CI[85], and auxiliary field quantum Monte Carlo[86] have been implemented to utilize GPUs. Within traditional CASSCF method active space up to (26e, 23o)[87] can be achieved, but (18e, 18o) is considered the limit for routine calculations[88]. GPU-accelerated variational 2-body reduced density matrix (v2RDM) CASSCF [89] has been used with (50e, 50o) active space for polyacenes. GPU-accelerated DMRG can treat active space of (113e, 76o) for FeMo cofactor[83] and (114e, 73o) for P-cluster[84], but only to solve the CI problem on GPUs. We also

highlight the adoption of GPU accelerated DMRG as a CI solver for CASSCF[90].

The memory bottleneck of electronic structure methods such as HF, DFT, MP2, and CASSCF is the processing of 4-center 2-electron (4c-2e) electron repulsion integrals (ERIs) which scales as $\mathcal{O}(N_{\text{ao}}^4)$, where N_{ao} is the number of atomic orbitals. While utilizing direct-SCF principles[91], where integrals are calculated on the fly, are common, within the realm of methods using precomputed integrals, this scaling makes the study of large systems quickly intractable with an increasing number of orbitals, often in the case of many inactive orbitals. Weakly correlated systems like water clusters or organic compounds have sparse ERIs and methods exploiting the sparsity of ERIs on GPUs[92, 93] can achieve significant speedups while keeping memory requirements low. However, sparsity may not exist when solving for organometallic compounds of heavier atoms with diffused orbitals.

Another way to reduce memory requirements is to approximate the 4c-2e integrals using Cholesky density fitting techniques [94] to decompose a 4c-2e ERI into a 3c-2e Cholesky vector. This reduces the memory requirement to $\mathcal{O}(N_{\text{aux}}N_{\text{ao}}^2)$ where N_{aux} is the number of auxiliary orbitals and is usually 5-10x of N_{ao}. Other methods to approximate ERI like chain of spheres[95] and semi-numerical-K[96] have also been developed. Although various single-reference methods with density fitting techniques have been explored in GPU accelerated computational chemistry codes, the use of density fitting with multi-reference methods on GPUs is less common, except for the work by Mullinax *et al.*[89] and a newer work from Wang *et al.*[97]

In this chapter, we discuss our efforts to accelerate LASSCF and LASSI. For each method, we discuss the identification of the bottlenecks, our acceleration philosophy,

and the results of the acceleration.

3.2 Theory and Identification of bottlenecks for LASSCF

Recall that in LASSCF, each fragment is treated as an embedded CASSCF problem. The total number of fragment embedding orbitals (N_{emb_f}) can be up to $2(N_{\mathrm{ac}_f} + N_{F_K})$ where N_{ac_f} is the number of user-selected fragment active orbitals and N_{F_K} is the number of user-selected fragment inactive orbitals. Practically, a calculation starts with performing a Hartree-Fock calculation and selection of active space orbitals for each fragment and spatial atoms on which each fragment is localized. This localization forms the initial guess for the LASSCF calculation. Now, for each fragment, we form the embedding space ("CASSCF processing") and solve the "fragment CASSCF" in each active space. Once CASSCF subproblems are solved, we must now reorthogonalize all active orbitals. Finally, we optimize the whole molecule inactive orbitals along with minimization of energy with respect to rotation between two active spaces ("Recombination").

These CASSCF subproblems or tasks require repeated calculations of effective potentials and ERIs with specific indices pattern of $g^{p_1 a_1}_{p_2 a_2}$ and $g^{p_1 p_2}_{a_1 a_2}$ where a and p respectively refer to active and embedded space orbitals for orbital optimization[98, 99]. For a realistic system, N_{emb_f} and N_{ac_f} do not scale with the system size since fragments necessary for modeling multireference nature would remain similar. Cases where larger fragments may need to be considered, such as extended aromatic systems will have N_{emb_f} and N_{ac_f} scale with the system size.

Once all embedded CASSCF tasks are completed, the active orbitals of all frag-

ments are not necessarily orthogonal since they are optimized independently and must be reorthogonalized. The active orbitals and CI vectors are then frozen and the whole molecule's inactive orbitals are optimized. Additionally, the energy is minimized with respect to rotation between two active subspaces (K, L) and their CI vectors. This corresponds to solving the equation

$$\{E^{(1)}\}_{\mathrm{CI}} = \{E^{(1)}\}_{a_L}^{a_K} = \{E^{(1)}\}_d^v = 0 \tag{3.1}$$

Eq. 3.1 is satisfied through a second order algorithm that optimizes

$$\mathbf{E}^{(1)} + \mathbf{E}^{(2)}\mathbf{x} = \vec{0}, \tag{3.2}$$

where $\mathbf{x}$ is a unitary generator containing all orbital and CI transformation variables, $\mathbf{E}^{(2)}$ is the second-order derivative. We omit details of exact forms of derivatives in Eq. 3.1 and refer the reader to ref. 34. Eq. 3.1 requires a HF-like effective potential repeatedly and ERIs of index pattern $g_{a_{L_1} a_{N_1}}^{a_{K_1} a_{M_1}}$.

Diagrammatically, the algorithm can be represented as in Figure 3.1. In the algorithm, the orange boxes represent the CASSCF tasks; the green boxes represent processing before and after a fragment CASSCF, e.g. embedding potential in Eq. 2.5; and the blue boxes represent the recombination algorithm.

We used LASSCF within a density-fitting framework that splits a 4c-2e integral

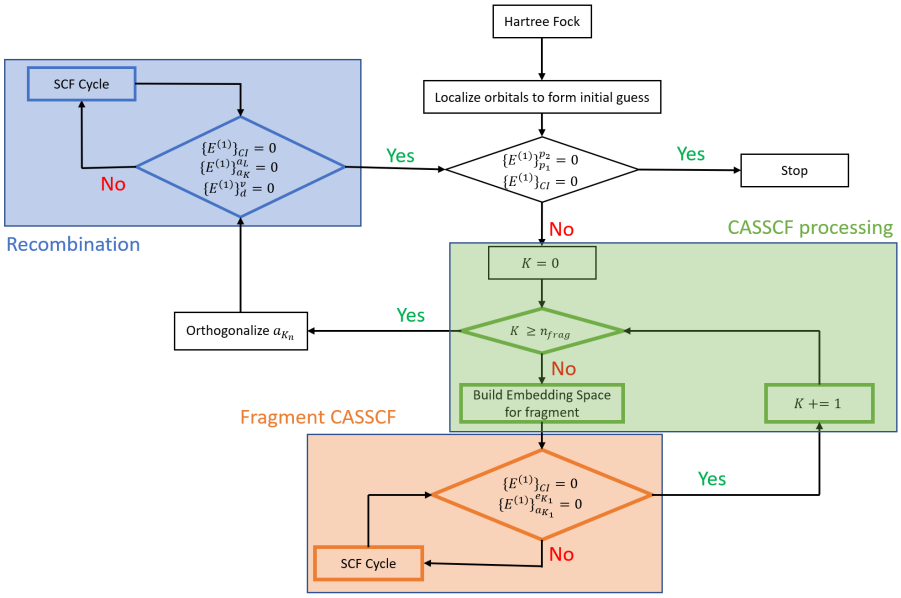


Figure 3.1: Flowchart of LASSCF algorithm. Highlighted boxes represented various phases of calculations. The green box represents constructing an embedding space for each fragment, orange box for performing a CASSCF task and blue box for active orbitals of a fragment are relaxed with respect to other fragments.

into 3c-2e integral[94] given as

$$\begin{aligned} g^{\mu_1\mu_3}_{\mu_2\mu_4} &\approx (\mu_1\mu_2|P)(P|Q)^{-1}(Q|\mu_3\mu_4) \\ &= ((\mu_1\mu_2|P)T)(T(Q|\mu_3\mu_4)) = b^P_{\mu_1\mu_2} b^P_{\mu_3\mu_4} \end{aligned} \tag{3.3}$$

where $b^P_{\mu_1\mu_2}$ are the three-center two-electron Cholesky vectors of size $\mathcal{O}(N_{\text{aux}}N^2_{\text{ao}})$, and T is $(P|Q)^{-\frac{1}{2}}$. We profiled CPU-only LASSCF calculations using simple Python timer functions within PySCF to collect the total time spent during the various stages of the algorithm. This was done for all systems described in section 4.2, and results for the iron-sulfur cluster are shown as a representative example in figure 3.2.

All embedded CASSCF calculations (in the orange boxes of Fig. 3.1) combined accounted for about 33% of the total wall time. Within fragment CASSCFs, the evaluation of effective potentials consumed 50% of the CASSCF wall time, while the computation of ERIs ($g^{p_1a_1}_{p_2a_2}$ and $g^{p_1p_2}_{a_1a_2}$) accounted for another 48%. The pre- and post- processing of CASSCF fragments (shown in the green boxes of Fig. 3.1) contributed 28% of the total wall time, with over 90% of this time spent on evaluating effective potentials. Finally, the recombination of all fragments (blue boxes of Fig. 3.1) consumed 37% of the total wall time with 72% dedicated to effective potential evaluations and 15% to LASSCF ERIs ($g^{p_1a_{M_1}}_{a_{L_1}a_{N_1}}$). A similar breakdown was observed for the other systems. Analyzing where time was spent in the code was repeated throughout the software development process to ensure priority was given to the next largest time-consuming step in the calculation.

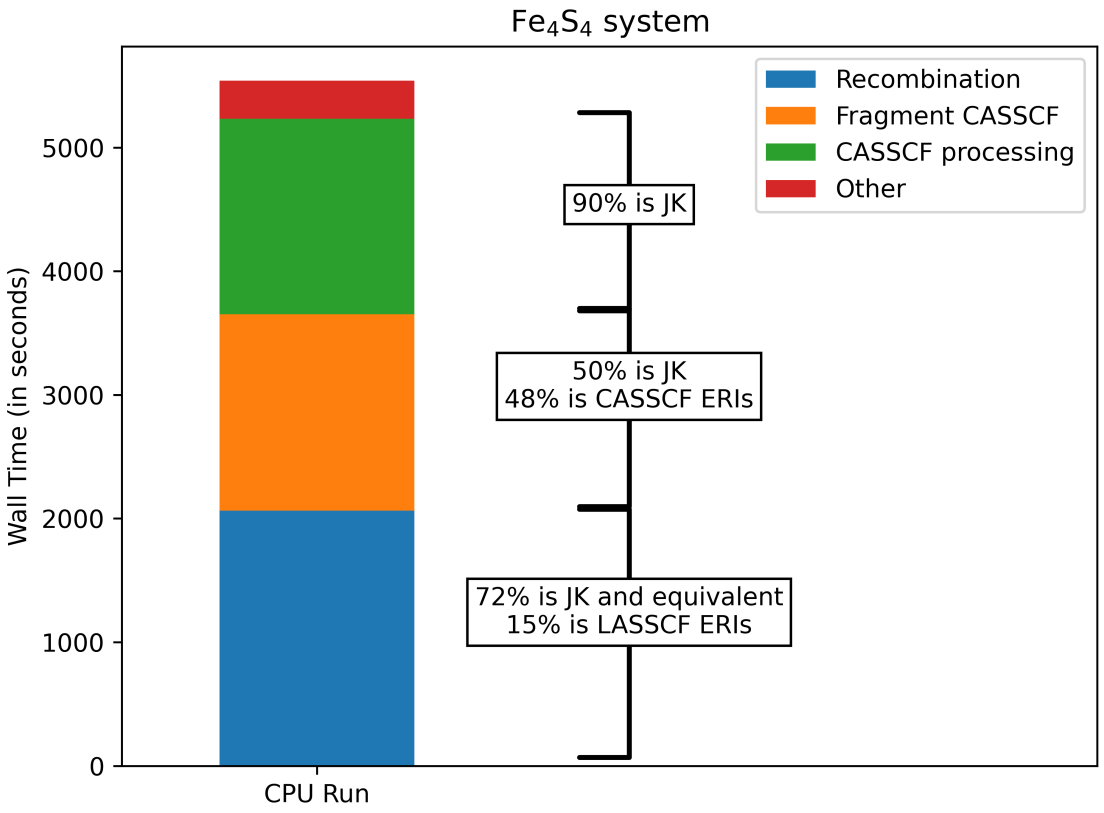


Figure 3.2: CPU run profile for iron-sulfur cluster. Computer details: Polaris compute node: One Milan CPU with 32 cores. The various colors in the plot correspond to areas of the flowchart in Fig. 3.1

3.3 Acceleration of bottlenecks

3.3.1 Calculation of effective potentials

The calculation of effective potentials is performed as follows:

$$\{v^{(jk)}\}_{ij} = J_{ij} - \{K_\sigma\}_{ij} \tag{3.4}$$

where J_{ij} are calculated as

$$v_P = b_{ij}^P D_{ij} \tag{3.5}$$

$$J_{ij} = v_P b_{ij}^P \tag{3.6}$$

and

$$v_{ik}^P = b_{ij}^P \{\gamma_\sigma\}_k^j \tag{3.7}$$

$$\{K_\sigma\}_{ij} = v_{ik}^P b_{jk}^P \tag{3.8}$$

Here i, j represent both atomic orbitals and embedded orbitals since effective potentials are calculated with both. The computation of the Coulomb matrix J and exchange matrix K is collectively referred to as a "JK call". The calculation of J has a complexity of $\mathcal{O}(N_{\text{aux}} N_{\text{ao}}^2)$ if performed for the full system (or $\mathcal{O}(N_{\text{aux}} N_{\text{emb}_f}^2)$ for a single fragment) for both steps, namely Eq. 3.5 and Eq. 3.6 with output of size $\mathcal{O}(N_{\text{ao}}^2)$ (or $\mathcal{O}(N_{\text{emb}_f}^2)$). The exchange matrix ($K$), on the other hand, has a complexity of $\mathcal{O}(N_{\text{aux}} N_{\text{ao}}^3)$ (or $\mathcal{O}(N_{\text{aux}} N_{\text{emb}_f}^3)$) for both steps (Eq. 3.7 and Eq. 3.8), an

order of magnitude more expensive than J, but with the same size of output ($\mathcal{O}(N_{\text{ao}}^2)$ (or $\mathcal{O}(N_{\text{emb}_f}^2)$)) as J. Performing singular value decomposition (SVD) on the density matrix could reduce the complexity of and memory requirements for the calculation of the K matrix. We show later that JK is no longer the largest computational bottleneck after being GPU-accelerated. That said these algorithmic improvements will be implemented in the future to further reduce the time-to-solution. For each effective potential calculation, the input is a density matrix and the AO (or embedded basis) Cholesky vectors and the output is J and K. For each call, we ensure that the relevant Cholesky vectors and density matrix are present on the GPUs, we calculate J and K, and transfer them back to the CPU.

Since the calculation of full K at once requires at least $2N_{\text{aux}}N_{\text{ao}}^2$ memory (one to store original AO basis ERI and one for the intermediate), it becomes taxing on the available computer memory. Most modern software, including PySCF, perform this calculation in blocks of size N_{blk} along P. We currently have kept the PySCF default $N_{\text{blk}} = 240$ for this work. Initial investigations showed smaller blocks help in load balancing, and examples of such explorations are presented in SI of 100. Tuning based on the number of GPUs or dynamically changing N_{blk} could help to ensure work is evenly distributed across GPUs. Dynamically changing the blocksize would require some data reorganization across GPUs (leveraging GPU-GPU links), but ensuring consistency with the rest of the PySCF code will be important.

The Cholesky vectors are often stored in triangular form of size $N_{\text{aux}}N_{\text{ao}}(N_{\text{ao}} + 1)/2$. After transferring a block of this lower-triangular matrix to the GPU, it must be unpacked to the square-matrix form before proceeding. Eq. 3.7 generates an

intermediate (v_{ik}^{P}) of the same size as this block. Furthermore, in order to carry out Eq. (3.8) efficiently, the Cholesky vector factor must be transposed. Therefore, the total peak memory usage for the build of $\{K_\sigma\}_{ij}$ is $N_{\text{aux}}N_{\text{ao}}^2/2 + 3N_{\text{blk}}N_{\text{ao}}^2$. We will discuss data transfer optimizations later.

3.3.2 Calculation of ERIs in fragment CASSCF

The CASSCF orbital optimization requires explicit ERI ($g_{a_{K_1}a_{K_2}}^{p_{K_1}p_{K_2}}$ and $g_{p_{K_2}a_{K_2}}^{p_{K_1}a_{K_1}}$)[99, 98] given by

$$b_{e_{K_2}p_{K_1}}^{P} = b_{e_{K_1}e_{K_2}}^{P} M_{e_{K_1}}^{p_{K_1}} \tag{3.9}$$

$$b_{p_{K_1}p_{K_2}}^{P} = b_{e_{K_2}p_{K_1}}^{P} M_{e_{K_2}}^{p_{K_2}} \tag{3.10}$$

$$g_{p_{K_2}a_{K_2}}^{p_{K_1}a_{K_1}} = b_{p_{K_1}p_{K_2}}^{P} b_{a_{K_1}a_{K_2}}^{P} \tag{3.11}$$

$$g_{a_{K_1}a_{K_2}}^{p_{K_1}p_{K_2}} = b_{p_{K_1}a_{K_1}}^{P} b_{p_{K_2}a_{K_2}}^{P} \tag{3.12}$$

Here, M is the matrix of MO coefficients that transforms from AO basis to MO basis. We refer to this kernel as AO2MO.

The complexity of the first two steps (Eq. 3.9 and Eq. 3.10) is $\mathcal{O}(N_{\text{aux}}N_{\text{emb}_f}^3)$ and of the last two steps (Eq. 3.11 and Eq. 3.12) is $\mathcal{O}(N_{\text{aux}}N_{\text{emb}_f}^2 N_{\text{ac}_f}^2)$. If the first two steps are performed on the GPU and the next two on the CPU, we transfer $b_{p_{K_1}p_{K_2}}^{P}$ (of size $\mathcal{O}(N_{\text{aux}}N_{\text{emb}_f}^2)$) back to the CPU. A GPU memory allocation of $2N_{\text{blk}}N_{\text{emb}_f}^2$ is required to perform the calculation.

If, instead, we perform Eq. 3.9, Eq. 3.10 and Eq. 3.11 on the GPU and Eq. 3.12 on the CPU, we pull $g_{p_{K_2}a_{K_2}}^{p_{K_1}a_{K_1}}$ and $b_{p_{K_1}a_{K_1}}^{P}$ ($N_{\text{emb}_f}^2 N_{\text{ac}_f}^2 + N_{\text{aux}}N_{\text{emb}_f}N_{\text{ac}_f}$ data)

back and require a memory allocation of $2N_{\text{blk}}N^2_{\text{emb}_f} + N^2_{\text{emb}_f}N^2_{\text{ac}_f}$ on GPUs. Since $N_{\text{aux}} \gg N^2_{\text{ac}_f}$ and $N_{\text{emb}_f} \gg N_{\text{ac}_f}$, this approach significantly reduces data transfer compared to the previous option, where only Eq. 3.9 and Eq. 3.10 were executed on the GPU. Furthermore, given that the JK call already requires $3N_{\text{blk}}N^2_{\text{ao}}$ memory and $N_{\text{ao}} \gg N_{\text{emb}_f}$ and $N_{\text{blk}} = 240 > N^2_{\text{ac}_f}$ for most practical cases, this strategy does not impose any additional memory overhead beyond what is required for the JK call.

Finally, performing all 4 steps on a GPU would transfer $2 \times N^2_{\text{emb}_f}N^2_{\text{ac}_f}$ data back to CPU and require a minimum GPU memory allocation of $2N_{\text{blk}}N^2_{\text{emb}_f} + 2N^2_{\text{ac}_f}N^2_{\text{emb}_f}$. In most practical cases, the data transferred is smaller than in the previous approach, as $N_{\text{emb}_f}N_{\text{ac}_f} < N_{\text{aux}}$. Currently, we execute the first three steps on the GPU and the final step on the CPU because Eq. 3.12 is not relatively expensive. This approach makes CASSCF ERIs no longer a bottleneck of LASSCF, although further optimizations may be explored in the future.

The `mrh` research code[101], which implements LASSCF, builds upon the PySCF[88] code. No modification or special version of PySCF is required and users of `mrh` can simply import the relevant PySCF modules and objects needed for LASSCF calculations. Within the LASSCF algorithm, low level functionalities, such as JK calls, are transparently accessed from PySCF. In designing the framework for GPU-accelerated LASSCF calculations, there was a strong emphasis on maintaining a similar level of transparency with PySCF to reduce the installation burden on users while enabling them to fully leverage GPU accelerators. Additionally, there was a strong emphasis on enabling performance-portable calculations, allowing users to take full advantage

of GPU accelerators from various vendors, including AMD, Intel, and NVIDIA. This approach aims to support a broader user community. From a software development perspective, the goal was to minimize code duplication while maintaining flexibility in optimizing the mapping of LASSCF and `mrh` to GPUs. This was achieved, in part, by relying on vendor-optimized matrix-multiply routines to ensure good performance.

The majority of the GPU-related code is self-contained in a stand-alone C/C++ library `libgpu`. Python bindings for the C++ functions are created using the header-only pybind11[102] library, facilitating data exchange between the two programming models. This lightweight Python/C++ interface helps keep the `mrh` code organized, as it is only utilized during GPU-enabled runs. The core development of multiconfigurational algorithms in the Python `mrh` code continues uninterrupted, while the algorithms can be patched to leverage accelerated core functionalities for speedup. A key benefit of this development approach is that other algorithms can also take advantage of the same core functionalities for acceleration, without additional development effort.

Adding new GPU-accelerated functionality is straightforward, and as demonstrated below, the overall application speedup is already promising across several multi-metallic systems on NVIDIA and Intel GPUs. We prioritized which functionalities to accelerate using profiling tools, and NVIDIA's profiling tools such as NVIDIA Nsight Systems[103] on Polaris and Intel's profiling tools such as VTune and Advisor with profilers such as THAPI and iprof to ensure multiple GPUs were utilized effectively. These decisions enabled the development of mini-apps for quicker and independent prototyping and optimization of GPU-accelerated kernels. The trade-off

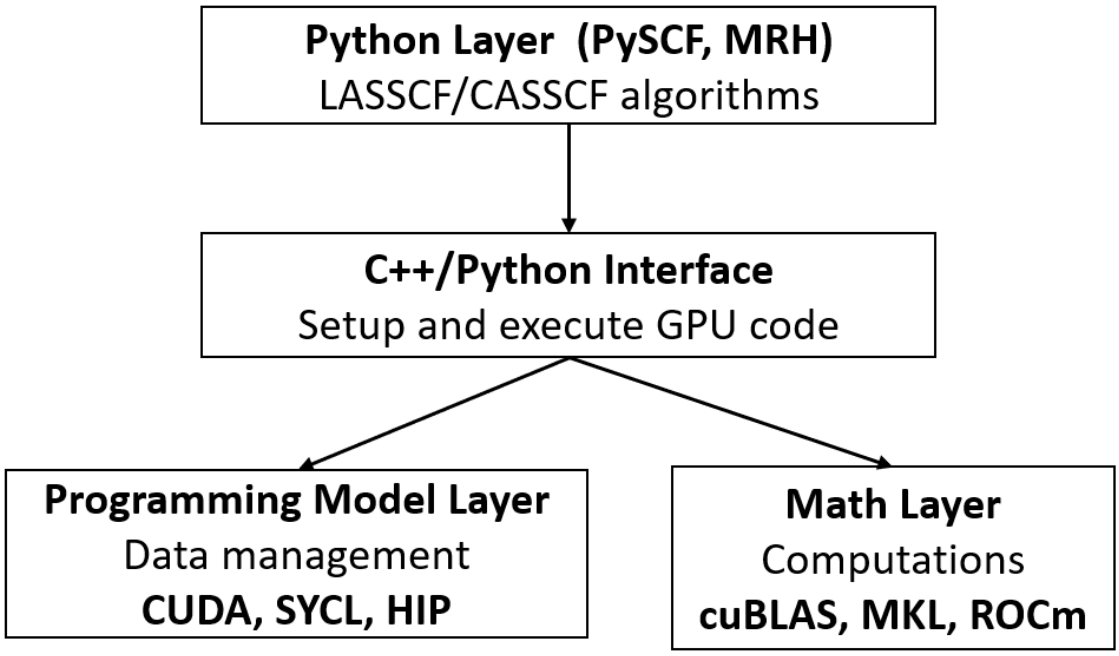


Figure 3.3: General code structure for offloading computational hotspots.

of focusing only on key, separate computational tasks is potentially additional data transfers to ensure the remaining CPU code runs correctly.

3.3.3 Python layer

The LASSCF algorithm leverages several functionalities provided by PySCF. For example, the CASSCF task is solved directly by PySCF drivers, and all effective potential calculations of LASSCF are performed with PySCF's JK engine.

An example file of how to run LASSCF is provided in Fig. 3.4. First, we instantiate PySCF's `mol` object with the geometry, charge and basis set information and perform a ROHF with density fitting. The LASSCF object `las` then uses the

```
from pyscf import gto, scf
mol=gto.M(atom='geom.xyz', basis=basis)
mf=scf.ROHF(mol).density_fit().run()
from mrh.my_pyscf.mcscf.lasscf_async import
↪  LASSCF
las=LASSCF(mf, (no1, no2), (ne1, ne2))
lo=las.set_fragments_
↪  ((atom_list1,atom_list2),mf.mo_coeff)
las.kernel(lo)
```

Figure 3.4: A sample code to run a LASSCF calculation with a `geom.xyz` geometry file, two fragments of active spaces with `(ne1,ne2)` electrons, `(no1,no2)` orbitals, localized on `(atom_list1,atom_list2)` atom numbers. LASSCF is started with an initial guess orbitals `lo`.

PySCF's mean field object (`mf`) along with, for this example, two fragments defined with `no1,no2` orbitals and `ne1,ne2` electrons localized on `atom_list1, atom_list2`, respectively. The orbitals are localized on each fragment, and used to perform the LASSCF calculation.

To avoid modifying the PySCF library, we monkey-patched the JK operation (eq. 3.5-3.8) and AO2MO functionalities (eq. 3.9 - 3.12) using decorators on existing PySCF functions, redirecting these calls to our GPU accelerated versions. This is identical to how `gpu4pyscf`[104, 105] modifies code executed at runtime. Additionally, we patched the `mol` object in PySCF to include a `use_gpu` flag, which readily enables branching for GPU usage without cluttering code too much. To minimize changes on the Python side of `mrh`, we used this flag to create if-else branches that direct the code to utilize the GPU-accelerated functionalities where applicable.

A sample code for running a GPU-accelerated calculation is provided in fig. 3.5. We import the `libgpu` library (subsection 3.3.4) to access GPU functionalities and call `patch_pyscf` that performs the monkey-patching described above. We initialize the GPU device(s) and pass the `use_gpu` flag in the SCF and LASSCF methods to enable the GPU-accelerated branches. Finally, after retrieving the relevant statistics for GPU usage, we free all allocated memory and release the GPUs. Overall, running a GPU-accelerated calculation is relatively straightforward requiring minimal changes to the Python script.

3.3.4 C++ layer

The C++ layer is written with portability in mind such that developers (and users) can take advantage of various accelerators without rewriting algorithms. The GPU accelerated C/C++ library `libgpu` is bound to python and exchanges data through pybind11, avoiding deep copies and instead using pointers to data when possible. The library has a concise programming model layer that is responsible for data management and streams/queues for scheduling work on the GPUs. The underlying programming model can be switched out to CUDA, SYCL, or HIP depending on the GPU accelerator. Similarly, math library abstractions for routine matrix multiplications (GEMM and Batched GEMM) are used. Other mathematical operations like transposes and packing/unpacking data were developed as hand-written kernels using CUDA as much of the initial development leveraged NVIDIA GPUs. These hand-written CUDA kernels are then translated with SYCLomatic[106] and HIPify[107] to generate the corresponding SYCL and HIP versions respectively. Only minor

```
from mrh.my_pyscf import libgpu
import pyscf
from gpu4mrh import patch_pyscf
from pyscf import gto, scf
gpu = libgpu.libgpu_init()
mol = gto.M(atom='geom.xyz', basis=basis,
↪  use_gpu = gpu)
mf = scf.ROHF(mol).density_fit().run()
from mrh.my_pyscf.mcscf.lasscf_async import
↪  LASSCF
las = LASSCF(mf, (no1, no2), (ne1, ne2),
↪  use_gpu = gpu)
lo = las.set_fragments_
↪  ((atom_list1,atom_list2),mf.mo_coeff)
las.kernel(lo)
libgpu.libgpu_destroy_device(gpu)
```

Figure 3.5: A sample code to run a GPU-accelerated LASSCF calculation. The LASSCF setup is similar as previous example, with `libgpu` imported for instructing code to use GPU-accelerated library, `gpu4mrh` being used to patch `pyscf` functionalities. `use_gpu` is tagged to LASSCF method that ensures all GPU accelerated branches are used.

modifications to the translated code are made to resolve compilation issues and for the purpose of this paper they have not been optimized further. The high-level algorithms implemented in libgpu, such as JK, leverage the interfaces (programming model and math) such that they only need to be written once, and automatically run on any supported GPU once the needed kernels have been translated. The goal with developing the software in this manner was not to recreate a robust abstraction framework like Kokkos[108, 109, 110], but remain agile while understanding how best to leverage GPUs in LASSCF calculations and keeping additional dependencies to a minimum. It is straightforward to add support for and explore external optimized libraries for select tasks, such as MAGMA[111] and cuTENSOR[112], without needing to edit the quantum chemistry algorithms. There is also a CPU backend available, but that is only used to aid in the initial development of new algorithms. The performance of the CUDA version on NVIDIA A100 GPUs and the SYCL version on Intel Max Series GPUs is discussed in section 4.2.

3.3.5 Algorithms for Performance

The JK and AO2MO functions are fundamental operations that are called thousands and hundreds of times, respectively, within a single LASSCF calculation. Because the implementation is not fully GPU-resident, the GPU portions of the calculation are interleaved with CPU operations. These functions require the same Cholesky vectors as input, and naively transferring them to the GPU every call would be extremely inefficient, as the data transfer time would negate any performance gains. While on-the-fly integral generation techniques on the GPU are available, we opted

to store all Cholesky vectors blockwise on the GPUs and use an efficient hashing scheme to access vector blocks. This hashing technique utilizes all GPUs on a node without relying on MPI.

To fully utilize all GPUs, we must ensure that kernels are launched asynchronously on GPUs, and any barriers are removed. Operations involving GPU memory or interactions with CPU pageable memory would typically require the CPU to wait until the operation completes before starting new tasks, such as scheduling additional work on the GPUs. To minimize this delay, new allocations are kept to a minimum, and existing allocations are reused whenever possible. Intermediate results from different kernels executed on GPUs are stored in a common scratch space. Data transfer is performed using pinned memory as necessary to optimize performance.

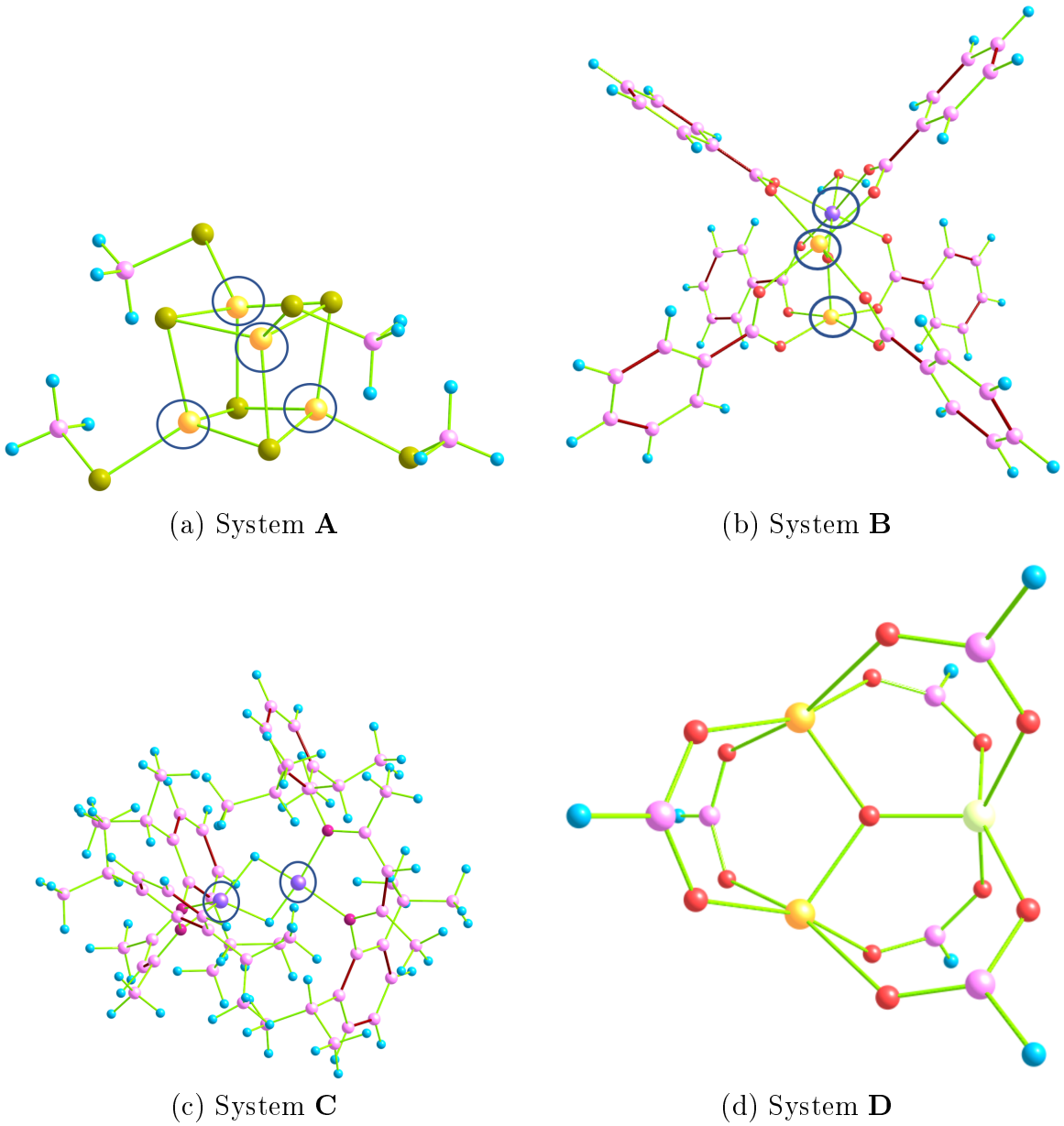
The calculations were performed on two of Argonne's Leadership Computing Facility (ALCF) supercomputers: Polaris and Aurora. Polaris compute nodes have 1 AMD "Milan" 32-core CPU and 4 NVIDIA A100 40GB GPUs. All A100s are linked to each other with NVLink. Aurora compute nodes have 2 Intel Xeon Max Series 52-core CPUs and 6 Intel Data Center Max Series GPUs code-named "Ponte Vecchio (PVC)". All PVCs are linked to each other with Xe link. Each Intel GPU in Aurora consists of two physical stacks that can be separately targeted to run independent work. All Aurora runs reported here distributed work across individual stacks (i.e. 12 separate queues of overlapping work when running on 6 GPUs). CPU runs are performed on Polaris with a single CPU using 32 threads. Polaris' GPU runs are performed with a single CPU using 32 threads and up to 4 A100 GPUs. Aurora runs are performed using a single CPU with 32 threads only, with up to 6 PVC GPUs. We

refer readers to the technical paper on Polaris supercomputer for specific details on the technical paper[113]. Technical paper for Aurora is currently under preparation. Briefly, the Polaris node Milan CPU's peak performance is 1.4 Tera floating point operations (TFLOPs), and each A100 has peak performance of 19.5 TFLOPs.

3.3.6 Systems under study

We used polyacetylenes of form $(C_2H_2)_nH_2$, where n is the number of monomers, for basic profiling, testing accuracy, understanding effects of performance with varying system parameters and limits of LASSCF calculations within given resources. This system can however be misleading and be an unreasonable approximation for the workload of a chemically meaningful system. This can lead make priorities skewed for acceleration when the goal is to accelerate LASSCF for meaningful systems. Three multi-metallic systems were considered to assess the performance of the GPU-accelerated LASSCF implementation. As a stress test of the algorithm, we were able to perform at least one LASSCF iteration for a (96e,96o) calculation for a 48 fragment polyacetylene with a GPU accelerated run within an hour.

The first system under consideration is an iron-sulfur (Fe_4S_4) cluster (Fig 3.6(a)). We refer to this as system **A** in the remainder of the manuscript. Conventionally, the iron sulfur tetramer is thought of as two iron sulfur dimers with $Fe^{+2.5}$ interacting antiferromagentically to form a singlet ground state, we designate two Fe centers as Fe^{+2} and the other two as Fe^{+3}. The local active spaces for Fe^{+3} and Fe^{+2} are (5e,10o) and (6e,10o) respectively, which include the 5 and 6 $3d$ electrons and all the $3d$ and $4d$ orbitals of each Fe center. This results in a total active space of (22e,40o),



(a) System **A**

(b) System **B**

(c) System **C**

(d) System **D**

Figure 3.6: Systems under study. (a) Fe_4S_4 cluster: System **A**, (b) Fe_2Ni trimetallic MOF node: System **B**, (c) Homobimetallic nickel hydride catalyst: System **C**, and (d) $AlFe_2$ trimetallic oxo complex: System **D**. Color: Atom - Yellow: Fe, Purple: Ni, Lime Green: Al, Olive Green: S, Pink: C, Red: O, Blue: H.

which is a typical size for systems with multiple metal atoms, such as the MoFe cofactor[84, 83] and Photosystem II [114]. The Fe and S atoms are described using a cc-pVTZ basis set, while for C and H, a cc-pVDZ basis is used. This gives the total number of orbitals as 660 and the total number of auxiliary orbitals as 4236.

Next, we performed LASSCF on the model of a node of the MIL-127 metal-organic framework (MOF) (system **B**) used in propylene oligomerization[115]. The structure (Fig. 3.6(b)) consists of one Ni^{+2} and two Fe^{+3} centers, bridged by six benzoate linkers and one H_2O molecule. The active spaces of Ni^{+2} and Fe^{+3}, (8e,10o) and (5e,10o) respectively, include the $3d$ electrons and $3d$ and $4d$ orbitals, resulting in a total active space of (18e,30o). The basis set used for Ni, Fe and central O is cc-pVTZ[116] and cc-pVDZ[116] is used for the rest of the atoms. The total number of orbitals in this system is 1164 and the total number of auxiliary orbitals is 8905.

The third test system is a homobimetallic nickel hydride catalyst (system **C**) shown in Fig. 3.6 (c). The system has two Ni^{+2} centers, corresponding to an active space of (8e,10o) on each atom which includes all $3d$ electrons and orbitals and $4d$ orbitals. This compound can be immobilized on silica support to be used for hydrogenation of unsaturated hydrocarbons.[117] The Ni atoms were modeled with the def2-TZVP basis while all other atoms with the def2-SVP basis. This gives 1378 orbitals in total and the total number of auxiliary orbitals is 10296.

The rationale for selecting these systems is to explore cases with different number of fragments, active space and basis set sizes. From **A** to **C** the number of fragments and total active space size decreases, while the number of basis functions increases. Strictly speaking, performance from **A** to **C** is not directly comparable as the total

active space, number of fragments, number of basis functions and complexity of each part (number of iterations in each SCF cycle of CASSCF and recombination) changes, making it difficult to predict the time-to-solution going from **A** to **B** to **C**.

After those heuristics, we present the results denoting the end-to-end time to converge a LASSCF calculation for **A** and **B** with the same initial guess both with CPU-only and GPU-accelerated implementations. The results presented below focus exclusively on performance profiling and the differences between GPU and CPU implementations. We computed total electronic energies for fixed geometries; these calculations do not explore any chemical properties of the systems. Their sole purpose is to demonstrate the feasibility of such computations.

While end-to-end time of a converged LASSCF calculation provides a way to evaluate an implementation's practical usefulness, it may be difficult to perform a detailed analysis of the performance of the accelerated implementation using this data for two reasons. First, the overall optimization may require different number of LASSCF cycles with the same input due to different convergence paths, including overcoming local minima, leading to different workloads (see SI fig. S1-S6). Second, the workload for various parts of a single LASSCF cycle, like CASSCF or recombination steps can vary substantially cycle to cycle (see SI fig S1-S6). For these reasons, we also present more robust tests of CPU-only and GPU-accelerated LASSCF iterations where two parameters, namely, maximum number of recombination cycles and total number of LASSCF cycles are arbitrarily fixed at 10 to give similar workloads for all runs for a given system.

We also performed a CASSCF calculation (not LASSCF) on a Al-Fe oxide cluster

(system **D**), as shown in Fig. 3.6. The rational for these calculations is that the effective potentials and CASSCF ERIs dominate the computational cost in fragment-based calculations. Therefore, accelerating these functionalities should also result in performant CASSCF calculations. We used an (11e,10o) active space, including all $3d$ orbitals and electrons for the two Fe centers. Hydrogen and carbon atoms were modeled with the cc-pVDZ[116] basis set, while oxygen, aluminum, and iron were modeled with the cc-pVTZ[116] basis set, resulting in a total of 674 atomic basis functions and a total of 3998 auxiliary basis functions.

Finally, for completeness, we demonstrate how varying the number of GPUs impacts the performance of LASSCF (system **A**) and CASSCF (system **D**). Analyzing the calculation profiles helps further identify potential serialization and its effect on performance.

The atomic coordinates of the systems, input files, output files, and a sample analyzing script for all calculations can be found in ref. 118.

3.3.7 LASSCF scaling heuristics with polyacytelene systems

The plots for all runs are presented in SI Sec. of Ref. [100]. As we increase the basis set from 6-31g to cc-pVDZ to cc-pVTZ while maintaining the same number of fragments and the total active space size where a total of ((24,24) active space is partitioned into 12 fragments of (2,2), the processing cost escalates most rapidly. This is because it predominantly comprises almost entirely of JK calls that scale with system size. Recombination costs go up with the system size, but contain other computations as well, making the scaling not so dramatic. Finally, CASSCF costs

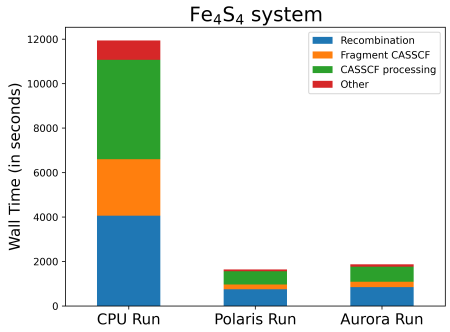
also increase, but tend to do so slowly since the embedding size does not increase linearly with system size. Consequently, we also see an increase in the acceleration as JK calls are highly accelerated.

Changing fragment active space sizes while keeping the total active space constant, i.e., a (24,24) active space being split into 2 spaces of (12,12), 3 spaces of (8,8), 4 spaces of (6,6), 6 spaces of (4,4) and 12 spaces of (2,2) increased the cost of CASSCF processing linearly with the number of fragments. CASSCF costs decrease due to decrease in FCI costs and possible simplification of fragment leading to a lower number of CASSCF iterations being performed. Recombination costs should remain about the same.

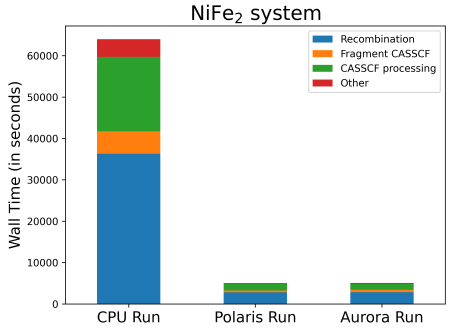
Finally, while varying the total active space as 2 spaces of (4,4), 2 spaces of (6,6), 2 spaces of (8,8) and 2 spaces of (12,12), we expect the cost of LASSCF to grow due to increase in FCI costs. However, the FCI problems are still very small compared to other parts of the algorithm. Therefore, no clear heuristics can be derived for this system.

3.3.8 Performance of LASSCF and CASSCF for multi-metallic systems

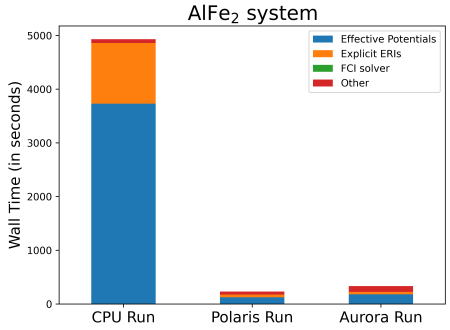
The performance of LASSCF and CASSCF till convergence is presented in Fig. 3.7. The figure presents a CPU-only run on a Polaris node, a GPU-accelerated run with Polaris (4 A100 GPUs) and with Aurora (4 PVC GPUs). Only 4 of the 6 GPUs on an Aurora node were used in these plots in an attempt to make an apples-to-apples comparison. The results are further discussed in the first two and third rows of table



(a) System **A** LASSCF performance with Polaris and Aurora Nodes.



(b) System **B** LASSCF performance with Polaris and Aurora Nodes.



(c) System **D** CASSCF performance with Polaris and Aurora Nodes.

Figure 3.7: Calculation profiles for: (a) LASSCF on system **A** with (22e,40o) active space,(b) LASSCF on system **B** with (18e,30o) active space, (c) CASSCF on system **D** with (11e,10o) active space, with Polaris and Aurora nodes. Resources used: Polaris: Milan CPU 32 Threads + 4 A100 GPUs; Aurora: Max Series CPU 32 Threads + 6 PVC GPUs with 2 queues per GPU.

3.1 for LASSCF and CASSCF respectively. We present the wall time for all runs, their speedups, and GPU active time, which shows how heavily GPUs are being used. Finally, we discuss the speedup of individual parts of LASSCF in table 3.2 in the first two columns.

LASSCF to Convergence: System **A** LASSCF is accelerated by about 7x for both architectures, and **B** LASSCF is accelerated by about 13x for both architectures. Going from **A** to **B**, the workload on recombination and CASSCF processing increases significantly due to increase in the N_{ao}, resulting in JK operations with substantially more work and ultimately leading to a higher speedups. Fragment CASSCFs, on the other hand, are similarly accelerated because the individual fragments are similar in embedding size and have similar active spaces.

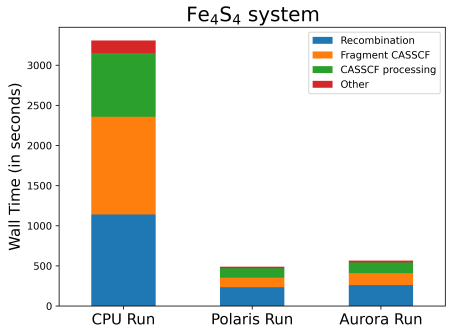
CASSCF to convergence: For system **D**, profiling CASSCF in manner similar to Fig. 3.2 showed that the effective potentials take about 77% of the wall time, and ERIs take about 22% of the wall time (Fig. 3.7(c)), the functionalities that we accelerated for LASSCF. Other parts of the calculation, including the exact FCI solver (not visible as a small green bar) are relatively very small for this system. The GPU-runs are accelerated by 15-20x. Since over 98% of the CASSCF runtime was from kernels that were accelerated, we see a very high relative speedup. This speedup is significantly higher compared to fragment CASSCFs ran within LASSCF since each cycle performs multiple CASSCFs of varying sizes and complexity, and as the number of LASSCF cycles goes on, fragment CASSCFs tend to converge very quickly in 3-4 iterations, leading to a much lower workload and hence lower speedups. Additionally, the embedding space size is significantly smaller in LASSCF's CASSCFs, leading to

smaller workloads. For a smaller CASSCF problem, we have presented absolute performance metrics in SI of Ref. 100.

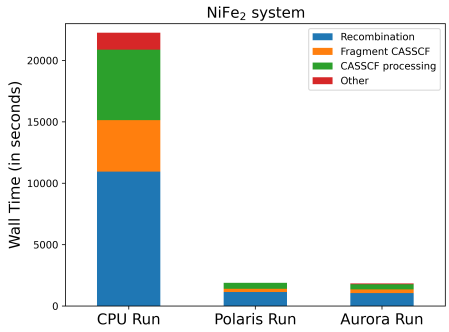
Controlled Workload LASSCF: For a more controlled workload comparison, each LASSCF run for **A**, **B** and **C** were executed with identical parameters: LASSCF maximum cycles were fixed to 10, CASSCF maximum cycles were fixed at 50, and recombination maximum cycles were fixed at 10. CASSCF or recombination may converge earlier than the maximum cycles but this setup limits unreasonably large number of iterations in one phase of the calculation, which may happen due to a variety of reasons such as local minimas or moving out of convergence basin altogether for a specific iteration. The results are presented in Fig. 3.8 and further discussed in the last three rows of table 3.1 and last three columns of table 3.2.

For system **A** (fig. 3.8(a)) and **B** (fig. 3.8(b)), the processing has the same speedup as running till convergence because the number of calls is the same for both CPU and GPU runs. We believe that the recombination speedup from unit runs are much closer to actual performance because the workload can vary substantially in production runs due to numerical convergence of the SCF algorithms.

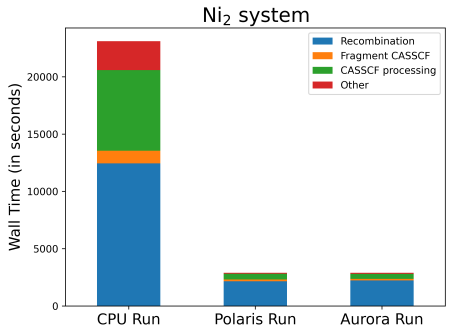
From **A** to **C**, the processing step gets more accelerated due to it being mostly a JK call that gets accelerated with increased workload. The recombination step performance increases from **A** to **B**, and decreases from **B** to **C**. From **A** to **B**, it's simply an increased workload leading to higher speedup. However, for **C**, the calculations have highly variable workloads among runs. For the specific run presented, the CPU run perform 37 total SCF cycles during the recombination step over 10 LASSCF cycles, compared to 51 of Polaris and 55 of Aurora run. Rerunning the



(a) System **A** LASSCF performance with Polaris and Aurora Nodes.



(b) System **B** LASSCF performance with Polaris and Aurora Nodes.



(c) System **C** LASSCF performance with Polaris and Aurora Nodes.

Figure 3.8: Calculation profiles for: (a) LASSCF on system **A** with (22e,40o) active space,(b) LASSCF on system **B** with (18e,30o) active space, (c) LASSCF on system **C** with (11e,10o) active space, with Polaris and Aurora nodes for controlled workloads. Resources used: Polaris: Milan CPU 32 Threads + 4 A100 GPUs; Aurora: Max Series CPU 32 Threads + 6 PVC GPUs with 2 queues per GPU.

same calculations may change the number of cycles significantly. Comparing per recombination SCF cycle speedups in system **C**, we get an approximate speedup of 8x, much closer to **B**. For fragment CASSCFs, we take all of them to have about the same speedups because the embedding space size and active space size is similar for all systems. The difference in cost comes from two areas. First, the number of CASSCF steps can differ significantly in the runs with same input based on convergence patterns, and second, often the CASSCF can converge quickly, sometimes in one or two cycles, within a LASSCF cycle, leading to insufficient workloads for performance to be apparent.

We note that a higher speedup is generally characterized by higher amount of time spent utilizing the GPUs, which is expected for larger workloads and when the CPU-GPU data transfers are minimized. The overall GPU usage is still low with considerable work still being done on the CPU, but we are achieving 7-10x speedups for LASSCF with 25% active time, and about 20x speedup for CASSCF with 50% active time. Individual bottleneck acceleration and overall acceleration does not exactly match the peak efficiency of processing power provided because of several reasons that include modest GPU utilization, CPU-only kernels, single-GPU kernels, communications overheads with GPU, and serialization and load imbalance with multi-GPU kernels. Effort is underway to further improve performance by moving more of the computation to GPUs and make existing computations more efficient.

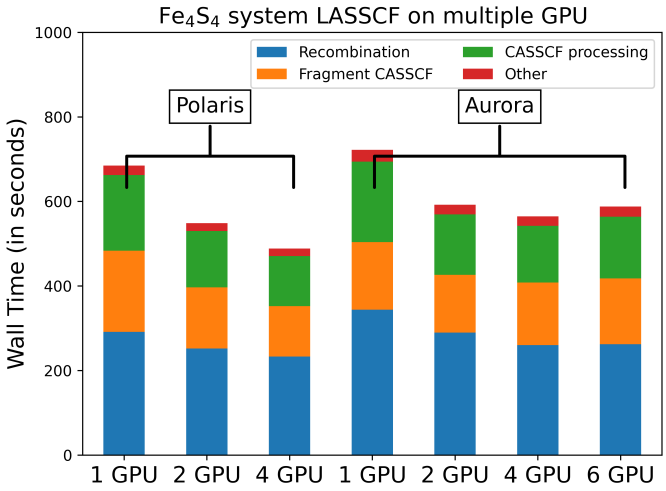
		CPU Only	GPU Wall Run Time (s)		GPU Time (s)	
Method	System	Wall Time (s)	Polaris	Aurora	Polaris	Aurora
LASSCF convergence	**A**	11938	1644 (7.3x)	1927 (6.2x)	445 (26%)	736 (38%)
	B	63993	5045 (12.6x)	5044 (12.6x)	1994(39%)	2057 (40.7%)
CASSCF	**D**	4931	230 (21.4x)	332 (14.9x)	146 (63%)	195 (59%)
LASSCF unit	**A**	3307	488 (6.8x)	564 (5.9x)	131 (27%)	204 (36%)
	B	22260	1842 (12.1x)	1833 (12.1x)	747 (40%)	753 (41%)
	C	23105	2895 (8x)	2214 (10.5x)	1217 (42%)	998 (45%)

Table 3.1: Performance of LASSCF and CASSCF on Polaris and Aurora nodes. Speedup shown in GPU Run Wall Time column in parenthesis. % active time for GPU is shown in GPU Time Column.

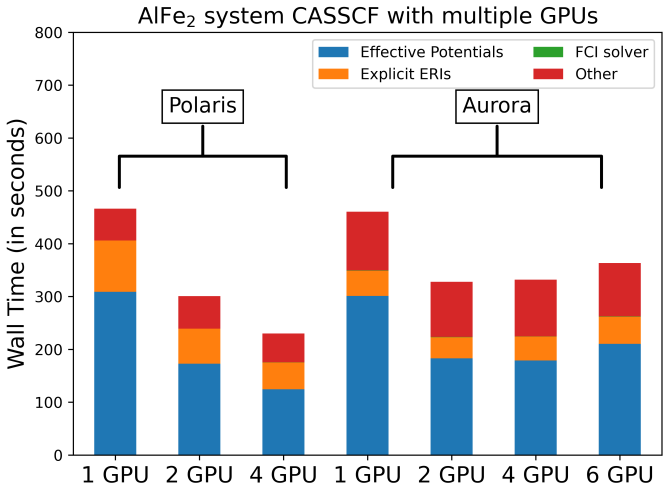
3.3.9 Performance scaling with multiple GPUs for LASSCF

For scaling tests, the Polaris runs are performed on 1 CPU with 32 cores and varying the number of GPUs as 1, 2 and 4. The Aurora runs are performed on 1 CPU with 32 cores and number of GPUs are varied as 1, 2, 4 and 6. We perform the same CASSCF run on system **D** as in (fig. 3.7(c)) subsection and the same LASSCF run on system **A** (fig. 3.8(a)). For each run, we plot the time for each phase of the calculation in Fig. 3.9 and the relevant statistics are presented in Table 3.3.

LASSCF and CASSCF scaling (Fig. 3.9), in the case of the Polaris node, performance increases visibly when going from 1 GPU to 4 GPU; however, the scaling is not ideal. For the Aurora node, end-to-end time decreases from 1 GPU to 2 GPUs, remains constant with 4 GPUs, and then increases again on 6 GPUs. The accumulation of results from multiple GPUs to CPU is serialized in this implementation and is one of the main reasons for the slowdown on 6 Aurora GPUs (i.e. 12 separate SYCL queues). This effect is compounded by load imbalance, where an unevenly distributed workload, such as 1 GPU getting three blocks of work and the rest getting two blocks would still cost three blocks' worth of time. A similar effect is seen in CASSCF runs but the penalty is more pronounced since the GPU accelerated kernels have started to account for the majority of run time. Removing these deficiencies is a work currently in progress to improve the code.



(a) LASSCF runs on system **A** with varying number of NVIDIA and Intel GPUs



(b) CASSCF runs on system **D** with varying number of NVIDIA and Intel GPUs

Figure 3.9: Scaling of (a) LASSCF and (b) CASSCF with varying GPUs. Resources used: Polaris: Milan CPU 32 Threads + 1/2/4 A100 GPUs; Aurora: Max Series CPU 32 Threads + 1/2/4/6 PVC GPUs with 2 queues per GPU.

		Convergence		Units		
LASSCF part	Node	**A**	**B**	**A**	**B**	**C**
Processing	Polaris	7.5	11.2	6.7	11.6	14.2
	Aurora	6.6	12.7	5.9	14.0	16.1
CASSCF	Polaris	11.8	13.0	10.2	16.0	7.0
	Aurora	10.6	9.5	8.2	13.7	8.5
Recombination	Polaris	5.4	12.8	4.9	9.6	5.8
	Aurora	4.8	12.8	4.4	10.4	5.6

Table 3.2: Speedup of individual parts of LASSCF to convergence and unit tests with Polaris and Aurora Nodes.

Method	n_{GPU}	Wall Time (s)		GPU Time (s) (Active time %)	
		Polaris	Aurora	Polaris	Aurora
CASSCF on **D**	1	466	461	374 (80%)	320 (69%)
	2	301	328	209 (69%)	195 (59%)
	4	230	332	146 (63%)	195 (59%)
	6	–	363	–	234 (64%)
LASSCF on **A**	1	685	722	323 (47%)	363 (50%)
	2	549	592	184 (33%)	229 (39%)
	4	488	565	131 (26%)	204 (36%)
	6	–	587	–	228 (39%)

Table 3.3: Scaling performance of LASSCF and CASSCF with multiple GPUs

3.4 Identification of bottlenecks for LASSIS

The goal of LASSIS is to approximate a large active-space CASCI wave function by partitioning the active orbitals into localized fragments. A LASSIS calculation proceeds in three stages.

1. Optimization of the CI vectors for each fragment statelet ("LASSIS state preparation") based on the heuristics discussed in Sec. 2.3;

2. Construction of the model-space Hamiltonian matrix, either as a dense matrix or as a callable matrix-vector product function ("Hamiltonian con- struction"); and

3. Diagonalization of the Hamiltonian, either via direct non-iterative diagonalization or using a Davidson iteration algorithm to obtain the SI vector for each target state.

Depending on the fragmentation scheme, the computational bottleneck arises from different stages of the calculation. For a small number of large fragments, the dominant cost is associated with state preparation and Hamiltonian construction (steps 1 and 2), whereas for many small fragments the dominant cost shifts to Hamiltonian diagonalization (step 3). The former scales primarily with the size of the fragment active spaces, while the latter scales with the dimension of the LASSIS model space, which grows rapidly with the number of fragments. To test the cost of the method, we use a toy example below where the ground state for a 12-mer of acetylene is calculated. All π and π^* orbitals are taken in the active space, making the total

active space equal to (24e, 24o). We split this into several options: 2 spaces of (12e, 12o) each, 3 spaces of (8e, 8o) each, 4 spaces of (6e, 6o) each and 6 spaces of (4e, 4o) each. The active spaces are localized on the relevant monomers, i.e., 6 units, 4 units, 3 units, and 2 units for the 2, 3, 4, and 6 fragment cases, respectively.

We present the end-to-end cost of converged calculations in Fig. 3.10. The cost of state preparation is dominant when we solve a problem with a few large fragment active spaces ((12e, 12o) x 2 fragments). The diagonalization step, on the other hand, grows more expensive as the number of fragments increases. In this example, we perform Davidson diagonalization and only obtain the ground state. The numbers of states in the model space for this molecule under the various fragmentation schemes are presented in Table 3.4.

System	Total Model States
(12e, 12o) x 2	305
(8e, 8o) x 3	3293
(6e, 6o) x 4	43162
(4e, 4o) x 6	5378777

Table 3.4: Problem size metrics for different LASSIS calculations with polyene examples.

The state preparation and Hamiltonian diagonalization steps of the algorithm both consist of iterative algorithms. (The Hamiltonian construction, by contrast, is a one-shot process that includes accumulating all TDMs and building the Hamiltonian.) Fig. 3.11 shows the approximate time cost per iteration of the calculations reported in Fig. 3.10. Interpreting this data requires some nuance. First, the number of iterative sub-problems to be solved in the state preparation step grows with the number of model states and hence with the number of fragments. These various

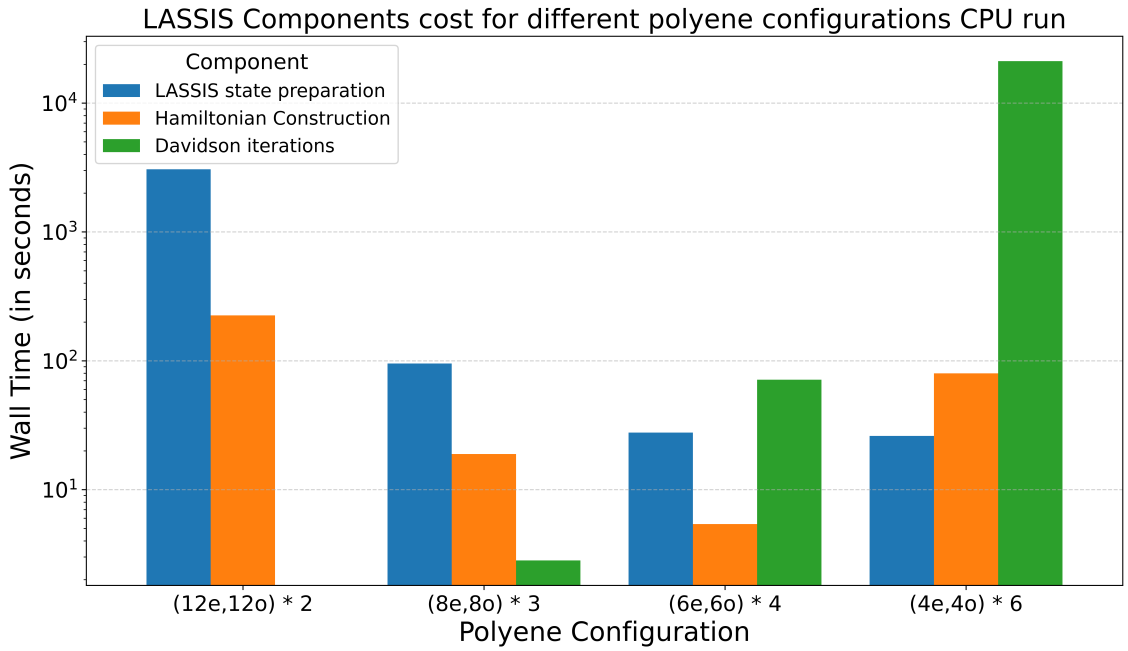


Figure 3.10: Cumulative cost of various parts of the LASSIS algorithm run on a CPU on Polaris, described in the Computational Resources Section. Note that the y-axis is logarithmic, so differences are much larger than perceived.

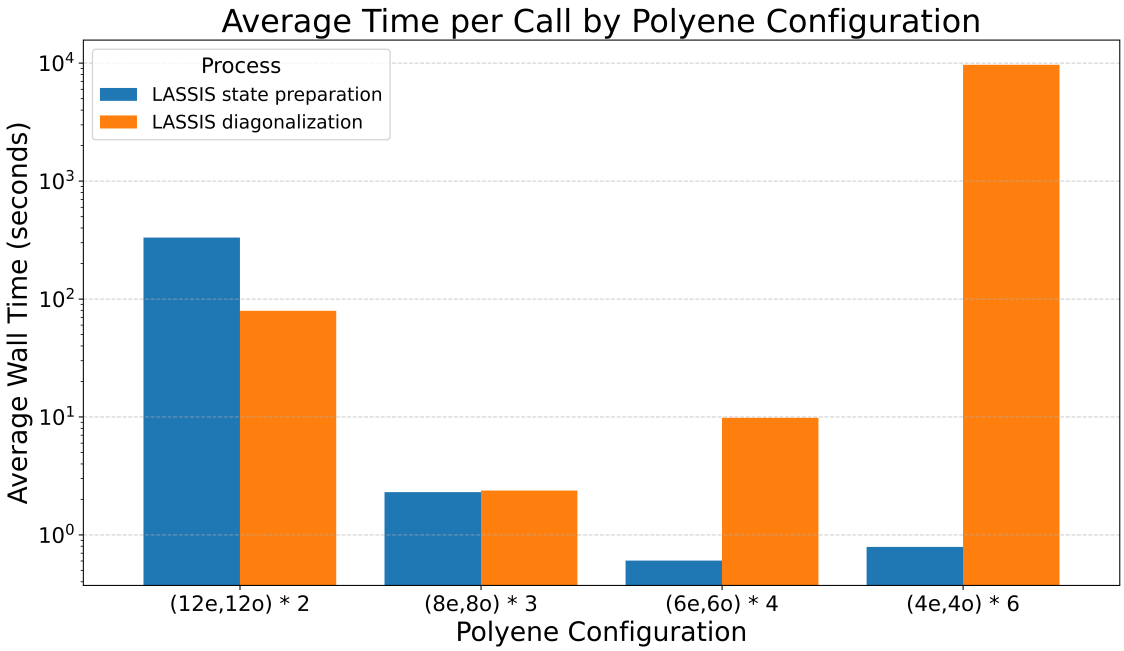


Figure 3.11: Approximate average cost per iteration during various parts of the LASSIS algorithm run on a CPU with Polaris resources, described in the Computational Resources Section.

sub-problems have different sizes and converge at different rates, and this level of detail is challenging to represent faithfully in a reasonably sized figure. Second, the cost of the optimization of product-state CI may depend on the quality of the initial guess and ease of convergence. Both of these factors combined may lead to total costs that may not conform to heuristics.

In Fig. 3.11, we present a rough average of time per iteration evaluated as the total time in the model space preparation phase divided by the total number of iterations for preparation of charge-transfer fragment statelets, lumping overheads with the cost. By comparing Figs. 3.10 and 3.11, we can conclude that the controlling aspect of the computational cost in both the few-fragment and many-fragment limits is the per-cycle cost of the individual steps within the iterative loop, rather than increasing difficulty of convergence of the iterative algorithm. We therefore direct our attention to optimizing the implementation of these individual iterative steps.

3.5 Acceleration of bottlenecks

For model states **j** and **i** represented as

$$|\mathbf{j}\rangle = |...klmno...\rangle \tag{3.13}$$

$$|\mathbf{i}\rangle = \left|...\bar{k}\bar{l}\bar{m}\bar{n}\bar{o}...\right\rangle \tag{3.14}$$

where k, l, m, n, o and $\bar{k}, \bar{l}, \bar{m}, \bar{n}, \bar{o}$ are ket and bra states of fragments K, L, M, N, O, respectively, the model-space Hamiltonian matrix element is

$$\begin{aligned}
\langle \mathbf{i}| \hat{H} |\mathbf{j}\rangle = &\sum_K H_k^{\bar{k}} \prod_L S_l^{\bar{l}} \\
&+ \sum_{K,L} H_{kl}^{\bar{k}\bar{l}} \prod_M S_m^{\bar{m}} \\
&+ \sum_{K,L,M} H_{klm}^{\bar{k}\bar{l}\bar{m}} \prod_N S_n^{\bar{n}} \\
&+ \sum_{K,L,M,N} H_{klmn}^{\bar{k}\bar{l}\bar{m}\bar{n}} \prod_O S_o^{\bar{o}}
\end{aligned} \tag{3.15}$$

where $S_k^{\bar{k}}$ is an overlap matrix $\langle \bar{k}|k\rangle$, and it must be understood that all fragment indices within a single term are unique ($K \neq L \neq \cdots$). This can be understood as the "operator part" (quantities labeled H) multiplied by the "overlap part" (product of overlap matrices, S). The "operator part" intermediates, in turn, are obtained by contracting one- and two-electron integrals with transition density matrices (TDMs):

$$H_k^{\bar{k}} = h_p T_p^{k\bar{k}} \tag{3.16}$$

$$H_{kl}^{\bar{k}\bar{l}} = h_{pq} T_p^{k\bar{k}} T_q^{l\bar{l}} \tag{3.17}$$

$$H_{klm}^{\bar{k}\bar{l}\bar{m}} = h_{pqr} T_p^{k\bar{k}} T_q^{l\bar{l}} T_r^{m\bar{m}} \tag{3.18}$$

$$H_{klmn}^{\bar{k}\bar{l}\bar{m}\bar{n}} = h_{pqrs} T_p^{k\bar{k}} T_q^{l\bar{l}} T_r^{m\bar{m}} T_s^{n\bar{n}} \tag{3.19}$$

where T represents TDMs; h_p, h_{pq}, etc. represent various one- and two-electron integrals; and the indices p, q, r, s refer to one or more molecular orbital (MO) indices

represented collectively.

When diagonalizing non-iteratively, all possible matrix elements $\langle \mathbf{i}|\hat{H}|\mathbf{j}\rangle$ are first computed and stored in a dense array, which is passed to LAPACK diagonalization functions. On the other hand, when diagonalizing iteratively, the Hamiltonian-vector product is implemented as a function:

$$\vec{r}_I = \hat{H}\,|\mathbf{j}\rangle\, C_{\mathbf{j}I} \tag{3.20}$$

where $\vec{r}_I$ is a response vector. For any one term in Eqs. (3.16)–(3.19), at most 4 fragments are relevant. Therefore, the compound index $\mathbf{j}$ is, in practice, reorganized as up to five indices z, k, l, m, n, where z is the combined state index for all the "spectator" fragments [labeled L, M, N, O, respectively, in the four terms of Eq. (3.15)]. Thus, the contribution to the response vector from the (for instance) 2-fragment interaction is written as

$$\langle \mathbf{i}|\vec{r}_I\rangle \leftarrow H_{kl}^{\bar{k}\bar{l}} \left(\prod_{M\in Z} S_m^{\bar{m}} \right) C_{zklI} \tag{3.21}$$

Note that neither the SI vector (C) nor the response vector ($\vec{r}$) can be stored in memory as a single contiguous rectangular array because, for instance, states with different total numbers of electrons are excluded. We return to the implications of this in a later section.

3.5.1 Calculation of Transition Density Matrices

For the calculations in Sec. 3.4 which are dominated by the "LASSIS state preparation" and "Hamiltonian construction" steps, the computational hotspot is the preparation of TDMs. This is because, in the "LASSIS state preparation" step, the most costly type of fragment statelet to optimize is the charge-transfer type, which as Ref. 61 described is obtained by a self-consistent loop over minor LASSI sub-problems in which matrices with elements like Eq. (3.15) are repeatedly built and diagonalized.

Because two statelets belonging to the same fragment may differ in number of electrons, spin, or both, LASSIS requires a variety of standard and non-standard TDMs:

$$T^{k\bar{k}}_{r,} = \langle \bar{k} | \, \hat{c}^\dagger_r \, | k \rangle \tag{3.22}$$

$$T^{k\bar{k}}_{pr,q} = \langle \bar{k} | \, \hat{c}^\dagger_r \hat{c}^\dagger_p \hat{c}_q \, | k \rangle \tag{3.23}$$

$$T^{k\bar{k}}_{\beta,\alpha} = \langle \bar{k} | \, \hat{c}^\dagger_\beta \hat{c}_\alpha \, | k \rangle \tag{3.24}$$

$$T^{k\bar{k}}_{\alpha,\beta} = \langle \bar{k} | \, \hat{c}^\dagger_\alpha \hat{c}_\beta \, | k \rangle \tag{3.25}$$

$$T^{k\bar{k}}_{pq,} = \langle \bar{k} | \, \hat{c}^\dagger_p \hat{c}^\dagger_q \, | k \rangle \tag{3.26}$$

$$T^{k\bar{k}}_{,pq} = \langle \bar{k} | \, \hat{c}_q \hat{c}_p \, | k \rangle \tag{3.27}$$

$$T^{k\bar{k}}_{p,q} = \langle \bar{k} | \, \hat{c}^\dagger_p \hat{c}_q \, | k \rangle \tag{3.28}$$

$$T^{k\bar{k}}_{pq,rs} = \langle \bar{k} | \, \hat{c}^\dagger_p \hat{c}^\dagger_q \hat{c}_s \hat{c}_r \, | k \rangle \tag{3.29}$$

together with their complex conjugates. Here, $\hat{c}^\dagger_p$ ($\hat{c}_p$) creates (annihilates) an electron in orbital p, α and β denote spin-up and spin-down spin orbitals, and p, q, r, s

represent general spin-orbital indices.

PySCF offers calculation of one-particle [Eq. (3.28)] and two-particle [Eq. (3.29)] TDMs. For such calculations, we need the CI vectors of both bra and ket, and the link index. A link index is a 3D array of size $[n_{\mathrm{det}}, n_{\mathrm{link}}, 4]$. This contains, for all possible determinants (n_{det}), for all possible single excitations (n_{link}), and the corresponding excitation definition, final address, and sign (i, a, str, sgn). Note that this is a much smaller quantity in size compared to a CI vector.

For calculations of other TDMs [Eqs. (3.22)–(3.27)], we can reuse the same code with some spoofing. Since the CI vectors on bra and ket sides would be imbalanced due to electrons leaving or entering, we pad these CI vectors with empty or occupied orbitals so that the bra and ket states exist in the same enlarged Hilbert space. For example, Eq. (3.22) can be represented as

$$\left\langle \bar{k} \right| \hat{c}_r^\dagger \left| k \right\rangle \rightarrow \left\langle \bar{k}^{(x)} \right| \hat{c}_r^\dagger \hat{c}_x \left| k_{(x)} \right\rangle , \tag{3.30}$$

where the "dummy" orbital x is always occupied in $\left|k_{(x)}\right\rangle$ and always empty in $\left|\bar{k}^{(x)}\right\rangle$. In the format in which CI vectors are stored in PySCF, this is essentially a matter of padding the CI vectors of $|k\rangle$ and $\left|\bar{k}\right\rangle$ with zeros in the appropriate rows and columns. The TDMs thus calculated on the right-hand side of Eq. (3.30) likewise contain many zeros. For each configuration of dummy orbitals in evaluating the various non-standard density matrices in Eqs. (3.22)–(3.27), a separate link index intermediate must also be constructed.

The standard TDMs [i.e., Eqs. (3.28) and (3.29)] are calculated as follows:

$$\langle \bar{k}|\hat{c}_p^\dagger \hat{c}_q|k\rangle = \langle \bar{k}|\xi_{pq}^{(1)}\rangle \tag{3.31}$$

$$\begin{aligned}\langle \bar{k}|\hat{c}_p^\dagger \hat{c}_r^\dagger \hat{c}_s \hat{c}_q|k\rangle =& \langle \xi_{pq}^{(2)}|\xi_{rs}^{(2)}\rangle \\ &- \frac{1}{2}(\langle \bar{k}|\hat{c}_p^\dagger \hat{c}_s|k\rangle + \langle \bar{k}|\hat{c}_q^\dagger \hat{c}_r|k\rangle)\end{aligned} \tag{3.32}$$

with vector intermediates

$$\left|\xi_{pq}^{(1)}\right\rangle = \hat{c}_p^\dagger \hat{c}_q\,|k\rangle \tag{3.33}$$

$$\left|\xi_{pq}^{(2)}\right\rangle = \left\langle \bar{k}\right|\hat{c}_p^\dagger \hat{c}_q \tag{3.34}$$

which are arrays with indices α, β ranging over spin-up and spin-down determinants as well as p, q orbital indices. The full size of the intermediate is $N_\alpha N_\beta N_{\mathrm{ac}_f}^2$. This can be made manageable with chunks over α and β determinants.

In the CPU version of the code, each CPU thread does the whole process [i.e., Eqs. (3.33), (3.34), and (3.22)–(3.29)]. Each thread allocates two buffers of size $b_{max} N_{\mathrm{ac}_f}^2$, where b_{max} is some predetermined maximum value set by the PySCF authors. This iterates over all α determinants in single increments, and all β determinants in block increments. In our previous work,[100] we allocated buffers of size $N_{\mathrm{blk}} N_{\mathrm{ao}}^2$. We block over alpha determinants based on the largest integer smaller than $N_{\mathrm{blk}} N_{\mathrm{ao}}^2 / N_\beta N_{\mathrm{ac}_f}^2$.

For Eqs. (3.28) and (3.29), we have same-sized CI vectors. The active space, and hence the CI vectors, are usually small in size in fragmentation methods. A (16e, 16o) active space would have a CI vector of size 1.3 GB in double precision, compared

to the GPU memory of 40 GB in an A100. In our tests, generation and transfer of the link index from CPU to GPU is not a leading cost and has not been implemented on GPUs. The resulting TDMs, in comparison to CI vectors, are negligible in memory cost. For calculation of the non-standard TDMs in Eqs. (3.22)–(3.27), we carry out the padding with zero rows and columns on the GPU; "artificially" larger CI vectors such as those describing $\left|\bar{k}^{(x)}\right\rangle$ and $\left|k_{(x)}\right\rangle$ are not transmitted explicitly.

The calculation of TDMs is farmed out to multiple GPUs, with each GPU being responsible for one TDM for one pair of fragment statelet Hilbert spaces at a time. Further optimization using broadcasting through GPU-GPU connections can be pursued in the future, but data transfer does not constitute a bottleneck currently.

3.5.2 Calculation of Matrix-Vector Products

For the calculations in Sec. 3.4 which are dominated by the "Davidson iterations" or "LASSIS diagonalization" steps, the computational hotspot is the evaluation of the Hamiltonian-vector product, $\vec{r}_I$ from Eq. (3.20).

Unfortunately, this operation is not as straightforwardly vectorizable as it may appear from Eq. (3.21), because some fragment statelets involve different numbers of electrons or local spin magnitudes, but only model states which can be coupled to the reference wave function by powers of the Hamiltonian should be included in the model space — so model states with different total numbers of electrons, for instance, are dropped. To accomplish this, we must consider each fragment statelet index k to actually consist of two sub-indices:

- q_K, consisting of various quantum numbers such as the total number of elec-

trons,

- and $n_K \geq 0$, an integer indexing the number of statelets in the given Hilbert space for fragment K.

Let $\vec{q} = \{\ldots, q_K, q_L, \ldots\}$ and $\vec{n} = \{\ldots, n_K, n_L, \ldots\}$ be the vectors of these indices over fragments, so that a single model state $\mathbf{j}$ can also be expressed as $\vec{q}\vec{n}$. A single value for $\vec{q}$ is a single rootspace, and the quantities on the right-hand side of Eq. (3.21) are actually stored in memory as

$$H_{kl}^{\bar{k}\bar{l}} \to H_{\bar{n}_K \bar{n}_L n_K n_L}^{\bar{q}_K \bar{q}_L q_K q_L}, \tag{3.35}$$

$$S_m^{\bar{m}} \to S_{\bar{n}_M n_M}^{\bar{q}_M q_M}, \tag{3.36}$$

$$C_{zklI} \to C_{n_Z n_K n_L}^{q_Z q_K q_L}, \tag{3.37}$$

and while these arrays are dense in the n subindices, they are ragged in the q subindices. Therefore, in practice, each set of q bra and ket indices is a separate rectangular array, and only the contractions over the n subindices are vectorized. This creates a major challenge for efficient GPU acceleration.

The evaluation of Eq. (3.20) is carried out by multiplying the SI vector first by the overlap matrix factors (S) and then by the Hamiltonian element factors (H); i.e., from right to left as written in Eq. (3.15). It is the final term, the contraction with the "H" intermediates, which is the main computational bottleneck, and so this dot product has been offloaded to GPUs. In the case of the 4-fragment term, the 8-index intermediate $H_{klmn}^{\bar{k}\bar{l}\bar{m}\bar{n}}$ is too large in practice to be stored in memory, so for that term

specifically we instead store the 6-index intermediate

$$H_{rs}^{k\bar{k},l\bar{l}} \equiv h_{pqrs} T_p^{k\bar{k}} T_q^{l\bar{l}}, \tag{3.38}$$

and evaluate the matrix-vector product in three steps:

$$\xi_{rs}^{\bar{z}k\bar{l}mnI} \leftarrow H_{rs}^{k\bar{k},l\bar{l}} (SC)_{\bar{z}klmnI}, \tag{3.39}$$

$$\xi_{s}^{\bar{z}k\bar{l}\bar{m}nI} \leftarrow \xi_{rs}^{\bar{z}k\bar{l}mnI} T_r^{m\bar{m}}, \tag{3.40}$$

$$\langle \mathbf{i} | \vec{r}_I \rangle \leftarrow \xi_{s}^{\bar{z}k\bar{l}\bar{m}nI} T_s^{n\bar{n}}, \tag{3.41}$$

where (SC) is the SI vector after multiplication by spectator fragment overlap matrices. Although the ξ intermediates have formally worse scaling than $H_{klmn}^{k\bar{l}\bar{m}\bar{n}}$ in the limit of many fragments due to the compound index $\bar{z}$, in this work, in practice, since we consider fragment sizes not much larger than 4, this reorganization saves substantial memory resources. This is currently being done with a single GPU.

3.5.3 Performance

As discussed in Section 3.4, the computational bottleneck for a given molecule can change based on the setup of a calculation. Timings for GPU-accelerated LASSIS calculations of the 12-oligomer of acetylene are presented in Fig. 3.12, to be compared to the timings reported in Figs. 3.10 and 3.11. We note that the speedup is more pronounced with larger active spaces, and using GPUs may be counterproductive with small active spaces.

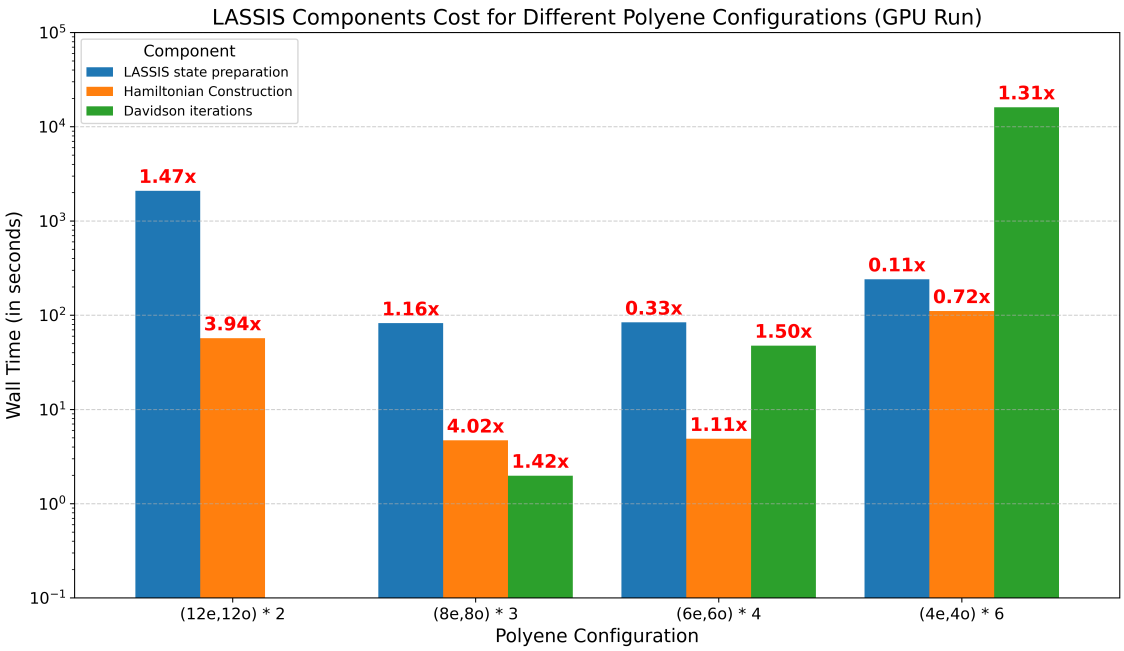


Figure 3.12: Cumulative cost of various parts of the LASSIS algorithm on a GPU on Polaris, described in the Computational Resources Section. The speedup ratio is highlighted on each relevant bar. Speedups are compared to Fig. 3.10.

More specifically, when the fragment active space is (12e, 12o) (8.5×10^5 elements in the CI vector) and (8e, 8o) (4900 elements in the CI vector), the state preparation and Hamiltonian construction are slightly improved with the use of GPUs, whereas it is counter-productive to use GPUs with much smaller active spaces of (6e, 6o) (400 elements in the CI vector) and (4e, 4o) (36 elements in the CI vector). The Davidson iterations, on the other hand, show very little speedup regardless of the active space partition.

Kremer's dimer is the tris-hydroxo-bridged chromium dimer represented in Fig. 3.13. For performing LASSIS on this system, we have used a similar setup as used in Ref. 61 with two active spaces of (15e, 11o) each, which is defined as all Cr 3d, all Cr 3p, all Cr 3s, and 2 ligand O 2p orbitals. Cr 3d electrons are kept in up spin in one fragment and down in the other, leading to the total active space in the reference LASSCF wave function of $((9\alpha, 6\beta), 11\text{o}) \wedge ((6\alpha, 9\beta), 11\text{o})$. The basis set is def2-SVP on C, H, O, and N atoms, and def2-TZVP on Cr atoms. Starting from LASSCF orbitals, for one state, the cost profile for LASSIS with and without GPU acceleration is given in Fig. 3.14. We see about a 2x overall speedup with GPU acceleration.

3.6 Conclusions

We have demonstrated an overall speedup of up to 10x for the LASSCF method using GPUs and up to 30x speedup in some fundamental operations. We were also able to run at least a full LASSCF cycle with a (96e,96o) active space on a polymer chain. We were able to converge iron-sulfur cluster with up to (22e,40o)

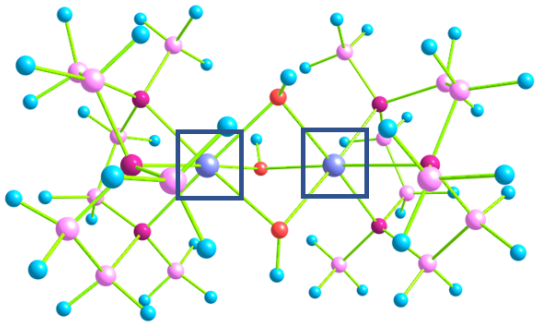


Figure 3.13: Structure of Kremer's Dimer. The purple boxed atoms are Cr, the red atoms between the two Cr atoms are O, the dark pink atoms are N, the light pink atoms are C, and the small light blue atoms are H. The two fragments selected for the calculation are the two Cr atoms.

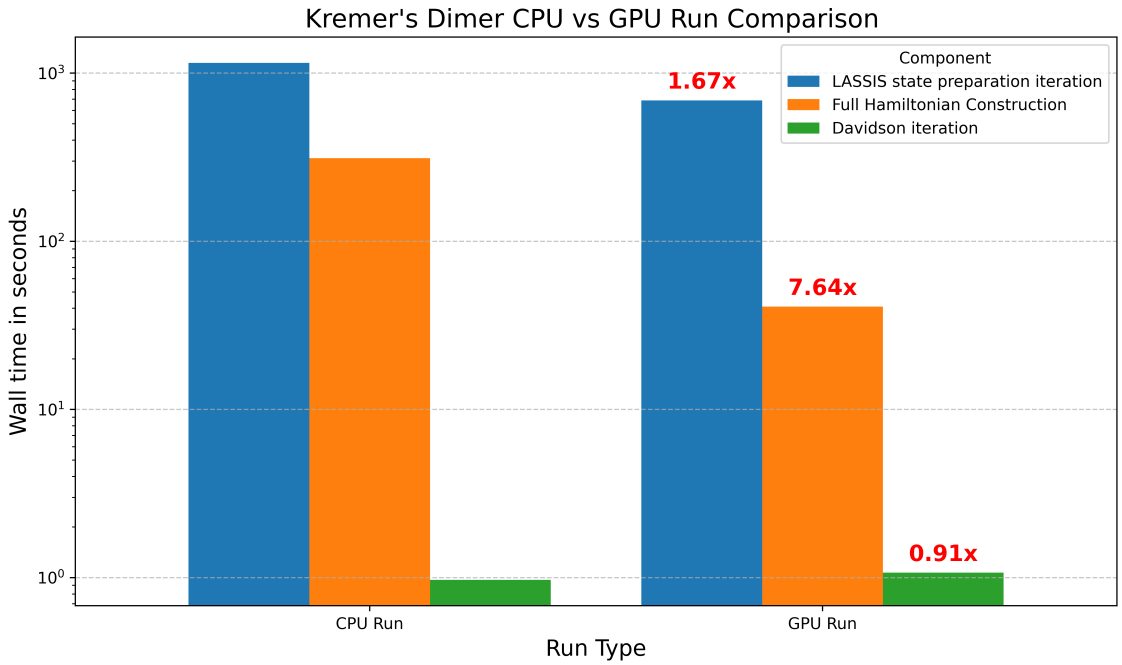


Figure 3.14: GPU-accelerated run for Kremer's Dimer for a single state on Polaris.

active space and 660 orbitals and also able to converge a MIL-127 MOF node with an (18e,30o) active space and about 1200 atomic orbitals. Focusing on speeding up low-level computational bottlenecks has yielded up to 20x speedup of CASSCF algorithm without additional effort. We have used density fitting algorithms to keep memory cost low and followed a code development philosophy to more easily explore GPU algorithms and retain some degree of portability. To that end, we demonstrate portable performance with NVIDIA A100 GPUs and Intel Max Series GPUs without further tuning (as well as on an AMD MI250 GPU not discussed here). Work continues in moving more bottlenecks to GPU-accelerated algorithms and improving the scaling efficiency of leveraging all GPUs on larger compute nodes.

Following a similar apporach, we have accelerated LASSIS using GPUs. We were able to accelerate TDMs about 7x for large active spaces of (15e, 11o) but performance decreases as the active space size decreases. On the other hand, the Hamiltonian diagonalization has not been accelerated appreciable, in part due to a highly irregular data access pattern and intense data transfer requirements. Further optimization of this kernel is high on the priority list. Overall several parts of the LASSIS workflow still run on the CPU. Due to this, the LASSIS has not seen acceleration comparable to LASSCF, but is still sufficient to enable production level workloads and enable new science, some of which we will discuss in the next chapter.

CHAPTER 4

LOW-LYING SPIN STATE ENERGIES OF IRON-SULFUR CLUSTERS WITH LOCALIZED ACTIVE SPACE STATE INTERACTION SINGLES AND PAIR DENSITY FUNCTIONAL THEORY

4.1 Introduction

Iron-sulfur clusters are ubiquitous in biology and play essential roles in electron transfer, respiration, photosynthesis, and catalyzing nitrogen fixation.[119] Among these, cubane-type [4Fe-4S] clusters are particularly important owing to their rich redox chemistry and ability to mediate charge-transfer processes.[120] Their remarkable reactivity is closely linked to the presence of multiple valence isomers and a dense manifold of low-lying electronic states arising from strong exchange interactions among partially occupied Fe $3d$ orbitals.[120] Because many of these states are thermally accessible under ambient conditions, they are thought to facilitate the diverse chemical functions of iron-sulfur proteins. The geometry of the clusters can alter the electronic structure, meaning slight changes to the geometry can provide access to different electronic states.[120] Understanding the low-energy electronic structure of $[Fe_4S_4(SMe)_4]^{2-}$ therefore remains a central challenge in bioinorganic chemistry.

The electronic structure of $[Fe_4S_4(SMe)_4]^{2-}$ is strongly multiconfigurational and requires a multireference description.[121] However, due to multiple partially filled

3d orbitals and the possible need for ligand orbitals,[2] the total size of the active space needed to model the cluster exceeds the conventional FCI limit of (18e, 18o).[122] Since the seminal work of Sharma and co-workers,[2] a variety of multireference approaches have been applied to iron-sulfur clusters, including density matrix renormalization group (DMRG),[2, 123] coupled-cluster valence bond theory (CCVB),[124] adaptive sampling configuration interaction (ASCI),[1] graphical unitary group approach full configuration interaction quantum Monte Carlo (GUGA-FCIQMC),[125, 126] and, more recently, auxiliary-field quantum Monte Carlo (AFQMC).[127, 128] In particular, Sharma *et al.*[2] used DMRG to report a dense manifold of low-energy states and argued that it could not be described by a simple spin Hamiltonian. In contrast, Dobrautz *et al.*[125] employed GUGA-FCIQMC and observed spin-ladder behavior in related all-ferric iron-sulfur systems ($[Fe_4S_4(SMe)_4]$), while Li Manni and co-workers investigated the coupling between local iron spin states in all-ferric clusters using the same method.[126] Mejuto-Zaera *et al.*[1] used ASCI with PT2 corrections and matched their results with DMRG to further demonstrate that the relative energies of the low-lying spin states are highly sensitive to structural details. These studies have significantly advanced the understanding of the ground-state electronic structure and magnetic interactions in iron-sulfur clusters. Nevertheless, a consistent description of the low-energy states of $[Fe_4S_4(SMe)_4]^{2-}$ remains elusive.

One possible reason is methodological. Most previous studies compute excited states individually. In systems with a dense manifold of nearly degenerate states, such a one-state-at-a-time approach does not guarantee a mutually orthogonal set of

eigenstates of the *same* effective Hamiltonian. As a result, different calculations may converge to distinct local minima, complicating the construction of a unified picture of the low-energy states. At the same time, active-space methods alone provide only a qualitative treatment of electron correlation, and quantitative predictions require the inclusion of correlation beyond the active space.[24]

The localized active space state-interaction singles (LASSIS)[100, 61] method provides a natural framework to address these challenges. By constructing and diagonalizing a common model-space Hamiltonian, LASSIS generates a manifold of mutually orthogonal eigenstates of the same Hamiltonian. Electron correlation outside the active space can then be incorporated efficiently through pair-density functional theory (PDFT),[24] avoiding the high computational cost associated with perturbative treatments while retaining quantitative accuracy.[129]

In this work, we apply LASSIS and PDFT to investigate the low-lying electronic states of $[Fe_4S_4(SMe)_4]^{2-}$. We show that the resulting low-lying energy levels organize into well-defined state manifolds that can be described by model spin Hamiltonians, presenting a coherent picture of the low-energy electronic structure that has not emerged clearly from previous multireference studies on $[Fe_4S_4(SMe)_4]^{2-}$. We further demonstrate that the choice of fragmentation has a significant effect on the nature of the excited states and that electron correlation outside the active space, included through PDFT, substantially increases the magnetic exchange coupling.

4.2 Results and Discussion

We considered two active spaces for $[Fe_4S_4(SMe)_4]^{2-}$. The smaller active space, (22e,20o), contains only the Fe $3d$ orbitals and was investigated using both two-fragment and four-fragment decompositions. In the two-fragment description, each fragment corresponds to a ferromagnetically coupled Fe^{3+}–Fe^{2+} pair (Fig. 4.1a). The corresponding active space is

$$((10\alpha, 1\beta), 10o) \wedge ((1\alpha, 10\beta), 10o)$$

where the electron occupations reflect the high-spin coupling within each Fe–Fe dimer. In the four-fragment description, each Fe center is treated as an independent fragment (Fig. 4.1b), giving

$$((5\alpha, 1\beta), 5o) \wedge ((5\alpha, 0\beta), 5o) \wedge ((1\alpha, 5\beta), 5o) \wedge ((0\alpha, 5\beta), 5o)$$

We also considered a larger (54e,36o) active space that includes sulfur orbitals commonly associated with double-exchange interactions.[2, 1] In this case, each fragment consists of a ferromagnetically coupled Fe^{3+}–Fe^{2+} pair together with the four sulfur atoms directly involved in the magnetic coupling pathway, resulting in the active space

$$((18\alpha, 9\beta), 18o) \wedge ((9\alpha, 18\beta), 18o)$$

Attempts to partition the (54e,36o) active space into four fragments led to numerical convergence difficulties in the state preparation step, and only the two-fragment

decomposition was considered for this larger active space.

For all calculations, high-spin ($2S + 1 = 19$) restricted open-shell Hartree-Fock (ROHF) orbitals were used as the starting orbitals. Localized active-space orbitals were generated using the atomic valence active space (AVAS) procedure,[130] followed by manual selection of the orbitals included in the final active space. For the (22e,20o) active space, a LAS self-consistent field (LASSCF) calculation[34, 100] with (22e,40o) active space including the Fe $4d$ shell with four fragments was performed and resorted to generate the stable $3d$ orbitals. Performing a LASSCF with (22e,20o) only leads to swapping out of doubly-occupied $3d$ orbitals. For the (54e,36o) active space we similarly performed ROHF, AVAS, manual selection of relevant orbitals and localization (meta-Löwdin[131]). We then performed a LAS configuration interaction on these localized orbitals and the resulting natural orbitals were used for LASSIS. Performing (22e,40o) LASSIS with two fragments led to very high memory requirements of CI calculation, and numerical convergence issues for the four fragment case in the Hamiltonian diagonalization step.

To examine the effect of geometry on the low-energy electronic structure, we considered two previously reported structures of $[Fe_4S_4(SMe)_4]^{2-}$. The key difference between these geometries lies in the relative distances between the ferromagnetically coupled Fe–Fe pairs and the antiferromagnetically coupled Fe–Fe pairs. Geometry **A** was taken from Mejuto-Zaera et. al[1] (geometry 2A in Ref. 1), where the structure was optimized using the TPSSh density functional for the high-spin ($2S + 1 = 19$) state with the aug-cc-pVDZ basis set on Fe and S atoms and the cc-pVDZ basis set on C and H atoms.[1] Geometry **B** was taken from Sharma et. al[2] . In that

study, the structure was optimized using the BP86 density functional with a TZV basis set for the high-spin ($2S+1=21$) state of the all-ferric $[Fe_4S_4(SMe)_4]$ cluster and subsequently used for calculations on $[Fe_4S_4(SMe)_4]^{2-}$. For consistency with Ref. 1, all calculations reported here employed the aug-cc-pVDZ basis set for Fe and S atoms and the cc-pVDZ basis set for C and H atoms.

The Fe–Fe distances for the two geometries are reported in Table 4.1. Although both structures exhibit the same magnetic coupling pattern, they differ significantly in the relative distances between the ferromagnetically and antiferromagnetically coupled Fe pairs.

For geometry **A**, the ferromagnetically coupled Fe pairs are shorter than the antiferromagnetically coupled pairs,

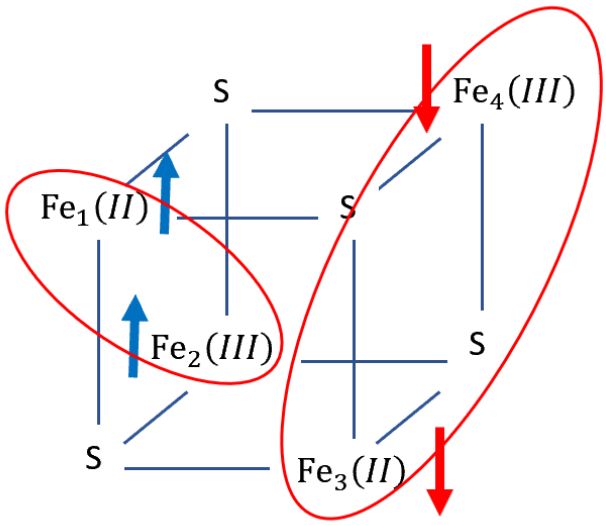
$$r_{\mathrm{Fe_1Fe_2}} \approx r_{\mathrm{Fe_3Fe_4}} < r_{\mathrm{Fe_1Fe_3}} \approx r_{\mathrm{Fe_1Fe_4}} \approx r_{\mathrm{Fe_2Fe_3}} \approx r_{\mathrm{Fe_2Fe_4}} \tag{4.1}$$

where $r_{\mathrm{Fe}_i\mathrm{Fe}_j}$ denotes the distance between Fe_i and Fe_j.

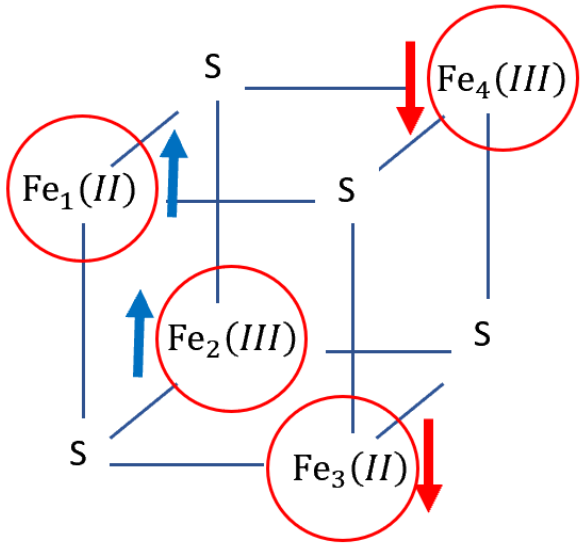
In contrast, geometry **B** exhibits the opposite trend,

$$r_{\mathrm{Fe_1Fe_2}} \approx r_{\mathrm{Fe_3Fe_4}} < r_{\mathrm{Fe_1Fe_3}} \approx r_{\mathrm{Fe_1Fe_4}} \approx r_{\mathrm{Fe_2Fe_3}} \approx r_{\mathrm{Fe_2Fe_4}}, \tag{4.2}$$

with the antiferromagnetically coupled Fe pairs now being shorter than the ferromagnetically coupled pairs.



(a) $[Fe_4S_4(SMe)_4]^{2-}$ with 2 fragments



(b) $[Fe_4S_4(SMe)_4]^{2-}$ with 4 fragments

Figure 4.1: Iron-sulfur clusters with different fragmentation schemes and spin arrangements. The blue arrows are up-spin on the Fe atom, the red arrows are down-spin on Fe. Each circle/oval denotes a fragment.

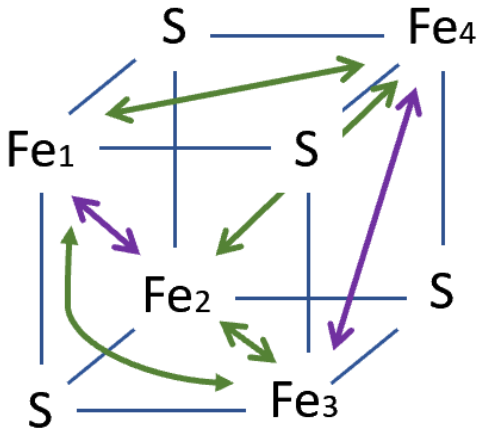


Figure 4.2: [4Fe-4S] schematic showing two different sets of Fe-Fe distances. Both purple distances are similar, and all four olive green distances are similar.

	Fe-Fe distances	
Pair	Geometry **A**	Geometry **B**
$r_{Fe_1Fe_3}$	3.09	2.75
$r_{Fe_1Fe_4}$	3.08	2.75
$r_{Fe_1Fe_2}$	2.77	2.85
$r_{Fe_2Fe_3}$	3.09	2.75
$r_{Fe_2Fe_4}$	3.09	2.75
$r_{Fe_3Fe_4}$	2.77	2.85

Table 4.1: Fe-Fe distances (in Å) in geometries **A** and **B**. See Fig. 4.2 for corresponding atom pairs in the cluster.

	LAS methods						Mejuto-Zaera et. al[1]	
	22e,20o 2frag		22e,20o 4frag		54e,36o 2frag		54e,36o	
2S+1	LASSIS	PDFT	LASSIS	PDFT	LASSIS	PDFT	ASCI+PT2$^{\alpha}$	DMRG$^{\beta}$
1	0	0	0	0	0	0	0±153.6	0
3	19	42	18	44	18	45	0±175.6	2
5	56	127	53	131	55	133	22±219.5	32
7	112	251	106	258	110	265	-	-
9	187	417	178	428	184	442	-	-
11	281	626	270	639	278	665	-	-
13	395	877	382	890	391	934	-	-
15	528	1173	515	1182	525	1252	-	-
17	680	1513	671	1514	680	1620	-	-
19	854	1898	850	1888	858	2040	1207$^{\gamma}$	-

Table 4.2: Energy (in cm^{-1}) of lowest 10 spin states with geometry **A** using various methods. α, β: Result from Mejuto-Zaera et. al[1] . α: Each spin state has orbitals optimized with CASSCF for the given spin with SP-ASCI as the FCI solver. β: Each spin state had orbitals optimized with CASSCF with DMRG as spin solver. γ: Orbitals are the same as S=0 state.

.	LAS methods						CASSCF		CASCI	CASSCF	CASCI
.	22e,20o 2frag		22e,20o 4frag		54e,36o 2frag		54e,36o			20e,20o$^{\eta}$	
2S+1	LASSIS	PDFT	LASSIS	PDFT	LASSIS	PDFT	ASCI+PT2$^{\alpha}$	DMRG$^{\beta}$	DMRG$^{\epsilon}$	GUGA - FCIQMC	
1	0	0	0	0	0	0	0 ± 154	0	0	0	0
3	63	119	62	123	60	124	331 ± 219	171	350	57	118
5	190	356	188	368	182	371	–346 ± 219	-990	-	172	348
7	380	709	377	729	366	740	-	-	-	344	681
9	636	1181	631	1213	615	1238	-	-	-	569	1097
11	957	1778	953	1823	931	1872	-	-	-	847	1585
13	1347	2503	1345	2560	1319	2649	-	-	-	1180	2130
15	1807	3363	1809	3428	1783	3581	-	-	-	1568	2719
17	2339	4365	2349	4432	2329	4681	-	-	-	2011	3331
19	2946	5517	2967	5576	2965	5966	3621$^{\gamma}$	4060$^{\delta}$	9750$^{\zeta}$	2506	3942
21$^{\theta}$	.	.	.	.	.	.	.	.	.	3053	4533

Table 4.3: Energy (in cm^{-1}) of lowest 10 spin states with geometry **B** using various methods. α, β: Results from Mejuto-Zaera et. al[1] using ASCI-PT2 and DMRG method. α: Orbitals for each spin state optimized separately with CASSCF using SP-ASCI as the FCI solver. β: Orbitals for each spin state optimized separately with CASSCF using DMRG as the FCI solver. γ, δ: Orbitals the same as S=0 state of the corresponding state (SP-ASCI and DMRG respectively). ϵ: Results from Sharma et. al[2] . Orbitals are visually selected from a localized basis. ζ: The number is visually derived from a graph. η: This is *not* $[4Fe\text{-}4S]^{+2}$ but $[Fe_4S_4(SMe)_4]$. Included as the only evidence of energy of the low lying states following a model Hamiltonian. θ: $2S + 1 = 21$ is only possible with all-ferric cluster.

For each active space, we first discuss the lowest-energy states for each spin multiplicity. The corresponding LASSIS energies are reported in Table 4.2 for geometry **A** and Table 4.3 for geometry **B**. Across both geometries, both active spaces ((22e,20o) and (54e,36o)) and both fragmentation schemes (two-fragment vs. four-fragment), the resulting spin-state energy ladders are consistent with a picture of two weakly interacting $S = 9/2$ Fe–Fe dimers, *via* exchange interactions. Inspection of the state interaction coefficients of the LASSIS wave functions confirms this interpretation.

The two fragmentation schemes provide different descriptions of the ferromagnetically coupled Fe pairs. In the four-fragment model, electron correlation between Fe1–Fe2 and Fe3–Fe4 is recovered through explicit charge-transfer and spin-flip configurations between individual Fe centers, as discussed in Sec. 2.3. In the two-fragment model, the same correlation is incorporated by treating each Fe–Fe pair as a single configuration-interaction problem. Despite these different descriptions, the two approaches yield very similar relative energies of the low-lying states (*vide infra*), indicating that the low-energy spin ladder is largely insensitive to the fragmentation scheme.

For geometry **A**, Table 4.2 reports the energies of the lowest states for all active spaces and fragmentations and compares them with the results of Mejuto-Zaera et. al[1] (that uses the same geometry). For the (22e,20o) active space, the two-fragment and four-fragment calculations yield nearly identical relative energies for the lowest states with spin $S = 0$ through $S = 9$. This agreement indicates that the low-energy spin ladder is largely insensitive to the choice of fragmentation, despite the different physical pictures underlying the two decompositions.

A similarly close agreement is observed upon enlarging the active space to (54e,36o). The relative energies obtained with the larger active space closely track those from the smaller (22e,20o) calculations. At first glance, the agreement between the two active spaces may appear surprising given that the (54e,36o) active space explicitly includes the sulfur orbitals commonly associated with double-exchange interactions. Nevertheless, our results are in good agreement with those reported by Mejuto-Zaera et. al[1] , who employed both ASCI+PT2 and DMRG calculations for different total spin states. In particular, the $S = 0 - S = 9$ energy gap obtained with LASSIS differs from the ASCI+PT2 values by only 20–30% across all active spaces considered here. This suggests that the essential physics governing the relative ordering of the low-lying spin states is already captured within the smaller active space.

To recover electron correlation outside the active space, we evaluated LASSIS-tPBE energies using the one- and two-body reduced density matrices obtained from the LASSIS wave functions. The resulting LASSIS-tPBE energies are reported in Tables 4.2 for geometry **A** and 4.3 for geometry **B**. Inclusion of PDFT systematically increases the separation between neighboring spin states by approximately a factor of two to two and a half for all active spaces considered. This effect is substantially larger than the differences between the LASSIS and ASCI+PT2 results, highlighting the importance of electron correlation outside the active space for quantitative predictions.

The results for geometry **B** are summarized in Table 4.3. Similar to geometry **A**, the three LASSIS models yield very similar relative energies for geometry **B**. However, the corresponding spin-state gaps are approximately three times larger

than those obtained for geometry **A**, and inclusion of PDFT increases the gaps by roughly a factor of two across all active spaces.

The sensitivity of the low-lying spin-state energies to the molecular geometry is consistent with the results of Mejuto-Zaera et. al[1] . In particular, the $S = 0$–$S = 9$ gap obtained with LASSIS differs from their ASCI+PT2 values by less than 30%. For the singlet–triplet and singlet–quintet gaps, our predictions are larger than those reported by Mejuto-Zaera *et al.* for geometry **A**, but smaller than the values reported by either Mejuto-Zaera *et al.* or Sharma *et al.* for geometry **B**.

We do not reproduce the quintet state below the singlet state reported in Ref. 1. One possible reason is that the orbitals in Ref. 1 were optimized separately for each spin state, whereas all states in the present work are derived from a common model-space Hamiltonian. Previous studies have shown that orbital optimization can significantly affect relative spin-state energies,[61] and, since $[Fe_4S_4(SMe)_4]^{2-}$ exhibits a high density of low-lying electronic states, different orbital optimizations may lead to qualitatively different state orderings.

To quantify the magnetic interactions, we fit the low-energy spin ladders to the Heisenberg–Dirac–van Vleck (HDvV) Hamiltonian,

$$(E_i)_{\mathrm{HDvV}} = -J(S_i)(S_i + 1) \tag{4.3}$$

A similar analysis was performed by Dobrautz *et al.*[125] for the all-ferric $[Fe_4S_4(SMe)_4]$ cluster at geometry **B** (Table 4.3) using GUGA-FCIQMC as the FCI solver for CASCI and CASSCF calculations where they show a good fit for a model Hamiltonian for the lowest *eleven* states.

Geometry	Geometry **A**		Geometry **B**	
Active Space	J_{LASSIS}	J_{PDFT}	J_{LASSIS}	J_{PDFT}
22e,20o 2frag	-9.45	-21.01	-32.47	-60.61
22e,20o 4frag	-9.30	-21.06	-32.59	-61.55
54e,36o 2frag	-9.44	-22.49	-32.32	-64.98

Table 4.4: Exchange coupling constant (J) (in cm^{-1}) for the two used geometries with various active spaces calculated over the ten lowest states with HDvV Hamiltonian. R^2, the coefficient of determination is calculated for all active spaces. $R^2 = 1 - \frac{\sum_i (E_i - E_{HDvV})^2}{\sum_i (E_i - E_{avg})^2}$ where E_i are the actual calculate energies, E_{HDvV} is the calculated energy from model Hamiltonian (eq. 4.3) and E_{avg} is the average of all actual energies. For all cases, $1 - R^2 < 10^{-3}$.

The resulting HDvV parameters are reported in Table 4.4. Across all active spaces, fragmentation schemes, and correlation treatments, the low-energy spin ladders are described well by these simple spin Hamiltonians since the coefficient of determination (R^2) is very close to unity (all model fits give $1 - R^2 < 10^{-3}$). This indicates that the relative energies of the low-lying states follow the spin-Hamiltonian behavior.

The extracted exchange couplings show a dependence on geometry. Geometry **B** yields $J \approx -32$ cm^{-1}, whereas geometry **A** yields $J \approx -10$ cm^{-1}. This trend can be rationalized by the Fe–Fe distances in the two structures. While the ferromagnetically coupled Fe pairs are slightly farther apart in geometry **B**, the antiferromagnetically coupled Fe pairs are substantially closer, leading to significantly stronger antiferromagnetic exchange interactions.

Inclusion of electron correlation outside the active space through PDFT approximately doubles the magnitude of J while preserving the overall structure of the spin ladder. The effect of PDFT on the exchange coupling is therefore considerably larger

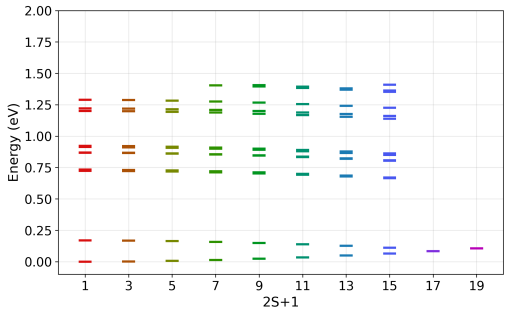
than the differences arising from either the choice of active space or the fragmentation scheme.

The same-spin excitation patterns for the (22e,20o) active space are shown in Fig. 4.3. The two-fragment calculations produce several distinct manifolds of excited states, whereas the four-fragment calculations yield a much denser set of states arising from the larger number of possible spin couplings among the four Fe centers. Despite these differences, the lowest state of each spin multiplicity appears at nearly the same energy in both fragmentation schemes. This observation further supports the picture of two weakly interacting Fe–Fe dimers governing the low-energy spin ladder. The corresponding results for the (54e,36o) active space and the PDFT calculations are presented in the Supporting Information.

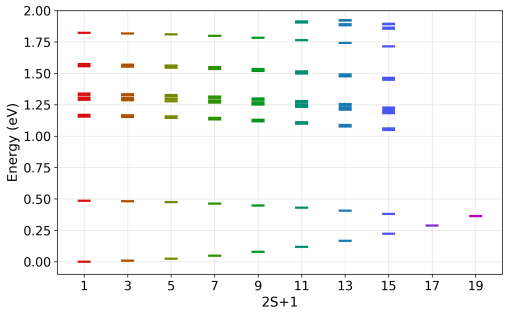
Because the two fragmentation schemes recover electron correlation in different ways, it is not possible to determine a priori which description is more appropriate. Nevertheless, the lowest state of each spin multiplicity occurs at nearly the same energy in both fragmentation schemes. This agreement suggests that the low-energy spin ladder can be understood in terms of two Fe–Fe dimers coupled through exchange interactions, while the higher density of states obtained in the four-fragment calculations primarily reflects a more explicit description of intra-dimer correlation.

4.3 Conclusions

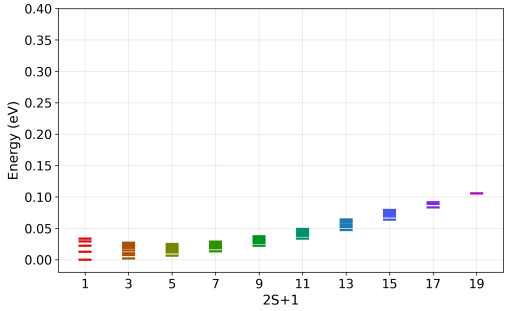
We have investigated the low-lying electronic states of the iron-sulfur cluster $[Fe_4S_4(SMe)_4]^{2-}$ using localized active space state interaction singles (LASSIS) and pair-density functional theory (PDFT). By constructing and diagonalizing a common model-space



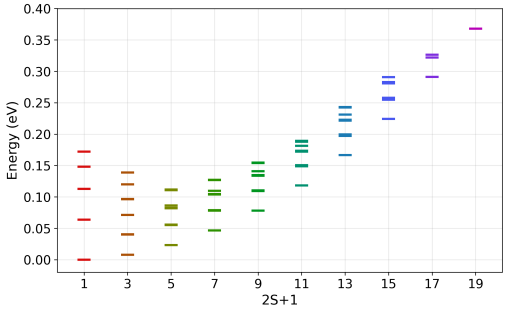
(a) Spin state energies with **A** with 2 fragments



(b) Spin state energies with **B** with 2 fragments



(c) Spin state energies with **A** with 4 fragments



(d) Spin state energies with **B** with 4 fragments

Figure 4.3: LASSIS spin state energies with (22e,20o) active space with two and four fragments with both geometries. Each bar represents a state. States of the same spin with very close energy appear on top of each other. For two-fragment cases, all states up to 2 eV are shown. For four-fragment cases, the lowest 8 states of each spin, or states with energy within 0.4 eV of the S=0 ground state, whichever is smaller, are shown. Adding PDFT in any of the LASSIS plots does not change the shape of the plot. PDFT results are presented in the SI.

Hamiltonian, LASSIS provides a unified description of low-lying states and avoids the inconsistencies that can arise when individual states are optimized independently.

Across all active spaces, geometries, and fragmentation schemes considered here, the relative energies of the lowest states with spins ranging from $S = 0$ to $S = 9$ exhibit a remarkably similar pattern. In particular, enlarging the active space from (22e,20o) to (54e,36o) and changing the fragmentation from two Fe–Fe dimers to four individual Fe centers has only a modest effect on the low-energy spin ladder. The resulting energies are well described by simple spin Hamiltonians, supporting a picture in which the low-energy electronic structure is governed primarily by interactions between two Fe–Fe dimers.

Inclusion of electron correlation outside the active space through PDFT produces substantial quantitative changes in the spin-state energetics and approximately doubles the magnetic exchange coupling.

CHAPTER 5

CONCLUSIONS

In this thesis, we have developed and applied GPU-accelerated implementations of localized active space methods to address the exponential scaling problem in multi-configurational quantum chemistry. We began by introducing the theoretical foundations of LASSCF and LASSI, demonstrating how fragment-based factorization enables the treatment of strongly correlated systems that would otherwise be inaccessible to conventional CASSCF.

Through systematic profiling of CPU-only implementations, we identified the primary computational bottlenecks: effective potential evaluations, orbital transformation computations for LASSCF and transition density matrices for LASSIS. By offloading these operations to GPUs, we achieved significant speedups, particularly for calculations with large fragment active spaces. For LASSCF with realistic workloads, we observed 5-10x speedup. This speedup is observed for a variety of computation architectures. For Kremer's dimer, we observed an overall 2x speedup with LASSIS.

Our application to the challenging $[Fe_4S_4(SMe)_4]^{2-}$ cluster demonstrated the practical utility of these methods. LASSIS provided a unified description of low-lying spin states, avoiding the inconsistencies that arise when individual states are optimized separately. Across two geometries, two active spaces, and two fragmentation schemes, the relative energies of states with spins $S = 0$ through $S = 9$ exhibited remarkably consistent patterns, highlighting the robustness of LASSIS. This is the first instance of all intermediate spin states being reported for this cluster. The

energies of all the spin states are all well described by simple Heisenberg spin Hamiltonians. Inclusion of dynamic correlation through PDFT approximately doubled the magnetic exchange coupling, highlighting the importance of correlation beyond the active space.

This work also provides a GPU-accelerated framework to perform multireference calculations for catalysis. In areas where several multireference calculations are needed, like benchmarking MC-PDFT functionals or active space selections, this work can accelerate data generation by an order of magnitude.

Looking forward, several directions merit further exploration. First, on the technical and programming side, the current GPU implementation can always be optimized further. The LASSIS acceleration efforts, while enabling highly complex calculations, still leave a lot of performance on the table. All calculations performed in this work only use one compute node. Use of MPI to go beyond more than one node could allow access to larger systems. Next, because the LAS framework relies on chemically intuitive choices for fragments and active spaces, it is primarily accessible to experts in multireference methods, with an even steeper learning curve for those unfamiliar with fragmentation techniques. The development of automated fragmentation procedures would make these methods more accessible to non-experts and enable systematic studies of many more chemically interesting systems. Finally, the framework could be extended to incorporate environmental effects, such as solvation or protein embedding, to model iron-sulfur clusters in their native biological environments.

The strategies developed here—identifying bottlenecks, targeting acceleration,

and managing data transfers—provide a general roadmap for GPU-accelerating other quantum chemistry methods. As computing architectures continue to evolve toward greater parallelism, the principles of efficient GPU utilization demonstrated in this work will remain relevant for the next generation of electronic structure codes.

CHAPTER 6

REFERENCES

REFERENCES

[1] Carlos Mejuto-Zaera, Demeter Tzeli, David Williams-Young, Norm M. Tubman, Mikuláš Matoušek, Jiri Brabec, Libor Veis, Sotiris S. Xantheas, and Wibe A. de Jong. The Effect of Geometry, Spin, and Orbital Optimization in Achieving Accurate, Correlated Results for Iron–Sulfur Cubanes. *J. Chem. Theory Comput.*, 18(2):687–702, February 2022.

[2] Sandeep Sharma, Kantharuban Sivalingam, Frank Neese, and Garnet Kin-Lic Chan. Low-energy spectrum of iron–sulfur clusters directly from many-particle quantum mechanics. *Nat. Chem.*, 6(10):927–933, 2014.

[3] Alexis T. Bell and Martin Head-Gordon. Quantum mechanical modeling of catalytic processes. *Annu. Rev. Chem. Biomol. Eng.*, 2(1):453–477, 2011. PMID: 22432627.

[4] Anna Krylov, Theresa L. Windus, Taylor Barnes, Eliseo Marin-Rimoldi, Jessica A. Nash, Benjamin Pritchard, Daniel G. A. Smith, Doaa Altarawy, Paul Saxe, Cecilia Clementi, T. Daniel Crawford, Robert J. Harrison, Shantenu Jha, Vijay S. Pande, and Teresa Head-Gordon. Perspective: Computational chemistry software and its advancement as illustrated through three grand challenge cases for molecular science. *J. Chem. Phys.*, 149(18):180901, 11 2018.

[5] Shenzhen Xu and Emily A. Carter. Theoretical insights into heterogeneous (photo)electrochemical co2 reduction. *Chem. Rev.*, 119(11):6631–6669, 2019. PMID: 30561988.

[6] Reinhard J Maurer, Christoph Freysoldt, Anthony M Reilly, Jan Gerit Brandenburg, Oliver T Hofmann, Torbjörn Björkman, Sébastien Lebègue, and Alexandre Tkatchenko. Advances in density-functional calculations for materials modeling. *Annu. Rev. Mater. Res.*, 49(1):1–30, 2019.

[7] Barbara Kirchner, Frank Wennmohs, Shengfa Ye, and Frank Neese. Theoretical bioinorganic chemistry: the electronic structure makes a difference. *Current opinion in chemical biology*, 11(2):134–141, 2007.

[8] Per EM Siegbahn and Margareta RA Blomberg. Transition-metal systems in biochemistry studied by high-accuracy quantum chemical methods. *Chem. Rev.*, 100(2):421–438, 2000.

[9] Per EM Siegbahn and Tomasz Borowski. Modeling enzymatic reactions involving transition metals. *Acc. Chem. Res.*, 39(10):729–738, 2006.

[10] Martin Karplus and J Andrew McCammon. Molecular dynamics simulations of biomolecules. *Nat. Struct. Bio.*, 9(9):646–652, 2002.

[11] Top 500 supercomputers - june 2024. Accessed: 2024-06-17.

[12] Attila Szabo and Neil S Ostlund. *Modern quantum chemistry: introduction to advanced electronic structure theory*. 1996.

[13] Pierre Hohenberg and Walter Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136(3B):B864, 1964.

[14] Walter Kohn and Lu Jeu Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140(4A):A1133, 1965.

[15] J. Čížek. On the correlation problem in atomic and molecular systems. calculation of wavefunction components in ursell-type expansion using quantum-field theoretical methods. *J. Chem. Phys.*, 45:4256–4266, 1966.

[16] J. Čížek. On the use of the cluster expansion and the technique of diagrams in calculations of correlation effects in atoms and molecules. *Adv. Chem. Phys*, 14:35–89, 1969.

[17] J Čížek and J Paldus. Correlation problems in atomic and molecular systems iii. rederivation of the coupled-pair many-electron theory using the traditional quantum chemical methods. *Int. J. Quantum Chem.*, 5(4):359–379, 1971.

[18] Björn O Roos, Peter R Taylor, and Per EM Siegbahn. A complete active space scf method (casscf) using a density matrix formulated super-ci approach. *Chem. Phys.*, 48(2):157, 1980.

[19] Konstantinos D Vogiatzis, Dongxia Ma, Jeppe Olsen, Laura Gagliardi, and Wibe A De Jong. Pushing configuration-interaction to the limit: Towards massively parallel mcscf calculations. *J. Chem. Phys.*, 147(18):184111, 2017.

[20] Kerstin. Andersson, Per Aake. Malmqvist, Bjoern O. Roos, Andrzej J. Sadlej, and Krzysztof. Wolinski. Second-order perturbation theory with a CASSCF reference function. *J. Phys. Chem.*, 94(14):5483–5488, July 1990. Publisher: American Chemical Society.

[21] Kerstin Andersson, Per-Åke Malmqvist, and Björn O. Roos. Second-order

perturbation theory with a complete active space self-consistent field reference function. *J. Chem. Phys.*, 96(2):1218–1226, January 1992.

[22] C. Angeli, R. Cimiraglia, S. Evangelisti, T. Leininger, and J.-P. Malrieu. Introduction of n-electron valence states for multireference perturbation theory. *J. Chem. Phys.*, 114(23):10252–10264, June 2001.

[23] Celestino Angeli, Renzo Cimiraglia, and Jean-Paul Malrieu. n-electron valence state perturbation theory: A spinless formulation and an efficient implementation of the strongly contracted and of the partially contracted variants. *J. Chem. Phys.*, 117(20):9138–9153, November 2002.

[24] Giovanni Li Manni, Rebecca K. Carlson, Sijie Luo, Dongxia Ma, Jeppe Olsen, Donald G. Truhlar, and Laura Gagliardi. Multiconfiguration Pair-Density Functional Theory. *J. Chem. Theory Comput.*, 10(9):3669–3680, September 2014. Publisher: American Chemical Society.

[25] Núria Queralt, David Taratiel, Coen De Graaf, Rosa Caballol, Renzo Cimiraglia, and Celestino Angeli. On the applicability of multireference second-order perturbation theory to study weak magnetic coupling in molecular complexes. *J. Comput. Chem.*, 29(6):994–1003, 2008.

[26] Daniel Aravena and Eliseo Ruiz. Spin dynamics in single-molecule magnets and molecular qubits. *Dalton Trans.*, 49(29):9916–9928, 2020.

[27] Carmen J Calzado, Celestino Angeli, David Taratiel, Rosa Caballol, and Jean-Paul Malrieu. Analysis of the magnetic coupling in binuclear systems. iii. the

role of the ligand to metal charge transfer excitations revisited. *J. Chem. Phys.*, 131(4), 2009.

[28] Lluís Blancafort and Alexander A Voityuk. Exciton delocalization, charge transfer, and electronic coupling for singlet excitation energy transfer between stacked nucleobases in dna: An ms-caspt2 study. *J. Chem. Phys.*, 140(9), 2014.

[29] Antonio Francés-Monerris, Javier Segarra-Martí, Manuela Merchán, and Daniel Roca-Sanjuán. Complete-active-space second-order perturbation theory (caspt2//casscf) study of the dissociative electron attachment in canonical dna nucleobases caused by low-energy electrons (0-3 ev). *J. Chem. Phys.*, 143 (21), 2015.

[30] Eduarda S Gil, Claudia B da Silva, Pablo A Nogara, Carolina H da Silveira, Joao BT da Rocha, Bernardo A Iglesias, Diogo S Lüdtke, Paulo FB Goncalves, and Fabiano S Rodembusch. Synthesis, photophysical characterization, casscf/caspt2 calculations and ct-dna interaction study of amino and azido benzazole analogues. *J. Mol. Liq.*, 297:111938, 2020.

[31] Soumen Ghosh, Andrew L Sonnenberger, Chad E Hoyer, Donald G Truhlar, and Laura Gagliardi. Multiconfiguration pair-density functional theory outperforms kohn–sham density functional theory and multireference perturbation theory for ground-state and excited-state charge transfer. *J. Chem. Theory Comput.*, 11(8):3643–3649, 2015.

[32] Davide Presti, Donald G Truhlar, and Laura Gagliardi. Intramolecular charge

transfer and local excitation in organic fluorescent photoredox catalysts explained by rasci-pdft. *J. Phys. Chem. C*, 122(22):12061–12070, 2018.

[33] Shreya Verma, Abhishek Mitra, Qiaohong Wang, Ruhee D'Cunha, Bhavnesh Jangid, Matthew R Hennefarth, Valay Agarawal, Leon Otis, Soumi Haldar, Matthew R Hermes, et al. Multireference embedding and fragmentation methods for classical and quantum computers: From model systems to realistic applications. *Chem. Rev.*, 126(1):184–203, 2026.

[34] Matthew R Hermes, Riddhish Pandharkar, and Laura Gagliardi. Variational localized active space self-consistent field method. *J. Chem. Theory Comput.*, 16(8):4923, 2020.

[35] Christopher J Cramer. *Essentials of computational chemistry: theories and models*. 2013.

[36] T Helgaker, P Jorgensen, and J Olsen. Molecular electronic-structure theory (john wiley & sons. *Ltd.: Chichester*, pages 405–425, 2000.

[37] Erwin Schrödinger. An undulatory theory of the mechanics of atoms and molecules. *Phys. Rev.*, 28(6):1049, 1926.

[38] Joseph Ivanic and Klaus Ruedenberg. Identification of deadwood in configuration spaces through general direct configuration interaction. *Theor. Chem. Acc.*, 106:339, 2001.

[39] Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg,

Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825): 357–362, 2020.

[40] Ryosuke Okuta, Yuya Unno, Daisuke Nishino, Shohei Hido, and Crissman Loomis. Cupy: A numpy-compatible library for nvidia gpu calculations. In *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[41] Riddhish Pandharkar, Matthew R Hermes, Christopher J Cramer, and Laura Gagliardi. Localized active space-state interaction: a multireference method for chemical insight. *J. Chem. Theory Comput.*, 18(11):6557–6566, 2022.

[42] Valay Agarawal, Daniel S King, Matthew R Hermes, and Laura Gagliardi. Automatic state interaction with large localized active spaces for multimetallic systems. *J. Chem. Theory Comput.*, 20(11):4654–4662, 2024.

[43] B. Huron, J. P. Malrieu, and P. Rancurel. Iterative perturbation calculations of ground and excited state energies from multiconfigurational zeroth-order wave functions. *J. Chem. Phys.*, 58(1973):5745, 1973.

[44] Renzo Cimiraglia and Maurizio Persico. Recent advances in multireference second order perturbation CI: The CIPSI method revisited. *J. Comput. Chem.*, 8(1):39, 1987.

[45] Josefa Miralles, Oscar Castell, Rosa Caballol, and Jean Paul Malrieu. Specific

CI calculation of energy differences: Transition energies and bond energies. *Chem. Phys.*, 172(1):33, 1993.

[46] Frank Neese. A spectroscopy oriented configuration interaction procedure. *J. Chem. Phys.*, 119(18):9428, 2003.

[47] Norm M. Tubman, Joonho Lee, Tyler Y. Takeshita, Martin Head-Gordon, and K. Birgitta Whaley. A deterministic alternative to the full configuration interaction quantum Monte Carlo method. *J. Chem. Phys.*, 145:044112, 2016.

[48] Adam A. Holmes, Norm M. Tubman, and C. J. Umrigar. Heat-bath configuration interaction: An efficient selected configuration interaction algorithm inspired by heat-bath sampling. *J. Chem. Theory Comput.*, 12(8):3674, 2016.

[49] Steven R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69(19):2863, 1992.

[50] Per Aake. Malmqvist, Alistair. Rendell, and Bjoern O. Roos. The restricted active space self-consistent-field method, implemented with a split graph unitary group approach. *J. Phys. Chem.*, 94(14):5477, 1990.

[51] Jeppe Olsen, Björn O. Roos, Poul Jörgensen, and Hans Jo/rgen Aa. Jensen. Determinant based configuration interaction algorithms for complete and restricted configuration interaction spaces. *J. Chem. Phys.*, 89(4):2185, 1988.

[52] Dongxia Ma, Giovanni Li Manni, and Laura Gagliardi. The generalized active space concept in multiconfigurational self-consistent field methods. *J. Chem. Phys.*, 135(4):044128, 2011.

[53] Konstantinos D. Vogiatzis, Giovanni Li Manni, Samuel J. Stoneburner, Dongxia Ma, and Laura Gagliardi. Systematic expansion of active spaces beyond the casscf limit: A gasscf/splitgas benchmark study. *J. Chem. Theory Comput.*, 11(7):3010, 2015.

[54] Samuel O. Odoh, Giovanni Li Manni, Rebecca K. Carlson, Donald G. Truhlar, and Laura Gagliardi. Separated-pair approximation and separated-pair pair-density functional theory. *Chem. Sci.*, 7:2399, 2016.

[55] Carlos a. Jiménez-Hoyos and Gustavo E. Scuseria. Cluster-based mean-field and perturbative description of strongly correlated fermion systems. Application to the 1D and 2D Hubbard model. *Phys. Rev. B*, 92:085101, 2015.

[56] Matthew R. Hermes and Laura Gagliardi. Multiconfigurational self-consistent field theory with density matrix embedding: The localized active space self-consistent field method. *J. Chem. Theory Comput.*, 15:972, 2019.

[57] Kizashi Yamaguchi, Hiroaki Fukui, and Takayuki Fueno. Molecular orbital (mo) theory for magnetically interacting organic compounds. ab-initio mo calculations of the effective exchange integrals for cyclophane-type carbene dimers. *Chem. Lett.*, 15(4):625, 1986.

[58] Jenny G Vitillo, Aditya Bhan, Christopher J Cramer, Connie C Lu, and Laura Gagliardi. Quantum chemical characterization of structural single fe (ii) sites in mil-type metal–organic frameworks for the oxidation of methane to methanol and ethane to ethanol. *ACS Catal.*, 9(4):2870–2879, 2019.

[59] Giovanni Ghigo, Björn O Roos, and Per-Åke Malmqvist. A modified definition of the zeroth-order hamiltonian in multiconfigurational perturbation theory (caspt2). *Chem. Phys. Lett.*, 396(1-3):142–149, 2004.

[60] Huanchen Zhai, Henrik R Larsson, Seunghoon Lee, Zhi-Hao Cui, Tianyu Zhu, Chong Sun, Linqing Peng, Ruojing Peng, Ke Liao, Johannes Tölle, and et. al. Block2: a comprehensive open source framework to develop and apply state-of-the-art dmrg algorithms in electronic structure and beyond. *J. Chem. Phys.*, 159(23):234801, 2023.

[61] Matthew Hermes, Jangid Bhavnesh, Valay Agarawal, and Laura Gagliardi. Localized active space state interaction singles. ChemRxiv, 2025.

[62] Melisa Alkan, Buu Q Pham, Daniel Del Angel Cruz, Jeff R Hammond, Taylor A Barnes, and Mark S Gordon. Liberi—a portable and performant multi-gpu accelerated library for electron repulsion integrals via openmp offloading and standard language parallelism. *J. Chem. Phys.*, 161(8), 2024.

[63] Ivan S Ufimtsev and Todd J Martinez. Quantum chemistry on graphical processing units. 1. strategies for two-electron integral evaluation. *J. Chem. Theory Comput.*, 4(2):222–231, 2008.

[64] Jorge L Galvez Vallejo, Calum Snowdon, Ryan Stocks, Fazeleh Kazemian, Fiona Chuo Yan Yu, Christopher Seidl, Zoe Seeger, Melisa Alkan, David Poole, Bryce M Westheimer, et al. Toward an extreme-scale electronic structure system. *J. Chem. Phys.*, 159(4), 2023.

[65] David B Williams-Young, Andrey Asadchev, Doru Thom Popovici, David Clark, Jonathan Waldrop, Theresa L Windus, Edward F Valeev, and Wibe A de Jong. Distributed memory, gpu accelerated fock construction for hybrid, gaussian basis density functional theory. *J. Chem. Phys.*, 158(23):234104, 2023.

[66] Loriano Storchi, Laura Bellentani, Jeff Hammond, Sergio Orlandini, Leonardo Pacifici, Nicoló Antonini, and Leonardo Belpassi. Acceleration of the relativistic dirac–kohn–sham method with gpu: A pre-exascale implementation of bertha and pybertha. *J. Chem. Theory Comput.*, 21(7):3460–3475, 2025.

[67] Leslie Vogt, Roberto Olivares-Amaya, Sean Kermes, Yihan Shao, Carlos Amador-Bedolla, and Alán Aspuru-Guzik. Accelerating resolution-of-the-identity second-order møller- plesset quantum chemistry calculations with graphical processing units. *J. Phys. Chem. A*, 112(10):2049–2057, 2008.

[68] Ryan Stocks, Elise Palethorpe, and Giuseppe MJ Barca. High-performance multi-gpu analytic ri-mp2 energy gradients. *J. Chem. Theory Comput.*, 20(6): 2505–2519, 2024.

[69] Wenjing Ma, Sriram Krishnamoorthy, Oreste Villa, and Karol Kowalski. Gpu-based implementations of the noniterative regularized-ccsd (t) corrections: applications to strongly correlated systems. *J. Chem. Theory Comput.*, 7(5): 1316–1327, 2011.

[70] A Eugene DePrince III and Jeff R Hammond. Coupled cluster theory on graphics processing units i. the coupled cluster doubles method. *J. Chem. Theory Comput.*, 7(5):1287–1295, 2011.

[71] Katharina Boguslawski, Filip Brzęk, Rahul Chakraborty, Kacper Cieślak, Seyedehdelaram Jahani, Aleksandra Leszczyk, Artur Nowak, Emil Sujkowski, Julian Świerczyński, Somayeh Ahmadkhani, Dariusz Kędziera, Maximilian H Kriebel, Piotr S Żuchowski, and Paweł Tecmer. Pybest: Improved functionality and enhanced performance. *Comput. Phys. Comm.*, 297:109049, 2024.

[72] Maximilian H Kriebel, Paweł Tecmer, Marta Gałynska, Aleksandra Leszczyk, and Katharina Boguslawski. Accelerating pythonic coupled-cluster implementations: A comparison between cpus and gpus. *J. Chem. Theory Comput.*, 20 (3):1130–1142, 2024.

[73] Andrey Asadchev and Mark S Gordon. New multithreaded hybrid cpu/gpu approach to hartree–fock. *J. Chem. Theory Comput.*, 8(11):4166–4176, 2012.

[74] Giuseppe MJ Barca, David L Poole, Jorge L Galvez Vallejo, Melisa Alkan, Colleen Bertoni, Alistair P Rendell, and Mark S Gordon. Scaling the hartree-fock matrix build on summit. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14. IEEE, 2020.

[75] Augustin Bussy, Ole Schütt, and Jürg Hutter. Sparse tensor based nuclear gradients for periodic hartree–fock and low-scaling correlated wave function methods in the cp2k software package: A massively parallel and gpu accelerated implementation. *J. Chem. Phys.*, 158(16), 2023.

[76] Ji Qi, Yingfeng Zhang, and Minghui Yang. A hybrid cpu/gpu method for hartree–fock self-consistent-field calculation. *J. Chem. Phys.*, 159(10), 2023.

[77] Daniel Stopper and Roland Roth. Massively parallel gpu-accelerated minimization of classical density functional theory. *J. Chem. Phys.*, 147(6), 2017.

[78] Weile Jia, Zongyan Cao, Long Wang, Jiyun Fu, Xuebin Chi, Weiguo Gao, and Lin-Wang Wang. The analysis of a plane wave pseudopotential density functional theory code on a gpu machine. *Comput. Phys. Comm.*, 184(1):9–18, 2013.

[79] Calum Snowdon and Giuseppe MJ Barca. An efficient ri-mp2 algorithm for distributed many-gpu architectures. *J. Chem. Theory Comput.*, 20(21):9394–9406, 2024.

[80] Edward G Hohenstein, Nathan Luehr, Ivan S Ufimtsev, and Todd J Martínez. An atomic orbital-based formulation of the complete active space self-consistent field method on graphical processing units. *J. Chem. Phys.*, 142(22), 2015.

[81] James W Snyder, Edward G Hohenstein, Nathan Luehr, and Todd J Martínez. An atomic orbital-based formulation of analytical gradients and nonadiabatic coupling vector elements for the state-averaged complete active space self-consistent field method on graphical processing units. *J. Chem. Phys.*, 143 (15), 2015.

[82] Edward G Hohenstein, Marine EF Bouduban, Chenchen Song, Nathan Luehr, Ivan S Ufimtsev, and Todd J Martínez. Analytic first derivatives of floating occupation molecular orbital-complete active space configuration interaction on graphical processing units. *J. Chem. Phys.*, 143(1), 2015.

[83] 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. *J. Chem. Theory Comput.*, 20(19):8397–8404, 2024.

[84] Chunyang Xiang, Weile Jia, Wei-Hai Fang, and Zhendong Li. Distributed multi-gpu ab initio density matrix renormalization group algorithm with applications to the p-cluster of nitrogenase. *J. Chem. Theory Comput.*, 20(2): 775–786, 2024.

[85] TP Straatsma, R Broer, S Faraji, RWA Havenith, LE Suarez, RK Kathir, M Wibowo, and C De Graaf. Gronor: Massively parallel and gpu-accelerated non-orthogonal configuration interaction for large molecular systems. *J. Chem. Phys.*, 152(6):064111, 2020.

[86] Yifei Huang, Zhen Guo, Hung Q Pham, and Dingshun Lv. Gpu-accelerated auxiliary-field quantum monte carlo with multi-slater determinant trial states. *arXiv preprint arXiv:2406.08314*, 2024.

[87] Hong Gao, Satoshi Imamura, Akihiko Kasagi, and Eiji Yoshida. Distributed implementation of full configuration interaction for one trillion determinants. *J. Chem. Theory Comput.*, 20(3):1185–1192, 2024.

[88] Qiming Sun, Xing Zhang, Samragni Banerjee, Peng Bao, Marc Barbry, Nick S. Blunt, Nikolay A. Bogdanov, George H. Booth, Jia Chen, Zhi-Hao Cui, Janus J.

Eriksen, Yang Gao, Sheng Guo, Jan Hermann, Matthew R. Hermes, Kevin Koh, Peter Koval, Susi Lehtola, Zhendong Li, Junzi Liu, Narbe Mardirossian, James D. McClain, Mario Motta, Bastien Mussard, Hung Q. Pham, Artem Pulkin, Wirawan Purwanto, Paul J. Robinson, Enrico Ronca, Elvira R. Sayfutyarova, Maximilian Scheurer, Henry F. Schurkus, James E. T. Smith, Chong Sun, Shi-Ning Sun, Shiv Upadhyay, Lucas K. Wagner, Xiao Wang, Alec White, James Daniel Whitfield, Mark J. Williamson, Sebastian Wouters, Jun Yang, Jason M. Yu, Tianyu Zhu, Timothy C. Berkelbach, Sandeep Sharma, Alexander Yu. Sokolov, and Garnet Kin-Lic Chan. Recent developments in the pyscf program package. *J. Chem. Phys.*, 153(2):024109, 2020.

[89] J Wayne Mullinax, Elvis Maradzike, Lauren N Koulias, Mohammad Mostafanejad, Evgeny Epifanovsky, Gergely Gidofalvi, and A Eugene DePrince III. Heterogeneous cpu+ gpu algorithm for variational two-electron reduced-density matrix-driven complete active-space self-consistent field theory. *J. Chem. Theory Comput.*, 15(11):6164–6178, 2019.

[90] Örs Legeza, Andor Menczer, Ádám Ganyecz, Miklós Antal Werner, Kornél Kapás, Jeff Hammond, Sotiris S Xantheas, Martin Ganahl, and Frank Neese. Orbital optimization of large active spaces via ai-accelerators, 2025.

[91] Jan Almlöf, Knut Fægri Jr, and Knut Korsell. Principles for a direct scf approach to licao–moab-initio calculations. *J. Comput. Chem.*, 3(3):385–399, 1982.

[92] Thomas Herault, Yves Robert, George Bosilca, Robert J Harrison, Cannada A

Lewis, Edward F Valeev, and Jack J Dongarra. Distributed-memory multi-gpu block-sparse tensor contraction for electronic structure. In *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 537–546. IEEE, 2021.

[93] Stefan Seritan, Christoph Bannwarth, Bryan S. Fales, Edward G. Hohenstein, Christine M. Isborn, Sara I. L. Kokkila-Schumacher, Xin Li, Fang Liu, Nathan Luehr, James W. Snyder Jr., Chenchen Song, Alexey V. Titov, Ivan S. Ufimtsev, Lee-Ping Wang, and Todd J. Martínez. Terachem: A graphical processing unit-accelerated electronic structure package for large-scale ab initio molecular dynamics. *Wiley Interdiscip. Rev. Comput. Mol. Sci.*, 11(2):e1494, 2021.

[94] Henrik Koch, Alfredo Sánchez de Merás, and Thomas Bondo Pedersen. Reduced scaling in electronic structure calculations using cholesky decompositions. *J. Chem. Phys.*, 118(21):9481–9484, 2003.

[95] Frank Neese, Frank Wennmohs, Andreas Hansen, and Ute Becker. Efficient, approximate and parallel hartree–fock and hybrid dft calculations. a 'chain-of-spheres' algorithm for the hartree–fock exchange. *Chem. Phys.*, 356(1-3): 98–109, 2009.

[96] Henryk Laqua, Travis H Thompson, Jörg Kussmann, and Christian Ochsenfeld. Highly efficient, linear-scaling seminumerical exact-exchange method for graphic processing units. *J. Chem. Theory Comput.*, 16(3):1456–1468, 2020.

[97] Ruiyan Wang, Yuanheng Wang, Lixin Lu, Diptarka Hait, and Todd J Martínez.

Complete active space self-consistent field with gpu-accelerated density fitting. *J. Chem. Theory Comput.*, 22(7):3398–3414, 2026.

[98] Hans-Joachim Werner and Peter J Knowles. A second order multiconfiguration scf procedure with optimum convergence. *J. Chem. Phys.*, 82(11):5053–5063, 1985.

[99] Hans-Joachim Werner and Wilfried Meyer. A quadratically convergent multiconfiguration–self-consistent field method with simultaneous optimization of orbitals and ci coefficients. *J. Chem. Phys.*, 73(5):2342–2356, 1980.

[100] Valay Agarawal, Rishu Khurana, Cong Liu, Matthew R. Hermes, Christopher Knight, and Laura Gagliardi. Enabling Multireference Calculations on Multimetallic Systems with Graphic Processing Units. *J. Chem. Theory Comput.*, 21(15):7378–7393, August 2025. Publisher: American Chemical Society.

[101] Matthew R. Hermes. mrh: Quantum chemistry tools, 2018. Commit ID: 49a5950.

[102] pybind11 Authors. pybind11. Accessed: 2025-02-18.

[103] NVIDIA Corporation. Nsight systems, . Accessed: 2025-02-18.

[104] Rui Li, Qiming Sun, Xing Zhang, and Garnet Kin Chan. Introducing gpu-acceleration into the python-based simulations of chemistry framework. *arXiv preprint arXiv:2407.09700*, 2024.

[105] Xiaojie Wu, Qiming Sun, Zhichen Pu, Tianze Zheng, Wenzhi Ma, Wen Yan, Xia Yu, Zhengxiao Wu, Mian Huo, Xiang Li, Weiluo Ren, Sheng Gong, Yumin

Zhang, and Weihao Gao. Enhancing gpu-acceleration in the python-based simulations of chemistry framework, 2024.

[106] SYCLomatic Authors. Syclomatic. Accessed: 2025-02-18.

[107] Hipify. Accessed: 2025-02-18.

[108] H. Carter Edwards, Christian R. Trott, and Daniel Sunderland. Kokkos: Enabling manycore performance portability through polymorphic memory access patterns. *J. Parallel Distrib. Comput.*, 74(12):3202–3216, 2014.

[109] Christian Trott, Luc Berger-Vergiat, David Poliakoff, Sivasankaran Rajamanickam, Damien Lebrun-Grandie, Jonathan Madsen, Nader Al Awar, Milos Gligoric, Galen Shipman, and Geoff Womeldorff. The kokkos ecosystem: Comprehensive performance portability for high performance computing. *Comput. Sci. Eng.*, 23(5):10–18, 2021.

[110] Christian R. Trott, Damien Lebrun-Grandié, Daniel Arndt, Jan Ciesko, Vinh Dang, Nathan Ellingwood, Rahulkumar Gayatri, Evan Harvey, Daisy S. Hollman, Dan Ibanez, Nevin Liber, Jonathan Madsen, Jeff Miles, David Poliakoff, Amy Powell, Sivasankaran Rajamanickam, Mikael Simberg, Dan Sunderland, Bruno Turcksin, and Jeremiah Wilke. Kokkos 3: Programming model extensions for the exascale era. *IEEE Trans. Parallel Distrib. Syst.*, 33(4):805–817, 2022.

[111] Rajib Nath, Stanimire Tomov, and Jack Dongarra. Accelerating GPU kernels for dense linear algebra. In *Proceedings of the 2009 International Meeting on*

High Performance Computing for Computational Science, VECPAR'10, Berkeley, CA, June 22-25 2010.

[112] NVIDIA Corporation. cutensor, . Accessed: 2025-02-18.

[113] *Polaris and Acceptance Testing*, 2023.

[114] Yuki Kurashige, Garnet Kin-Lic Chan, and Takeshi Yanai. Entangled quantum electronic wavefunctions of the mn4cao5 cluster in photosystem ii. *Nat. Chem.*, 5(8):660–666, 2013.

[115] Benjamin Yeh, Saumil Chheda, Steven D Prinslow, Adam S Hoffman, Jiyun Hong, Jorge E Perez-Aguilar, Simon R Bare, Connie C Lu, Laura Gagliardi, and Aditya Bhan. Structure and site evolution of framework ni species in mil-127 mofs for propylene oligomerization catalysis. *J. Am. Chem. Soc.*, 145(6): 3408–3418, 2023.

[116] Thom H Dunning Jr. Gaussian basis sets for use in correlated molecular calculations. i. the atoms boron through neon and hydrogen. *J. Chem. Phys.*, 90 (2):1007–1023, 1989.

[117] Frank Czerny, Keith Searles, Petr Sot, Johannes F Teichert, Prashanth W Menezes, Christophe Coperet, and Matthias Driess. Well-defined, silica-supported homobimetallic nickel hydride hydrogenation catalyst. *Inorg. Chem.*, 60(8):5483–5487, 2021.

[118] Valay Agarawal. Gpu4lasscf data. Accessed: 2025-04-17.

[119] Helmut Beinert, Richard H. Holm, and Eckard Münck. Iron-Sulfur Clusters: Nature's Modular, Multipurpose Structures. *Science*, 277(5326):653–659, August 1997.

[120] Brighton A. Skeel and Daniel L. M. Suess. Iron-sulfur clusters: the road to room temperature. *J. Biol. Inorg. Chem.*, 30(2):151–159, March 2025.

[121] Seunghoon Lee, Joonho Lee, Huanchen Zhai, Yu Tong, Alexander M. Dalzell, Ashutosh Kumar, Phillip Helms, Johnnie Gray, Zhi-Hao Cui, Wenyuan Liu, Michael Kastoryano, Ryan Babbush, John Preskill, David R. Reichman, Earl T. Campbell, Edward F. Valeev, Lin Lin, and Garnet Kin-Lic Chan. Evaluating the evidence for exponential quantum advantage in ground-state quantum chemistry. *Nat. Comm.*, 14(1):1952, April 2023.

[122] Francesco Aquilante, Jochen Autschbach, Rebecca K. Carlson, Liviu F. Chibotaru, Mickaël G. Delcey, Luca De Vico, Ignacio Fdez. Galván, Nicolas Ferré, Luis Manuel Frutos, Laura Gagliardi, Marco Garavelli, Angelo Giussani, Chad E. Hoyer, Giovanni Li Manni, Hans Lischka, Dongxia Ma, Per Åke Malmqvist, Thomas Müller, Artur Nenov, Massimo Olivucci, Thomas Bondo Pedersen, Daoling Peng, Felix Plasser, Ben Pritchard, Markus Reiher, Ivan Rivalta, Igor Schapiro, Javier Segarra-Martí, Michael Stenrup, Donald G. Truhlar, Liviu Ungur, Alessio Valentini, Steven Vancoillie, Valera Veryazov, Victor P. Vysotskiy, Oliver Weingart, Felipe Zapata, and Roland Lindh. Molcas 8: New capabilities for multiconfigurational quantum chemical calculations across the periodic table. *J. Comput. Chem.*, 37(5):506–541, 2016.

[123] Huanchen Zhai, Chenghan Li, Xing Zhang, Zhendong Li, Seunghoon Lee, and Garnet Kin-Lic Chan. Classical solution of the FeMo-cofactor model to chemical accuracy and its implications, January 2026. arXiv:2601.04621 [physics.chem-ph].

[124] David W. Small and Martin Head-Gordon. Independent amplitude approximations in coupled cluster valence bond theory: Incorporation of 3-electron-pair correlation and application to spin frustration in the low-lying excited states of a ferredoxin-type tetrametallic iron-sulfur cluster. *J. Chem. Phys.*, 149(14): 144103, October 2018.

[125] Werner Dobrautz, Oskar Weser, Nikolay A. Bogdanov, Ali Alavi, and Giovanni Li Manni. Spin-Pure Stochastic-CASSCF via GUGA-FCIQMC Applied to Iron–Sulfur Clusters. *J. Chem. Theory Comput.*, 17(9):5684–5703, September 2021.

[126] Giovanni Li Manni, Werner Dobrautz, Nikolay A. Bogdanov, Kai Guther, and Ali Alavi. Resolution of Low-Energy States in Spin-Exchange Transition-Metal Clusters: Case Study of Singlet States in [Fe(III)4S4] Cubanes. *J. Phys Chem. A*, 125(22):4727–4740, June 2021.

[127] Don Danilov, Brad Ganoe, Leon Otis, Zhi Gong, Zixiang Lu, and James Shee. Selecting optimal unrestricted Hartree-Fock trial wavefunctions for phaseless auxiliary-field quantum Monte Carlo: Accuracy and limitations in modeling three iron-sulfur clusters, May 2026. arXiv:2605.03981 [physics.chem-ph].

[128] Eirik F. Kjønstad, Huanchen Zhai, James Shee, Sandeep Sharma, and Garnet Kin-Lic Chan. Can phaseless auxiliary-field quantum Monte Carlo with broken symmetry trials describe iron-sulfur clusters?, May 2026. arXiv:2605.03270 [physics.chem-ph].

[129] Jacob J Wardzala, Matthew R Hennefarth, Valay Agarawal, Bhavnesh Jangid, Aniruddha Seal, Matthew R Hermes, Daniel S King, and Laura Gagliardi. Multireference methods for chemistry and materials science: automated active spaces, efficient dynamic correlation, and extended systems. *Chem. Rev.*, 126 (8):4592–4618, 2026.

[130] Elvira R. Sayfutyarova, Qiming Sun, Garnet Kin-Lic Chan, and Gerald Knizia. Automated Construction of Molecular Active Spaces from Atomic Valence Orbitals. *J. Chem. Theory Comput.*, 13(9):4063–4078, September 2017.

[131] Qiming Sun and Garnet Kin Lic Chan. Exact and optimal quantum mechanics/molecular mechanics boundaries. *J. Chem. Theory Comput.*, 10(9): 3784–3790, 2014.