Efficient Implementation of Configuration Interaction and Selected CI using Reinforcement Learning

Ojas Singh

A dissertation submitted for the partial fulfilment of BS-MS dual degree in Science

Indian Institute of Science Education and Research, Mohali

December 2021

Ojas Singh

Efficient Implementation of Configuration Interaction and Selected CI using Reinforcement Learning

Master Thesis, December 14, 2021
Reviewers: Dr. R. Ramesh and Dr. K. R. Shamasundar
Supervisor: Dr. P. Balanarayan

Indian Institute of Science Education and Research, Mohali
Department of Chemical Sciences
Knowledge city, Sector 81, SAS Nagar, Manauli
140306, Mohali

Certificate of Examination

This is to certify that the dissertation titled "**Efficient Implementation of Configuration Interaction and Selected CI using Reinforcement Learning"** submitted by **Ojas Singh** (Reg. No. MS16084) for the partial fulfilment of BS-MS dual degree programme of the institute, IISER Mohali, has been examined by the thesis committee duly appointed by the Institute. The committee finds the work done by the candidate satisfactory and recommends that the report be accepted.

Dr. R. Ramesh	Dr. K. R. Shamasundar	Dr. P. Balanarayan
(Committee member)	(Committee member)	(Supervisor)

Mohali, December 14, 2021

Declaration

The work presented in this dissertation has been carried out by me under the guidance of Dr. P. Balanarayan at the Indian Institute of Science Education and Research, Mohali.

This work has not been submitted in part or in full for a degree, a diploma, or a fellowship to any other university or institute. Whenever contributions of others are involved, every effort is made to indicate this clearly, with due acknowledgement of collaborative research and discussions. This thesis is a bonafide record of original work done by me and all sources listed within have been detailed in the bibliography.

Ojas Singh
(Candidate)
Mohali, December 14, 2021

In my capacity as the supervisor of the candidates project work, I certify that the above statements by the candidate are true to the best of my knowledge.

Dr. P. Balanarayan
(Supervisor)
Mohali, December 14, 2021

Acknowledgement

Foremost, I would like to express my wholehearted gratitude to my thesis supervisor Dr. P. Balanarayan for his constant support and unending kindness throughout the duration of this research work. I am sincerely thankful to him for introducing me to this fascinating area of research. I would also like to congratulate him on building a stalwart team of researchers in the lab who are passionate about their research.

I would like to thank Dr. R. Ramachandran and Dr. K. R. Shamasundar for being a part of my thesis committee, and for their insightful comments and mentorship that paved the way for my dissertation work. I am thankful to Prof. J. Gowrishankar, director, IISER Mohali, for providing me the opportunity to utilize the excellent infrastructure in this premier research institute for the efficient execution of my work.

I am immensely grateful to the lab members who have always maintained a keenly enthusiastic environment that is conducive to learning. I would especially like to thank Alkit Gugalia and Prashant Raj for all the brainstorming sessions that have helped me every step of the way. Their constructive advice in all academic matters has been invaluable.

I would not be here without my beloved family. I am indebted to my parents for nurturing me all these wonderful years. I would especially like to acknowledge them and my loving sisters for being so understanding when I could not be available due to being caught up with work. This dissertation would not have reached its completion without their unconditional support.

This thesis would not have been possible without the treasured company of a number of wonderful individuals. I express my heartiest gratitude to my friends Aman Singh Katariya, Abhijit Pati, Aaditya Mishra, Vivek Shukla, Abhishek Purohit, and Hunarpreet Kaur for the stimulating discussions and

making this a blessed and blissful experience. I extend my appreciation to Pritam bhaiya for his integral advice during the course of my study. I am profoundly grateful to Harjasnoor Kakkar for the insightful conversations within and beyond science, for being my greatest confidante and a source of comfort that cannot be emphasized enough in this brief mention. I would also like to acknowledge the creators of CS:GO, and Isaac, my cat, who would never know how valuable their presence has been to the progress of this thesis.

Finally, I would like to thank IISER Mohali and the Department of Chemical Sciences for infrastructure and facilities required for research purposes and DST-KVPY for fellowship.

Abstract

In order to account for the missing electron correlation from the Hartree-Fock method, the Configuration Interaction method uses a variational wave function that is a linear combination of configuration state functions (CSFs) built from spin orbitals. Full CI scales unfavorably with the number of orbitals and electrons relative to other orbital methods. Hence, implementation of CI is done in Rust using numerous Code Optimizations, making this implementation extremely efficient. To deal with an even bigger systems, selected Configuration Interaction is explored, and a very promising Reinforcement Learning-based method has been implemented and improved further for excited states energies.

List of Figures

List of Tables

List of Algorithms

Contents

1 Introduction

Quantum mechanics is the characterization of the interaction of light and matter at an atomic scale. A state of a quantum system can be described by a wavefunction $\Psi(r,t)$, which can be determined at any point in time by solving the Schrödinger equation [1], given the initial state of the system $\Psi(r,t0)$,

$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \hat{H}\Psi(\mathbf{r},t) \tag{1.1}$$

1.1 The Quantum Many-Body Problem

The Schrödinger equation[1.1] can be analytically solved only for a few simple quantum systems such as the hydrogen atom. However, the problem results in an analytically unsolvable Schrödinger equation with an increase in the number of constituent particles in the system. This is known as the electronic quantum many-body problem, which is the central topic in quantum chemistry. In order to solve chemistry for such systems, numerical approximations are required.

1.1.1 Hartree-Fock Theory

One of the simplest approximation in ab-initio wavefunction theory is the Hartree-Fock (HF) method [2] [3]. In HF theory, the exact N-electron wavefunction is approximated by only a single Slater determinant. [4], $|\Phi_{HF}\rangle$.

The one-electron spin-orbitals, denoted as $\{\phi_k^\sigma\}$, from which $|\Phi_{HF}\rangle$ are variationally optimized to minimize the energy expectation value

$$E_{HF} = \min_{\phi_k} \left\langle \Phi_{HF}\left(\phi_k^\sigma\right) |\hat{H}| \Phi_{HF}\left(\phi_k^\sigma\right) \right\rangle \tag{1.2}$$

The variational optimization in the space of $\{\phi_k^\sigma\}$ consists of unitary transformation within the spin-orbital basis. So $|\Phi_{HF}\rangle$ can parametrized as

$$|\Phi_{HF}\rangle = \mathrm{e}^{-\hat{\kappa}}|0\rangle, \quad \text{with} \quad \hat{\kappa} = \sum_{ij,\sigma\tau} \kappa_{IJ} a_{i,\sigma}^\dagger a_{j,\tau} \tag{1.3}$$

with $I = (i,\sigma), J = (j,\tau)$ and $\kappa_{IJ} = -\kappa_{JI}^*$ forming an anti-Hermitian matrix of orbital rotation parameters, to ensure $\mathrm{e}^{\hat{k}}$ to be unitary. $|0\rangle$ in Eq.[1.3] is an arbitrary initial Slater determinant. With the Ansatz[1.3] Eq.[1.2] is then minimized with respect to the rotation parameters κ_{IJ}. In this formulation the constraint $\left\langle \phi_i^\sigma \mid \phi_j^\tau \right\rangle = \delta_{ij}\delta_{\sigma\tau}$ is automatically fulfilled, since unitary transformations will leave the spin-orbitals orthonormal. From Eq.[1.3] orbital rotations can be formulated as single excitation operators. In the final HF solution, these single excitations will be minimized leading to the Brillouin's theorem, that the matrix elements between single excitations and the HF states are zero [5].

The HF method solves the Schrödinger equation of a single electron in the mean-field created by the remaining $N-1$ electrons. The correlation between electrons is treated in an averaged manner, so the single N- body problem can be separated into N one-body problems. The averaged interaction between electrons in the Hartree-Fock method does not precisely represent the physical systems. However, for many systems, the HF method provides results that are not very far from the exact solution for a given basis set.

1.1.2 Correlation Energy

Within the HartreeFock method, only a single Slater determinant approximates the antisymmetric wave function. Exact wave functions, however, cannot generally be expressed as single determinants. The dynamic behav-

ior of electrons in close proximity leads to a total electronic energy difference between the exact and Hartree-Fock solutions. The difference is called the correlation energy, a term coined by Löwdin [8]. The correlation energy is given by the difference between the exact solution and the HF energy.

$$E_{corr} = E_{exact} - E_{HF} \tag{1.4}$$

Electronic correlation is related to the repulsion of electrons due to Coulombic interaction. In weakly correlated systems i.e. systems wherein the electrons scarcely interact, the mean-field HF model is an accurate description and is sufficient to yield excellent results. Whereas in the case of strongly correlated systems, the HF method fails since the complex correlated motion of electrons can not be characterized by a mean-field approach. All methods post-Hartree-Fock focus on determining the missing correlation energy.

To correctly characterize the chemical properties of a system dominated by strong correlation effects, where the mean-field approach fails, more elaborate treatments are required to capture the dynamic interaction between electrons. Several theories and approaches that expand on the result obtained by the mean-field solution are categorized under the name of post-Hartree-Fock methods. The most commonly used approximative method is the density functional theory (DFT), developed by Kohn and Sham [6] in 1965. Density functional theory simplifies the quantum many-body problem by reformulating it in terms of electron number density. However, DFT does not provide a high degree of accuracy either, whereas the accuracy and simplicity of the Configuration Interaction method is quite compelling.

1.1.3 Configuration Interaction

The configuration interaction (CI) [7] approach approximates the eigenfunction $|\Phi_0\rangle$ of the molecular Hamiltonian[1.5] as a linear expansion of N-electron basis functions (where N is the number of electrons in the system), i.e.,

$$|\Psi\rangle = \sum_I c_I |\Phi_I\rangle \tag{1.5}$$

The linear expansion coefficients c_l are the CI coefficients. Substituting this linear expansion into the electronic Schrödinger equation, one obtains [9] a matrix form more suitable for computation:

$$\mathrm{Hc} = E\mathrm{Sc} \tag{1.6}$$

where the Hamiltonian operator $\hat{H}$ has been replaced by a matrix H and the CI wavefunction $|\Psi\rangle$ has been replaced by a column vector of coefficients c. In principle, this "matrix mechanics" formulation is equivalent to the original electronic Schrödinger equation; [1] hence it is said that CI constitutes an "exact theory". In practise, however, the matrix equations are not perfect due to the expansion of the equation[1.5] must be truncated to a finite number of terms. The elements of the Hamiltonian matrix are given by $H_{IJ} = \left\langle \Phi_I | \hat{H} | \Phi_J \right\rangle$ and $\mathbf{S}$ is the overlap matrix with elements $S_{IJ} = \langle \Phi_I \mid \Phi_J \rangle$. If orthonormal functions $|\Phi_I\rangle$ are used for the expansion, then $\mathbf{S}$ becomes a unit matrix and the equation becomes an eigenvalue equation. The electronic ground state is represented by the lowest-energy solution, and excited electronic states are represented by higher-energy solutions.

According to the antisymmetry principle, a wavefunction describing a system of fermions must be antisymmetric with respect to the interchange of spin and spatial coordinates for any pair of electrons. This requirement is satisfied by making the expansion functions Slater determinants. A Slater determinant in which the one-electron functions $\phi_i, \phi_j, \ldots, \phi_k$ are occupied may be written

$$\Phi = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_i(\mathbf{x}_1) & \phi_j(\mathbf{x}_1) & \ldots & \phi_k(\mathbf{x}_1) \\ \phi_i(\mathbf{x}_2) & \phi_j(\mathbf{x}_2) & \ldots & \phi_k(\mathbf{x}_2) \\ \vdots & \vdots & & \vdots \\ \phi_i(\mathbf{x}_N) & \phi_j(\mathbf{x}_N) & \ldots & \phi_k(\mathbf{x}_N) \end{vmatrix} \tag{1.7}$$

and abbreviated as $|\phi_i \phi_j \ldots \phi_k\rangle$. Such a determinant satisfies the antisymmetry principle, since the interchange of coordinates for a pair of electrons translates to the swapping of rows of the determinant, which introduces a change in sign. An electronic wavefunction can be represented exactly by equation (1) if the expansion contains all possible Slater determinants formed from a complete set of one-electron functions $\{\phi\}$ [1.5]. Such a procedure called complete CI [10]. Since a complete set of orbitals will typ-

ically be infinite, a complete CI is practically impossible to perform. However, if the one-electron basis set is truncated, then only a finite number of determinants can be formed. Using every available determinants in the expansion constitutes a full CI method, and the resulting eigenvalues and eigenfunctions are exact within the space spanned by the given basis set. Although full CI results are computationally extremely costly, but they are essential for benchmarking more approximate methods. Unfortunately, even with an imperfect one-electron basis, a full CI is computationally unmanageable for any but the smallest systems, due to the vast number of N-electron basis functions required. The CI space must be reduced, in such a way that the approximate CI wavefunction and energy are as close as possible to the exact values. The Hartree-Fock technique, which provides the best single-configuration approximation to the wavefunction that can be generated from a given basis set of one-electron orbitals(usually atom centered and hence called atomic orbitals, or AOs), is by far the most prevalent approximation. This give in a set of molecular orbitals (MOs) which are linear combinations of the AOs :

$$\phi_i(\mathbf{x}_1) = \sum_{\mu} C_{\mu}^{i} \chi_{\mu}(\mathbf{x}_1) \tag{1.8}$$

where $\chi_{\mu}(\mathbf{x}_1)$ denotes an atomic orbital and C_{μ}^{i} is an SCF coefficient. The CI space can then be expanded according to substitution or "excitation" level relative to the HartreeFock "reference" determinant, i.e.,

$$|\Psi\rangle = c_0 |\Phi_0\rangle + \sum_{ia} c_i^a |\Phi_i^a\rangle + \sum_{a<b,i<j} c_{ij}^{ab} \left|\Phi_{ij}^{ab}\right\rangle + \sum_{a<b<c,i<j<k} c_{ijk}^{abc} \left|\Phi_{ijk}^{abc}\right\rangle + \ldots \tag{1.9}$$

where $|\Phi_i^a\rangle$ means the Slater determinant formed by replacing spin-orbital i in $|\Phi_0\rangle$ with spin orbital a, etc. This is the form of the full CI wave function. The restrictions on the sumnation indices (e.g., $a < b, r < s$, etc.) insure that a given excited determinant included in the sum only once.

1.1.4 Matrix Elements in Terms of One- and Two-electron Integrals

Slater's Rules

The matrix elements $H_{IJ} = \left\langle \Phi_I | \hat{H} | \Phi_J \right\rangle$ can be expressed in terms of one- and two-electron integrals. If we implement Slater determinants, the matrix elements may be evaluated using Slater's rules (also known as the Slater-Condon rules)[11][12][13] if a common set of one-electron orbitals are used for all determinants and if these orbitals are orthonormal. If nonorthogonal orbitals are employed then the more complicated Löwdin rules [14] apply.

Slater's rules are described here in terms of spin-orbitals, which are functions of the spin and spatial coordinates of a single electron. The one-electron integrals are written as

$$[i|\hat{h}|j] = \int \phi_i^* \left(\mathbf{x}_1\right) \hat{h} \left(\mathbf{x}_1\right) \phi_j \left(\mathbf{x}_1\right) d\mathbf{x}_1 \tag{1.10}$$

and the two-electron integrals, in Mulliken notation, are written as

$$[ij \mid kl] = \int \phi_i^* \left(\mathbf{x}_1\right) \phi_j \left(\mathbf{x}_1\right) \frac{1}{\mathbf{r}_{12}} \phi_k \left(\mathbf{x}_2\right)^* \phi_l \left(\mathbf{x}_2\right) d\mathbf{x}_1 d\mathbf{x}_2 \tag{1.11}$$

To use Slaters rules, we see how many spin orbitals they differ by and employ the following rules:

1. Identical Determinants:

$$\left\langle \Phi_1 | \hat{H} | \Phi_1 \right\rangle = \sum_m^N [i|\hat{h}|i] + \sum_{i>j}^N \{[ii \mid jj] - [ij \mid ji]\} \tag{1.12}$$

2. Determinants that Differ by One Spin Orbital:

$$\begin{aligned} |\Phi_1\rangle &= |\cdots i \cdots\rangle \\ |\Phi_2\rangle &= |\cdots j \cdots\rangle \\ \left\langle \Phi_1 | \hat{H} | \Phi_2 \right\rangle &= [i|\hat{h}|j] + \sum_k^N \{[ij \mid kk] - [ik \mid kj]\} \end{aligned} \tag{1.13}$$

3. Determinants that Differ by Two Spin Orbitals:

$$\begin{aligned} |\Phi_1\rangle &= |\cdots ij \cdots\rangle \\ |\Phi_2\rangle &= |\cdots kl \cdots\rangle \\ \left\langle \Phi_1 | \hat{H} | \Phi_2 \right\rangle &= [ik \mid jl] - [il \mid jk] \end{aligned} \tag{1.14}$$

Some of the terms mentioned above vanish after integrating over spin coordinates, and for pair of determinants that differ by more than two spin orbitals has a matrix element of zero. A derivation of given rules can be found in the book by Szabo and Ostlund [5]. The rules for evaluating elements of the Hamiltonian matrix in a CSF basis are more complicated and are generally derived using second quantization [10][15].

Second Quantization

The matrix elements can be expressed in a more general form.

$$H_{IJ} = \sum_{pq}^{n} \gamma_{pq}^{IJ} (p|\hat{h}|q) + \frac{1}{2} \sum_{pqrs}^{n} \Gamma_{pqrs}^{IJ} (pq \mid rs) \tag{1.15}$$

where the use of parenthesis rather than square brackets indicates a change from spin orbital notation to spatial orbital notation. Hence, the factor of $1/2$ in the two-electron term. The constants γ_{pq}^{IJ} and Γ_{pqrs}^{IJ} are called the one and two-electron coupling coefficients, respectively. The CI energy in terms of these coefficients is

$$E = \sum_{IJ}^{CI} c_I \left[\sum_{pq}^{n} \gamma_{pq}^{IJ} (p|\hat{h}|q) + \frac{1}{2} \sum_{pqrs}^{n} \Gamma_{pqrs}^{IJ} (pq \mid rs) \right] c_J \tag{1.16}$$

where we assume that the CI coefficients are real. The one- and two-electron reduced density matrices are defined as

$$\begin{aligned} \gamma_{pq} &= \sum_{IJ}^{CI} c_I c_J \gamma_{pq}^{IJ} \\ \Gamma_{pqrs} &= \sum_{IJ}^{CI} c_I c_J \Gamma_{pqrs}^{lJ} \end{aligned} \tag{1.17}$$

Using these definitions, the energy may be stated more compactly as

$$E = \sum_{pq}^{n} \gamma_{pq}(p|\hat{h}|q) + \frac{1}{2}\sum_{pqrs}^{n} \Gamma_{pqrs}(pq \mid rs) \tag{1.18}$$

The factor of $1/2$ is sometimes absorbed into the definition of the two-electron coupling coefficient and reduced density matrix. These coupling coefficients are generally derived using second quantization [5][16], in which the Hamiltonian is written as

$$\hat{H} = \sum_{pq}^{2n} a_p^\dagger a_q [p|\hat{h}|q] + \frac{1}{2}\sum_{pqrs}^{2n} a_p^\dagger a_r^\dagger a_s a_q [pq \mid rs] \tag{1.19}$$

where $a_p^\dagger$ and a_p are the creation and annihilation operators, respectively, for an electron in spin orbital p. The second-quantized form of the Hamiltonian is unaffected by the number of electrons. If the spatial component of α and β spin orbitals are identical, then the second-quantized Hamiltonian in terms of the following shift operators, which Paldus has shown[10] to be generators of the unitary group:

$$\hat{E}_{ij} = a_{i\alpha}^\dagger a_{j\alpha} + a_{i\beta}^\dagger a_{j\beta} \tag{1.20}$$

Due to the anticommutation relations of creation and annihilation operators,

$$\hat{E}_{ij}^\dagger = \hat{E}_{ji} \tag{1.21}$$

$$\left[\hat{E}_{ij}, \hat{E}_{kl}\right] = \hat{E}_{il}\delta_{jk} - \hat{E}_{kj}\delta_{il} \tag{1.22}$$

The resulting Hamiltonian in terms of these operators is

$$\hat{H} = \sum_{pq}^{n} h_{pq}\hat{E}_{pq} + \frac{1}{2}\sum_{pq+s}^{n} (pq \mid rs)\left(\hat{E}_{pq}\hat{E}_{rs} - \delta_{qr}\hat{E}_{ps}\right) \tag{1.23}$$

where $h_{pq} = (p|\hat{h}|q)$ and the one- and two-electron coupling coefficients can be written as

$$\begin{aligned} \gamma_{pq}^{IJ} &= \left\langle \Phi_I \left| \hat{E}_{pq} \right| \Phi_J \right\rangle \\ \Gamma_{pqrs}^{IJJ} &= \left\langle \Phi_I \left| \hat{E}_{pq}\hat{E}_{rs} - \delta_{qr}\hat{E}_{ps} \right| \Phi_J \right\rangle \end{aligned} \tag{1.24}$$

Furthermore, using equations [1.21] and [1.22],

$$\begin{aligned} \gamma_{pq}^{IJ} &= \left(\gamma_{qp}^{JI}\right)^* \\ \Gamma_{pqrs}^{IJ} = \Gamma_{rspq}^{IJ} &= \left(\Gamma_{srqp}^{JI}\right)^* = \left(\Gamma_{qpsr}^{JI}\right)^* \end{aligned} \tag{1.25}$$

1.1.5 Reducing the Size of the CI Space

The main disadvantage of FCI is its severely unfavourable scalability, which is combinatorial in the number of spatial orbitals n and electrons N, since the Hilbert space size, which is the number of possible determinants, is given by all the possibilities to distribute N electrons in 2n orbitals

$$N_{SD} = \begin{pmatrix} 2n \\ N \end{pmatrix} = \frac{(2n)!}{N!(2n-N)!} \tag{1.26}$$

So it is necessary to select only the most important N-electron functions.

Truncation by Excitation Level

The CI expansion is commonly truncated according to excitation level; normally, the expansion is truncated (for computational tractability) at doubly-excited configurations. As the Hamiltonian contains only two-body terms, only singles and doubles can interact directly with the reference; this is a result of Slater's rules [1.1.4]. Additionally, the matrix elements of singly substituted determinants with the reference are zero when canonical SCF orbitals are used, according to Brillouin's theorem. Hence, we can expects double excitations to make the largest contributions to the CI wavefunction after the reference, state. Still singles, triples, etc., do not interact directly with the reference, they can still contribute of the CI wavefunction because they mix with the doubles. Despite singles are much lower importance to the energy than doubles, they are still included in CI procedure because of their comparatively small number and because of their higher importance in describing one-electron properties.

$$|\Phi_0\rangle = c_0\,|\Psi_0\rangle + c_S|S\rangle + c_D|D\rangle + c_T|T\rangle + c_Q|Q\rangle + \cdots \tag{1.27}$$

After singles and doubles, triples and quadruples are the most essential determinants because only these can interact directly with doubles. With increasing substitution or excitation level relative to the reference, the contribution of a determinant to the final CI wavefunction is projected to fall off, given that the reference provides a good zeroth-order description of the desired electronic state. Singles and doubles give 95% of the correlation energy at the equilibrium geometries of the molecules. In terms of energy, quadruple excitations are more important than triple excitations. The CISD and CISDT methods perform noticeably worse at stretched geometries, but the CISDTQ method recovers a very high (and nearly constant) fraction of the correlation energy, implying that CISDTQ should provide reliable results for energy differences across potential energy surfaces for molecules as long as no more than two bonds are broken at the same time (simultaneously breaking three bouds would require up to sextuple substitutions).

Selected Configuration Interaction

Long ago, it was discovered that only a small number of determinants significantly contributes to the wave function even within a predefined subspace of determinants[18][19]. As a result, selecting determinants is a natural idea given in the late 1960s by Bender and Davidson[20] and Whitten and Hackmeyer[21]. Selected Configuration Interaction methods operate on the same principles as traditional CI approaches, except that determinants are chosen from the entire set of determinants based on their estimated contribution to the FCI wave function (estimated significance of a determinant may be based off perturbative or energetic heuristic considerations), rather than a priori based on an occupation or excitation criterion.
The most common approach is CIPSI developed by Huron, Rancurel and Malrieu,[25] that iteratively selects external determinants (determinants which are not present in the variational space) using a perturbative criterion. Typical procedure look like

Procedure

The variational wave function $\left|\Psi^{(n)}\right\rangle$ is defined over a set of determinants $\{|D_I\rangle\}^{(n)}$ in which we diagonalize $\hat{H}$.

$$\left|\Psi^{(n)}\right\rangle = \sum_I c_I^{(n)} |D_I\rangle \tag{1.28}$$

The determinants in $\{|D_I\rangle\}^{(n)}$ will be characterized as internal.
1. For all external determinants $|\alpha\rangle \notin \{|D_I\rangle\}^{(n)}$, compute the EpsteinNesbet first-order or second-order perturbative contribution to the energy

$$e_\alpha = \frac{\left\langle \Psi^{(n)}|\hat{H}|\alpha\right\rangle^2}{E^{(n)} - \langle\alpha|\hat{H}|\alpha\rangle} \tag{1.29}$$

$E^{(n)}$ is the variational energy of the wave function at the current iteration. Here, any other perturbation theory could be used to estimate e_α.
2. An estimate of the total missing correlation energy can be computed by summing all the e_α contributions

$$\begin{aligned} E_{\mathrm{PT2}} &= \sum_\alpha e_\alpha \\ E_{\mathrm{FCI}} &\approx E + E_{\mathrm{PT2}} \end{aligned} \tag{1.30}$$

3. $\{|\alpha\rangle\}_\star^{(n)}$, the subset of determinants $|\alpha\rangle$ with the largest contributions e_α, is added to the variational space

$$\{|D_I\rangle\}^{(n+1)} = \{|D_I\rangle\}^{(n)} \cup \{|\alpha\rangle\}_\star^{(n)} \tag{1.31}$$

4. Go to iteration $n+1$, or exit on some criterion (number of determinants in the wave function, low $E_{\mathrm{PT2}}, \dots$).
Because such a method may be applied to any state, it can treat both ground and excited states.

Problem with Truncated CI and sCI

If the excitation level under consideration is less than the highest possible, the CI technique is neither size-consistent nor size-extensive[17][23]. A method is size consistent if correctly describes the additive energy separation if a system is split into two noninteracting subsystem, e.g. due to distance in dissociation processes. Let $AB = A + B$ be a super-system consisting of two separated subsystems A and B, described by the Hamiltonians $\hat{H}_{AB} = \hat{H}_A + \hat{H}_B$ and $\left[\hat{H}_A, \hat{H}_B\right] = 0$. The total energy is additive separable,

while the wavefunction of the total system $|AB\rangle$ is multiplicative separable:

$$\hat{H}_{AB}|AB\rangle = \left(\hat{H}_A + \hat{H}_B\right)|A\rangle|B\rangle = E_A|A\rangle|B\rangle + E_B|A\rangle|B\rangle = (E_A + E_B)\,|AB\rangle \tag{1.32}$$

The CI method is not size-consistent[24], due to the linear wavefunction Ansatz in [1.9], except if all levels of possible excitations are taken into consideration[22].
A method is size extensive if it correctly describes the linear scaling of extensive properties, i.e. the energy, with the number of electrons. Size-extensivity and size-consistency are not mutually exclusive properties. In the interacting limit, size-extensivity is actually a essential and sufficient condition for size-consistency.

1.2 Selected CI Using Machine Learning

As the CI Hamiltonian Matrix is extremely large and highly sparse for bigger molecules and significant progress has been made in dealing with this type of problem on high performance computers using advanced sparse matrix techniques and efficient iterative methods. However, the size of the problems we can currently solve is still limited by the amount of available computational resources. Reducing CI space using selected Configuration or Truncation methods is a good way to deal with such problem.
The vital ingredient in Selected Configuration Interaction is the searching/ranking of determinants. Although numerous heuristics exist to fulfill these objectives, because these methods does not account the full underlying structure of the CI Hamiltonian, the solutions found may be far from optimal. Reinforcement learning approach try to solve the sCI problem by addressing the CI problem onto a sequential decision-making process, in which the agent learns which determinants to include and which to ignore, yielding a compressed wave function with good accuracy.[26]

The purpose of sCI approaches is to minimise the expectation value of the molecular electronic Hamiltonian spanned by a subset of all potential determinants, we are looking for the best solution to

$$\min_{\|\Psi_{\mathrm{CI}}\|_0 \leq k} \frac{\left\langle \Psi_{\mathrm{CI}} | \hat{H} | \Psi_{\mathrm{CI}} \right\rangle}{\left\langle \Psi_{\mathrm{CI}} \mid \Psi_{\mathrm{CI}} \right\rangle} \tag{1.33}$$

where $\|\Psi_{\mathrm{CI}}\|_0$ is the cardinality (the number of non-zero elements) of the CI vector. The goal of selected CI methods is to find the solution to Eq. [1.33] where k is the maximum number of determinants(userdefined parameter and depend on desired accuracy) to be considered in the sCi problem. If k is equal to the dimension of all possible determinants, the solution to Eq. [1.33] is equivalent to the FCI problem. Finally, this problem can be thought of as a subset of a combinatorial optimization problem. What is the ideal combination of determinants that minimises the expected value of the Hamiltonian given a subset of possible determinants to include?

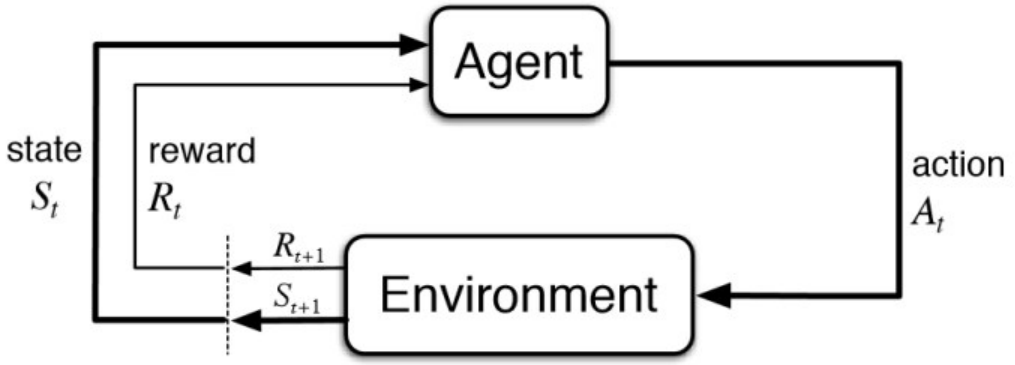


Fig. 1.1: Basic Diagram of Reinforcement Learning

An agent is trained in reinforcement learning to do a set of actions in order to maximise a reward[28]. The reinforcement learning process is divided by episodes in which the agent explores its environment and receives rewards or punishments for its actions. To map the sCI method onto a reinforcement learning framework, the state, environment, actions, and rewards must be defined. The state is the current collection of selected determinants to be included in the sCI calculation. The environment is the set of all Slater determinants that can exist. During each step of an episode, the local reward

r can be determined by the change in the obtained eigenvalue λ_{new} from the previous step λ_{old} , that is

$$r = \lambda_{\text{old}} - \lambda_{\text{new}} \tag{1.34}$$

Although there are various different RL approaches that might be utilised to learn the optimal state-action value function $Q(s, a)$. Our focus is on the Q-learning approach because of its simplicity and ability to learn the optimal policy while following a different exploration policy. The learned function $Q(s, a)$ in Q-learning returns the value of performing a particular action a out of a state s. For a given set s of determinants that make up the current sCI space, possible actions a are removing the p-th determinant from the current set and replacing it with a determinant q that is outside the current sCI space, i.e.,

$$a = (p, q), \text{ for } p \in s \text{ and } q \notin s \tag{1.35}$$

So, once the optimal $Q(s, a)$ is determined, it is easy to acquire the optimal set of determinants from any state s by following the policy(s) for a particular state, i.e.,

$$\pi(s) = \underset{a}{\text{argmax}} Q(s, a) \tag{1.36}$$

Although the ultimate goal is to learn the best policy in Eq. [1.36] by learning the optimal state-action value function $Q(s, a)$, the behavioral policy - that is, the actions taken during training - need not follow the policy that is being learned in Eq. [1.36]. This is undesirable in that the agent will not explore actions which may appear sub-optimal yet may ultimately lead to the global optimal $Q(s, a)$: this is the exploration-exploitation trade off. Q-learning, because it is off-policy has the flexibility to follow physics-inspired and potentially more efficient search policies, all the while ensuring it learns the optimal $Q(s, a)$ and $\pi(s)$.

The method's mathematical and algorithmic details are further discussed in chapter [3].

Bibliography

[1] E. Schrödinger, *An Undulatory Theory of the Mechanics of Atoms and Molecules*, Physical Review **79** (1926), 734.

[2] V. Fock, *Näherungsmethode zur Lösung des quantenmechanischen Mehrkörperproblems.* **61(1)** (1930), 126.

[3] D. R. Hartree., *The Wave Mechanics of an Atom with a Non-Coulomb Central Field.*: *Part I. Theory and Methods.*, Mathematical Proceedings of the Cambridge Philosophical Society **24(1)** (1928), 89.

[4] J. C. Slater, *Note on Hartrees Method.*, Phys. Rev. **35** (1930), 210.

[5] A. Szabo and N. S. Ostlund., *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory.*, McGraw-Hill, New York, 1989.

[6] W. & Sham Kohn L. J, *Self-Consistent Equations Including Exchange and Correlation Effects.*, Vol. 140, 1965.

[7] C. D. Sherrill and H. F. Schaefer, *The Configuration Interaction Method: Advances in Highly Correlated Approaches.*, Vol. 34, 1999.

[8] Per-Olov Löwdin, *Quantum Theory of Many-Particle Systems. III. Extension of the Hartree-Fock Scheme to Include Degenerate Systems and Correlation Effects*, Physical Review **97** (1955), 1509.

[9] R. McWeeny, *Methods of Molecular Quantum Mechanics.*, Academic Press **2** (1992).

[10] J. Paldus, J.Chem. Phys. **61** (1974), 5321.

[11] J. C. Slater, Phys. Rev. **34** (1929), 1293.

[12] E. U. Condon, Phys. Rev. **36** (1930), 1121.

[13] J. C. Slater, Phys. Rev. **38** (1931), 1109.

[14] P. O. Löwdin, Phys. Rev. **97** (1955), 1474.

[15] I. Shavitt, *Int. J. Quantum Chem.*, Symp **61** (1978), 5321.

[16] J. Olsen, *The method of second quantization*: *in Lecture Notes in Quantum Chemistry: European Summer School in Quantum Chemistry, edited by B. 0. Roos*, Vol. 58 of Lecture Notes in Chemistry, pages 37-87, Springer-Verlag, New York, 1992.

[17] R. J. Bartlett, *Many-Body Perturbation Theory and Coupled Cluster Theory for Electron Correlation in Molecules*, Annu. Rev. Phys. Chem **32** (1981), 359.

[18] Laimutis Bytautas and Klaus Ruedenberg, *A priori identification of configurational deadwood*, Chemical Physics **356(1-3)** (2009), 65-75.

[19] Farnaz Heidar-Zadeh James S.M. Anderson and Paul W. Ayers, *Breaking the curse of dimension for the electronic schrodinger equation with functional analysis*, Computational and Theoretical Chemistry (2018).

[20] Charles F. Bender and Ernest R. Davidson, *Studies in configuration interaction: the first-row diatomic hydrides*, Phys. Rev. (1969).

[21] J. L. Whitten and Melvyn Hackmeyer, *Configuration interaction studies of ground and excited states of polyatomic molecules. i. the CI formulation and studies of formaldehyde*, The Journal of Chemical Physics (1969).

[22] W. Duch and G. H. F. Diercksen, *Size-extensivity corrections in configuration interaction methods*, J. Chem. Phys. **104(4)** (1994), 3018.

[23] P. R. Taylor., *Coupled-cluster Methods in Quantum Chemistry*: *In B. O. Roos, editor, Lecture Notes in Quantum Chemistry II: European Summer School in Quantum Chemistry, page 125*, Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

[24] S. R. Langhoff and E. R. Davidson, *Configuration interaction calculations on the nitrogen molecule*, Int. J. Quantum Chem. **8** (1974), 61.

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

[26] Joshua J. Goings and Hang Hu and Chao Yang and Xiaosong Li, *Reinforcement Learning Configuration Interaction*, Journal of Chemical Theory and Computation **17** (2021), no. 9, 5482–5491, DOI 10.1021/acs.jctc.1c00010.

[27] Li Zhou and Lihao Yan and Mark A. Caprio and Weiguo Gao and Chao Yang, *Solving the k-sparse Eigenvalue Problem with Reinforcement Learning*, arXiv physics.comp-ph (2020).

[28] R. S.; Barto Sutton, *Reinforcement Learning*: *An Introduction*, MIT press, 2018.

2

Implementation of Configuration Interaction Method

In this chapter the numerical implementation of the configuration Interaction (CI) method is presented in detail. For the theory behind the method, see Chapter [1]. We start by describing the basic structure of the overall implementation, and afterwards present specific details.

A huge part of the workload was the development of code for CI Hamiltonian construction from determinants and Molecular orbital integrals. The most important aspects of developing scientific software are development speed, maintainability and performance. For performance and maintainability the Rust was chosen as the programming language. Rust is a highly efficient, low level language with guarantee memory-safety and thread-safety.[1]

For matrix operations and linear algebra the excellent Rust library ndarray-linalg[2] is used. Which calls upon the highly optimized linear algebra functions in LAPACK[3] and BLAS[4].

2.1 Overall Structure of the Code

For all Post-Hartree operations, we need MO integrals (obtained from AO to MO transformation); here, we used Psi4[6] API to call SCF procedure on given molecular geometry and basis set (input). Psi 4 has a MintsHelper class for computing integrals.

Integrals were sent to our program, where we initialized all determinants according to specified CI space truncation (e.g., CIS, CISD ..., FCI). Using

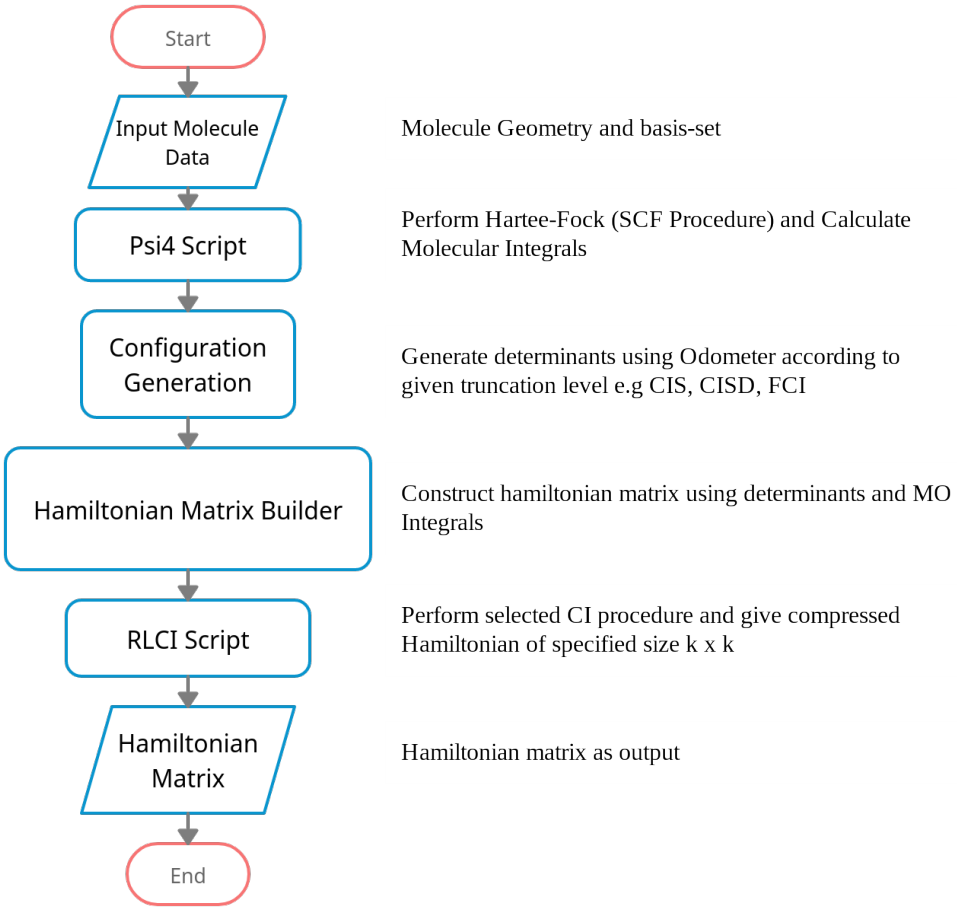


Fig. 2.1: The Basic flow of the program

MO integrals and generated determinants, program construct CI Hamiltonian Matrix which then sent to RLCI Script for selected CI procedure. RLCI returns a smaller Hamiltonian as output which is more tractable computationally; we can then direct diagonalize the matrix for energies or sent it to another program for time-propagation.

2.2 A description of the Basic Functionality

It is critical for performance that some basic operations are carried out efficiently. Notably, the computing of Hamiltonian matrix elements. This raises

significant concerns regarding the data structures employed to describe the two-electron integrals and determinants, as well as their algorithmic implications.

In the following subsections we discuss the representation of determinants and implementation of odometer along with a class of bit-array algorithm essential for construction of Hamiltonian matrix.

2.2.1 Representation of determinants

For many-body wave functions, product wavefunctions are the most straightforward choice.

$$\Psi\left(x_1, x_2, x_3, \ldots, x_A\right) \approx \phi_1\left(x_1\right) \phi_2\left(x_2\right) \phi_3\left(x_3\right) \ldots \tag{2.1}$$

We need antisymmetric wavefunctions since we have fermions, for example.

$$\Psi\left(x_1, x_2, x_3, \ldots, x_A\right) = -\Psi\left(x_2, x_1, x_3, \ldots, x_A\right) \tag{2.2}$$

etc. This is accomplished formally by using the determinantal formalism

$$\Psi\left(x_1, x_2, \ldots, x_A\right) = \frac{1}{\sqrt{A!}} \det \begin{vmatrix} \phi_1\left(x_1\right) & \phi_1\left(x_2\right) & \ldots & \phi_1\left(x_A\right) \\ \phi_2\left(x_1\right) & \phi_2\left(x_2\right) & \ldots & \phi_2\left(x_A\right) \\ \vdots & & & \\ \phi_A\left(x_1\right) & \phi_A\left(x_2\right) & \ldots & \phi_A\left(x_A\right) \end{vmatrix} \tag{2.3}$$

Product wavefunction + antisymmetry = Slater determinant.
The Pauli principle is built in determinant properties (the exchange of any two rows or columns results in a change in sign; consequently, no two rows or columns can be the same).
However, Determinants beyond N = 4 quickly become unmanageable in practise. To simplify calculations, we use the occupation representation or second quantization.

A Slater Determinants can be written in Fock space as a string of creation operators applied to the vacuum state $|0\rangle$.

$$|\Phi_\alpha\rangle = |\phi_{\mathrm{w}_1} \cdots \phi_{\mathrm{ax}}\rangle \tag{2.4}$$

Also be written

$$|\Phi_{\mathrm{a}}\rangle = a^{\dagger}_{\mathrm{a}_1} \cdots a^{\dagger}_{\mathrm{a}_N} |0\rangle \tag{2.5}$$

where $|0\rangle$ is called vacuum state and $a^{\dagger}$ is a creation operator. When a creation operator $a_i^{\dagger}$ acts on a determinant, it fills state i in the Slater determinants Fock space:

$$a_i^{\dagger}|0\rangle = |\phi_i\rangle \tag{2.6}$$

If the state is already occupied, the operation returns zero:

$$a_i^{\dagger}\,|\phi_i\rangle = 0 \tag{2.7}$$

To remove states we introduce the annihilation operator, a.

$$\begin{aligned} a_i \mid \phi_j\} &= \begin{cases} |0\rangle \\ 0, \quad \text{for } i \neq j, \end{cases} \\ a_i|0\rangle &= 0. \end{aligned} \tag{2.8}$$

The use of creation and annihilation operators to express states is called second quantization. Here, because of the fermionic nature of electrons, a permutation of two contiguous creation operators results in a change of sign,

$$\begin{aligned} a_i\,|\Phi_{a,b,\ldots,\ldots}\rangle &= \langle -1)^{n_p}\,|\Phi_{a,b,\ldots}\rangle \\ a_i^{\dagger}\,|\Phi_{u,b,\ldots,}\rangle &= (-1)^{n_p}\left|\Phi_{a,\hat{b},-,\ldots,\ldots}\right\rangle \end{aligned} \tag{2.9}$$

where n_p is the number of filled states before i.

Occupation Number Representation

The occupation number representation is a method of describing a Slater determinant that uses a string of numbers that are either zero or one. If position I is set to one, the ith orbital is occupied. An representation of a 4-particle determinant:

$$\Phi_{3,6,10,13} = |0001001000100100\rangle$$

Such a representation is very convenient when working with computers due to the binary representation of numbers. We use arrays of bit-packed 64-bit (128-bit and 512-bit, depending on need) unsigned integers provided by the "bit-array" crate. Before choosing such data structure from a performance

point of view, some benchmark was done between possible alternatives using XOR operation as CI algorithm is heavily used XOR to find the difference between states. Bit array implementation outperforms boolean-based arrays primarily due to its fast fetch time and cache efficiency despite its CPU overhead in indexing.

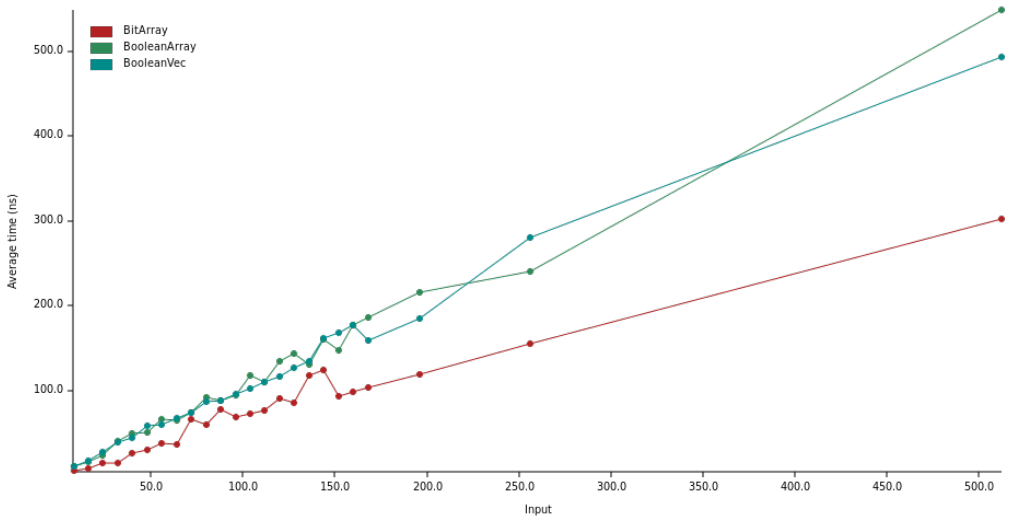
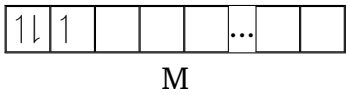


Fig. 2.2: Relative latency in XOR operation between two determinant vs size of determinant for BitArray, BooleanArray and BooleanVec implementation.

2.2.2 States Generation

To generate configuration space we have to permute N electron to M basis functions. below is a example of ground configuration of system with N = 3, alpha and beta spins can be easily calculated as we know beforehand if the system is for Singlet or Triplet states.



or more clear representation would be to use different position for different spins (using odd-even for alpha and beta respectively.):

This configuration can be expressed in a list with `[1,2]` for alpha and `[1]` for beta. then for each spins we will create excitation states using odometer algorithm. for full configuration space we will just multiply alpha states set with beta states set and represent mixed states in BitArray format.

- FCI use Odometer to generate all possible states.
- CIS and CISD use a straight algorithm as taking only singles and doubles from all possible states are inefficient.

Algorithm 1 Odometer

Require: state, M
Ensure: nextstate

$N = state.len()$
for $j = N, \ldots, 1$ **do**
  if $state[j] < M - N + j$ **then**
    $l = state[j]$
    for $k = j, \ldots, N$ **do**
      $state[k] = 1 + l + k - j$
    end for
  end if
end for
$state[:] = 0$
return state

2.2.3 Phase Factor

Because of the data structure we are using for determinants, we also need to compute a phase(sign) factor. The sign change (or phase factor) from Eq. [2.9] can be computed efficiently using the algorithm 2. .

Algorithm 2 Sign Function

Require: state, i
Ensure: nextstate

Initialize N_p
for $k = 1, 2, \ldots, i$ **do**
    if $state[k]$ **then**
        N_p++
    end if
end for
if N_p is even **then**
return 1
else
return -1
end if

List of additional Functions

To find $\hat{T}$ so that $\hat{T}|I\rangle = |J\rangle$, which yields the i, j, k, l indices of the associated twoelectron integrals. A list of determinant functions is implemented on BitArray.

1. getOrbitalMixedIndexList
2. getUniqueOrbitalsInMixIndexListsPlusSign
3. getCommonOrbitalsInMixedSpinIndexList

Their implementations can be found in source code.[B.1]

2.3 Matrix Representation of the Hamiltonian

The matrix elements $H_{IJ} = \left\langle \Phi_I | \hat{H} | \Phi_J \right\rangle$ can be expressed in terms of one- and two-electron integrals. Most of the matrix elements are zero. We get non-Zero elements when the difference between Configurations is less than or equal to two. By difference, we mean [XOR] between the bitstates.

Algorithm 3 Difference count function

Input: state1 and state2
Output: Difference

```
Let state = state1 XOR state2
return state.count( )/2
```

For difference less than or equal to two we calculate matrix elements are evaluated using Slater's rules (also called the Slater-Condon rules).[1.1.4]

Algorithm 4 Compute Hamiltonian Matrix

Input: states, hspin, vspin, M
Output: H

```
Let nslater = states.len()
initialize H = H_(nslater,nslater)
for m = 1, 2, ..., nslater do
    for n = m ..., nslater do
        Let numuniqueorbitals = totalDiff(states[n], states[m]);
        match numuniqueorbitals
        if numuniqueorbitals = 0 then
            H_(n,m) = calcElementDiffIn_0(states[n],states[m])
            H_(m,n) = calcElementDiffIn_0(states[n],states[m])
        else if numuniqueorbitals = 1 then
            H_(n,m) = calcElementDiffIn_1(states[n],states[m])
            H_(m,n) = calcElementDiffIn_1(states[n],states[m])
        else if numuniqueorbitals = 2 then
            H_(n,m) = calcElementDiffIn_2(states[n],states[m])
            H_(m,n) = calcElementDiffIn_2(states[n],states[m])
        else
            H_(n,m) = 0
            H_(m,n) = 0
        end if
    end for
end for
return H
```

1. Identical Determinants:
Find list l of occupied orbitals in the determinant.

$$\left\langle \Phi_1 | \hat{H} | \Phi_1 \right\rangle = \sum_{\text{m in list l}} h[m][m] + \sum_{\text{[m in list l]}} \sum_{\text{[n in list l]>m}} V[m][n][m][n] \qquad (2.10)$$

2. Determinants that Differ by One Spin Orbital:
Find the unique orbitals m and p in both determinants Find the sign from the position of unique orbitals
Find the list of common orbitals in both determinants lc :

$$\left\langle \Phi_1 | \hat{H} | \Phi_2 \right\rangle = \text{sign} * \left(h[m][p] + \sum_{[n \text{ in list } lc]} V[m][n][p][n] \right) \tag{2.11}$$

3. Determinants that Differ by Two Spin Orbitals:
Find the list of unique orbitals in both determinants, m and n in determinant 1 and p and q in determinant 2 . Find the sign from the position of these unique orbitals using sign fuction[2.2.3],

$$\left\langle \Phi_1 | \hat{H} | \Phi_2 \right\rangle = \text{sign} * (V[m][n][p][q]) \tag{2.12}$$

2.4 Parallelization and Optimizations

The optimised implementation of algorithms presented in this chapter, we are able to efficiently compute the matrix elements of $\hat{H}$, to further improve performance and to leverage multiple core counts of the modern CPUs, code is parallelized.
As each element $H_{IJ} = \left\langle \Phi_I | \hat{H} | \Phi_J \right\rangle$ of the Hamiltonian can be independently calculated, so we can parallelize the outer most loop using parallel iterator of rayon (https://crates.io/crates/rayon) [7].

Memory Limitation

Each element of the Matrix is a double-precision Complex float, so it will take 16 bytes of storage. Memory required to store the Full Matrix of size $(nConf, nConf)$ is :

$$Memory_{GB} = nConf^2 * 16 * 10^{-9}$$

So with 32 GB of RAM, we can store CI Matrix of up to 44,000 Configurations.

Sparse Matrix Format

we can store Hamiltonian matrix in sparse matrix format to deal with memory limitation. we used sprs (https://crates.io/crates/sprs) library its support CSR/CSC matrix format. Matrix operation is slower when using sparse format because of CPU overhead, but we can tackle large matrices.

Bibliography

[1] Jeffrey M. Perkel, *Why scientists are turning to Rust*, Nature **588** (2020), no. 7836, 185–186, DOI 10.1038/d41586-020-03382-2.

[2] Rust-Ndarray, *rust-ndarray/ndarray-linalg: Linear algebra package for rust-ndarray using LAPACK binding*, crates.io.

[3] *LAPACK Users' Guide*, Third, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.

[4] L Susan and Petitet Blackford Antoine and Pozo, *An updated set of basic linear algebra subprograms (BLAS)*, ACM Transactions on Mathematical Software **28** (2002), no. 2, 135–151.

[5] *Intel Math Kernel Library. Reference Manual*, Intel Corporation, 2009.

[6] Justin M. Turney and Andrew C. Simmonett and Robert M. Parrish and Edward G. Hohenstein and Francesco A. Evangelista and Justin T. Fermann and Benjamin J. Mintz and Lori A. Burns and Jeremiah J. Wilke and Micah L. Abrams and Nicholas J. Russ and Matthew L. Leininger and Curtis L. Janssen and Edward T. Seidl and Wesley D. Allen and Henry F. Schaefer and Rollin A. King and Edward F. Valeev and C. David Sherrill and T. Daniel Crawford, *Psi4: an open-source ab initio electronic structure program*, Wiley Interdisciplinary Reviews: Computational Molecular Science **2** (2011), no. 4, 556–565, DOI 10.1002/wcms.93.

[7] Rayon-Rs, *rayon-rs/rayon: Rayon: A data parallelism library for Rust*, crates.io.

3

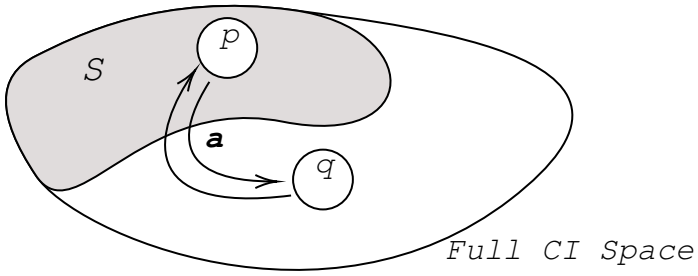
Reinforcement Learning Configuration Interaction

As Hamiltonian size grows bigger when dealing with larger systems, we needed to compress the CI space to account for only the most important determinants. Energetically, we can rank the contribution of a particular determinant using the perturbation theory. Selecting top rank determinants is a good start. Still, we may not get the best possible description as this method only exploits the little-known information about a particular determinant contribution. But with a reinforcement learning algorithm, its exploration-exploitation characteristic can lead to an optimal CI Space.

After detailing the mathematical aspects of the reinforcement learning configuration interaction (RLCI) method and a basic algorithm[1], this chapter presents a modification that accounts for missing energies of excited states in the algorithm. As in the original method, only ground state energy contributions are taken, Hence, poor description of the excited state.

3.1 Reinforcement Learning Method

A basic mapping of selected configuration interaction problem to reinforcement learning term is already mentioned in chapter [1.2]. Continuing the details of $Q(s, a)$ function,



A linear approximation to $Q(s,a) \approx Q_{\mathbf{w}}(s,a)$ is used, as the dimension of $Q(s,a)$ is $\dim(s) \times \dim(a)$ –which is too large to store and utilize directly.

$$Q_{\mathbf{w}}(s,a) = \sum_{i \in N_{\text{det}}} w_i f_i(s,a) = \mathbf{w}^\top \mathbf{f} \tag{3.1}$$

Here, $Q_{\mathbf{w}}(s,a)$ is the inner product between weights w_i and feature vectors $f_i(s,a)$, which is defined below. The following is one possible definition of the feature vectors $f_i(s,a)$, and it is the one utilised in this work.

$$f_i(s,a) = \begin{cases} 1 & \text{if } i \in s \text{ and } i \neq p, \text{ or if } i \notin s \text{ and } i = q \\ -1 & \text{if } i \in s \text{ and } i = p \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

which then reduces Eq [3.1] to a difference in weights w_i,

$$Q_{\mathbf{w}}(s,a) = \sum_{i \in s'} w_i - w_p \tag{3.3}$$

where s' is the active set of determinants subsequent to taking the action $a = (p,q)$. The information in $Q_{\text{w}}(s,a)$ is completely captured in the weights w. Moreover, the weights w, which are independent of s and a, can be interpreted as asserting a ranking for the set of all possible determinants. Eq. [3.2] defines feature vectors as converting the predicted value of an action $a = (p,q)$ to the anticipated change in energy by eliminating determinant p and replacing it with determinant q.

In Q-learning, the weights w are updated at every step according to the conventional update rule [2]

$$w_i \leftarrow w_i + \alpha \frac{dQ_{\mathbf{w}}(s,a)}{dw_i}\delta \tag{3.4}$$

where the Bellman error δ is

$$\delta = r + \gamma \max_{a'} Q_{\mathbf{w}}(s',a') - Q_{\mathbf{w}}(s,a) = r + \gamma \mathbf{w}^\top \mathbf{f}' - \mathbf{w}^\top \mathbf{f} \tag{3.5}$$

and f′ denotes the feature vector after taking the action $a'.\alpha$ is the learning rate and γ is the discount factor, which are user-defined parameters and both should take values in $(0,1]$. Larger values of γ favor rewards in the future, rather than immediate rewards which are given by smaller values of γ.

Although the update in Eq. [3.15] appears to be a gradient descent method, for approximate state-action value functions such as that used in Eq. [3.1], this is no longer the case.[6] To see this, assume that the update in Eq. [3.15]has been mapped onto the general gradient descent expression

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial \mathbf{J}(\mathbf{w})}{\partial w_i} \tag{3.6}$$

where $\mathbf{J}(\mathbf{w})$ is some loss function. In the case of Eq. [3.15], we have

$$\begin{aligned} \frac{\partial \mathbf{J}(\mathbf{w})}{\partial w_i} &= -\frac{\partial Q_{\mathbf{w}}(s,a)}{\partial w_i}\delta \\ &= \frac{\partial \mathbf{w}^\top \mathbf{f}}{\partial w_i}\left(\mathbf{w}^\top \mathbf{f} - \left(r + \gamma \mathbf{w}^\top \mathbf{f}'\right)\right) \\ &= f_i\left(\mathbf{w}^\top \mathbf{f} - \left(r + \gamma \mathbf{w}^\top \mathbf{f}'\right)\right) \end{aligned} \tag{3.7}$$

However, to be a suitable gradient descent method, the second derivatives must be symmetric. Yet for approximate $Q_{\mathrm{w}}(s,a)$ it can be shown that this is not the case:

$$\frac{\partial^2 \mathbf{J}(\mathbf{w})}{\partial w_j \partial w_i} = f_i\left(f_j - \gamma f_j'\right); \quad \frac{\partial^2 \mathbf{J}(\mathbf{w})}{\partial w_i \partial w_j} = f_j\left(f_i - \gamma f_i'\right) \tag{3.8}$$

Therefore, the weights w are not guaranteed to converge in approximate Q-learning. However, it is possible to modify the weight update in Eq. [3.15]

in order to yield a suitably convergent learning algorithm. To this end, we modify the Greedy-GQ method of Ref [4] , which relies on minimizing not the Bellman error directly, but rather the Bellman error projected onto the basis of feature vectors. This is to compensate the gradient due to the lack of an incomplete basis.

Therefore, the loss function $\mathbf{J}(\mathbf{w})$ is chosen to minimize the projected Bellman error, rather than the Bellman error (Eq. [3.1]) itself[2][4]

$$\begin{aligned}\mathbf{J}(\mathbf{w}) &= \|\mathbf{I}\cdot\delta\|^2 \\ &= \left(\delta\cdot\mathbf{f}^\top\right)\left(\mathbf{f}\cdot\mathbf{f}^\top\right)^{-1}(\delta\cdot\mathbf{f})\end{aligned} \tag{3.9}$$

which leads to the gradient expression

$$\begin{aligned}\frac{1}{2}\frac{\partial\mathbf{J}(\mathbf{w})}{\partial w_i} &= \delta\cdot(\gamma\mathbf{f}'-\mathbf{f})\cdot\mathbf{f}^\top\left(\mathbf{f}\cdot\mathbf{f}^\top\right)^{-1}\mathbf{f} \\ &= -\delta\cdot\mathbf{f}\cdot\mathbf{f}^\top\left(\mathbf{f}\cdot\mathbf{f}^\top\right)^{-1}\mathbf{f}+\gamma\mathbf{f}'\cdot\mathbf{f}^\top\left(\mathbf{f}\cdot\mathbf{f}^\top\right)^{-1}\mathbf{f} \\ &= -\delta\cdot\mathbf{f}+\gamma\mathbf{f}'\cdot\mathbf{f}^\top\cdot\mathbf{v}\end{aligned} \tag{3.10}$$

where we define a new vector of auxiliary weights $\mathbf{v}$ in order to avoid the computational overhead of inverting matrices of feature vectors, namely

$$\mathbf{v} = \left(\mathbf{f}\cdot\mathbf{f}^\top\right)^{-1}(\delta\cdot\mathbf{f}) \tag{3.11}$$

The vector $\mathbf{v}$ need not be explicitly formed, and will be learned on-the-fly during the training as will be shown. Thus, with the above results, the new update formula for w is

$$\mathbf{w}\leftarrow\mathbf{w}+\alpha\cdot\left(\delta\cdot\mathbf{f}-\gamma\cdot\left(\mathbf{f}^\top\mathbf{v}\right)\cdot\mathbf{f}'\right) \tag{3.12}$$

and

$$\mathbf{v}\leftarrow\mathbf{v}+\beta\cdot\left(\delta-\mathbf{f}^\top\mathbf{v}\right)\cdot\mathbf{f} \tag{3.13}$$

$\mathbf{v}$ is of the same dimension as $\mathbf{w}$ and may be initialized to zero at the beginning of the RLCI algorithm. The update for $\mathbf{v}$ in Eq. [3.13] is derived from the Least Mean Square (LMS) rule that seeks $\mathbf{v}$ so as to minimize the squared error $\left(\mathbf{f}^\top\mathbf{v}-\delta\right)^2$. This modifies the update scheme to account for the fact that the linear approximation to $Q(s,a)$ is not complete.

3.2 RLCI Algorithm

Here we summarize the algorithm used to generate the results discussed in subsequent sections.

1. From the given full Hamiltonian $H_{(m,m)}$, initialize the set s through the greedy probing algorithm proposed in Ref [3] or we can choose k determinants to form set s randomly too. Summarily, the greedy probing method builds up the set s incrementally through a perturbation-based approach until $\dim(s) = k$, at which the procedure terminates. The additional determinants to be added to s are calculated using the first-order perturbative wave function estimate, which is also employed in other sCI approaches such as CIPSI and ASCI.

$$c_i^{(1)} = \frac{\sum_{j \neq i} H_{ij} c_j^{(0)}}{(E^{(0)} - H_{ii})} \tag{3.14}$$

where H_{ij} are the Hamiltonian matrix elements between determinants i, j, and $E^{(0)}$ is the energy for the Hamiltonian in the current determinant basis indexed by j with eigenvector components $c_j^{(0)}$. At each iteration of the greedy probing algorithm, the determinant i corresponding to the largest value of $\left|c_i^{(1)}\right|$ is added to the set s at each iteration. The technique for selecting determinants does not have to be confined to adding a single determinant at each iteration, and determinants can be added in "batches", adding the 5 determinants with the largest magnitude of $\left|c_i^{(1)}\right|$ to the set s until a dimension of k is reached.

2. Given the initial set s, initialize $\mathbf{w}$ using the magnitude of the eigenvector components for the $k \times k$ Hamiltonian submatrix spanned by the determinants in s. For values of $w_i \notin s$, the initial value is estimated using the Epstein-Nesbet perturbation theory expression in Eq. [3.14]. The values of $w_i \notin s$ and $w_i \in s$, combine to form the initial vector $\mathbf{w}$, are not on the same scale as they are estimated from different procedures. Therefore, each component of w_i (i.e., the component $w_i \in s$ and the component $w_i \notin s$ are each normalized individually to the unit vector and then scaled by its proportion of $\mathbf{w}$. This ensures that the initial values of w_i spanning the full determinant space are on the same scale approximately. The auxiliary vector $\mathbf{v}$ is initialized to zero, following Ref [4] .

3. The behavioural search policy is as follows once it has been initiated. In order to effectively eliminate and expand the active determinant space s, two ranked lists are generated: the first list S_1 contains candidate determinants within the current space to be removed (given by small value in w) and the second list S_2 contains determinants outside the current space s to be added (given by high value in Eq. [3.14]). The dimension of S_1 is the same as the selected subspace k, and nominally the dimension of S_2 is $(N_{\text{det}} - k)$, but for efficiency, S_2 should be limited to the most important external determinants.

$$r = \lambda_{\text{old}} - \lambda_{\text{new}} \tag{3.15}$$

4. Once the action a is decided, the search policy ends and the local reward r is computed according to Eq. [3.15] and the weights w and $\mathbf{v}$ are updated according to Eq. [3.12] and Eq. [3.13], respectively. Upon taking an action $a = (p, q), S_1$ can be updated by removing p and adding q, and the next candidate external determinant in S_2 can be considered. This prevents the need to re-construct S_1 and S_2 at each step in an episode. The episode terminates when once the space is exhausted, or no further suitable candidate actions can be found .

5. These training steps are iterated through until the completion of an episode. To reinitialize the state for a new training episode, the largest k values of w may be used. It is useful to occasionally and randomly initialize with the best s obtained between the training procedure.

Algorithm 5 Reinforcement Learning Configuration Interaction (RLCI)

Input: matrix H , number of determinants k
Output: $H_{(k,k)}$

Initialize learning rate α, discount rate γ, exploration rate τ, weights **w**, and **v**;
for episode in 1,2,. . . ,maxepisode **do**
    Select an initial state s;
    construct S_1 from $i \in s$ with smallest w_i's;
    construct S_2 from $j \in s$ with largest $|c_j|$'s;
    for $j = 1, 2 \ldots, |S_2|$ **do**
        for $i = 1, 2 \ldots, |S_1|$ **do**
            $s' = \left(\left(s \backslash S_{1[i]}\right) \cup S_2[j]\right)$
            Compute smallest eigenvalue λ' of $H\,(i \in s', i \in s')$;
            Generate a random number $\epsilon \sim U(0, 1)$;
            if $\lambda' < \lambda_{best} \cdot (1 - \tau \cdot \epsilon)$ **then**
                Let $p = S_1[i], q = S_2[j]$,
                take action $a = (p, q)$ to get new state s'
                Evaluate local reward r;
                Update **w** and **v**;
            end if
        end for
    end for
end for
return $H\,(i \in s', i \in s')$

3.3 Excited State Consideration

The Eq [3.14] can be re-written as for Excited state n,

$$c_i^{(1^{st})} = \frac{\sum_{j \neq i} H_{ij} c_{(n)j}^{(Zero)}}{(E^{(n)} - H_{ii})} \tag{3.16}$$

Using this equation, we redefined the reward term in RLCI algorithm.

Algorithm 6 RLCI For Excited State (RLCIe)

Input: matrix H , number of determinants k, exicted states to consider n
Output: $H_{(k,k)}$

Initialize learning rate α, discount rate γ, exploration rate τ, weights **w**, and **v**;
for episode in 1,2,. . . ,maxepisode **do**
    Select an initial state s;
    construct S_1 from $i \in s$ with smallest w_i's;
    construct S_2 from $j \in s$ with largest $|c_j|$'s;
    for $j = 1, 2 \ldots, |S_2|$ **do**
        for $i = 1, 2 \ldots, |S_1|$ **do**
            $s' = \left(\left(s \backslash S_{1[i]}\right) \cup S_2[j]\right)$
            Compute n smallest eigenvalue of $H\ (i \in s', i \in s')$;
            Evaluate χ'
            Generate a random number $\epsilon \sim U(0, 1)$;
            if $\chi' < \chi_{best} \cdot (1 - \tau \cdot \epsilon)$ **then**
                Let $p = S_1[i], q = S_2[j]$,
                take action $a = (p, q)$ to get new state s'
                Evaluate local reward r;
                Update **w** and **v**;
            end if
        end for
    end for
end for
return $H\ (i \in s', i \in s')$

Where $\chi = a_0\lambda_0 + a_1\lambda_1 + \ldots + a_n\lambda_n$ and $a_0, a_1, \ldots, a_n$ are parameters tuned by user for desired accuracy trade-off between ground state and excited state energies.

Bibliography

[1] Joshua J. Goings and Hang Hu and Chao Yang and Xiaosong Li, *Reinforcement Learning Configuration Interaction*, Journal of Chemical Theory and Computation **17** (2021), no. 9, 5482–5491, DOI 10.1021/acs.jctc.1c00010.

[2] R. S.; Barto Sutton, *Reinforcement Learning*: *An Introduction*, MIT press, 2018.

[3] T. M.; Van Beeumen Hernandez R.; Caprio, *A Greedy Algorithm for Computing Eigenvalues of a Symmetric Matrix with Localized Eigenvectors.*, Numer. Linear Algebra Appl. (2020), e2341.

[4] Hamid Reza and Szepesvári Maei Csaba and Bhatnagar, *Toward Off-Policy Learning Control with Function Approximation*, Proceedings of the 27th International Conference on International Conference on Machine Learning, 2010, pp. 719-726.

[5] Li Zhou and Lihao Yan and Mark A. Caprio and Weiguo Gao and Chao Yang, *Solving the k-sparse Eigenvalue Problem with Reinforcement Learning*, arXiv physics.comp-ph (2020).

[6] E. Barnard, *Temporal-difference Methods and Markov Models*, EEE Trans. Syst. Man Cybern. (1993).

4

Results and Benchmarks

In this chapter, the produced program has been compared with the outcomes of Psi4 and Psi4Numpy. The parallel efficiency of the matrix builder program has also been investigated. Furthermore, excited state energies are compared at different levels of truncation and RLCI.

4.1 CI Code Validation and Performance

In order to validate the code, the output should match with an existing program with enough precision. We have used Psi4's Configuration Interaction implementation available under Psi4Numpy[1][2] and compared CISD energies of H2O on 3-21G basis set. The output from both programs in table [4.1] matches up to 13 decimal places.

Energies	Our Program	Psi4
0	-83.700550808339386	-83.700550808339301
1	-83.404670828791424	-83.404670828791438
2	-83.373941389977816	-83.373941389977816
3	-83.328260831176323	-83.328260831176365
4	-83.327828332155931	-83.327828332155917
5	-83.305502961188509	-83.305502961188452
6	-83.266883991337153	-83.266883991337110
7	-83.263106353199291	-83.263106353199333
8	-83.206812088270595	-83.206812088270567
9	-83.202856530639806	-83.202856530639735

Tab. 4.1: H2O 3-21G CISD compared to psi4Numpy

The produced code needs to be performant for tackling standard size systems on a routine basis. A considerable part of the workload was optimizing the produced program. To compare performance, we only considered Psi4Numy[2] (as all other implementation are iterative and does not produce a full matrix). This benchmark used an Oxygen atom with CC-pVDZ basis set and recorded CISD matrix construction time. We used a 6 Core AMD Ryzen 5 1600 @ 3.775GHz system to perform this benchmark, and the CISD matrix size is 2221x2221. From table [4.2], we can see that our implementation is more efficient than Psi4Numpy.

	Our Program	**Psi4Numpy**
Configuration Generation Time	250 μs	4 ms
Matrix Building Time	35 ms	15 s

Tab. 4.2: Oxygen cc-pVDZ [3s2p1d] $\rightarrow$ 14 function (CISD)

4.2 Parallel Speedup and Timings

The primary reason for such good performance is the parallelization of the code. The parallel efficiency of the code is demonstrated in table [4.3]. For this benchmark, we used a 48 core Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz and computed CISD matrix of Sulphur atom with an aug-cc-pVDZ basis set. The CISD matrix size is 32985x32985 in this case, So a total of $\sim$ 18 GB RAM is used just to store the generated matrix.

Here are more benchmark of the matrix builder program. In these cases sparse matrix format is used to store the output matrix due to RAM limitation of the system.

4.3 Truncation and RLCI Energies

As system size increases, it is very difficult to deal with their large sparse matrices. So, here we have tested truncation and Selected CI(RLCI) accuracies

Cores	Compute Hamiltonian Matrix(s)
1	33.9
2	17.6
4	9.9
6	6.8
8	5.2
12	3.6
18	2.8
24	2.5
30	2.11
36	1.84
42	1.79
48	1.69

Tab. 4.3: Sulphur CISD aug-cc-pVDZ [5s4p2d] → 27 function

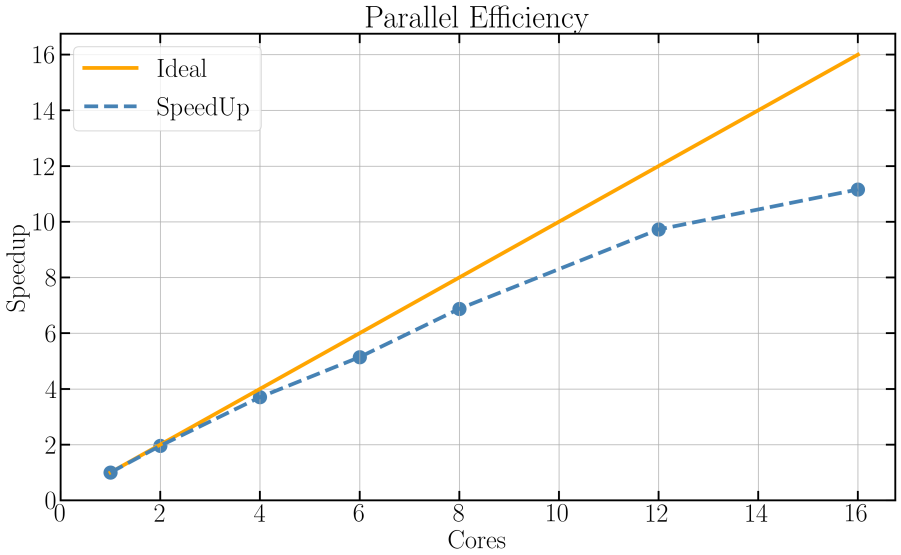


Fig. 4.1: Parallel efficiency of Matrix builder program.

Basis Set	Determinants	Matrix Build Time
cc-pVDZ	12636	371 ms
aug-cc-pVDZ	45361	9.635 s
cc-pVTZ	98316	15 48.83 s

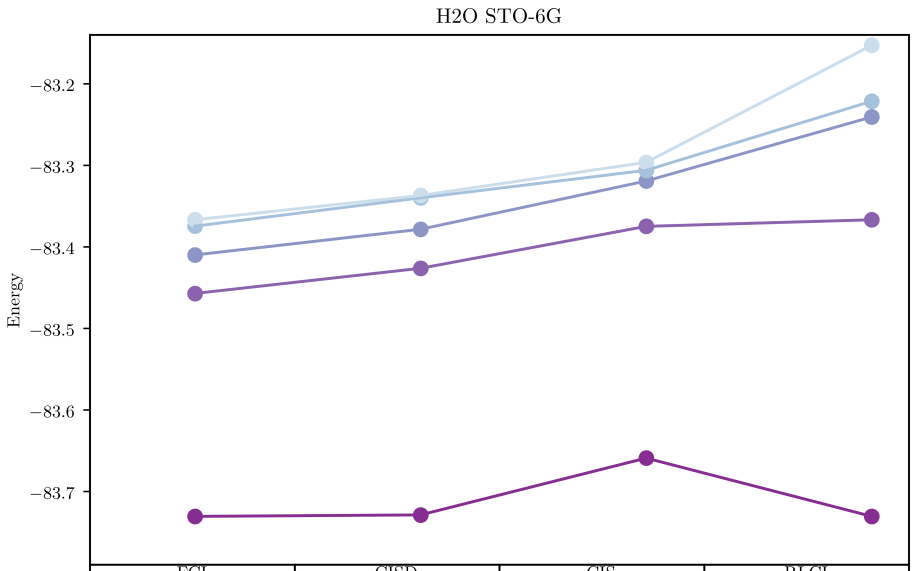
Tab. 4.4: H2O (CISD)

Basis Set	Determinants	Matrix Build Time
cc-pVDZ	20161	911 ms
aug-cc-pVDZ	70876	26.33 s
cc-pVTZ	157116	137.85 s

Tab. 4.5: NH3 (CISD)

for ground state and first few excited states.

H2O (STO-6G) is used as a test case to show the difference between Full CI, Singles, Singles and Doubles, RLCI[3]. Where FCI matrix is of the size 441x441 and the CISD size is 141x141 and for CIS its only 21x21. Here, we have chosen $k = 141$ (same as CISD matrix size) in RLCI procedure, to have the best comparison between both cases.

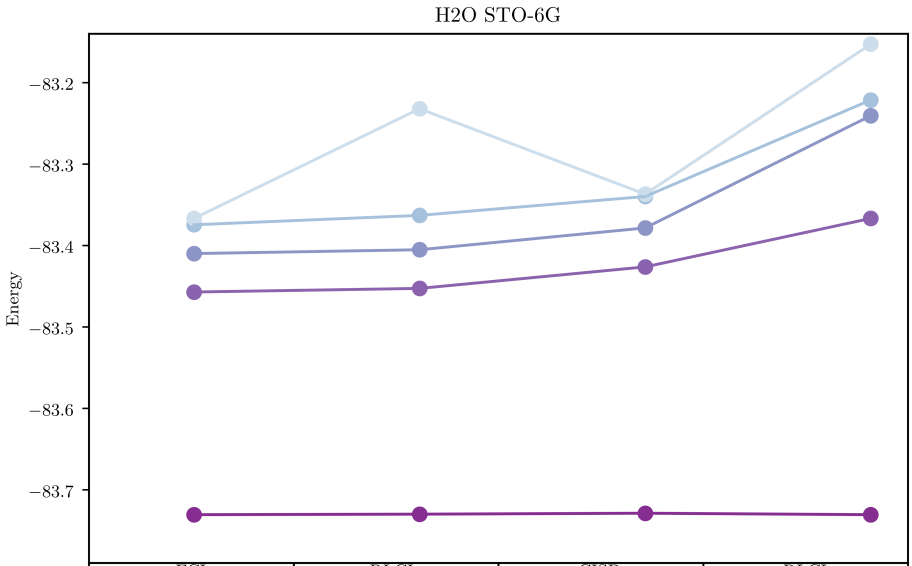


	FCI	CISD	CIS	RLCI
Ground State	-83.73064858705442	-83.72885582306627	-83.6591543777603	-83.73064858705445
1st Excited State	-83.45715919604183	-83.42631129133007	-83.37485689131812	-83.36677961822564
2nd Excited State	-83.40993222019571	-83.37845038230702	-83.31920689760453	-83.24059905619171
3rd Excited State	-83.37457501298181	-83.33984034574272	-83.30609264167188	-83.22131135240478
4th Excited State	-83.36677961822565	-83.33696824101654	-83.29635082620099	-83.15270729531406

Fig. 4.2: Energies for different excited states of H2O (STO-6G) using different levels of truncation and RLCI

RLCI with Excited State Consideration

In table [4.2], RLCI results are way off for excited states, as the original method does not consider excited state contribution in reward function. In section [3.3] improvment of the original method is presented. Using the corrected method (RLCIe) the energies obtained for all the specified states are very close to an FCI calculation.



	FCI	RLCIe	CISD	RLCI
Ground State	-83.73064858705442	-83.73001092349575	-83.72885582306627	-83.73064858705445
1st Excited State	-83.45715919604183	-83.45272065799396	-83.42631129133007	-83.36677961822564
2nd Excited State	-83.40993222019571	-83.40524053524487	-83.37845038230702	-83.24059905619171
3rd Excited State	-83.37457501298181	-83.36301899012997	-83.33984034574272	-83.22131135240478
4th Excited State	-83.36677961822565	-83.23205732435953	-83.33696824101654	-83.15270729531406

Fig. 4.3: Energies for different excited states of H2O (STO-6G) using FCI, RLCIe, CISD, RLCI(original)

The energy of the 4th excited state in RLCIe is off. because we have given the input as $n = 4$ in $\chi = a_0\lambda_0 + a_1\lambda_1 + \ldots + a_n\lambda_n$ with $a_0 = 1.0, a_1 = 0.8, a_2 = 0.6, a_3 = 0.4$. The produced RLCIe script is flexible with respect to the excited states and their accuracy.

Bibliography

[1] Justin M. Turney and Andrew C. Simmonett and Robert M. Parrish and Edward G. Hohenstein and Francesco A. Evangelista and Justin T. Fermann and Benjamin J. Mintz and Lori A. Burns and Jeremiah J. Wilke and Micah L. Abrams and Nicholas J. Russ and Matthew L. Leininger and Curtis L. Janssen and Edward T. Seidl and Wesley D. Allen and Henry F. Schaefer and Rollin A. King and Edward F. Valeev and C. David Sherrill and T. Daniel Crawford, *Psi4: an open-source ab initio electronic structure program*, Wiley Interdisciplinary Reviews: Computational Molecular Science **2** (2011), no. 4, 556–565, DOI 10.1002/wcms.93.

[2] Daniel G. A. Smith and Lori A. Burns and Dominic A. Sirianni and Daniel R. Nascimento and Ashutosh Kumar and Andrew M. James and Jeffrey B. Schriber and Tianyuan Zhang and Boyi Zhang and Adam S. Abbott and Eric J. Berquist and Marvin H. Lechner and Leonardo A. Cunha and Alexander G. Heide and Jonathan M. Waldrop and Tyler Y. Takeshita and Asem Alenaizan and Daniel Neuhauser and Rollin A. King and Andrew C. Simmonett and Justin M. Turney and Henry F. Schaefer and Francesco A. Evangelista and A. Eugene DePrince and T. Daniel Crawford and Konrad Patkowski and C. David Sherrill, *Psi4NumPy: An Interactive Quantum Chemistry Programming Environment for Reference Implementations and Rapid Development*, Journal of Chemical Theory and Computation **14** (2018), no. 7, 3504–3511, DOI 10.1021/acs.jctc.8b00286.

[3] Joshua J. Goings and Hang Hu and Chao Yang and Xiaosong Li, *Reinforcement Learning Configuration Interaction*, Journal of Chemical Theory and Computation **17** (2021), no. 9, 5482–5491, DOI 10.1021/acs.jctc.1c00010.

5

Conclusion

The quantum many-body problem, which is one of the most challenging problems in quantum chemistry, has only been solved for a limited number of test systems. Although Hartree-Fock is used as an approximation for the intractable problem, it fails to address the electron correlation due to many-body interactions. Strongly correlated systems call for methods that take into account correlation energy, and the exact numerical many-body method known as Configuration Interaction has been explored. CI uses various configurations in order to allow for spatial correlations in electron motion.

Due to FCI's severely unfavorable scalability, its implementation needed to be performant for tackling standard-size systems on a routine basis. So, the implementation of CI is done with numerous Code Optimizations, and its specific details are present in chapter [2]. From the benchmark in section [4.2], we conclude that the produced program is highly efficient in building CI matrices. As system size increases, the size of CI matrices increases factorially, and it is hard to deal with such large matrices even with supercomputers. So, truncation and selected CI are explored to reduce the CI space. In section [4.3], a comparison is done between different truncation levels, and energies of excited states are compared with a new selected CI method that uses reinforcement learning. The ground state energy of the RLCI procedure is comparable to an FCI calculation, but it is way off for excited states. Hence, in section [3.3], a correction to the original method is discussed, and results from the corrected procedure are shown in section [4.3]. Presented RLCIe script gives control over which states the user wants to include in the selection procedure and with what accuracy. Program and scripts, produced in this thesis, adds another tool to the sCI methods and can potentially help develop new methods to solve challenging problems in electronic structure theory.

Future Avenues

1. When non-orthogonal orbitals are used, the more difficult Lowdin rules[1] are used rather than Slater's rules[2][3]. So, implement Lowdin rules in matrix builder program.
2. Improve Q-Learning approximate function used in RLCI with deep neural network and further improve overall script implementation.
3. Build a Time-Dependent Configuration Interaction program using (t, t') method.[4]

Bibliography

[1] Per-Olov Löwdin, *Quantum Theory of Many-Particle Systems. III. Extension of the Hartree-Fock Scheme to Include Degenerate Systems and Correlation Effects*, Physical Review **97** (1955), 1509.

[2] J. C. Slater, Phys. Rev. **34** (1929), 1293.

[3] E. U. Condon, Phys. Rev. **36** (1930), 1121.

[4] Prashant Raj and Alkit Gugalia and P. Balanarayan, *Quantum Dynamics with Explicitly Time-Dependent Hamiltonians in Multiple Time Scales: A New Algorithm for (t, tı) and (t, tı, t) Methods in Laser–Matter Interactions*, Journal of Chemical Theory and Computation **16** (2019), no. 1, 35–50, DOI 10.1021/acs.jctc.9b00863.

A

Appendix

A.1 Second Quantization

A Slater determinant is written in Fock space as

$$|\Phi_\alpha\rangle = |\phi_{\mathrm{w}_1} \cdots \phi_{\mathrm{ax}}\rangle$$

Such a Slater determinant can also be written

$$|\Phi_\mathrm{a}\rangle = a^\dagger_{\mathrm{a}_1} \cdots a^\dagger_{\mathrm{a}_N}|0\rangle$$

where $|0\rangle$ is called the vacuum state and $a^\dagger$ is a creation operator. The index α_1 refers to a set of quantum numbers used to describe a single-particle state, and α is the collection of all quantum numbers used in the Slater determinant. $\alpha = \alpha_1, \cdots, \alpha_N$, Such operators are very similar to the excitation operators we encountered when solving the Schrdinger equation for the harmonic oscillator potental for one particle, but instead of increasing the energy, it now creates a state. When a creation operator $a_i^\dagger$ acts on a Slater determinant, it fills state i in the Slater determinants Fock space:

$$a_i^\dagger|0\rangle = |\phi_i\rangle$$

If the state is already occupied, the operation returns zero:

$$a_i^\dagger\,|\phi_i\rangle = 0$$

To remove states we introduce the annihilation operator, a.

$$a_i \mid \phi_j\} = \begin{cases} |0\rangle \\ 0, \quad \text{for } i \neq j, \end{cases}$$

$$a_i|0\rangle = 0.$$

The creation and annihilation operators are related by being their Hermitian conjugates:

$$a_i^\dagger = (a_i)^\dagger$$

The following anti-commutation rules apply to the creation and annibilation operators:

$$\left\{a_i^\dagger, a_j\right\} = \delta_{ij}$$
$$\left\{a_i^\dagger, a_j^\dagger\right\} = 0$$
$$\{a_i, a_j\} = 0$$

The use of creation and annihilation operators to express states is called second quantization. For a state containing more than one particle we must be careful with the ordering of creation and annihilation operators, due to the commutations between the operators. Using the commutation rules governing creation and annihilation operators we find that,

$$a_i \left|\Phi_{a,b,\ldots,\ldots}\right\rangle = \langle -1)^{n_p} \left|\Phi_{a,b,\ldots}\right\rangle$$
$$a_i^\dagger \left|\Phi_{u,b,\ldots,}\right\rangle = (-1)^{n_p} \left|\Phi_{a,\hat{b},-,\ldots,\ldots}\right\rangle$$

where n_p is the number of filled states before i. This can be shown by writing, $|\Phi_{a,b,\ldots,i,-}\rangle$ in terms of creation operators and employing the cothmutation rules given above.

A.2 The Hamiltonian Operator in Second Quantization

We start by splitting the Hamiltonian in a one- and two-body part:

$$\hat{H} = \hat{H}_0 + \hat{H}_I$$

where $\hat{H}_0$ is the one-body Hamiltonian and $\hat{H}_I$ is the two-body Hamiltonian. In second quantization the one-body operator, $\hat{H}_0$ is written as

$$\hat{H}_0 = \sum_{pq} \left\langle \phi_p | \hat{h} | \phi_q \right\rangle a_p^\dagger a_q$$

The matrix elements are typically evaluated as integrals in position space:

$$\left\langle \phi_p | \hat{h} | \phi_q \right\rangle = \int \phi_p^\dagger(\mathbf{r}) h(\mathbf{r}) \phi_q(\mathbf{r}) d\mathbf{r}.$$

For a two-body operator, $\hat{V}_I$, the second quantization form is

$$\hat{V}_I = \frac{1}{2} \sum_{pqrs} \left\langle \phi_p \phi_q | \hat{V} | \phi_r \phi_s \right\rangle a_p^\dagger a_q^\dagger a_s a_r$$

with the matrix elements being

$$\left\langle \phi_p \phi_q | \hat{V} | \phi_r \phi_s \right\rangle = \iint \phi_p^\dagger(\mathbf{r}_1) \, \phi_q^\dagger(\mathbf{r}_2) \, V(\mathbf{r}_1, \mathbf{r}_2) \, \phi_r(\mathbf{r}_1) \, \phi_s(\mathbf{r}_2) \, d\mathbf{r}_1 d\mathbf{r}_2$$

if evaluated in position space. The full Hamiltonian is written as

$$\hat{H} = \sum_{pq} \left\langle \phi_p | \hat{h} | \phi_q \right\rangle a_p^\dagger a_q + \frac{1}{2} \sum_{pqrs} \left\langle \phi_p \phi_q | \hat{V} | \phi_r \phi_s \right\rangle a_p^\dagger a_q^\dagger a_s a_r$$

B

Appendix

B.1 Codes

The code developed as part of this thesis is found at `https://github.com/Ojas-Singh/TDCI` under a MIT License.

B.2 Compiler and Packages Used

Compiler used are rustc 1.57.0-nightly and Packages used can be found in cargo.toml file.

```
[dependencies]
colored = "2.0.0"
num = "0.4.0"
num-iter = "0.1.42"
typenum = "1.14.0"
bit-array = "0.4.4"
rayon = "1.5.1"
mimalloc = { version = "0.1.26", default-features = false }
sysinfo = "0.19.2"
combination = "0.1.5"
ndarray = "0.15.3"
ndarray-linalg = {git = "https://github.com/Ojas-Singh/ndarray-linalg",
                  branch = "master" , features = ["intel-mkl-static"] }

[dependencies.pyo3]
version = "0.14.5"
features = ["auto-initialize"]
```

B.3 Compilation of code

```
git clone https://github.com/Ojas-Singh/TDCI.git
cd TDCI/
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
rustup default nightly
export RUSTFLAGS="-C target-cpu=native"
cargo build --release
```

executable binary under /target/release/TDCI

To set Threads limit set environment variable `RAYON_NUM_THREADS` to number of desired threads. eg for 12 core CPU

```
export RAYON_NUM_THREADS="12"
```