UNIVERSITY OF LEEDS

Development of Methodology, Functionality and Optimisation Tools for the Fermionic "Zombie" Coherent State Method

Oliver Aidan Bramley

Submitted in accordance with the requirements for the degree of
Doctor of Philosophy

in the

Faculty of Engineering and Physical Sciences
School of Chemsitry

September 2024

Declaration of Authorship

The candidate confirms that the work submitted is their own, except where work which has formed part of jointly authored publications has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

The work in the final section of Chapter 2; Chapter 3; the first two sections of Chapter 4; the first section of Chapter 5 and sections 4 and 5 of Appendix B have appeared in The Journal of Chemical Physics, 2022, Volume 156, Page 174116, by O. A. Bramley, T. J. H. Hele and D. V. Shalashilin. The candidate was responsible for writing the program which generated the results, the running of the simulations, creating the graphs and writing the majority of the text. The contribution of the other authors was the derivation of the equations to generate Zombie states from the vacuum state, the algorithmic algebra for the two electron Hamiltonian and number operator; the idea to use imaginary time propagation and Gram-Schmidt orthogonalisation along with an edit of the paper.

The work in Chapter 4 and Chapter 5 is currently in preparation to be published with the manuscripts authored by O. A. Bramley, T. J. H. Hele and D. V. Shalashilin. The candidate was responsible for writing the program which generated the results, the running of the simulations and creating the graphs and the writing of the text. The candidate independently developed the idea of using Gradient Descent and derived the algorithmic process used. The candidate also developed the use of Gram-Schmidt orthogonalisation with multiple Zombie state wave functions. The other authors have contributed some of the text and editing.

"The electron: may it never be of any use to anybody!"

– J. J. Thompson [1]

[1] A slogan of his Cavendish Laboratory (early 1900s) [1].

Acknowledgements

Firstly, I would like to thank my supervisor Prof. Dmitrii Shalashilin for his continual enthusiasm, support and scientific insight. I am grateful for how much he has taught me about quantum mechanics and numerical methods. He has always shown great confidence and faith in me and my work without which I certainly would not be in the position I am today. He afforded me the freedom to shape the direction of the Zombie states method while providing invaluable guidance and support when necessary.

I am grateful for the assistance from my current and former colleagues in the Quantum and Classical Molecular Dynamics group. Dr. Timothy Hele was very helpful at the beginning of the project when getting to grips with the Zombie states method. Ryan Brook has given me useful advice especially when working through some of the mathematical concepts. Dr. Christopher Symonds was invaluable when I first joined the group, during my Masters project in 2018, particularly when learning how to use Fortran. His MCE code has been an useful example of modular program design and best practise which I have incorporated into my own work.

I am indebted to Harry Tata and Alessio Zakaria who first explained the concept of Gradient Descent to me. They both generously gave their time to answer my various questions and were a useful sounding board as the project went on.

I would like to acknowledge the funding provided by the Engineering and Physical Sciences Research Council and the University of Leeds. I would also like to acknowledge that this work was undertaken on ARC3, part of the High Performance Computing facilities at the University of Leeds.

I would like to acknowledge and thank my parents, Andrea and Ian, for all the support they have given me. Their belief and encouragement that I was capable has helped me through the many years of education. I am grateful to both my siblings who have provided more company over the last four years than any of us probably expected – Eve during the pandemic and Alex over the last year. Finally, I would like to thank my partner Caitlin for all she has done for me throughout the PhD. She has encouraged me during the hardest parts and happily let me talk, at length, in minutiae detail about the project. She has supported me through the writing process, taking up the slack so I could focus on writing, which has made a difficult process so much easier.

UNIVERSITY OF LEEDS

Abstract

Faculty of Engineering and Physical Sciences
School of Chemsitry
Doctor of Philosophy
by Oliver Aidan Bramley

Zombie states are a novel addition to the Coupled Coherent States (CCS) family of methods. These have been successfully used for simulation of the dynamics of various quantum systems which primarily have been bosonic. Usually, such methods would be generalised, to describe fermions, by application of fermionic coherent states. These are constructed using either Grassmann algebra or a Lie group with appropriate topological properties which are not well suited to numeric calculations. Zombie states are constructed as a superposition of "dead" and "alive" electronic states. Antisymmetry is preserved by the addition of a simple sign-change rule to the creation and annihilation operators. CCS methods are characterised by their high chemical accuracy and low computational cost when compared to similar methods. Therefore, Zombie states are intended to maintain these properties for electron dynamic simulations.

This thesis presents developments to the formulation and implementation of the Zombie states method which are verified by application to a range of chemical systems. Firstly, the Zombie states method is formulated using the general coherent state definition; it is shown how this naturally gives rise to the sign-change rule. It is then demonstrated that imaginary time propagation can be used to find exact ground state energies with minimal computational cost, which greatly improves the method's practicality. Smaller basis sets incur lower computational costs. However, an incomplete sized basis set of random Zombie states does not accurately recover the ground state. Thus, necessitating the development of sampling and optimisation techniques. Hence, gradient descent is adopted to find optimal sets of Zombie state amplitudes that can recover the ground state energy using a small basis set. Finally, the Zombie state method is extended to find excited states by incorporation of Gram-Schmidt orthogonalisation. These excited states energies can be found with minimal additional computational cost and with no requirement for a reference state.

Contents

List of Figures

List of Tables

List of Abbreviations

TDSE	**T**ime **D**ependent **S**chrödinger **E**quation
TISE	**T**ime **I**ndependent **S**chrödinger **E**quation
HF	**H**artree **F**ock
DFT	**D**ensity **F**unctional **T**heory
RHF	**R**estricted closed-shell **H**artree-**F**ock
UHF	**U**nrestricted **H**artree-**F**ock,
ROHF	**R**estricted **O**pen-Shell **H**artree-**F**ock
SCF	**S**elf-**C**onsistent **F**ield
MCSCF	**M**ulti-**C**onfigurational **S**elf-**C**onsistent **F**ield
CASSCF	**C**omplete **A**ctive **S**pace **S**elf-**C**onsistent **F**ield
STO	**S**later-**T**ype **O**rbital
GTO	**G**aussian-Type **O**rbital
CGF	**C**ontracted **G**aussian **F**unctions
cc	**C**orrelation-**C**onsistent
HOMO	**H**ighest **O**ccupied **M**olecular **O**rbital
LUMO	**L**owest **U**noccupied **M**olecular **O**rbital
CC	**C**oupled-**C**luster
CCSD	**C**oupled-**C**luster **S**ingles-and-**D**oubles
CCSDT	**C**oupled-**C**luster **S**ingles-**D**oubles-and-**T**riplets
MBPT	**M**any-**B**ody **P**erturbation **T**heory
EOM-CC	**E**quations-**O**f-**M**otion-**C**oupled-**C**luster
CCLR	**C**oupled-**C**luster - **L**inear **R**esponse
CI	**C**onfiguration **I**nteraction
MRCI	**M**ulti-**R**eference **C**onfiguration **I**nteraction
FCI	**F**ull **C**onfiguration **I**nteraction

QMC	**Q**uantum **M**onte **C**arlo
FCIQMC	**F**ull **C**onfiguration **I**nteraction **Q**uantum **M**onte **C**arlo
MCCI	**M**onte **C**arlo **C**onfiguration **I**nteraction
CSF	**C**onfiguration **S**tate **F**unction
CCS	**C**oupled **C**oherent **S**tates
ZS	**Z**ombie **S**tate
MCE	**M**ulti-**C**onfigurational **E**hrenfest
AIMCE	**A**b ***initio*** **M**ulti-**C**onfigurational **E**hrenfest
BCH	**B**aker-**C**ampbell-**H**ausdorff
MCTDH	**M**ulti-**C**onfigurational **T**ime-**D**ependent **H**artree
MCTDHF	**M**ulti-**C**onfigurational **T**ime-**D**ependent **H**artree-**F**ock
GD	**G**radient **D**escent
GSO	**G**ram-**S**chmidt **O**rthogonalisation
OMP/OpenMP	**O**pen **M**ulti-**P**rocessing
MPI	**M**essage **P**assing **I**nterface
SIMD	**S**ame **I**nstruction **M**ultiple **D**ata

Notation

Throughout this thesis the following conventions are used:

a,b	Basis function, index given in superscript in parentheses, i.e. $\zeta^{(a)}$
i,j,k,l	Orbital index, given as subscript, i.e. ζ_i.
m,n	Wave function index, given as a subscript, i.e. Ψ^m.
ζ	Is a one electron Zombie state representing a single orbital, where $\|\zeta\rangle = a_1\|1\rangle + a_0\|0\rangle$ is a coherent state.
$\boldsymbol{\zeta}$	Is a multi-electron Zombie state where $\|\boldsymbol{\zeta}\rangle = \|\zeta_1\zeta_2\ldots\zeta_i\rangle$.
N_{xx}	Total number of a parameter for example N_{el} is the total number of electrons and N_{bf} is the total number of basis functions.

All other notation is specified in the text.

For Caitlin

Chapter 1
Introduction

Theoretical simulation of chemical systems is now a well established scientific discipline and widely integrated into many stages of experimental work. Simulations can be used as a predictive tool to shape the direction of new research; as a point of comparison, to verify results, or to find results beyond what is currently experimentally possible. All theoretical work sits within one of three categories: Classical, considering only Newton's laws of motion; Quantum, where all objects are treated on a fully quantum level and Semi-classical, which treats some particles classically and others as quantum. For any chemical system there must be a description of its electrons which being quantum particles means, beyond treating them as a fixed negative charge or field, they must be described using quantum mechanics. Thus, it is necessary to find solutions for the Schrödinger equation which becomes increasingly more difficult as the size of the system grows. The so-called curse of dimensionality means that as the system size increases the computational cost of finding solutions grows exponentially. Hence, developing methods with lower scaling and computational cost is an active field of research with a large body of historic work underpinning it. A standard framework for formulating a theoretical method was set out by John Pople in 1973 [2]. These five stages have become the standard in quantum chemistry, forming a key part of his 1998 Nobel Prize lecture, summarised subsequently [3].

1. **Target**

   The model should have a target level of accuracy. Models aiming to be quantitative should be looking to replicate experimental accuracy and energies such as heats of formation or ionization potentials which should be within 1 kcal/-mole.

2. **Formulation**

   The method must be precisely formulated and generalised so it can be applied universally to any system.

3. **Implementation**

   The model has to be implemented in a way that is reasonable in computational time and cost.

4. **Verification**

   The model must be tested against known chemical facts.

5. **Prediction**

   Following verification, the model can then be applied to unknown chemical problems.

Though simply stated no electronic structure method currently manages to fully satisfy all of these conditions either requiring a loss of accuracy to be practical or maintaining exactness which limits the applications due to unreasonable costs. Of course, a method that meets all of these conditions would be ideal but using this framework allows existing methods to be compared fairly and properly assess their inherent trade-offs. The starting point of many discussions of electronic structure theory is the Born-Oppenheimer approximation which allows the electronic and nuclear parts of the energy to be calculated separately. Henceforth, only solutions to the electronic Schrödinger equation need to be considered which is predominately achieved by using either a Hartree-Fock (HF) or post-Hartree-Fock method or Density Functional Theory (DFT).

The Hartree-Fock method is an Ab *initio* method fully defined in the early 1930s [4–6]. The method describes the electrons in a molecule using a so-called Slater determinant to maintain the antisymmetric property of the electrons; early use of Hartree products was demonstrated by both Slater and Fock to not fully satisfy this quality. The set of wave functions describing each electron together give the total energy of the system which is then minimised using the variational method. This is achieved by application of the Self Consistent Field method usually solved iteratively in an algorithmic process to improve the description of each orbital. The mean field of the other particles in the molecule is considered and the energy minimised which improves the orbital being considered. This process is then repeated for each orbital until the energy has been minimised. The scope for using HF beyond the very simplest systems was greatly limited until much faster computers became available in the 1950s. Nonetheless, HF serves as basis for post-Hartree-Fock and related methods. Configuration interaction takes a linear combination of Slater determinants, each representing a different electronic configuration. This reaches the full-CI limit, when a complete set of all possible configurations is used. However, this is computationally expensive for all but the smallest systems. Multi-configurational active space SCF methods try to remedy this by introducing three classifications for spatial orbitals: core, always doubly occupied; active, partially occupied and virtual, always empty. In the complete active space SCF (CASSCF) method a linear

combination of all Slater determinants that describe the possible electron configurations created when the core orbitals are occupied, and the remaining electrons are distributed across the active orbitals. Less computationally expensive post-Hartree-Fock methods have been developed including many-body perturbation theory which adds a small perturbation to the ground state wave function to account for electron correlation. The coupled cluster theory starts with an exponential ansatz for the wave function often using a Hartree-Fock determinant as a reference wave function. However, all of these methods carry a computational cost that scales exponentially with degrees of freedom making them unsuitable for even seemingly modest systems.

Density Functional Theory takes a different approach, mapping the many-body problem onto a single-body one. DFT is centred around two key theorems developed by Hohenberg and Kohn in 1964 [7]. This first states that the ground state electron density uniquely determines the external potential and so the total system energy. Secondly, the ground state energy can be found variationally if an exact energy functional of the electron density can be found. In other words, with the right functional the electron density can be described and this electron density can then describe the system. The functional can be written as a sum of the electron kinetic energy, electron-electron interaction and electron-nucleus interaction. This can be solved once the external potential is known. But the external potential depends on the system and so it is not possible to write a universal expression of the functional equation. Kohn and Sham created a practical framework to carry out DFT calculations using a system of non-interacting particles that gives the same density as the real system of interacting particles [8]. The internal energy functional is written as the sum of the kinetic energy of the fictional non-interacting system; the electron density of this system and the exchange-correlation functional contain the unknown parts of the total energy. The description of the exchange-correlation functional requires density functional approximations which are not theoretically exact and only empirically shown to be effective. So, some would argue it is not a truly Ab *initio* method due to the fitting of some correlation functionals. Further, DFT lacks the systematic quantitative accuracy found in HF based methods. In many cases this is not a problem and DFT has become an extremely popular method used widely across many fields of research. However, DFT not being formulated in a way that is entirely general and HF type methods suffering from large computational costs at high levels of accuracy are problems that are unlikely to be fixed. Therefore, it is worth considering the building of a new theory explicitly designed to avoid these limitations.

Zombie states (ZS) are such a theory, first presented in 2018 by Shalashilin, they offer a novel approach to electronic structure theory that is not based on Hartree-Fock like Slater determinants. The method is constructed to main theoretical exactness while also utilising electronic basis sets like HF methods to maintain a hi-

erarchy of quantitative accuracy [9]. Zombie states like much of Shalashilin and his group's work has been based around the Coupled Coherent States (CCS) method [10]. Coherent states are an appealing quantum object, allowing system states to be described as a superposition of constituent states which can be a better reflection of the quantum nature of the overall system. This property was first applied to electromagnetic radiation, describing photons using coherent states of the quantum harmonic oscillator [11, 12]. In CCS the wave function is described using a basis of coherent states which are coupled via time propagation equations. This was then generalised to multiple electronic states in the form of the Multi-Configurational Ehrenfest (MCE) method. The first version of MCE showed excellent agreement with model system values [13]. A second version of MCE was developed to separate the inter- and intra-function coupling. This gave rise to the Ab *initio* Multi-Configurational Ehrenfest (AIMCE) method, which could be be propagated along a potential energy surface calculated "on-the-fly", allowing the simulation of real systems [14]. However, the change of ansatz necessitated the development of multiple sampling techniques so MCEv2 could replicate results, for model systems, that the first version of MCE had previously achieved [15]. Most work with CCS and MCE has focused on distinguishable systems particularly with bosons but some initial work showed CCS could be adapted to deal with fermion dynamics [16].

The electronic ground state of a molecule is never exactly a single configuration due to the constant movement of electrons in a chemical system. Hence, why the exact, full configuration interaction, wave function is constructed using a linear combination of all configurations. This motivates use of coherent states to describe a fermionic system – a supposition of coherent states replacing the linear combination of configurations. Thus, the Zombie states method is proposed to describe a fermionic system using a wave function of coherent states, that are superpositions of all possible electron configurations. Each spin orbital in a Zombie coherent state is described by a superposition of the occupied and unoccupied state. So the Zombie wave function consists of functions that are, by definition, made up of all configurations. This should make it possible to describe systems with the accuracy of a full-CI calculation but without the computationally prohibitive size. Further, the existing apparatus developed for CCS and MCE should then be easily applicable to fermionic systems. This would hopefully allow accurate simulation of dynamics while replicating the lower computational cost CCS type methods have demonstrated when compared to similarly accurate methodologies [13, 17]. Fermionic coherent states are not a new concept but are typically used to describe a specific electronic configuration as a superposition of spin states [18, 19]. There are two ways these fermionic coherent states are constructed, using either Grassmann algebra or the Gilmore Perelomov general coherent state definition [20–22]. Elements of Grassmann algebra are mathematical objects constructed to anticommute with

each other which ensures the antisymmetric property of the fermions is maintained. Gilmore Perelomov's general method, on the other hand, utilises Lie groups with an appropriate topological structure to define and maintain the necessary properties of the coherent states. However, neither construction lends itself to numeric simulations. The general coherent state definition allows all coherent states to be constructed in the same manner which can be used to contextualise Zombie states when compared to other types of coherent states. Using this definition, it is possible to show how Zombie states are constructed using SU(2) coherent states and how this naturally maintains the antisymmetric property and behaviour during creation and annihilation operations.

The primary aim of this thesis is to detail the work undertaken to further develop the Zombie states method. Within Pople's framework this work aims to increase the universality of the method and verify this by benchmarking results, for a range of chemical systems, to full-CI energies. Further, the practical implementation of the method is considered to minimise computational expense and execution time. However, the Zombie states method is still in its infancy and will require further work, beyond this, to be used and trusted for prediction. Therefore, this thesis is also intended to serve as single starting point for subsequent development. Hence, additional explanation, particularly of certain mathematical concepts, is given to aid understanding. Chapter 2 will lay out some integral building blocks from the Hartree-Fock method necessary for understanding the ZS method as well as a discussion of some current methods aiming to recover full-CI energies. A general definition for coherent states is given which is used to contextualise all the coherent states used in the CCS family of methods. This is also used to show how fermionic coherent states are constructed and compared to the Grassmann algebra construction. Within this context the Zombie states method is then presented combining the original formulation and building ZS from the vacuum state [9, 23]. Zombie states are then constructed using the general coherent state definition. This makes clear how Zombie states not only compare to other fermionic coherent state constructions but also individual and linear combinations of Slater determinants. In Chapter 3 imaginary time propagation is introduced as the primary method for efficiently optimising ZS coefficients to find the ground state energy of a system. Chapter 4 forms the most significant results of this thesis detailing the development of techniques to reduce the size of the ZS basis set. This starts with simple amplitude biasing which leads to the adoption of the optimisation technique Gradient Descent (GD). The underlying theory and algorithmic developments are presented followed by verification of the method by application to a variety of chemical systems. In Chapter 5 the ZS method is extended to find excited states using Gram Schmidt orthogonalisation. Finally, conclusions and the outlook for future work is presented. The appendices give additional theoretical background information and details of the Zombie states

program. The structure and function of the ZS program is discussed including the development of the algorithms used.

Chapter 2
Background and Theory

2.1 Introduction

All discussions of quantum mechanics naturally start with the Schrödinger equation. A system can be described by a wave function, $|\Psi(t)\rangle$. The use of the bra-ket notation here allows this to be done in terms of a chosen basis, which is often position. The Time-Dependent Schrödinger Equation (TDSE),

$$i\hbar\frac{d}{dt}|\Psi(t)\rangle = \hat{H}|\Psi(t)\rangle \tag{2.1}$$

is used to evolve $|\Psi(t)\rangle$. The system Hamiltonian operator $\hat{H}$ corresponds to the total energy of the system so a complete description of the system is maintained as the wave function is evolved from an initial state at $t = 0$ by,

$$|\Psi(t)\rangle = e^{-i\hat{H}t/\hbar}|\Psi(0)\rangle. \tag{2.2}$$

Through this process the energy states of the system are changed which describes the chemical process being investigated. However, for some systems the energy states are constant with time, such as the energy levels of a molecule at equilibrium. As such the time dependence of the wave function can be removed; it can be written in terms of the basis of energy eigenstates

$$|\Psi(t)\rangle = \sum_n A_n e^{-iE_n t/\hbar}|\Psi_{E_n}\rangle, \tag{2.3}$$

which can be substituted into Eq. (2.2). This can be simply rearranged to give the Time-Independent Schrödinger Equation (TISE)

$$\hat{H}|\Psi\rangle = E|\Psi\rangle. \tag{2.4}$$

E is the exact total energy associated with the eigenstate $|\Psi\rangle$, the $|\Psi_{E_n}\rangle$ notation has been simplified to $|\Psi\rangle$ as the TISE is used exclusively throughout this thesis to

consider molecular electronic states.

The Hamiltonian for a multi-electron system with N_{el} electrons and N_{nu} nuclei can be written as

$$\hat{H} = -\sum_{i=1}^{N_{el}} \frac{1}{2}\nabla_i^2 - \sum_{A=1}^{N_{nu}} \frac{1}{2M_A}\nabla_A^2 - \sum_{i=1}^{N_{el}}\sum_{A=1}^{N_{nu}} \frac{Z_A}{r_{iA}} + \sum_{i=1}^{N_{el}}\sum_{j>i}^{N_{el}} \frac{1}{r_{ij}} + \sum_{A=1}^{N_{nu}}\sum_{B>A}^{N_{nu}} \frac{Z_A Z_B}{R_{AB}} \tag{2.5}$$

where $r_{iA} = |\mathbf{r}_i - \mathbf{R}_A|$ is the distance between nucleus A and the i^{th} electron; r_{ij} is the distance between electrons i and j; R_{AB} is the inter-nucleus distance between nuclei A and B. M_A is the atomic mass of nucleus A and Z_A its charge, the operator is in atomic units. Moving left to right the Hamiltonian contains terms representing the electron kinetic energy, $\hat{T}_e$; the kinetic energy of the nuclei, $\hat{T}_n$; the Coulomb attraction between nuclei and electrons, $\hat{V}_{ne}$; the electron-electron repulsion, $\hat{V}_{ee}$ and the nucleus-nucleus repulsion, $\hat{V}_{nn}$. So, the Hamiltonian can be written as sum of each of these operators' contributions

$$\hat{H} = \hat{T}_e + \hat{T}_n + \hat{V}_{ne} + \hat{V}_{ee} + \hat{V}_{nn} \tag{2.6}$$

It is impossible to solve Eq. (2.4) exactly in all but a few limited cases and so begins the process of applying truncations and approximations to simplify (in relative terms) the problem.

The first and most widely used approximation is the Born-Oppenheimer approximation which separates the nuclear and electronic contributions to the total molecular energy. The mass of the nuclei being so much larger than the electrons means that their motion occurs on different time scales, electrons moving far faster than nuclei if given the same momentum. This allows the nucleus to be given a fixed position i.e. $\hat{T}_n = 0$. The fixed position of the nuclei means the nucleus-nucleus repulsion, V_{nn}, can be considered constant. Thus, the molecular Hamiltonian can then be written as a sum of the nuclear and electronic parts and each part solved independently of each other

$$\hat{H} = -\sum_{i=1}^{N_{el}} \frac{1}{2}\nabla_i^2 - \sum_{i=1}^{N_{el}}\sum_{A=1}^{N_{nu}} \frac{Z_A}{r_{iA}} + \sum_{i=1}^{N_{el}}\sum_{j>i}^{N_{el}} \frac{1}{r_{ij}} + V_{nn}. \tag{2.7}$$

This Hamiltonian can then be used to construct the Schrödinger equation, solutions of which correspond to different energy states of a molecular system.

This chapter is split into three main sections, Hartree-Fock, Coherent states and Zombie states, each focusing on a different way to generate wave functions that can be used to solve the Schrödinger equation made using Eq. (2.7). The Hartree-Fock method is foundational to all electronic structure theory, including the Zombie states method, and so a detailed description of it forms the first part of this chapter. Significantly, the key antisymmetric character of fermionic systems is introduced

through the use of Slater determinant wave functions and parametrised in the second quantisation approach by anticommutation of the creation and annihilation operators. Further, the use of electronic basis sets is introduced to describe the physical shape of the orbitals and also the process for constructing a Hamiltonian matrix using Eq. (2.7). Finally, the self-consistent method is detailed showing how the HF method uses its Slater determinant wave function and basis set to find solutions to the electronic Schrödinger equation. However, the original HF method is not capable of recovering scientifically significant results. Thus, the post-Hartree-Fock methods, Møller-Plesset theory and Coupled cluster theory are also discussed. The configuration-interaction (CI) approach is also explained, introducing the full configuration interaction limit that is used extensively throughout this work to benchmark the Zombie state method. Two methods that aim to recover full-CI energies while using significantly smaller basis sets are full-CI Quantum Monte Carlo (FCIQMC) and Monte Carlo-CI (MCCI) which are a useful point of comparison to the ZS method [24, 25]. Moreover, it will be shown in subsequent chapters that the Zombie states method borrows well tested processes already used in FCIQMC.

Post-Hartree Fock methods have been developed because a single Slater determinant does not exactly describe a state of system, rather a single configuration of electrons in the available spin orbitals. The electronic ground state of a system is not one configuration of electrons but a superposition of multiple. Therefore, a better description of the ground state could be made by using functions that are not a fixed and specific configuration of electrons. Coherent states are such a function and have been an active area of quantum mechanical research since the early 1960s [11]. This initial work was catalogued by Klauder and Skagerstam in Ref. [26] and more recently by Combescure and Didier in Ref. [20]. The Coupled Coherent States (CCS) method and related techniques have been extensively applied to a variety of different theoretical and real-world systems. CCS has been applied to the molecular dynamics of CHD_3 and the absorption spectrum of pyrazine [10, 27]. To allow problems with multiple dimensions to be investigated CCS was generalised to produce the multi-configurational Ehrenfest (MCE) method. Initial verification of the method was achieved through simulations of the model spin-boson system [13, 28]. But it has also been applied to a variety of real molecular systems for example the photodynamics of pyrrole and the excited state dynamics of a phenylene ethynylene dendrimer [29–31]. As CCS and MCE have been developed various sampling techniques have been added to the methods to improve their convergence [14, 15, 32, 33]. CCS methods have primarily focused on distinguishable particles but recently the CCS has been extended to describe indistinguishable bosons by employing second quantisation [34, 35]. Thus, within the context of this previous work it is logical to want to extend the CCS family of methods to also include fermionic systems hence, the novel Zombie states method. By describing fermionic systems

using coherent states the techniques developed in CSS for distinguishable particles can be easily transferred to indistinguishable fermions. Further, CCS methods do not scale exponentially because they utilise Monte Carlo sampling which, theoretically, scales quadratically. This lower scaling means it is possible to simulate much larger systems without incurring such a prohibitively large computational cost while maintaining high accuracy. In fact, the bottle neck in MCE calculations of real molecules is the repeated electronic structure calculations needed for the trajectories to propagate along. Full-CI calculations also scale exponentially with the number of basis functions and electrons, so developing a fermionic coherent state method would hopefully allow similar reductions in computational cost seen for bosons using MCE. Ultimately, this could mean molecular simulations with all particles treated on an equal footing using a CCS type method meaning accuracy and computational costs far lower than what is currently possible.

Therefore, the second part of this chapter focuses on the current standard approach to fermionic coherent states. A general definition for a coherent state is given which forms a framework for describing the coherent states used in CCS and MCE and fermionic coherent states. Some of the mathematical concepts used are possibly novel to a mathematically minded chemist rather than a mathematician so a brief précis is given in Appendix A which should give enough background information to sufficiently understand the section and begin further research. There is a particular focus on $SU(2)$ coherent states because these are used in the MCE method and also underpin the Zombie states method's construction. Fermionic coherent states are then constructed using the general definition. It is also demonstrated how an alternative construction utilises elements of Grassmann algebra to maintain the fermionic antisymmetry property. In the final part of the chapter the Zombie states method is presented demonstrating how ZS functions can be constructed from a vacuum state. Significantly, Zombie states are constructed using the general coherent state definition. The three fermionic coherent state constructions can then be compared highlighting how the constructions can describe analogous states but also their inherent differences.

2.2 Hartree-Fock Theory

2.2.1 Spin

The Hamiltonian in Eq. (2.7) only accounts for the spatial electronic coordinates and so it is necessary to introduce spin. Electrons are fermionic particles and obey the Pauli exclusion principle: no two electrons can occupy the same quantum state. Two spin functions are introduced, up $\alpha(\omega)$ and down $\beta(\omega)$,

$$\langle\alpha|\alpha\rangle = \int \alpha^*(\omega)\alpha(\omega)d\omega = \int \beta^*(\omega)\beta(\omega)d\omega = \langle\beta|\beta\rangle = 1 \tag{2.8}$$

$$\langle\alpha|\beta\rangle = \int \alpha^*(\omega)\beta(\omega)d\omega = \int \beta^*(\omega)\alpha(\omega)d\omega = \langle\beta|\alpha\rangle = 0. \tag{2.9}$$

Eq. (2.9) show that these two functions are complete, i.e. defined for all possible values and are orthonormal, meaning $\alpha(\omega)$ and $\beta(\omega)$ are normalised and orthogonal to each other. These equations also introduce Dirac notation, a ket representing a vector function with the corresponding bra being its complex conjugate. Therefore, each electron is described by three spatial coordinates $\mathbf{r}$ and one spin coordinate ω which can be summarised as $\mathbf{x} = \{\mathbf{r},\omega\}$. Hence, an N-electron system has a wave function that is dependent on $\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N$ that can be written as $\phi(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N)$. To account for the spin "A many-electron wave function must be antisymmetric with respect to the interchange of the coordinate x (both space and spin) of any two electrons"[36]

$$\phi(\mathbf{x}_1,\ldots,\mathbf{x}_i,\ldots,\mathbf{x}_j,\ldots,\mathbf{x}_N) = -\phi(\mathbf{x}_1,\ldots,\mathbf{x}_j,\ldots,\mathbf{x}_i,\ldots,\mathbf{x}_N). \tag{2.10}$$

From this antisymmetric principle comes the adherence to the Pauli exclusion principle.

2.2.2 Hartree Products

A single electron wave function can be constructed as a product of one-particle orbital functions

$$\Psi^{HP}(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N) = \chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2)\ldots\chi_k(\mathbf{x}_N). \tag{2.11}$$

This is the Hartree product and each function, $\chi_i(\mathbf{x})$, depends on both spatial and spin coordinates which can be called spin orbitals [4].

$$\left.\begin{aligned}\chi_{2i-1}(\mathbf{x}) &= \psi_i(\mathbf{r})\alpha(\omega)\\ \chi_{2i}(\mathbf{x}) &= \psi_i(\mathbf{r})\beta(\omega)\end{aligned}\right\} i = 1,2,\ldots,N_{el} \tag{2.12}$$

However, if the number of electrons in the system is increased beyond one, the Hartree product no longer satisfies the antisymmetric principle which was independently pointed out by Slater and Fock [5, 37].

2.2.3 Slater Determinants

Considering the simplest multi-electron case: two electrons and two orthonormal spin orbitals, it is possible to construct two Hartree products

$$\Psi_{12}^{HP}(\mathbf{x}_1,\mathbf{x}_2) = \chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2) \tag{2.13a}$$

$$\Psi_{21}^{HP}(\mathbf{x}_2,\mathbf{x}_1) = \chi_i(\mathbf{x}_2)\chi_j(\mathbf{x}_1). \tag{2.13b}$$

These can be combined linearly, $2^{-1/2}$ ensuring normalisation

$$\Psi_{12}(\mathbf{x}_1,\mathbf{x}_2) = 2^{-1/2}(\chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2) - \chi_i(\mathbf{x}_2)\chi_j(\mathbf{x}_1)). \tag{2.14}$$

Slater showed [38] that this antisymmetric wave function could be written as a determinant

$$\Psi_{12}(\mathbf{x}_1,\mathbf{x}_2) = 2^{-1/2}\begin{vmatrix}\chi_i(\mathbf{x}_1) & \chi_j(\mathbf{x}_1)\\ \chi_i(\mathbf{x}_2) & \chi_j(\mathbf{x}_2)\end{vmatrix}. \tag{2.15}$$

Which can be generalised for an N-electron system

$$\Psi(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N) = (N!)^{-1/2}\begin{vmatrix}\chi_i(\mathbf{x}_1) & \chi_j(\mathbf{x}_1) & \ldots & \chi_k(\mathbf{x}_1)\\ \chi_i(\mathbf{x}_2) & \chi_j(\mathbf{x}_2) & \ldots & \chi_k(\mathbf{x}_2)\\ \vdots & \vdots & & \vdots\\ \chi_i(\mathbf{x}_N) & \chi_j(\mathbf{x}_N) & \ldots & \chi_k(\mathbf{x}_N)\end{vmatrix}. \tag{2.16}$$

Different spin orbitals are in each column and different electrons are in each row. To change the position of an electron in the Slater determinant representation would require the swapping of two columns in the determinant which would introduce a negative sign. A sign change is also introduced if rows are swapped. Thus, the Slater determinant mathematically ensures the preservation of the antisymmetric property when the position of an electron is changed. A normalised Slater determinant can be written succinctly,

$$\Psi(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N) = |\chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2)\ldots\chi_k(\mathbf{x}_N)\rangle \tag{2.17}$$

which is equivalent to the Slater determinant in Eq. (2.16). The normalisation constant is included but not shown and only the diagonal elements are shown for succinctness. Therefore, swapping elements in this notation is the equivalent to swap-

ping columns in the Slater determinant giving the expected antisymmetric property,

$$|\chi_i\chi_j\cdots\chi_k\rangle = -|\chi_j\chi_i\cdots\chi_k\rangle. \tag{2.18}$$

2.2.4 Matrix Elements

The Hamiltonian matrix for Eq. (2.7) is constructed by its operation between two Slater determinants (formed of orthonormal orbitals), $\langle\Psi^{(A)}|\hat{H}|\Psi^{(B)}\rangle$ and can be explicitly written in two parts the first having single electron dependence, $\mathfrak{O}_1$ and the second being dependent on two electrons, $\mathfrak{O}_2$,

$$\hat{H} = \mathfrak{O}_1 + \mathfrak{O}_2 = -(\sum_{i=1}^{N_{el}} \frac{1}{2}\nabla_i^2 + \sum_{A=1}^{N_{nu}} \frac{Z_A}{r_{iA}}) + \sum_{i=1}^{N_{el}}\sum_{j>i}^{N_{el}} \frac{1}{r_{ij}} \tag{2.19}$$

the nuclear-nuclear repulsion has been omitted as it is a constant. It is first necessary to introduce some notation for both the one- and two-electron parts. The one-electron operators are now shown together as "h" and the following bra-ket notation is used to represent an integral,

$$\langle i|h|i\rangle = \langle\chi_i(\mathbf{x}_1)|h|\chi_i(\mathbf{x}_1)\rangle = \int \chi_i^*(1)h(1)\chi_i(1)d\mathbf{x}_1. \tag{2.20}$$

In a similar way the two-electron part uses the bra-ket notation for the integral,

$$\langle ij|ij\rangle = \langle\chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2)|\chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2)\rangle = \int \chi_i^*(1)\chi_j^*(2)\frac{1}{r_{12}}\chi_i(1)\chi_j(2)d\mathbf{x}_1 d\mathbf{x}_2 \tag{2.21}$$

the operator $\frac{1}{r_{ij}}$. For both the one- and two-electron part there are three types of result dependent on the spin orbitals in $|\Psi^{(A)}\rangle$ and $|\Psi^{(B)}\rangle$.

1. $|\Psi^{(A)}\rangle = |\Psi^{(B)}\rangle$ – Slater determinants have identical spin orbitals.
2. $|\Psi^{(A)}\rangle = |\chi_i(1)\chi_j(2)\ldots\rangle$ and $|\Psi^{(B)}\rangle = |\chi_k(1)\chi_j(2)\ldots\rangle$ – Slater determinants differ by one spin orbital only.
3. $|\Psi^{(A)}\rangle = |\chi_i(1)\chi_j(2)\ldots\rangle$ and $|\Psi^{(B)}\rangle = |\chi_k(1)\chi_l(2)\ldots\rangle$ – Slater determinants differ by more than one spin orbital.

The one-electron Hamiltonian part can take the following values:

$$\langle\Psi^{(A)}|\mathfrak{O}_1|\Psi^{(B)}\rangle = \begin{cases} \sum_i^{N_{el}}\langle i|h|i\rangle & \text{Case 1} \\ \sum_i^{N_{el}}\langle i|h|j\rangle & \text{Case 2} \\ 0 & \text{Case 3} \end{cases} \tag{2.22}$$

where h is the one electron operator. Similarly for the two-electron part

$$\langle \Psi^{(A)} | \mathfrak{O}_2 | \Psi^{(A)} \rangle = \begin{cases} 1/2 \sum_i^{N_{el}} \sum_j^{N_{el}} \langle ij|ij\rangle - \langle ij|ji\rangle & \text{Case 1} \\ \sum_j^{N_{el}} \langle ij|kj\rangle - \langle ij|jk\rangle & \text{Case 2} \\ 0 & \text{Case 3} \end{cases} \tag{2.23}$$

A full derivation of these results can be found in Appendix C.1.

2.2.5 Second Quantisation

Second quantisation takes the antisymmetric properties of the electronic wave function – the Slater determinants – and transfers them to the creation and annihilation operators [6, 39, 40]. This gives rise to a modified Hamiltonian equation and allows Slater determinants to just describe orbital occupancy. The one- and two-electron integrals only have to be calculated once before calculating matrix elements. This process is more practical for numerical applications hence, it is subsequently used throughout this thesis. The creation operator, $a_j^\dagger$, is defined on an arbitrary Slater determinant as

$$a_j^\dagger |\chi_k \dots \chi_l\rangle = |\chi_j \chi_k \dots \chi_l\rangle \tag{2.24}$$

$a_j^\dagger$ creates an electron in spin orbital j. The anticommutation relation can then be derived again

$$\begin{aligned} &(a_i^\dagger a_j^\dagger + a_j^\dagger a_i^\dagger)|\chi_k \dots \chi_l\rangle = |\chi_i \chi_j \chi_k \dots \chi_l\rangle + |\chi_j \chi_i \chi_k \dots \chi_l\rangle = \\ &|\chi_i \chi_j \chi_k \dots \chi_l\rangle - |\chi_i \chi_j \chi_k \dots \chi_l\rangle = 0 \Rightarrow \end{aligned} \tag{2.25}$$

$$a_i^\dagger a_j^\dagger + a_j^\dagger a_i^\dagger = 0 = \{a_i^\dagger, a_j^\dagger\}. \tag{2.26}$$

Note it is not possible to create two electrons in the same spin orbital so $a_j^\dagger a_j^\dagger |\chi_k \dots\rangle = 0$. The annihilation operator, a_j can then be defined as the adjoint of $a_j^\dagger$ so $(a_j^\dagger)^\dagger = a_j$. The annihilation works counter to the creation operator and only on the spin orbital immediately to the left requiring Slater determinant columns to be interchanged

$$a_j |\chi_k \chi_j \chi_l\rangle = -a_j |\chi_j \chi_k \chi_l\rangle = -|\chi_k \chi_l\rangle = |\chi_l \chi_k\rangle. \tag{2.27}$$

As with the creation operator the anticommutator can be derived analogously to give

$$a_j a_i + a_i a_j = 0 = \{a_j, a_i\}. \tag{2.28}$$

Further, the operator relations can be defined and then combined with their respective anticommutation relations to give the anticommutation relation between both operators

$$a_i^\dagger a_i + a_i^\dagger a_i = 1 = \{a_i, a_i^\dagger\} \tag{2.29a}$$

$$a_i^\dagger a_j + a_j^\dagger a_i = 0 = \{a_i, a_j^\dagger\} i \neq j \tag{2.29b}$$

$$a_i^\dagger a_j + a_j^\dagger a_i = \delta_{ij} = \{a_i, a_j^\dagger\}. \tag{2.29c}$$

Next the concept of the empty vacuum state, $|0\rangle$, is introduced. It is defined to be normalised, $\langle 0|0\rangle = 1$. Any Slater determinant can be represented as a set of creation operators acting on the vacuum state

$$|\psi_1\rangle = |\chi_l \chi_k\rangle = a_l^\dagger a_k^\dagger |0\rangle \tag{2.30a}$$

$$|\psi_2\rangle = |\chi_j \chi_i\rangle = a_j^\dagger a_i^\dagger |0\rangle \tag{2.30b}$$

which is shown here for two electrons in each Slater determinant but can easily be extended to include any number of electrons. Hence, the overlap between two Slater determinants can be found by repeated action of the operators to the right

$$\begin{aligned}
\langle \psi_1 | \psi_2 \rangle &= \langle 0 | a_k a_l a_j^\dagger a_i^\dagger | 0 \rangle = \langle 0 | a_k (\delta_{lj} - a_j^\dagger a_l) a_i^\dagger | 0 \rangle = \delta_{lj} \langle 0 | a_k a_i^\dagger | 0 \rangle - \langle 0 | a_k a_j^\dagger a_l a_i^\dagger | 0 \rangle \\
&= \delta_{lj}\delta_{ik} \langle 0|0\rangle - \delta_{lj} \langle 0 | a_i^\dagger a_k | 0 \rangle - \delta_{li} \langle 0 | a_k a_j^\dagger | 0 \rangle + \langle 0 | a_k a_j^\dagger a_i^\dagger a_l | 0 \rangle \\
&= \delta_{lj}\delta_{ik} \langle 0|0\rangle - \delta_{li}\delta_{kj} \langle 0|0\rangle + \delta_{li} \langle 0 | a_j^\dagger a_k | 0 \rangle \\
&= \delta_{lj}\delta_{ik} - \delta_{li}\delta_{kj}.
\end{aligned} \tag{2.31}$$

This derivation uses the fact $a_l a_j^\dagger = \delta_{lj} - a_j^\dagger a_l$ by simple rearrangement of Eq. (2.29c) and note the action of the annihilation operator, on the vacuum state, to the right causing terms to disappear. It is now possible to write the one- and two-electron operators in terms of creation and annihilation operators

$$\mathfrak{O}_1 = \sum_{ij} \langle i|h|j\rangle a_i^\dagger a_j \tag{2.32}$$

$$\mathfrak{O}_2 = \frac{1}{2} \sum_{klij} \langle kl|ji\rangle a_k^\dagger a_l^\dagger a_i \hat{a}_j. \tag{2.33}$$

These operators can then be used to define the second quantisation Hamiltonian equation

$$\hat{H} = \sum_{ij} h_{ij} a_i^\dagger a_j + \frac{1}{2} \sum_{klij} a_k^\dagger \hat{a}_l^\dagger W_{klji} a_i \hat{a}_j \tag{2.34}$$

with $h_{ij} = \langle i|h|j\rangle$ and $W_{klji} = \langle kl|ji\rangle$ being the one- and two-electron integrals.

2.2.6 Basis Sets

Slater determinants give a convenient way to describe the occupation of molecular orbitals while preserving the integral anti-symmetric property of the electrons. However, in an actual molecule the space the electrons inhabit is a real space and shape and so this has to be reflected in any simulation. To do this the concept of an electronic basis set is introduced which introduces two approximations via the way each orbital is constructed and the number of orbitals in the basis set. The basis set consists of a set of known functions that can be taken as linear combinations to describe an orbital. So molecular orbitals can be constructed by taking linear combinations of the constituent atomic orbitals. The number of basis functions, in most cases, is a finite set which sets the number of possible orbitals. The basis set limit is approached as the number of basis functions is increased towards the complete infinite basis set.

2.2.6.1 Slater-Type Orbitals

Slater-type orbitals (STO)s, named after John C. Slater, are similar to analytical solutions to the Schrödinger equation for the hydrogen atom with the following form

$$\chi_{\zeta,n,l,m}(r,\theta,\phi) = NY_{l,m}(\theta,\phi)r^{n-1}e^{-\zeta r} \tag{2.35}$$

where N is a normalisation constant, $Y_{l,m}$ is the spherical harmonics which depends on quantum numbers l and m_l; the radial part is given by r^{n-1} where n is a quantum number and the size of the function is controlled by the Slater exponent ζ [41]. At short and long distances from the nucleus they display the correct behaviour: a finite slope at $r = 0$ and a slow decay as $r \longrightarrow \infty$. This is due to their similarity to the hydrogen atom – they are based on a physical truth. Evaluation of their integrals can, however, be computationally expensive.

2.2.6.2 Gaussian-Type Orbitals

Gaussian-type orbitals have become the most popular basis set due to the way they can be simply combined [42]. A single Gaussian orbital function has the form

$$\begin{aligned} \chi_{\alpha,l_x,l_y,l_z}(x,y,z) &= Nx^{l_x}y^{l_x}z^{l_z}e^{-\alpha r^2} \text{ (Cartesian)} \\ \chi_{\alpha,n,l,m}(r,\theta,\phi) &= NY_{l,m}(\theta,\phi)r^{n-1}e^{-\alpha r^2} \text{ (Spherical)} \end{aligned} \tag{2.36}$$

where α is the Gaussian orbital exponent controlling the width. l_x, l_y, l_z are non-negative and sum $l = l_x + l_y + l_z$ to be equivalent to the angular momentum quantum number [43, 44]. GTOs do not have the correct behaviour at both short and long

distance from the centre having a more rapid decay that STOs as $r \longrightarrow \infty$ and having a zero slope when $r = 0$. But two Gaussians can be multiplied together to give another Gaussian centred between the two original functions which makes them simpler to combine than STOs when constructing molecular orbitals from constituent atomic orbitals. To overcome the limitations of GTOs a single orbital is constructed using a linear combination of primitive GTOs, so they better resemble STOs. These are called contracted Gaussian functions (CGF)

$$\chi^{CGF} = \sum_i a_i \chi_i^{GTO}. \tag{2.37}$$

These CGFs are variationally optimised with respect to the Hartree-Fock energy of free atoms which will give a fixed set of coefficients and exponents in the primitive GTOs. The optimal CGFs can then be used to construct the various types of Gaussian-type basis sets.

2.2.6.3 Pople Basis Sets

The most basic, minimal, basis set is a single-ζ basis set where each orbital is represented by one basis function. The most common form of this minimal basis is the STO-nG, proposed by Pople, with n being the number of primitive GTOs used to construct each CGF for each orbital [45]. But this representation is often insufficient to properly describe each orbital. Thus, the number of CGFs for each orbital can be increased giving double-ζ(DZ), triple-ζ(TZ) and quadruple-ζ(QZ) basis sets with two, three and four functions respectively. These higher ζ sets are usually used in split-valence sets where core orbitals used single-ζ functions and the valence orbitals, more important in dictating the chemistry, have the higher-ζ basis functions. These basis sets take the form n-ijG for DZ basis sets and n-ijkG for the triple-ζ sets. n is the number of primitive Gaussian functions used for the core orbitals, i, j (and k) give the number of primitive functions used in the first, second (and third) CGFs of the valence orbitals.

Further modifications can be made to these basis sets to improve their description of molecular orbitals. Polarisation functions are added to the Gaussian atomic orbitals to account for the fact when bonded s-orbitals take on some p-character, p-orbitals acquire d-character and so on. Originally this polarisation was shown by n-ijG* or n-ijG** the '*' symbol meaning polarisation of all heavy atoms and '**' polarisation of both the heavy and light atoms. This notation is inflexible, restricting the number and type of polarisation functions and so an improved notation is introduced. For example, the 3-21G** basis with polarisation of all atoms is equivalent to 3-21G(d,p) showing d-type functions added to the heavy atoms and p-type functions added to the hydrogens. In systems with loosely bound electrons such as anions or excited states diffuse functions can be added to better describe this

behaviour. They have small exponents and a rate of decay that slows with increasing distance from the nucleus. Diffuse functions are usually of s- and p- type and their inclusion is shown by adding a '+' before the G for inclusion on the heavy atoms and '++' for their addition to both the heavy and light atoms. Therefore, the 6-311+G(2df,2pd) basis would have 6 primitive Gaussians to represent the core orbitals and the valence orbitals are represented with a set of three CGFs made up of three, one and one Gaussian primitives. Two sets of d-orbitals and one set of f-type functions as well as diffuse functions are added to the heavy atoms and two sets of p-type and one of d-type functions are added to the hydrogens.

2.2.6.4 Correlation-consistent Basis Sets

Correlation-consistent (cc) basis sets were developed by Dunning and co-workers to converge post-Hartree Fock wave function methods [46–50]. The functions are optimised using configuration interaction single and double excited wave functions. These basis sets are denoted cc-pVXZ (X=D, T, Q, 5...), with 'cc-p' meaning correlation-consistent polarised and the 'V' meaning it is a valence orbital only set and 'XZ' being the X-ζ basis like in the Pople basis sets. The core-core and core-valence electron correlation can be recovered by the addition of functions with large exponents which is denoted as cc-pCVXZ, the 'C' showing the inclusion of core orbitals. Further, diffuse functions can also be added for all orbital types already present in the basis which is illustrated by the 'aug' prefix giving the notation aug-cc-pVDZ.

2.2.7 The Hartree-Fock Approximation

The Hartree-Fock approximation produces solutions to the electronic Schrödinger equation and form the starting point for the Post-Hartree Fock methods as well as the Zombie state method. The method was first presented in 1928 by Hartree and it was then given a more substantial theoretical footing by independent work by Gaunt and Slater [4, 51, 52]. Through use of the variation method a set of equations are constructed that can be used to find the best set of spin orbitals that minimise the ground state energy using the results from Eq. (2.22) and Eq. (2.23)

$$E_0 = \langle \Psi_0 | \hat{H} | \Psi_0 \rangle = \sum_i \langle i|h|i\rangle + \frac{1}{2}\sum_{ij} \langle ij|ij\rangle - \langle ij|ji\rangle \tag{2.38}$$

assuming the Ψ_0 is normalised. The spin orbitals, $\{\chi_i\}$, are systematically varied subject to the constraint they remain orthonormal i.e. $\langle \chi_j | \chi_i \rangle = \delta_{ij}$. This results in

a set of one-electron Fock equations

$$\hat{f}_i\chi_i = \epsilon_i\chi_i \tag{2.39}$$

where $\hat{f}_i$ gives the effective one-particle Hamiltonian and ϵ_i is the orbital energy for spin orbital χ_i.

2.2.7.1 The Fock Operator

The Fock operator is the sum of the core Hamiltonian operator, $\hat{h}_i$, and the effective one-electron Hartree Fock operator, V_i^{HF},

$$\hat{f}_i = \hat{h}_i + V_i^{HF} = -\frac{1}{2}\nabla_i^2 - \sum_A^{N_{nu}} \frac{Z_A}{r_{iA}} + \sum_j (\hat{J}_j(1) - \hat{K}_j(1)) \tag{2.40}$$

$\hat{h}_i$ is the kinetic and potential energy for attraction to the nuclei for a single chosen electron. V_i^{HF} describes the electron-electron interactions, and consists of the Coulomb, $\hat{J}_j$, and exchange, $\hat{K}_j$, operators. The Coulomb operator is defined as

$$\hat{J}_j(1) = \int \chi_j^*(2)\chi_j(2)\frac{1}{r_{12}}d\mathbf{x}_2 = \int |\chi_j(2)|^2 \frac{1}{r_{12}}d\mathbf{x}_2. \tag{2.41}$$

If electron 2 occupies χ_j then the two-electron potential experienced by electron χ_i is obtained by averaging this interaction over all space and spin coordinates, $\mathbf{x}_2$, weighted by the probability $d\mathbf{x}_2|\chi_j(2)|^2$ electron 2 occupies the volume $d\mathbf{x}_2$ at $\mathbf{x}_2$. This is summed over $i \neq j$ to give the total averaged potential acting on electron χ_i by the $N-1$ electrons in the other spin orbitals. Taking Eq. (2.39) and extracting just the exchange operator part

$$\hat{K}_j(1)\chi_i(1) = \left[\int \chi_j^*(2)\chi_i(2)\frac{1}{r_{12}}d\mathbf{x}_2\right]\chi_j(1) \tag{2.42}$$

the exchange operator can be defined with the single electron being made explicit. It can be seen that the operator acting on $\chi_i(1)$ causes the electrons to be exchanged and so there is no uniquely defined potential for $\hat{K}_j(\mathbf{x}_1)$ at $\mathbf{x}_1$ as it depends on χ_i throughout all space. The Coulomb operator is localised as $\hat{J}_j$ can be defined independently of χ_i.

The Fock operator forms a set of coupled non-linear differential equations where each spin orbital is a solution depending on all the other electrons. Therefore, they are solved iteratively starting from guess set of orbitals to allow V_i^{HF} to be calculated. The Fock equations can then be solved which generates a new set of orbitals. This process can then be repeated until the difference between the input and new orbital set is below a predefined threshold.

2.2.7.2 Spin

How electrons are distributed across the set of spin orbitals dictates the spin of the system which in turn gives different expressions for the HF equations. These distributions can be classified as either closed- or open-shell depending on all the electrons being paired up in orbitals or not. These expressions of the HF equations use spatial representations of the electrons rather than the basis that includes spin that has been used up until this point. So, integrals are over spatial coordinates, $\mathbf{r}$, with a spatial orbital containing at most two electrons. So, it is necessary to introduce some analogous notation to the bra-ket integrals used in Eq. (2.22) and Eq. (2.23) to calculate Hamiltonian matrix elements. The integral over spatial orbitals for the one-electron part of the Hamiltonian, Eq. (2.7), are written as

$$(i|h|i) = (\psi_i(\mathbf{r}_1)|h|\psi_i(\mathbf{r}_1)) = \int \psi_i^*(1)h(1)\psi_i(1)d\mathbf{r}_1, \tag{2.43}$$

the round brackets indicating spatial rather than spin orbitals. Similarly for the two-electron part of the Hamiltonian,

$$(ij|kl) = (\psi_i(\mathbf{r}_1)\psi_j(\mathbf{r}_1)|\psi_k(\mathbf{r}_2)\psi_l(\mathbf{r}_2)) = \int \psi_i^*(1)\psi_j(1)\frac{1}{r_{12}}\psi_k^*(2)\psi_l(2)d\mathbf{r}_1 d\mathbf{r}_2. \tag{2.44}$$

2.2.7.2.1 Restricted Closed-shell Equations

The restricted close-shell (RHF) case occurs when all electron spins are paired. Thus, it is possible to convert the spin orbitals to spatial orbitals which gives a Fock operator of the form

$$\hat{f}_i^{RHF} = h_i + \sum_j^{N_{el}/2} 2\hat{J}_j^{RHF}(1) - \hat{K}_j^{RHF}(1) \tag{2.45}$$

where the Coulomb and exchange operators are defined as

$$\hat{J}_j^{RHF}(1) = \int \psi_j^*(2)\frac{1}{r_{12}}\psi_j(2)d\mathbf{r}_2 = \int |\psi_j(2)|^2\frac{1}{r_{12}}d\mathbf{r}_2 \tag{2.46}$$

$$\hat{K}_j^{RHF}(1)\psi_i(1) = \left[\int \psi_j^*(2)\frac{1}{r_{12}}\psi_i^*(2)d\mathbf{r}_2\right]\psi_j(1). \tag{2.47}$$

As N_{el} is the number of electrons the sum in the Fock operator is now over the $N_{el}/2$ occupied spatial orbitals. The energy for a single closed shell determinant, $\langle\Psi_0| = \langle\psi_1\tilde{\psi}_1\ldots\psi_{N_{el}/2}\tilde{\psi}_{N_{el}/2}|$ is given by

$$E^{RHF} = \langle\Psi_0|\hat{H}|\Psi_0\rangle = 2\sum_i(i|h|i) + \sum_{ij}2(ii|jj) - (ij|ji) = 2\sum_i h_{ii} + \sum_{ij}2\hat{J}_{ij}^{RHF} - \hat{K}_{ij}^{RHF} \tag{2.48}$$

where

$$\hat{J}_{ij}^{RHF} = \int \psi_i^*(i)\psi_j^*(j)\frac{1}{r_{ij}}\psi_i(i)\psi_j(j)d\mathbf{r}_i d\mathbf{r}_j \tag{2.49}$$

is the Coulomb energy between electrons i and j in the i-th and j-th orbitals, made up of the pairwise interactions between all the electrons. The exchange energy

$$\hat{K}_{ij}^{RHF} = \int \psi_i^*(i)\psi_j^*(j)\frac{1}{r_{ij}}\psi_i(j)\psi_j(i)d\mathbf{r}_i d\mathbf{r}_j \tag{2.50}$$

consists of only pairwise interactions between electrons with parallel spins.

2.2.7.2.2 Unrestricted Hartree Fock

For open-shell systems the Unrestricted Hartree Fock (UHF) method is commonly used where any orbital is allowed to be left unpaired. The set of unrestricted spin orbitals can be defined in terms of two sets of spatial orbitals

$$\chi_k = \begin{cases} \psi_j^\alpha(\mathbf{r})\alpha(\omega) \\ \psi_j^\beta(\mathbf{r})\beta(\omega) \end{cases} \tag{2.51}$$

accounting for the unpaired spin up or down electrons. Either set can then be substituted into Eq. (2.39) and its respective spin integrated out. The α spin case has the Fock operator

$$\hat{f}_i^\alpha = h_i + \sum_j^{N^\alpha} \hat{J}_j^\alpha(1) - \hat{K}_j^\alpha(1) + \sum_j^{N^\beta} \hat{J}_j^\beta(1). \tag{2.52}$$

where N^α is the number of orbitals occupied by α spin electrons and N^β the number of electrons of β spin. The Coulomb and exchange operators can then be defined

$$\hat{J}_j^\alpha(1) = \int \psi_j^{\alpha*}(2)\frac{1}{r_{12}}\psi_j^\alpha(2)d\mathbf{r}_2 \tag{2.53}$$

$$\hat{K}_j^\alpha(1)\psi_i(1) = \left[\int \psi_j^{\alpha*}(2)\frac{1}{r_{12}}\psi_i^\alpha(2)d\mathbf{r}_2\right]\psi_j^\alpha(1). \tag{2.54}$$

The set of operators for the opposite β spin case have an analogous form. The ground state energy for the unrestricted electronic energy is then,

$$\begin{aligned} E^{UHF} =& \sum_i^{N^\alpha}(i|h^\alpha|i) + \sum_i^{N^\beta}(i|h^\beta|i) + \frac{1}{2}\sum_i^{N^\alpha}\sum_j^{N^\beta}(\hat{J}_{ij}^{\alpha\alpha} - \hat{K}_{ij}^{\alpha\alpha}) + \\ & \frac{1}{2}\sum_i^{N^\alpha}\sum_j^{N^\beta}(\hat{J}_{ij}^{\beta\beta} - \hat{K}_{ij}^{\beta\beta}) + \sum_i^{N^\alpha}\sum_j^{N^\beta}\hat{J}_{ij}^{\alpha\beta}. \end{aligned} \tag{2.55}$$

The Slater determinant is an eigenfunction of the $\hat{S}_z$ spin operator but not of the $\hat{S}_z^2$. The method allows artificial mixing of states of different spins called spin contamination which can give inaccurate energies. But due to the simplicity and computational efficiency UHF is widely used and a variety of methods have been developed to eliminate the spin contamination [53, 54].

2.2.7.2.3 Restricted Open-shell Equations

The final way for dealing with open-shell systems is restricted open-shell HF (ROHF). Antisymmetrised products are created from a doubly occupied closed-shell core, ψ_C and a partially occupied open shell chosen from a set, ψ_O. The total wave function is then made up by summing different products which are created by different subsets of ψ_O. The combined set of orbitals is $\psi = (\psi_C, \psi_O)$ which is assumed to be orthonormal. Thus, ψ_C and ψ_O are orthonormal and $\langle \psi_C^{(a)} | \psi_O^{(b)} \rangle = 0$, $\psi_C^{(a)} \in \psi_C$ and $\psi_O^{(b)} \in \psi_O$ [55]. This gives an energy

$$\begin{aligned} E^{ROHF} =& 2\sum_k H_{kk} + \sum_{kl}(2J_{kl} - K_{kl}) + f[2\sum_m H_{mm} \\ &+ \sum_{mn}(2aJ_{mn} - bK_{mn}) + 2\sum_{km}(2J_{km} - K_{km}], \end{aligned} \tag{2.56}$$

the indices k, l have been used for closed-shell orbitals and m, n for the open shell orbitals which is consistent with Ref. [55]. f is the fraction of occupied spin orbitals in the open shell; a and b are coupling coefficients [56]. So, the energy sum is made up of a closed-shell, an open-shell and an interaction part. This method is used far less frequently than UHF due to its higher computational cost.

2.2.7.3 Roothaan-Hall Equations

Finding direct solutions to the Fock equations is still difficult even when using the RHF or UHF methods to eliminate the spin. Roothaan and Hall independently proposed a method of solving the integro-differential into a set of algebraic equations by introducing basis functions [57, 58]. The specifics of these functions are detailed in the previous section 2.2.6. The unknown molecular orbitals of the system are expanded in the set of N_{bf} known atomic basis functions, here the RHF case is used.

$$\psi_i = \sum_{\mu=1}^{N_{bf}} C_{\mu i}\phi_\mu. \tag{2.57}$$

This can then be substituted into Eq. (2.39) using ν as the index yields

$$\hat{f}_i \sum_{\nu}^{N_{abf}} C_{\nu i}\phi_\nu(1) = \epsilon_i \sum_{\nu}^{N_{abf}} C_{\nu i}\phi_\nu(1). \tag{2.58}$$

This can be multiplied by $\phi_\mu^*(1)$ and integrated to give a matrix equation

$$\sum_\nu C_{\nu i} \int d\mathbf{r}_1 \phi_\mu^*(1) f_i \phi_\nu(1) = \epsilon_i \sum_\nu C_{\nu i} \int d\mathbf{r}_1 \phi_\mu^*(1) \phi_\nu(1) \tag{2.59}$$

These are the Roothaan-Hall equations and can be written compactly as a matrix equation

$$\mathbf{FC} = \mathbf{SC}\epsilon. \tag{2.60}$$

$\mathbf{S}$ is the overlap matrix of the basis set, is Hermitian and of size $K \times K$. The Fock matrix $\mathbf{F}$ is also a $K \times K$ Hermitian matrix and $\mathbf{C}$ is a $K \times K$ square matrix containing the expansion coefficients $C_{\nu i}$. Each column of $\mathbf{C}$ describes the molecular orbitals. For a closed-shell molecule the total charge density for a fully occupied molecular orbital ψ_i is

$$\rho(\mathbf{r}) = 2 \sum_i^{N/2} |\psi_i(\mathbf{r})|^2. \tag{2.61}$$

This can be written in terms of the expansion coefficients as the density matrix

$$P_{\mu\nu} = 2 \sum_i^{N/2} C_{\mu i} C_{\nu i}^*. \tag{2.62}$$

Fock matrix can then be written in terms of the density matrix $\mathbf{P}$

$$\begin{aligned} F_{\mu\nu} &= H_{\mu\nu}^{core} + \sum_i^{N/2} \sum_{\lambda\sigma} C_{\lambda i} C_{\sigma i}^* [2(\mu\nu|\sigma\lambda) - (\mu\lambda|\sigma\nu)] \\ &= H_{\mu\nu}^{core} + \sum_{\lambda\sigma} P_{\lambda\sigma} [(\mu\nu|\sigma\lambda) - \frac{1}{2}(\mu\lambda|\sigma\nu)] \\ &= H_{\mu\nu}^{core} + G_{\mu\nu} \end{aligned} \tag{2.63}$$

$H_{\mu\nu}^{core}$ is the core-Hamiltonian one-electron part, $\mathbf{G}$ is the two-electron part depending on the density matrix $\mathbf{P}$ and a set of two electron integrals. The basis set used is usually not orthogonal i.e. the overlap matrix $\mathbf{S}$ has off diagonal elements but it is possible to find a transformational matrix $\mathbf{X}$ that does form and orthogonal set

$$\mathbf{X}^\dagger \mathbf{S} \mathbf{X} = \mathbf{1}. \tag{2.64}$$

Once $\mathbf{X}$ has been found a new coefficient matrix $\mathbf{C}'$ can be defined in terms of the old coefficients $\mathbf{C}' = \mathbf{X}^{-1}\mathbf{C}$ and substituted into the Roothaan-Hall equations

$$\mathbf{FXC}' = \mathbf{SXC}'\epsilon \tag{2.65}$$

$$(\mathbf{X}^\dagger \mathbf{F} \mathbf{X})\mathbf{C}' = (\mathbf{X}^\dagger \mathbf{S} \mathbf{X})\mathbf{C}'\epsilon \tag{2.66}$$

A new Fock matrix can then be defined $\mathbf{F}' = \mathbf{X}^\dagger \mathbf{F} \mathbf{X}$ and so

$$\mathbf{F}'\mathbf{C}' = \mathbf{C}'\epsilon \tag{2.67}$$

These are called the transformed Roothaan-Hall equations which can be used iteratively to define new sets of $\mathbf{C}'$ and orbital energies ϵ.

2.2.7.4 Self Consistent Field Method

It is now possible to fully define the Self Consistent Field process using the Roothaan-Hall equations.

1. Define a system by its nuclear coordinates, atomic numbers and number of electrons N and choose a basis set $\{\phi_\mu\}$.
2. Calculate the overlap matrix $\mathbf{S}$ and one- and two-electron integrals $H_{\mu\nu}^{core}$ and $(\mu\nu|\sigma\lambda)$.
3. Diagonalise $\mathbf{S}$ to obtain $\mathbf{X}$.
4. Guess the coefficients of the density matrix $\mathbf{P}$.
5. Calculate matrix $\mathbf{G}$ and form the Fock matrix using Eq. (2.63).
6. Calculate the transformed Fock matrix using $\mathbf{F}' = \mathbf{X}^\dagger \mathbf{F} \mathbf{X}$.
7. Diagonalise $\mathbf{F}'$ to find $\mathbf{C}'$ and ϵ.
8. Calculate new coefficients, $\mathbf{C} = \mathbf{X}\mathbf{C}'$.
9. Form a new density matrix using Eq. (2.62).
10. Repeat the process from step 5 until convergence is achieved i.e. the newly calculated density matrix is the same as $\mathbf{P}$ from the previous step (or within a predetermined tolerance).

The initial guess for the density matrix $\mathbf{P}$ can be a zero matrix which would neglect the two-electron component of the Fock matrix however this does not always lead to convergence. There are of course a variety of different ways to make a better initial guess such as using a superposition of atomic HF density matrices or atomic potentials or the extended Hückle method where orbitals can be obtained from diagonalising an effective one-particle Hamiltonian [59–61]. If a complete basis set $\{\phi_\mu\}$ is used then the expansion of the molecular orbitals would be exact and the lowest variational energy, the Hartree Fock limit E_{HF}, would be obtained. In reality a finite basis will be used and the single Slater determinant wave function formed by the occupied spin orbitals is the best approximation of the ground state.

Significantly, the motion of one electron can affect that of another electron. However, though the mean field approximation this behaviour is not fully captured in HF theory. Therefore, it is possible two electrons could occupy the same space which is, of course, unphysical. Further, it is possible to construct configurations of electrons that, though different, have the exact or nearly degenerate HOMO and LUMO orbitals. By using a single determinant these configurations that could contribute a similar weight to the system means the non-dynamic correlation is excluded in the HF calculation. These contributions, though relatively small, can be hugely important giving completely different chemistry. Thus, E_{HF} is always an upper bound to the true ground state energy.

2.2.8 Post-Hartree-Fock

It is necessary to extend the basic Hartree Fock theory to account for dynamic electron correlation. In Many-Body Perturbation theory (MBPT) the electron correlation is treated as a small perturbation to the ground state wave function. Møller-Plesset theory, a form of Rayleigh Schrödinger perturbation theory, starts form the eigenvalue problem for an electronic state, n [62].

$$\hat{H}|\Psi_n\rangle = (\hat{H}_0 + \hat{V})|\Psi_n\rangle = E_n|\Psi_n\rangle \tag{2.68}$$

where $\hat{H}_0$ is the sum of Fock operators from Eq. (2.40) and the perturbation is defined as

$$\hat{V} = \hat{H} - \hat{H}_0 = \hat{H} - \sum_i^N \hat{f}_i. \tag{2.69}$$

The wave function can be expanded as a power series

$$(\hat{H}_0 + \lambda\hat{V})|\Psi_n^{(0)} + \lambda\Psi_n^{(1)} + \ldots\rangle = (E_n^{(0)} + \lambda E_n^{(1)} + \lambda^2 E_n^{(2)})|\Psi_n^{(0)} + \lambda\Psi_n^{(1)} + \ldots\rangle \tag{2.70}$$

with $\Psi_n^{(0)} = \Psi_{HF}$ making the reference determinant the Hartree Fock Slater determinant. The different order perturbations can then be found

$$E_n^{(0)} = \langle\Psi_n^{(0)}|\hat{H}_0|\Psi_n^{(0)}\rangle \tag{2.71a}$$
$$E_n^{(1)} = \langle\Psi_n^{(0)}|\hat{V}|\Psi_n^{(0)}\rangle \tag{2.71b}$$
$$E_n^{(2)} = \langle\Psi_n^{(0)}|\hat{V}|\Psi_n^{(1)}\rangle. \tag{2.71c}$$

The first order correction to the wave function is given by

$$|\Psi_n^{(1)}\rangle = \sum_{m\neq n} \frac{\langle\Psi_m^{(0)}|\hat{V}|\Psi_n^{(0)}\rangle}{E_n^{(0)} - E_m^{(0)}}|\Psi_m^{(0)}\rangle \tag{2.72}$$

with all other eigenstates contributing. Therefore

$$E_n^{(2)} = \sum_{m \neq n} \frac{\left|\langle \Psi_m^{(0)} | \hat{V} | \Psi_n^{(0)} \rangle\right|^2}{E_n^{(0)} - E_m^{(0)}}. \tag{2.73}$$

Coupled cluster (CC) theory also starts with a reference wave function, $|\psi_0\rangle$ [63, 64]. This state is usually the HF determinant although other wave functions can be used

$$|\Psi\rangle = e^{\hat{T}}|\psi_0\rangle = (1 + \hat{T}\frac{1}{2}\hat{T}^2 + \frac{1}{3!}\hat{T}^3)|\psi_0\rangle. \tag{2.74}$$

The cluster operator $\hat{T}$ can be written as a sum of operators that generate excitations from a single, $\hat{T}_1$ to an N-tuple, $\hat{T}_N$ for the reference wave function containing N electrons

$$\hat{T} = \hat{T}_1 + \hat{T}_2 + \ldots + \hat{T}_N \tag{2.75}$$

The energy of the system is calculated

$$E = \langle \psi_0 | e^{-\hat{T}} H e^{\hat{T}} | \psi_0 \rangle = \langle \psi_0 | \bar{H} | \psi_0 \rangle \tag{2.76}$$

which has unknown amplitudes which can be found by solving

$$\langle \psi^* | \bar{H} | \psi_0 \rangle = E \langle \psi^* | \psi_0 \rangle = 0. \tag{2.77}$$

In practice the sum of the operator $\hat{T}$ is truncated. Using just $\hat{T}_1$ and $\hat{T}_2$ gives just single and double excitations and is called the coupled cluster singles-and-doubles (CCSD) method. Higher order excitations can then be approximated, for example $\hat{T}_2^2$ gives the quadruple excitation. CCSD can be extended to include triple excitations (CCSDT) which is computationally much more expensive. To remedy this the triple excitations can be calculated using a non-iterative many-body perturbation theory, CCSD(T), the bracketed letter indicating which excitations have been calculated using MBPT. CC methods have been applied to adsorption and reaction energies on surfaces for atoms and molecules [65–69]. While also being used to calculate excitation energies and a variety of spectra [70–75].

2.2.8.1 Full Configuration Interaction

Alternatively, a variational approach can be taken to account for electron correlation. Configuration interaction (CI) constructs a wave function from a linear combination of Slater determinants

$$\Psi_{CI} = \sum_{m=0} c_m|\psi_m\rangle = c_0|\psi_0\rangle + \sum_{m=1}^{A} c_m|\psi_m\rangle + \sum_{m>A}^{B} c_m|\psi_m\rangle + \cdots + \sum_{m>B}^{\frac{N!}{K!(N-K)!}} c_m|\psi_m\rangle \tag{2.78}$$

Each ψ_i in Ψ_{CI} has a different configuration of electrons with the maximum number of configurations for a system with N electrons and K spin orbitals is $\frac{N!}{K!(N-K)!}$. This is the full CI limit and will exactly solve the electronic Schrödinger equation. Even for a small system the number of possible configurations in the full-CI space can be extremely large so truncations are often necessary. To choose these truncated sets a single HF Slater determinant is used as a reference configuration. Then configurations containing a specified type of excitation can be generated, for example CI singles-and-doubles (CISD) only includes single and double excitations of the reference determinant. It is also possible to use more than one reference Slater determinant which is called multi-reference CI (MRCI). The CI wave function coefficients are then found by variational minimisation of the ground state energy

$$E_{CI} = \min_{\Psi_{CI}} \frac{\langle \Psi_{CI}|\hat{H}_0|\Psi_{CI}\rangle}{\langle \Psi_{CI}|\Psi_{CI}\rangle} \tag{2.79}$$

2.2.8.2 Methods for Optimal Truncation of Configuration Space

Using a full-CI basis set is appealing due to the exact solutions to the Schrödinger equation that can be obtained but for most reasonable systems this is computationally expensive and potentially impossible. However, a significant number of configurations will contribute a negligible amount to the system and can be ignored. A key way to tackle this problem is by introducing an active space containing the most important orbitals which can be occupied by a set of active electrons. Though, intuitively, configurations with electrons in higher energy spin orbitals are less likely to be of importance they cannot be systematically ruled out; by construction the active space excludes configurations with these higher energy orbital occupations. Hence, Quantum Monte-Carlo (QMC) methods have been developed to allow important configurations with populated orbitals above the active space which smooths the transition between the two regions. Green's function MC and similarly Diffusion MC propagate walkers in continuum space which removes the need for a one-electron

basis but are limited by not preserving the sign change needed to describe the antisymmetry of the fermionic wave function [76, 77]. Using a fixed node approximation solves the antisymmetry problem but subsequent work has been limited [78].

A more practical and effective approach is to use a Monte-Carlo method to select configurations which is achieved by a random walk in the manifold of Slater determinants. Alavi and co-workers developed the Full Configuration-Interaction Quantum Monte-Carlo (FCIQMC) method which uses a long-time propagation in imaginary time with random walkers to stochastically describe the full CI wave function [79, 80]. This takes the wave function $\Psi_0^{FCI} = \sum_i D_i|D_i\rangle$ and uses a coupled linear first-order differential equation for the coefficients in terms of a K matrix,

$$-\frac{C_i}{d\tau} = (K_{ii} - S)C_i + \sum_{j\neq i} K_{ij}C_j. \tag{2.80}$$

S is an energy shift parameter used to control the rate of population change. The K matrix is defined by

$$K_{ij} = \langle D_i|H|D_j\rangle - E_{HF}\delta_{ij}, \tag{2.81}$$

which has positive diagonal elements. A set of "walkers" are then used to perform the stochastic integration by projecting them through the Slater determinant space. This process is controlled as such,

1. Spawn – For each walker on D_i a coupled determinant D_j, that has the normalised probability $p_{gen}(\boldsymbol{j}|\boldsymbol{i})$, is selected and a child walker is attempted to be spawned using,

$$p_s(\boldsymbol{j}|\boldsymbol{i}) = -\frac{\delta\tau|K_{ij}|}{p_{gen}(\boldsymbol{j}|\boldsymbol{i})}. \tag{2.82}$$

2. Death/cloning – Each parent walker is then removed with probability

$$p_d(\boldsymbol{j}) = \delta\tau(K_{ii} - S). \tag{2.83}$$

3. Annihilate – All surviving walker pairs with opposite sign on the same determinant are removed.

The coefficients are then proportional to the signed population of walkers on the corresponding determinant $C_i \propto N_i$. The simulation allows the shift parameter to change until the coefficients are unchanging with time, so

$$\sum_j K_{ij}C_j = SC_i \tag{2.84}$$

where S is the correlation energy of the ground state. The method evolves to the

Fermionic ground state with the annihilation step avoiding the fixed node approximation [79]. The total number of walkers grows and then plateaus, the height of which is used as a measure of the number of walkers needed to converge the energy. Using FCIQMC full-CI energies have been recovered for computationally expensive systems such as the neutral and cationic elements from Li to Mg [79, 81–83]. The method has also been shown to effective at recovering the energy of N_2 in the cc-pVDZ basis and in cc-pVTZ and cc-pVQZ basis sets which is beyond what is achievable with a regular full-CI calculation [80]. It has also been shown that FCIQMC can find excited state energies with little additional computational cost [24]. More recently FCIQMC has been applied to systems of bosons and also mixed particle systems [84–86].

Monte Carlo Configuration Interaction (MCCI), developed by Greer, also utilises a Monte-Carlo procedure to build a compact wave function of important configurations [25, 87]. Unlike FCIQMC, configuration state functions (CSF) are used rather than Slater determinants which take the form,

$$|\psi\rangle = \mathcal{A}\mathcal{O}_{S,M_s}\Xi(\boldsymbol{R})\Theta(\boldsymbol{\sigma}) \tag{2.85}$$

where $\mathcal{A}$ is an antisymmetrizer and $\mathcal{O}_{S,M_s}$ is a spin projection operator. $\Xi(\boldsymbol{R})$ and $\Theta(\boldsymbol{\sigma})$ are primitive spatial and spin functions which are formed by making Hartree products. Alternatively, two sets A_α and B_β containing equal numbers of orbitals can be used to construct CSFs. If the two sets have identical spatial orbitals then

$$|\psi\rangle = |A_\alpha; B_\beta\rangle \tag{2.86}$$

without the need for spin projection. This is equivalent to a Slater determinant with each spatial orbital doubly occupied. When A_α and B_β do differ, the CSF has the form,

$$|\psi\rangle = \frac{1}{\sqrt{2}}[|A_\alpha; B_\beta\rangle + |B_\beta; A_\alpha\rangle]. \tag{2.87}$$

The single particle orbitals are assumed to be orthogonal and so by the reasoning discussed in section 2.2.4 if $|\psi_A\rangle$ and $|\psi_B\rangle$ differ by more than two orbitals the Hamiltonian matrix element is zero. The CI wave function can be written in terms of a reference function,

$$|\psi\rangle = (c_0 + \sum_{ij} h_{ij} a_i^\dagger a_j + \sum_{i<j}\sum_{k<l} a_k^\dagger \hat{a}_l^\dagger W_{klji} a_i \hat{a}_j + \ldots)|\psi_0\rangle. \tag{2.88}$$

Using a configuration A, single or double substitution gives a new state B so

$$H_{AB} = \langle\psi_A|\hat{H}|\psi_b\rangle \neq 0. \tag{2.89}$$

MCCI works by taking a trial configuration and randomly generating a set of single and double substitutions. The CI problem is then solved using the set of configurations performing diagonalisation to find eigenvalues and eigenvectors. A selection process then decides if the configurations should be kept or discarded. Configurations are removed by either having a coefficient below a cut-off value or by calculating the contribution a CSF has to an energy eigenvalue,

$$\Delta E_k = \frac{(E - H_{KK})c_k^2}{\sum_A c_A^2 - c_K^2} \tag{2.90}$$

where c_K is the coefficient of the CSF in question and H_{KK} is the corresponding Hamiltonian matrix element. This is the change in energy if the CSF is removed from the CI expansion and all other coefficients are kept constant. This process is repeated until the number of configurations stops increasing i.e. all the significant configurations have been found or an energy criterion is reached.

MCCI was first applied to the single point energy of water and then the dissociation energy of water and HF [87, 88]. Significant work by Coe and Paterson *et. al* continued the development of MCCI demonstrating it could generate potential energy curves, near chemical accuracy, for a range of molecules including N_2 and CH_4 [88]. Accuracy and convergence time, particularly at longer bond lengths, were improved by introducing second-order perturbation theory and using natural orbitals [89]. Natural orbitals are eigenfunctions of the first-order reduced density matrix and converge better than HF molecular orbitals [90]. MCCI has also been used to accurately simulate multipole moments [91]. Excited states have been computed following the inclusion of state averaging [92]. Both FCIQMC and MCCI shows good agreement to the full-CI values when calculating ionisation energies with MCCI needing a significantly smaller number of basis functions compared to the full-CI space [91]. As these quantum MC methods are refined, the full-CI energies of larger and more difficult systems can be approached [93].

2.3 Coherent States

In this section Coherent states are reviewed with specific attention to their application to fermionic systems and electronic structure. Coherent states were first introduced by Schrödinger as specific quantum states of the quantum harmonic oscillator when looking for solutions to the Schrödinger equation that satisfied the correspondence principle. These Coherent states were minimal uncertainty wave packets i.e. quantum solutions that most closely resembled classical behaviour [94]. However, despite being part of the genesis of quantum mechanics, research into Coherent states was limited to condensed matter physics and particle physics [95–98]. The active field of research properly began in 1963 with work by Glauber and

Sudarshan in the field of quantum optics, describing light quantum mechanically [11, 99]. Glauber proposed in Ref. [12] that coherent states for the electromagnetic field could be constructed using one of three definitions:

1. Coherent states $|\alpha\rangle$ are eigenstates of the harmonic-oscillator annihilation operator,

$$a|\alpha\rangle = \alpha|\alpha\rangle \tag{2.91}$$

   where α is a complex number.

2. Application of a displacement operator,

$$D(\alpha) = e^{\alpha a^\dagger - \alpha^* a}, \tag{2.92}$$

   on the harmonic vacuum state obtains a coherent state

$$|\alpha\rangle = D(\alpha)|0\rangle. \tag{2.93}$$

3. Coherent states are quantum states with a minimum uncertainty relationship,

$$(\Delta p)^2(\Delta q)^2 = (1/2)^2, \tag{2.94}$$

   defining position and momentum operators as $\hat{q} = \frac{1}{\sqrt{2}}(a + a^\dagger)$ and $\hat{p} = \frac{1}{i\sqrt{2}}(a - a^\dagger)$ respectively. Then

$$(\Delta f)^2 = \langle\alpha|(\hat{f} - \langle\hat{f}\rangle)^2|\alpha\rangle\langle\hat{f}\rangle) = \langle\alpha|\hat{f}|\alpha\rangle \tag{2.95}$$

Although it should be noted that work by Klauder showed that coherent states could be expressed as eigenvectors of the annihilation operator that form an overcomplete set [100]. Glauber's work was driven by finding the complete factorisation of the electromagnetic field correlation functions. However, not all systems can be described by the harmonic oscillators used by Glauber, which motivates a generalised definition of coherent states [19]. In the 1970s work by Perelomov and Gilmore developed the mathematical structure of coherent states based on the theory of Lie groups and set a general definition for coherent states based on Glauber's second definition using the vacuum state [21, 22].

Firstly, I will briefly discuss why generalisation of the Glauber's first and third coherent state definitions were found to not be particularly useful and did not become the standard. To generalise Glauber's first definition one would construct coherent states from eigenstates of the annihilation operator. This method was adopted by Barut and Girardello giving a non-Hermitian eigenvalue problem with complex eigenvalues [101]. Gilmore showed that this definition would mean coherent

states could not be constructed for Hilbert spaces of finite dimension [102]. Further, the coherent states defined in this way do not resemble physical states unless the commutator of the annihilation operator and the creation operator commutator are multiples of the identity operator. This generally limits the application of this definition to the electromagnetic field essentially Glauber's initial definition [19]. The coherent states constructed by the minimum uncertainty definition states are not unique as can be seen in Fig. 2.1. However, this definition was still generalised to produce "intelligent" states by Aragone and co-workers with further work by Nieto and co-workers Aragone [103–105]. Beyond the limitation of not producing unique states the generalised definition only allows for coherent states to be constructed using the standard photon creation and annihilation operators which leads to limited applications [19].

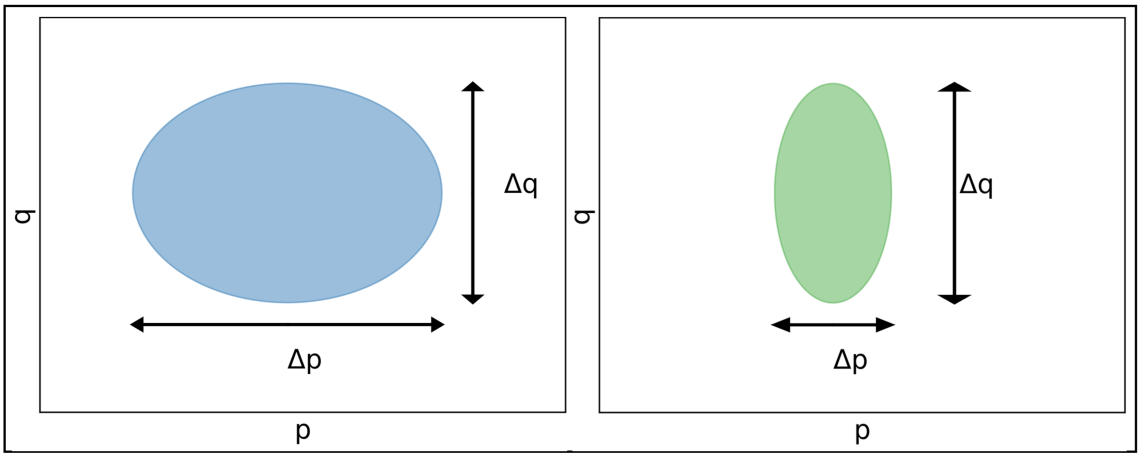


Figure 2.1: Plot showing that the minimum uncertainty relationship $(\Delta p)^2(\Delta q)^2 = (1/2)^2$ does not produce unique coherent states. The circle has $\Delta p = \Delta q = 1/2$ whereas the ellipse has $\Delta p = 1/4$, $\Delta q = 1$. Both the circle and the ellipse give the same minimum uncertainty but clearly the different shapes demonstrate how the coherent states generated in this manner are not necessarily unique.

2.3.1 General Definition of Coherent States

The general coherent state definition was developed by Gilmore and Perelomov[21, 102, 106, 107]. An arbitrary quantum-dynamic system can be defined so its Hamiltonian and transition operators, $\{A\}$, can be expressed in terms of the set of operators, $\{T_i\}$,

$$H = H(T_i), \tag{2.96a}$$

$$A = A(T_i). \tag{2.96b}$$

$\{T_i\} \equiv \boldsymbol{g}$ is a complete set of operators which means the commutator of any two operators in the set is also a member of the set,

$$\forall T_i, T_j \in \boldsymbol{g},\ [T_i, T_j] \in \boldsymbol{g}. \tag{2.97}$$

The Hamiltonian (in the mean-field approximation) can be classified as taking either a linear or quadratic form,

$$H = \sum_i d_i T_i \tag{2.98}$$

$$H = \sum_i c_i T_i + \sum_{i,j} c_{ij} T_i T_j. \tag{2.99}$$

Note how the second quantisation Hamiltonian in Eq. (2.4.5) is of the form in Eq. (2.99) with the set of operators $\{a_i^\dagger a_j, a_i^\dagger a_j^\dagger, a_i a_j\}$. A Lie group, G, with its accompanying algebra $\mathfrak{g}$ is spanned by the translation operators

$$[T_i, T_j] = \sum_{k=1} c_{i,j}^k T_k, \tag{2.100}$$

the structure coefficients are produced in a similar way as in Eq. (A.12). The Hamiltonian defines a Hilbert space, $\mathscr{H}$ which can be written as a unitary irreducible representation of the group G. $U(\mathscr{H}^\Lambda)$ is the set unitary operations on $\mathscr{H}^\Lambda$ found by the mapping

$$U : G \to U(\mathscr{H}^\Lambda) \tag{2.101}$$

which satisfies the condition $U(g)U(a) = U(ga) \quad \forall g, a \in G$. To be irreducible, a closed subspace, κ, of $\mathscr{H}^\Lambda$, is defined

$$U(g) \subseteq \kappa \tag{2.102}$$

which if defined for all $g \in G$ then $\kappa = \mathscr{H}^{\Lambda}$, and Γ^{Λ} is used to represent the irreducible set $U(\mathscr{H}^{\Lambda})$. Finally, a reference state within the Hilbert space is defined as normalising to unity, $\langle \Phi_0 | \Phi_0 \rangle = 1$.

A subgroup, H, of G has elements h that leave the reference state invariant up to a phase factor

$$h|\Phi_0\rangle = |\Phi_0\rangle e^{i\phi(h)} \tag{2.103}$$

which is also called the stability group. Using the reasoning given in section A.4 the coset space is defined with elements $\Omega \in G/H$, which have the property $g = \Omega h$. A coherent state can now be defined,

$$g|\Phi_0\rangle = \Omega h|\Phi_0\rangle = \Omega|\Phi_0\rangle e^{i\phi(h)} \tag{2.104}$$

which can be written

$$|\Lambda, \Omega\rangle = \Omega|\Phi_0\rangle \tag{2.105}$$

This representation preserves the algebra and topological properties of the coset space G/H which in itself has been constructed to be homogeneous to the group G [19]. This is in line with the definition given in Ref. [20] for a coherent state system being "an orbit for an irreducible group action in a Hilbert space" – the orbit being the irreducible representation of the Lie Group. Gilmore and Perelomov set different conditions for the choice of group, Hilbert space and reference state which are discussed further in Ref. [19].

2.3.2 Canonical Coherent States of the Harmonic Oscillator

The Hamiltonian for the quantum harmonic oscillator is

$$\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2} m\omega^2 \hat{q}^2 \tag{2.106}$$

where ω is the angular frequency of the oscillator and $\hat{q}$, $\hat{p}$ are the position and momentum operators respectively. The creation and annihilation operators are defined,

$$a^{\dagger} = \sqrt{\frac{m\omega}{2\hbar}} (\hat{q} - \frac{i}{m\omega} \hat{p}) \tag{2.107}$$

$$a = \sqrt{\frac{m\omega}{2\hbar}} (\hat{q} + \frac{i}{m\omega} \hat{p}). \tag{2.108}$$

The position and momentum operators can then be defined in terms of a and $a^\dagger$,

$$\hat{q} = \sqrt{\frac{\hbar}{2m\omega}}(a^\dagger + a) \tag{2.109}$$

$$\hat{p} = i\sqrt{\frac{\hbar m\omega}{2}}(a^\dagger - a). \tag{2.110}$$

The canonical commutation relation is $[\hat{q}, \hat{p}] = i\hbar$ which can be used along with the definition of the number operator $\hat{N} = a^\dagger a$ to define the set of commutation relations,

$$[\hat{N}, a^\dagger] = a^\dagger, \quad [\hat{N}, a] = -a, \quad [a, a^\dagger] = \boldsymbol{I}. \tag{2.111}$$

This means the Hamiltonian can be written as

$$\hat{H} = \hbar\omega(\hat{N} + \frac{1}{2}). \tag{2.112}$$

It is now possible to start defining coherent states using the general definition. The set of operators $\{\hat{N}, a^\dagger, a, \boldsymbol{I}\}$ span the Lie algebra h_4 which is used by the Heisenberg-Weyl (Lie) group H_4 [108]. The Hilbert space for the Lie group, H_4 is spanned by the number eigenstates,

$$\hat{N}|n\rangle = n|n\rangle \tag{2.113}$$

$$|n\rangle = \frac{(a^\dagger)^n}{\sqrt{n!}}|0\rangle \tag{2.114}$$

where $|0\rangle$ is the vacuum state. The Hamiltonian acting on $|n\rangle$ gives the energy eigenstates,

$$\hat{H}|n\rangle = \hbar\omega(n + \frac{1}{2})|n\rangle \tag{2.115}$$

which makes the vacuum state, $|0\rangle$ also the ground state and the reference state used to generate the coherent states. The stability group for H_4 is $U(1) \otimes U(1)$. $U(1)$ is the unitary group that consists of all complex numbers with absolute value 1

$$U(1) = \{z \in \mathbb{C} : |z| = 1\}. \tag{2.116}$$

The stability group is spanned by $\{\hat{N}, \boldsymbol{I}\}$ and $h \in U(1) \otimes U(1)$ is defined

$$h = e^{i(\varphi_{\boldsymbol{I}}\boldsymbol{I} + \varphi_{\hat{N}}\hat{N})} \tag{2.117}$$

with $\varphi_{\boldsymbol{I}}, \varphi_{\hat{N}}$ arbitrary coefficients this leaves the reference state invariant up to a phase factor,

$$h|0\rangle = |0\rangle e^{i\varphi_{\boldsymbol{I}}}. \tag{2.118}$$

The coset space $H_4/U(1)\otimes U(1)$ with elements Ω

$$g = \Omega h = Dh \tag{2.119}$$

with

$$D(\alpha) = e^{\alpha a^\dagger - \alpha^* a} \tag{2.120}$$

where α is an arbitrary complex number. Thus, a coherent state, $|\alpha\rangle$ can be defined

$$g|0\rangle = D(\alpha)h|0\rangle = D(\alpha)|0\rangle e^{i\varphi_{\boldsymbol{I}}} = |\alpha\rangle e^{i\varphi_{\boldsymbol{I}}}. \tag{2.121}$$

It is useful to state the Baker-Campbell-Hausdorff (BCH) identity which holds when the commutator $[A, B]$ commutes with both A and B

$$e^{A+B} = e^A e^B e^{-(1/2)[A,B]} \tag{2.122}$$

The overlap between two coherent states is

$$\langle\alpha|\beta'\rangle = exp(\alpha^*\beta - \frac{1}{2}(\alpha^*\alpha + \beta^*\beta)) \tag{2.123}$$

and the identity in terms of coherent states is given by

$$\boldsymbol{I} = \frac{1}{\pi}\int d^2\alpha |\alpha\rangle\langle\alpha| \tag{2.124}$$

as the coherent states are over-complete [26]. The displacement operator $D(\alpha)$ can be reformulated using the BCH identity

$$e^{\alpha a^\dagger - \alpha^* a} = e^{\alpha^*\alpha/2} e^{-\alpha^* a} e^{\alpha a^\dagger} = e^{-\alpha^*\alpha/2} e^{\alpha a^\dagger} e^{-\alpha^* a}. \tag{2.125}$$

Using this result the coherent state can be expanded in terms of number eigenstates

$$|\alpha\rangle = D(\alpha)|0\rangle = e^{-\alpha^*\alpha/2}\sum_0^\infty \frac{(\alpha a^\dagger)^n}{n!}|0\rangle = e^{-\alpha^*\alpha/2}\sum_0^\infty \frac{\alpha^n}{\sqrt{n!}}|n\rangle \tag{2.126}$$

which implies

$$|\alpha\rangle = e^{\frac{-|\alpha|^2}{2} + \alpha a^\dagger}|0\rangle \tag{2.127}$$

and

$$\langle\alpha|n\rangle = \frac{(\alpha^*)^n}{\sqrt{n!}} e^{-\alpha^*\alpha/2} \tag{2.128}$$

the arbitrary phase factor has been omitted in these equations. An arbitrary state can also be expanded in terms of the number eigenstates,

$$|\Psi\rangle = \sum_n c_n |n\rangle = \sum_N c_n \frac{(a^\dagger)^n}{n!}|0\rangle \tag{2.129}$$

the over-completeness relation can then be used,

$$|\Psi\rangle = \boldsymbol{I}|\psi\rangle = \frac{1}{\pi}\int d^2\alpha |\alpha\rangle\langle\alpha|\Psi\rangle = \frac{1}{\pi}\int d^2\alpha |\alpha\rangle \sum_N c_n \frac{(\alpha^*)^n}{\sqrt{n!}}. \tag{2.130}$$

The 1D harmonic oscillator Hamiltonian is of quadratic form, as in Eq. (2.99), so all Hamiltonians of the same form will have coherent states generated in the same way. In fact, using this general method can yield coherent states analogous to those detailed in the Coupled Coherent States method [109]. The general Hamiltonian is denoted $\hat{H}_{ord}(a^\dagger, a)$ as the powers of $a^\dagger$ precede those of the annihilation operator. Using an arbitrary coordinate parameter, γ, the creation and annihilation operators are then defined

$$a = \sqrt{\frac{\gamma}{2}}\hat{q} + \frac{i}{\hbar}\sqrt{\frac{1}{2\gamma}}\hat{p}, \quad a^\dagger = \sqrt{\frac{\gamma}{2}}\hat{q} - \frac{i}{\hbar}\sqrt{\frac{1}{2\gamma}}\hat{p}. \tag{2.131}$$

The so-called $|z\rangle$ notation is used where z is a complex number [26, 110, 111] which is equivalent to α in the previous notation

$$z = \alpha = \sqrt{\frac{\gamma}{2}}q + \frac{i}{\hbar}\sqrt{\frac{1}{2\gamma}}p, \quad z^* = \alpha^* = \sqrt{\frac{\gamma}{2}}q - \frac{i}{\hbar}\sqrt{\frac{1}{2\gamma}}p. \tag{2.132}$$

Hence, using Eq. (2.121) derived from the general method, a canonical Gaussian coherent state of the harmonic oscillator can be defined,

$$D(z)|0\rangle = e^{za^\dagger - z^* a}|0\rangle = e^{\frac{-|\alpha|^2}{2} + \alpha a^\dagger}|0\rangle = |z_{HO}\rangle. \tag{2.133}$$

The coherent state can then be defined in terms of x,

$$\begin{aligned}\langle x|z_{HO}\rangle &= \langle x|e^{\frac{-|z|^2}{2} + za^\dagger}|0\rangle \\ &= \sqrt[4]{\frac{\gamma}{\pi}}\exp\left(-\frac{\gamma}{2}(x-q)^2 + \frac{i}{\hbar}p(x-q) + \frac{ipq}{2\hbar}\right)\end{aligned} \tag{2.134}$$

by using the reasoning in Ref. [112] which demonstrates the equivalence of the CCS coherent states and those derived using the general method [109]. This is similar to the CCS method in general terms as in Ref. [113].

2.3.3 SU(2) Coherent States

The 1D harmonic oscillator Hamiltonian can now be generalised to the multi-dimensional case with an external field, γ, added

$$\hat{H} = \sum_i \hbar\omega_k a_i^\dagger a_i + \epsilon\sigma_0^{(i)} + \sum_{i,k} \gamma[\frac{\sigma_+^{(i)}}{\sqrt{N}} a_k + \frac{\sigma_-^{(i)}}{\sqrt{N}} a_k^\dagger]. \tag{2.135}$$

This can be simplified to just look at the two-level system by approximating the field as classical,

$$\hat{H} = \sum_i \Delta E\sigma_0^{(i)} + \gamma\sigma_+^{(i)} + \gamma^*\sigma_-^{(i)} \tag{2.136}$$

where $\{\sigma_0^{(i)}, \sigma_+^{(i)}, \sigma_-^{(i)}\}$ are spin operators with the commutation relation

$$[\sigma_0^{(i)}, \sigma_\pm^{(j)}] = \sigma_\pm^{(i)}\delta_{ij}. \tag{2.137}$$

Many particle operators can then be defined

$$J_0 = \sum_i \sigma_0^{(i)}, \quad J_\pm = \sum_i \sigma_\pm^{(j)} \tag{2.138}$$

which in turn have the commutation relations,

$$[J_0, J_\pm] = \pm J_\pm, \quad [J_+, J_-] = 2J_0. \tag{2.139}$$

The Hamiltonian can be rewritten in terms of these new operators,

$$\hat{H} = \Delta E J_0 + \gamma J_+ + \gamma^* J_- \tag{2.140}$$

which is a two-level system, with dynamical group $SU(2)$. The general process can then be applied by first defining the Hilbert space $\{|jm\rangle, m = -j, -j+1, \ldots, j-1, j\}$ where j is an integer for bosons or half integer for fermions. These are the Dicke states [114],

$$|jm\rangle = \sqrt{\frac{(2j)!}{(2j)!(j-m)!}} (J_+)^{j+m} |j-j\rangle \tag{2.141}$$

and the action of the operators is defined,

$$J^2|jm\rangle = j(j+1)|jm\rangle, \tag{2.142}$$

$$j_0|jm\rangle = m|jm\rangle. \tag{2.143}$$

The reference state is the lowest weight state, $|j-j\rangle$,

$$\hat{H}|j-j\rangle = \Delta E J_0|j-j\rangle = -\Delta E j|j-j\rangle \tag{2.144}$$

so $E_{min} = -\Delta E j$. The stability group for $SU(2)$ is $U(1)$ which has elements

$$h = \exp(i\alpha J_0) \tag{2.145}$$

which leaves the reference state invariant up to a phase factor,

$$h|j-j\rangle = |j-j\rangle e^{-i\alpha j}. \tag{2.146}$$

The coset space $SU(2)/U(1)$ with elements Ω can then be defined in the usual way

$$\Omega(\xi) = \exp(\xi J_+ - \xi^* J_-). \tag{2.147}$$

It is well known that $SU(2)/U(1)$ has the geometry of a two-dimensional sphere which allows the operators to be defined as matrices

$$J_+ \to \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix},\ J_- \to \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}\ J_0 \to \begin{pmatrix} 1/2 & 0 \\ 0 & -1/2 \end{pmatrix}. \tag{2.148}$$

Ω can then be defined

$$\Omega(\xi) \to \Omega = \exp\begin{pmatrix} 0 & \xi \\ -\xi^* & 0 \end{pmatrix} = \begin{pmatrix} cos|\xi| & \frac{\xi}{|\xi|}sin|\xi| \\ -\frac{\xi^*}{|\xi|}sin|\xi| & cos|\xi| \end{pmatrix}. \tag{2.149}$$

ξ can then be rewritten

$$\xi = \frac{\theta}{2}e^{-i\varphi} \quad 0 \le \theta \le \pi, 0 \le \varphi \le 2\pi. \tag{2.150}$$

Finally, the coherent states can be generated,

$$g|j-j\rangle = \Omega(\xi)h|j-j\rangle = |j,\xi\rangle e^{-i\alpha j} \tag{2.151}$$

which corresponds to a point on a three-dimensional sphere i.e. a point on its boundary which is a two-dimensional surface. There are a number of different expressions that can be generated for $D(z)$ for example,

$$\begin{aligned} D(z) &= \exp(\xi J_+ - \xi^* J_-) \\ &= \exp(\tau J_+)\exp[ln(1+|\tau|^2)J_0]\exp(-\tau^* J_-) \\ &= \exp(-\tau^* J_-)\exp[-ln(1+|\tau|^2)J_0]\exp(\tau J_+) \end{aligned} \tag{2.152}$$

where $\tau = tan(\frac{\theta}{2})e^{i\varphi}$ is the "normal" form [19, 107]. Using the normal form and Eq. (2.141) a coherent state can be written as,

$$\begin{aligned} |z\rangle &= D(z)|j-j\rangle = (1+|\tau|^2)^{-j}\exp(\tau J_+)|j-j\rangle \\ &= (1+|\tau|^2)^{-j}\sum_{0}^{\infty}\frac{(\tau J_+)^n}{n!}|j-j\rangle \\ &= \sum_{m=-j}^{+j}\sqrt{\frac{(2j)!}{(2j)!(j-m)!}}(cos(\theta/2))^{j-m}(sin(\theta/2))^{j+m}e^{-i(j+m)\varphi}|jm\rangle \end{aligned} \quad . \tag{2.153}$$

Thus an $SU(2)$ coherent state is formed out of the angular momentum coherent states. More explicitly, for the case $j = 1/2$, a coherent state can be defined by Eq. (2.153) to give,

$$|z_{\frac{1}{2}}\rangle = cos(\theta/2)|\frac{1}{2}\frac{-1}{2}\rangle + sin(\theta/2)e^{-i\varphi}|\frac{1}{2}\frac{1}{2}\rangle. \tag{2.154}$$

Alternatively, the coherent states can be derived explicitly from the reference state

$$|\Phi_0\rangle = |j-j\rangle = \begin{pmatrix}1\\0\end{pmatrix}. \tag{2.155}$$

The elements of the coset space are given by

$$\Omega = \begin{pmatrix} cos(\theta/2) & -e^{i\varphi}sin(\theta/2) \\ e^{-i\varphi}sin(\theta/2) & cos(\theta/2)\end{pmatrix}. \tag{2.156}$$

The arbitrary phase can be eliminated by setting $e^{-i\alpha j} = 1$ because only the difference between the phases is significant [115]. Using Eq. (2.105), an $SU(2)$ coherent state can be generated

$$|z\rangle = \Omega|\Phi_0\rangle = \begin{pmatrix} cos(\theta/2) \\ e^{i\varphi}sin(\theta/2)\end{pmatrix} \tag{2.157}$$

which can be more explicitly written as

$$|\zeta_{SU(2)}(\theta,\varphi)\rangle = \cos(\theta)|1\rangle + \sin(\theta)e^{i\varphi}|0\rangle. \tag{2.158}$$

The differing sign in the exponential between Eq. (2.154) and Eq. (2.158) is arbitrary and can be rectified by using different parametrisations. The coherent states generated in his way are analogous to those used in the multi-configurational Ehrenfest method which are used to describe two different quantum states or a single mixed occupied/unoccupied state in terms of the angular momentum states [9]. In the case of Ab *initio* MCE the coherent states are still made of the angular momentum states but generated using standard electronic structure theory to describe a real

system [116]. Further in Ref. [33] it was shown how the Hamiltonian and equations of motion could be derived for MCE in terms of Generalised Coherent States (GCS); could be used in the MCE method; they were equivalent to a single Ehrenfest trajectory. Significantly, it has been demonstrated that the general definition of an $SU(2)$ coherent can be generalised to produce $SU(n)$ coherent states [115, 117–119]. Using states generated on an equal footing, regardless of dimension, makes it much simpler to choose an appropriate size of spin coherent state for a specific system and so broadens a methods application.

The $SU(2)$ coherent states are often referred to as spin coherent states [120]. As such they can be used to describe a single fermion system. The system Hamiltonian is dependent on only a single pair of creation and annihilation operators. In this case the $SU(2)$ group is generated by the set of operators [100],

$$[a^\dagger, a] = 2(a^\dagger a - \frac{1}{2}), \quad [a^\dagger a - \frac{1}{2}, a] = -a, \quad [a^\dagger a - \frac{1}{2}, a^\dagger] = a^\dagger. \tag{2.159}$$

These operators are one-to-one with the angular momentum operators for spin $\frac{1}{2}$ and a coherent state can be generated in an analogous way to Eq. (2.154). In CCS an N-dimensional coherent state is constructed as product of 1D harmonic oscillators,

$$|\boldsymbol{z}\rangle = \prod_a |z_{HO}^{(a)}\rangle. \tag{2.160}$$

But like Hartree product in Eq. (2.11) a simple product of spin half coherent states, $|z_{\frac{1}{2}}\rangle$, does not preserve antisymmetry when constructing a multi-electron system. In the following section two standard solutions to construct multi-body fermionic coherent states are presented which then leads to the Zombie states method.

2.3.4 Standard Fermionic Coherent States

In this section two ways to construct many-fermionic coherent states are considered. Firstly, the general method using dynamical groups is used to generate fermionic coherent states. An alternative method that constructs coherent states using Grassmann algebra is also introduced. The basic idea of Grassmann algebra is introduced but more mathematical detail is given in Appendix A. The equations and their notation have been adapted from Ref. [20] in particular which has been supplemented by Refs. [18, 121, 122].

2.3.4.1 General Many-Fermion Coherent States

Balian and Brezin showed that fermionic coherent states of an even degree can be generally defined by first identifying the dynamical group $G = SO(2n)$ [123]. The coset space $G/H = SO(2n)/U(n)$ can be used to generate coherent states of an even degree, over a Hilbert space of dimension n [124, 125]. The corresponding Lie algebra $\mathfrak{g} = so(2n)$ is spanned by the operators

$$\begin{aligned} a_i^\dagger a_j - \frac{1}{2}\delta_{ij}, 1 \leq i,j \leq r \\ a_i a_j, 1 \leq i \neq j \leq r \\ a_i^\dagger a_j^\dagger, 1 \leq i \neq j \leq r \end{aligned} \tag{2.161}$$

and

$$[a_i^\dagger a_j - \frac{1}{2}\delta_{ij}, a_k^\dagger a_l - \frac{1}{2}\delta_{kl}] = \delta_{jk}(a_i^\dagger a_l - \frac{1}{2}\delta_{il}) - \delta_{il}(a_k^\dagger a_j - \frac{1}{2}\delta_{jk}) \tag{2.162a}$$

$$[a_i^\dagger a_j - \frac{1}{2}\delta_{ij}, a_k^\dagger a_l^\dagger] = \delta_{jk} a_i^\dagger a_l^\dagger - \delta_{jl} a_i^\dagger a_k^\dagger \tag{2.162b}$$

$$[a_i a_j, a_i^\dagger a_j^\dagger] = \delta_{ik}(a_l^\dagger a_j - \frac{1}{2}\delta_{lj}) + \delta_{lj}(a_k^\dagger a_i - \frac{1}{2}\delta_{ki}) - \delta_{li}(a_k^\dagger a_j - \frac{1}{2}\delta_{kj}) - \delta_{ki}(a_l^\dagger a_i - \frac{1}{2}\delta_{li}) \tag{2.162c}$$

are their commutation relations [19]. The basis sets for this space are $|\chi_1, \chi_2, \ldots, \chi_{N_{orb}}\rangle$ where $\chi_i = 0, 1$ then $N_{el} = \sum_{i=1}^{N_{orb}} \chi_i$ which is the number of occupied orbitals [126]. An even number of electrons allows coherent states to be generated from the vacuum whereas an odd number of electrons requires a different reference state,

$$|0\rangle = |0, 0, \ldots, 0\rangle \text{ for } N_{el} \text{ even} \tag{2.163}$$

$$|1\rangle = |1, 0, \ldots, 0\rangle \text{ for } N_{el} \text{ odd.} \tag{2.164}$$

The regular procedure to define a coherent state can then be carried out by first defining elements of the coset space. For simplicity just the even N_{el} case is considering, although an analogous process for N_{el} odd is possible. It was shown that $SO(2n)/U(n)$ is isomorphic to $Spin(2n)/U(n)$

$$\Omega = \exp(\sum_{1 \leq j \neq k \leq n} (\eta_{jk} a_j^\dagger a_k^\dagger - \eta_{jk}^* a_k a_j)). \tag{2.165}$$

where η_{jk} is a complex number [124]. The exterior algebra of the coset space has an anticommuting structure thus Ω can then be used to generate a coherent state $|z_{SO(2n)}\rangle = \Omega|0\rangle$ with the Pauli exclusion principle preserved. A one-to-one matrix representation of Ω can also be constructed [124, 127]. η is an $n \times n$ antisymmetric

complex matrix which represents the operator

$$\sum_{1\leq j\neq k\leq n} \eta_{jk} a_j^\dagger a_k^\dagger. \tag{2.166}$$

This can be constructed by,

$$a_i^\dagger a_l^\dagger \leftrightarrow E_{i,n+l} - E_{l,n+i} \tag{2.167}$$

$$a_i a_l \leftrightarrow E_{n+i,l} - E_{n+l,i} \tag{2.168}$$

for $1 \leq i, l \leq n$ and $E_{i,l}$ is a $2n \times 2n$ matrix with $+1$ in column i and row l and zeros everywhere else. Ω is then equivalent to,

$$\Omega \rightarrow \begin{pmatrix} \sqrt{\boldsymbol{I}_n - \xi\xi^\dagger} & \xi \\ -\xi^\dagger & \sqrt{\boldsymbol{I}_n - \xi^\dagger\xi} \end{pmatrix} \tag{2.169}$$

$\boldsymbol{I}_n$ is an $n \times n$ identity matrix and

$$\xi = \eta \frac{sin(\sqrt{\eta^\dagger\eta})}{\sqrt{\eta^\dagger\eta}}. \tag{2.170}$$

This matrix can then be used to generate fermionic coherent states. A similar process is available for the $SO(2n+1)$ Lie group which describes fermionic states with an even number of fermions distributed across an odd number of orbitals [19]. There is however little literature on fermionic coherent states of odd degree or considering the full Fock space i.e. the infinite dimensional case [128].

2.3.4.2 Grassmann Many-Fermion Coherent States

The use of Grassmann generators is now considered as a method for defining fermionic coherent states. An N dimensional system can be described by the usual creation and annihilation operators with the expected anticommutation relations. A Grassmann algebra denoted $\mathfrak{G}_N^C$ is also defined. Grassmann generators are complex numbers that have an anticommuting property with other Grassmann generators but commute with ordinary complex numbers. Thus,

$$\xi_i\xi_j = -\xi_j\xi_i, \quad \xi_i\alpha = \alpha\xi_i \quad \forall\xi_i, \xi_j \in \mathfrak{G}_N^C, \forall\alpha \in \mathbb{C}, \tag{2.171}$$

which for completeness gives the anticommutation relations,

$$\{\xi_i, \xi_j\} = 0, \quad \{\xi_i^*, \xi_j\} = 0, \quad \{\xi_i^*, \xi_j^*\} = 0. \tag{2.172}$$

Consequently, the square of a Grassmann generator vanishes, $(\xi_i)^2 = 0$. $\mathfrak{G}_N^C$ contains $2N$ Grassmann generators as each has a conjugate, $(\xi_i)^\dagger = \xi_i^*$. It is also defined that the Grassmann generators anticommute with the fermionic creation and annihilation operators (and commute with the bosonic)

$$\{\xi_i, a_j\} = 0, \quad \{\xi_i, a_j^\dagger\} = 0. \tag{2.173}$$

Further the Hermitian conjugate reverses the order of the operators and Grassmann generators,

$$(a_i \xi_j a_k^\dagger \xi_l^*)^\dagger = \xi_l a_k \xi_j^* a_i^\dagger. \tag{2.174}$$

As in Ref. [18] a unitary displacement operator can be defined for the Grassmann generators, $\mathfrak{G}_N^C$, with $\boldsymbol{\xi}$ used as shorthand,

$$\begin{aligned} D(\boldsymbol{\xi}) &= exp(\sum_i a_i^\dagger \xi_i - \xi_i^* a_i) \\ &= \prod_i exp(a_i^\dagger \xi_i - \xi_i^* a_i) \\ &= \prod_i [1 + a_i^\dagger \xi_i - \xi_i^* a_i + (a_i^\dagger a_i - \frac{1}{2}) \xi_i^* \xi_i], \quad \boldsymbol{\xi} = (\xi_1, \ldots, \xi_N) \end{aligned} \tag{2.175}$$

which is the multi-particle analogue to the displacement operator from Eq. (2.92) although with elements of Grassmann algebra rather than the complex numbers Glauber originally used [12]. The product is introduced in Eq. (2.175) because of the commutation relations,

$$\alpha_j a_j = -a_j \alpha_j, \quad \forall \alpha_j = \xi_j, \xi_j^* \tag{2.176a}$$

$$[a_j^\dagger \xi_j, \xi_k^* a_k] = \xi_j \xi_k^* \delta_{jk} \tag{2.176b}$$

$$[a_j, a_k^\dagger \xi_k] = \xi_k \delta_{jk} \tag{2.176c}$$

$$[a_j, a_k \xi_k] = 0. \tag{2.176d}$$

The translation property of these operators is

$$D^\dagger(\boldsymbol{\xi}) a_i D(\boldsymbol{\xi}) = a_i + \xi_i \tag{2.177a}$$

$$D^\dagger(\boldsymbol{\xi}) a_i^\dagger D(\boldsymbol{\xi}) = a_i^\dagger + \xi_i^* \tag{2.177b}$$

Using the BCH identity the displacement operator can be written in either normally or antinormally ordered form

$$D_{Normal}(\boldsymbol{\xi}) = D(\boldsymbol{\xi})exp(\frac{1}{2}\sum_i \xi_i^*\xi_i) = D(\boldsymbol{\xi})e^{\frac{\boldsymbol{\xi}^*\cdot\boldsymbol{\xi}}{2}} \tag{2.178a}$$

$$D_{Antinormal}(\boldsymbol{\xi}) = D(\boldsymbol{\xi})exp(\frac{-1}{2}\sum_i \xi_i^*\xi_i) = D(\boldsymbol{\xi})e^{\frac{-\boldsymbol{\xi}^*\cdot\boldsymbol{\xi}}{2}}. \tag{2.178b}$$

with the notation $\sum_i \xi_i^*\xi_i = \boldsymbol{\xi}^*\cdot\boldsymbol{\xi}$ used. Hence, a coherent state is defined as

$$\psi_{\boldsymbol{\xi}} = D(\boldsymbol{\xi})|0\rangle = |\boldsymbol{\xi}\rangle \tag{2.179}$$

where $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_N)$ is a set of Grassmann generators with each ξ_j corresponding to a spin orbital. Like in the second quantisation representation of Slater determinants the fermionic coherent states are constructed from the vacuum state $|0\rangle$. Using Eq. (2.175) a coherent state can be written in the form

$$\begin{aligned} |\boldsymbol{\xi}\rangle = D(\boldsymbol{\xi})|0\rangle &= \prod_i [1 + a_i^\dagger\xi_i - \xi_i^* a_i + (a_i^\dagger a_i - \frac{1}{2})\xi_i^*\xi_i]|0\rangle \\ &= \prod_i (1 + a_i^\dagger\xi_i - \frac{1}{2}\xi_i^*\xi_i)|0\rangle \\ &= exp(\sum_i (a_i^\dagger\xi_i - \frac{1}{2}\xi_i^*\xi_i)|0\rangle \end{aligned} \tag{2.180}$$

Further, using the translation property, defined in Eq. (2.177a), the action of annihilation operator,

$$\begin{aligned} a_i|\boldsymbol{\xi}\rangle &= a_i D(\boldsymbol{\xi})|0\rangle \\ &= D(\boldsymbol{\xi})D^\dagger(\boldsymbol{\xi})a_i D(\boldsymbol{\xi})|0\rangle \\ &= D(\boldsymbol{\xi})(a_i + \xi_i)|0\rangle = \xi_i D(\boldsymbol{\xi})|0\rangle \\ &= \xi_i|\boldsymbol{\xi}\rangle \end{aligned} \tag{2.181}$$

shows that the fermionic coherent states are eigenstates of the annihilation operator – Glauber's first definition of a coherent state. Further, the full state can be defined as $|1\rangle = \prod_i a_i^\dagger|0\rangle$ and Eq. (2.177b) can be used to show that $|\boldsymbol{\xi}\rangle'$ is an eigenstate of the creation operators

$$\begin{aligned} a_i^\dagger|\boldsymbol{\xi}\rangle' &= a_i^\dagger D(\boldsymbol{\xi})|1\rangle \\ &= D(\boldsymbol{\xi})D^\dagger(\boldsymbol{\xi})a_i^\dagger D(\boldsymbol{\xi})|1\rangle \\ &= D(\boldsymbol{\xi})(a_i^\dagger + \xi_i^*)|1\rangle = \xi_i^\dagger D(\boldsymbol{\xi})|1\rangle \\ &= \xi_i|\boldsymbol{\xi}\rangle'. \end{aligned} \tag{2.182}$$

Thus $|\boldsymbol{\xi}\rangle'$ can be defined explicitly,

$$|\boldsymbol{\xi}\rangle' = \prod_i (1 - \xi_i^* a_i + \frac{1}{2}\xi_i^* \xi_i)|1\rangle. \tag{2.183}$$

The inner product of two fermionic coherent states,$|\boldsymbol{\xi}\rangle$ and $|\boldsymbol{\gamma}\rangle$ constructed using the sets $\boldsymbol{\xi}$ and $\boldsymbol{\gamma}$ of Grassmann generators respectively,

$$\langle\boldsymbol{\xi}|\boldsymbol{\gamma}\rangle = \exp\sum_i (\xi_i^*\gamma_i - 1/2(\xi_i^*\xi_i + \gamma_i^*\gamma_i) = \exp(\boldsymbol{\xi}^* \cdot \boldsymbol{\xi} - 1/2(\boldsymbol{\xi}^*\boldsymbol{\xi} + \boldsymbol{\gamma}^* \cdot \boldsymbol{\gamma}). \tag{2.184}$$

Using this the following relation is found,

$$\langle\boldsymbol{\xi}|\boldsymbol{\gamma}\rangle\langle\boldsymbol{\gamma}|\boldsymbol{\xi}\rangle = \exp\left[-\sum_i (\xi_i^* - \gamma_i^*)(\xi_i - \gamma_i)\right] = \prod_i \left[1 - (\xi_i^* - \gamma_i^*)(\xi_i - \gamma_i)\right]. \tag{2.185}$$

In the second quantisation approach a Slater determinant with N electrons can be built from the vacuum state using N creation operators which exist in an N-dimensional Fock space labelled $\mathscr{H}_F^N$. For the fermionic coherent state analogue, it is also necessary to create an N-dimensional space, $\mathscr{H}^N$, to contain the state. This is a subspace of $\mathfrak{G}_{2N}$ constructed using holomorphic functions i.e. $\mathfrak{G}_N^C$ which gives it the following properties,

$$\psi \in \mathscr{H}^N \Rightarrow \tag{2.186}$$

$$\frac{\partial}{\partial \xi_j^*}\psi = 0, \quad \forall\, 1 \leq j \leq N. \tag{2.187}$$

$\mathscr{H}^N$ has the basis $\{\xi^\epsilon, \epsilon \in \mathfrak{E}[\mathbf{N}]\}$, with $\mathfrak{E}[\mathbf{N}]$ being the indexing set as previously defined. An unambiguous description of the specific fermionic state is given by an ordered set of creation operators acting on the vacuum state. The ordered set of creation operators gives a configuration that can be written in a position basis and so it is now possible to construct a multi-electron fermionic coherent state that is an equivalent representation to a Slater determinant constructed from the same set of creation operators. This can be summarised as being a map from the Fock space to the $\mathscr{H}^N$ a Hilbert space,

$$f_\psi : \mathscr{H}_F^N \mapsto \mathscr{H}^N f(|\psi(r_1, r_2, \ldots, r_N)\rangle) = \psi(\xi_1)\psi(\xi_2)\ldots\psi(\xi_N) = \psi(\boldsymbol{\xi}) \tag{2.188}$$

where $|\psi\rangle$ is an arbitrary vector in the Fock space its electronic configuration described by a position vector. f_ψ is a one-to-one mapping from the position basis to the Grassmann generator products. A fermionic state can then be specified using the

fermionic coherent state definition and the known action of the creation operator,

$$|\psi\rangle = \sum_{\epsilon\in\mathfrak{E}[\mathbf{N}]} c_\epsilon(\psi) a_\epsilon^\dagger |0\rangle \tag{2.189}$$

using this definition and Eq. (2.180) a fermionic coherent state with a specified set of occupied orbitals is represented by,

$$\psi(\boldsymbol{\xi}) = \langle \boldsymbol{\xi}|\psi\rangle = e^{\frac{-\boldsymbol{\xi}^*\cdot\boldsymbol{\xi}}{2}} \sum_{\epsilon\in\mathfrak{E}[\mathbf{N}]} c_\epsilon(\psi)\xi^{*\epsilon} = e^{\frac{-\boldsymbol{\xi}^*\cdot\boldsymbol{\xi}}{2}} \sum_{\epsilon\in\mathfrak{E}[\mathbf{N}]} \psi_\epsilon \tag{2.190}$$

Thus, a fermionic coherent state can be constructed from the vacuum state and is also shown to be equivalent to the Slater determinant representation. $\mathscr{H}^N$ is a Hilbert space for the scalar product,

$$\langle\psi|\varphi\rangle = \int e^{\boldsymbol{\xi}^*} e^{\boldsymbol{\xi}} \psi^*(\boldsymbol{\xi})\varphi(\boldsymbol{\xi}) d\boldsymbol{\xi} d\boldsymbol{\xi}^* \tag{2.191}$$

which makes $\mathscr{H}^N$ a Hilbert space with $2N$ dimensions over $\mathbb{C}$ and is now isomorphic to the Fock space $\mathscr{H}_F^N$. The creation and annihilation operators can then be defined

$$(a_j^\dagger\psi)(\boldsymbol{\xi}) = \xi_j\psi(\boldsymbol{\xi}) \tag{2.192a}$$

$$a_j\psi(\boldsymbol{\xi}) = \frac{\partial}{\partial\xi_j}\psi(\boldsymbol{\xi}). \tag{2.192b}$$

Operators in in the Fock space can also be written as numbers in the $\mathfrak{G}_{2N}$ space. A second set of Grassmann generators $\{\boldsymbol{\gamma},\boldsymbol{\gamma}^*\}$ which have the same properties as $\{\boldsymbol{\xi},\boldsymbol{\xi}^*\}$ are defined. Then an operator $\hat{A}$ acting on a fermionic coherent state is,

$$\hat{A}\psi(\boldsymbol{\xi}) = \int K_A(\boldsymbol{\xi},\boldsymbol{\gamma}^*)\psi(\boldsymbol{\xi}) e^{\boldsymbol{\gamma}^*\cdot\boldsymbol{\gamma}} d\boldsymbol{\gamma} d\boldsymbol{\gamma}^*, \forall\psi\in\mathscr{H}^N. \tag{2.193}$$

$K_A \in \mathbb{C}[\xi,\gamma^*]$ is the so-called integral kernel that can be computed

$$K_A(\boldsymbol{\xi},\gamma^*) = \sum_{\epsilon,\epsilon'\in\mathfrak{E}[\mathbf{N}]} \langle\psi_{\epsilon'}|\hat{A}|\psi_\epsilon\rangle \psi_{\epsilon'}(\boldsymbol{\xi})\psi_\epsilon(\boldsymbol{\gamma})^* \tag{2.194}$$

the sums are over the indexing sets and so $\psi_\epsilon(\boldsymbol{\gamma}) = f(|\psi_\epsilon\rangle)$. Which gives the result that all linear operators have a unique decomposition

$$\hat{A} = \sum_{\epsilon,\epsilon'\in\mathfrak{E}[\mathbf{N}]} A_{\epsilon\epsilon'} a^{\dagger\epsilon'} a^\epsilon \tag{2.195}$$

where $A_{\epsilon\epsilon'} \in \mathbb{C}$.

For a single mode a complete set of operators are formed by a, $a^\dagger$, $a^\dagger a - 1/2$ and the identity. Each operator can be written as an integral over the displacement

operators,

$$\int d\boldsymbol{\xi}^* d\boldsymbol{\xi}(\boldsymbol{\xi}^*\boldsymbol{\xi})D((\xi)) = \int d\xi^* d\xi \xi^* \xi[1+a^\dagger\xi-\xi^* a+(a^\dagger a-\frac{1}{2})\xi^*\xi] = \boldsymbol{I} \quad (2.196)$$

$$\int d\boldsymbol{\xi}^* d\boldsymbol{\xi}(\boldsymbol{\xi})D((\xi)) = \int d\xi^* d\xi(\xi)[1+a^\dagger\xi-\xi^* a+(a^\dagger a-\frac{1}{2})\xi^*\xi] = a \quad (2.197)$$

$$\int d\boldsymbol{\xi}^* d\boldsymbol{\xi}(-\boldsymbol{\xi}^*)D((\xi)) = \int d\xi^* d\xi(-\xi^*)[1+a^\dagger\xi-\xi^* a+(a^\dagger a-\frac{1}{2})\xi^*\xi] = a^\dagger \quad (2.198)$$

$$\int d\boldsymbol{\xi}^* d\boldsymbol{\xi} D((\xi)) = \int d\xi^* d\xi[1+a^\dagger\xi-\xi^* a+(a^\dagger a-\frac{1}{2})\xi^*\xi] = a^\dagger a-\frac{1}{2}. \quad (2.199)$$

Thus, the displacement operators form a complete set of operators for that mode. Which is the same set of operators as in that generate an SU(2) coherent state for single fermion. This can be easily generalised for the multimode case to give the set of operators,

$$\{a_i a_j, a_i^\dagger a_j^\dagger, a_i^\dagger a_j - \frac{1}{2}\delta_{ij}\} \quad (2.200)$$

defined for $1 \leq i,j \leq N$ and form a complete set for all modes. These are the same set of operators in Eq. (2.161) that generate the $SO(2n)$ Lie group. In fact, it is possible to project a set of $2N$ Grassmann generators on to a Lie algebra of $so(4n)$, half the generators being even and the other odd [129]. Thus, the Grassmann and general definition of fermionic coherent states ensures that any state that is eigenstate of the number operator must remain as such.

2.4 Zombie states

The two standard ways of generating fermionic coherent states discussed, use the natural anticommuting properties of either the $SO(2n)$ Lie or Grassmann generators to ensure the correct antisymmetry of the fermionic state. Both constructions also create coherent states that are eigenstates of the number operator which is strictly conserved. It has also been shown how a single fermion can be described using an $SU(2)$ coherent state that is a superposition of the two spin states of the fermion. The Zombie states method takes a different approach describing a single fermion state as an $SU(2)$ coherent state of a two-level system which is defined as a superposition of the occupied and unoccupied states. This can then be extended to describe a multi-fermion state.

The Zombie state method will be formalised showing how action of ZS creation and annihilation operators can be used to compute the second quantisation Hamiltonian [9]. Further, it will be demonstrated that Zombie states can be constructed from a vacuum state [23]. Zombie states method will then be constructed using analogous apparatus to the general coherent states method. This then allows direct comparison between the construction of general fermionic coherent states, Grass-

mann algebra coherent states and Zombie states. Note that due to the introduction of Zombie state coefficients, creation and annihilation operators are represented as both $b_i^\dagger$ and b_i where necessary to avoid confusion and keep the notation consistent with previously published work.

2.4.1 Construction

The i^{th} spin orbital of a Zombie state is given as

$$|\zeta(a_{1i}, a_{0i})\rangle = a_{1i}|1_i\rangle + a_{0i}|0_i\rangle \tag{2.201}$$

This is a coherent state consisting of antisymmetrized superpositions of "dead" and "alive" electronic states rather than the two spin states of an electron [9]. $|1_i\rangle$ corresponds to their being an electron occupying spin orbital i and $|0_i\rangle$ to the i^{th} spin orbital being empty. In a similar way to Eq. (2.16) an N-particle Slater determinant can be constructed from one-electron Zombie states

$$|\boldsymbol{\zeta}\rangle = |\zeta_1\zeta_2\ldots\zeta_N\rangle \tag{2.202}$$

which is described by $2N$ coefficients

$$|\boldsymbol{\zeta}\rangle = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1(i-1)} & a_{1i} & a_{1(i+1)} & \ldots & a_{1N} \\ a_{01} & a_{02} & \ldots & a_{0(i-1)} & a_{0i} & a_{0(i+1)} & \ldots & a_{0N} \end{bmatrix}. \tag{2.203}$$

Note that in $a_{m_i i}$, $m_i = 0$ refers to the ith spin orbital being 'dead' (unoccupied) and $m_i = 1$ to the ith spin orbital being 'alive' (occupied), and i refers to the spin orbital number – this is the same notation used in Refs. [9, 23]. As detailed in Ref. [130] and shown in Eq. (2.16) a given electronic occupancy can be written as an antisymmetrised Slater determinant which is equivalent to Eq. (2.213). This can be equivalently stated using second quantisation notation

$$|\varphi\rangle = \prod_{k\ \mathrm{occ}} \hat{b}_k^\dagger |0\rangle \tag{2.204}$$

where $|0\rangle$ is the vacuum state, ensuring the creation operators are applied in the reverse order they appear in the Slater determinant. Trivially Eq. (2.204) can be rewritten as

$$|\varphi\rangle = \prod_{j=1}^{N} [(1-m_j)\hat{I} + m_j \hat{b}_j^\dagger]|0\rangle \tag{2.205}$$

and generalized to

$$|\boldsymbol{\zeta}\rangle = \prod_{j=1}^{N} (a_{0j}\hat{I} + a_{1j}\hat{b}_j^\dagger)|0\rangle, \tag{2.206}$$

where a_{0j} and a_{1j} are complex scalar coefficients. The same Slater determinant is recovered if $a_{1j} = m_j$ and $a_{0j} = (1 - m_j)$ as found in Eq. (2.205). The Zombie operator and its adjoint are then defined

$$\hat{z}_j =: a_{0j}\hat{I} + a_{1j}\hat{b}_j^\dagger, \tag{2.207a}$$
$$\hat{z}_j^\dagger =: a_{0j}^*\hat{I} + a_{1j}^*\hat{b}_j, \tag{2.207b}$$

giving rise to commutator and anti-commutator relations

$$[\hat{z}_j, \hat{z}_k] = 2\hat{b}_j^\dagger\hat{b}_k^\dagger a_{1j}a_{1k} \tag{2.208a}$$
$$[\hat{z}_j^\dagger, \hat{z}_k^\dagger] = 2\hat{b}_j\hat{b}_k a_{1j}a_{1k} \tag{2.208b}$$
$$\{\hat{z}_j, \hat{z}_k\} = 2(a_{0j}a_{0k}\hat{I} + a_{1j}a_{0k}\hat{b}_j^\dagger + a_{0j}a_{1k}\hat{b}_k^\dagger)\hat{I} \tag{2.208c}$$
$$\{\hat{z}_j^\dagger, \hat{z}_k\} = 2(a_{0j}a_{0k}\hat{I} + a_{1j}a_{0k}\hat{b}_j + a_{0j}a_{1k}\hat{b}_k). \tag{2.208d}$$

Therefore, Eq. (2.206) can be rewritten

$$|\boldsymbol{\zeta}\rangle = \prod_{j=1}^{N} \hat{z}_j|0\rangle. \tag{2.209}$$

The notation $\hat{z}_j \equiv \hat{z}_j(a_{0j}, a_{1j})$ is introduced for neatness.

2.4.2 Creation and Annihilation Operators

The action of creation and annihilation operators acting on a single orbital is defined

$$\hat{b}|\boldsymbol{\zeta}\rangle = \hat{b}(a_1|1\rangle + a_0|0\rangle) = 0|1\rangle + a_1|0\rangle \tag{2.210}$$
$$\hat{b}^\dagger|\boldsymbol{\zeta}\rangle = \hat{b}^\dagger(a_1|1\rangle + a_0|0\rangle) = a_0|1\rangle + 0|0\rangle. \tag{2.211}$$

When extended to the multi-electron case an additional sign change rule is required which preserves the antisymmetry of the fermionic function

$$\hat{b}_i^\dagger|\boldsymbol{\zeta}^{(b)}\rangle = \begin{bmatrix} -a_{11}^{(b)} & -a_{12}^{(b)} & \dots & -a_{1(i-1)}^{(b)} & a_{0i}^{(b)} & a_{1(i+1)}^{(b)} & \dots & a_{1N}^{(b)} \\ a_{01}^{(b)} & a_{02}^{(b)} & \dots & a_{0(i-1)}^{(b)} & 0 & a_{0(i+1)}^{(b)} & \dots & a_{0N}^{(b)} \end{bmatrix}, \tag{2.212a}$$

$$\hat{b}_i|\boldsymbol{\zeta}^{(b)}\rangle = \begin{bmatrix} -a_{11}^{(b)} & -a_{12}^{(b)} & \dots & -a_{1(i-1)}^{(b)} & 0 & a_{1(i+1)}^{(b)} & \dots & a_{1N}^{(b)} \\ a_{01}^{(b)} & a_{02}^{(b)} & \dots & a_{0(i-1)}^{(b)} & a_{1i}^{(b)} & a_{0(i+1)}^{(b)} & \dots & a_{0N}^{(b)} \end{bmatrix}. \tag{2.212b}$$

The operators $\hat{b}_i^\dagger$ and $\hat{b}_i$ not only act on the i-th orbital but also change sign of all the alive amplitudes a_{1j} for all orbitals with $j < i$ leaving the dead amplitudes unchanged. A standard Hartree Fock configuration $|\varphi_{m_e}^{(j)}\rangle$ which corresponds to m_e electrons can be written as a Zombie state with "binary" amplitudes of dead and alive states and m_e ones in the upper row

$$|\varphi_{m_e}^{(j)}\rangle = \begin{bmatrix} 1 & 0 & \dots & 0 & 1 & 0 & \dots & 1 \\ 0 & 1 & \dots & 1 & 0 & 1 & \dots & 0 \end{bmatrix}. \tag{2.213}$$

In this case the sign change rule becomes equivalent to the Wigner-Jordan rule, but it can be treated just like any other ZS. As previously shown, Slater determinants can be constructed by a set of creation operations on the vacuum state. Similarly, the action of the creation and annihilation operators can be extended to allow Zombie states to be constructed from the vacuum state. First the action on a single Zombie operator is defined for $j \neq k$,

$$\hat{b}_j \hat{z}_k(a_{0k}, a_{1k}) = \hat{b}_j(a_{0k}\hat{I} + a_{1k}\hat{b}_k^\dagger) = (a_{0k}\hat{I} - a_{1k}\hat{b}_k^\dagger)\hat{b}_j = \hat{z}_k(a_{0k}, -a_{1k})\hat{b}_j, \tag{2.214a}$$
$$\hat{b}_j^\dagger \hat{z}_k(a_{0k}, a_{1k}) = \hat{b}_j^\dagger(a_{0k}\hat{I} + a_{1k}\hat{b}_k^\dagger) = (a_{0k}\hat{I} - a_{1k}\hat{b}_k^\dagger)\hat{b}_j^\dagger = \hat{z}_k(a_{0k}, -a_{1k})\hat{b}_j^\dagger, \tag{2.214b}$$

and if $j = k$,

$$\begin{aligned} \hat{b}_j \hat{z}_j(a_{0j}, a_{1j}) &= \hat{b}_j(a_{0j}\hat{I} + a_{1j}\hat{b}_j^\dagger) = a_{0j}b_j - a_{1k}(1 - \hat{b}_j^\dagger \hat{b}_j) \\ &= \hat{z}_j(a_{0j}, -a_{1j})\hat{b}_j + \hat{z}_j(a_{1j}, 0), \end{aligned} \tag{2.215a}$$
$$\hat{b}_j^\dagger \hat{z}_j(a_{0j}, a_{1j}) = \hat{b}_j^\dagger(a_{0j}\hat{I} + a_{1j}\hat{b}_j^\dagger) = a_{0j}\hat{b}_j^\dagger = \hat{z}_j(0, a_{0j}), \tag{2.215b}$$

using the standard anticommutator relations Eq. (2.29c).

As before this can be generalised to the action on a whole Zombie state (which is the product of Zombie operators as in Eq. (2.209))

$$\begin{aligned} \hat{b}_j|\boldsymbol{\zeta}\rangle &= \hat{b}_j \prod_{k=1}^{N} \hat{z}_k(a_{0k}, a_{1k}) \\ &= \left[\prod_{k=1}^{j-1} \hat{z}_k(a_{0k}, -a_{1k})\right] \left[\hat{z}_j(a_{0j}, -a_{1j})\hat{b}_j + \hat{z}_j(a_{1j}, 0)\right] \left[\prod_{k=j+1}^{N} \hat{z}_k(a_{0k}, a_{1k})\right] |0\rangle \\ &= \left[\prod_{k=1}^{j-1} \hat{z}_k(a_{0k}, -a_{1k})\right] \hat{z}_j(a_{1j}, 0) \left[\prod_{k=j+1}^{N} \hat{z}_k(a_{0k}, a_{1k})\right] |0\rangle \end{aligned} \tag{2.216}$$

where the $\hat{z}_j(a_{0j}, -a_{1j})\hat{b}_j$ term vanishes because $\hat{b}_j$ tries to destroy an electron which

is not there. Hence,

$$\begin{aligned}\hat{b}_j^\dagger|\boldsymbol{\zeta}\rangle &= \hat{b}_j^\dagger \prod_{k=1}^{N} \hat{z}_k(a_{0k}, a_{1k}) \\ &= \left[\prod_{k=1}^{j-1} \hat{z}_k(a_{0k}, -a_{1k})\right] \hat{z}_j(0, a_{0j}) \left[\prod_{k=j+1}^{N} \hat{z}_k(a_{0k}, a_{1k})\right] |0\rangle.\end{aligned} \tag{2.217}$$

The annihilation operation, Eq. (2.216), is equivalent to the equations derived from the sign changing rules, Eq. (2.214a) and Eq. (2.215a) and the same is true for the creation operator, Eq. (2.217), to Eq. (2.214b) and Eq. (2.215b). This equivalence can be made more explicit since, $\{a_{m_j j}\} = \mathbf{a}$ then $|\zeta\rangle \equiv |\zeta(\mathbf{a})\rangle$ and so these results can be stated in terms of Zombie coefficients $\{a_{n_j j}\}$ by stating $|\zeta^{(j)}(\mathbf{a}^{(j)})\rangle = \hat{b}_j|\zeta(\mathbf{a})\rangle$. Thus,

$$a_{0k}^{(j)} = \begin{cases} a_{0k} & k \neq j \\ a_{1k} & k = j \end{cases} \tag{2.218a}$$

$$a_{1k}^{(j)} = \begin{cases} -a_{1k} & k < j \\ 0 & k = j \\ a_{1k} & k > j \end{cases} \tag{2.218b}$$

and if $|\boldsymbol{\zeta}^{(j)}(\mathbf{a}^{(j)})\rangle = \hat{b}_j^\dagger|\boldsymbol{\zeta}(\mathbf{a})\rangle$ then,

$$a_{0k}^{(j)} = \begin{cases} a_{0k} & k \neq j \\ 0 & k = j \end{cases} \tag{2.219a}$$

$$a_{1k}^{(j)} = \begin{cases} -a_{1k} & k < j \\ a_{0k} & k = j \\ a_{1k} & k > j \end{cases} \tag{2.219b}$$

2.4.3 Overlap of Two Zombie states

The overlap, $\langle\boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}^{(b)}\rangle$, between two Zombie states, each formed by the action of M creation operators, can now be computed. Using Eq. (2.209) and defining

$$|\boldsymbol{\zeta}_j^{(b)}\rangle = \prod_{k=j}^{N} \hat{z}_k|0\rangle \tag{2.220a}$$

$$|\boldsymbol{\zeta}_1^{(b)}\rangle \equiv |\boldsymbol{\zeta}^{(b)}\rangle, \tag{2.220b}$$

$$|\boldsymbol{\zeta}_{N+1}^{(b)}\rangle \equiv |0\rangle. \tag{2.220c}$$

Therefore,

$$\begin{aligned}\langle\boldsymbol{\zeta}_j^{(a)}|\boldsymbol{\zeta}_j^{(b)}\rangle &= \langle\boldsymbol{\zeta}_{j+1}^{(a)}|\hat{z}_j^{(a)*}\hat{z}_j^{(b)}|\boldsymbol{\zeta}_{j+1}^{(b)}\rangle \\ &= \langle\boldsymbol{\zeta}_{j+1}^{(a)}|a_{0j}^{(a)*}a_{0j}^{(b)}\hat{I} + a_{0j}^{(a)*}a_{1j}^{(b)}\hat{b}_j^\dagger + a_{1j}^{(a)*}a_{0j}^{(b)}\hat{b}_j + a_{1j}^{(a)*}a_{1j}^{(b)}\hat{b}_j\hat{b}_j^\dagger|\boldsymbol{\zeta}_{j+1}^{(b)}\rangle \\ &= (a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)})\langle\boldsymbol{\zeta}_{j+1}^{(a)}|\boldsymbol{\zeta}_{j+1}^{(b)}\rangle\end{aligned} \tag{2.221}$$

The second term in Eq. (2.221) is of the form $\langle\zeta_{j+1}^{(a)}|\hat{b}_j^\dagger|\zeta_{j+1}^{(b)}\rangle$ with neither state having a contribution from electron j the $\hat{b}_j^\dagger$ operator acting to the left will try to destroy a non-existent electron giving zero. Similarly, the third term in Eq. (2.221) vanishes, but the fourth term survives. Combining Eq. (2.220c) and Eq. (2.221) gives,

$$\langle\boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}^{(b)}\rangle = \prod_{j=1}^{N}(a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)}) \tag{2.222}$$

Which can be succinctly written as

$$\Omega_{ab} = \langle\boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}^{(b)}\rangle = \prod_{i=1}^{M}\sum_{m_i=0,1} a_{m_i i}^{(a)*}a_{m_i i}^{(b)}. \tag{2.223}$$

2.4.4 Normalisation

For a Zombie state to be normalised Eq. (2.222) must equal unity when $|\boldsymbol{\zeta}^{(a)}\rangle = |\boldsymbol{\zeta}^{(b)}\rangle$

$$\prod_{j=1}^{N}|a_{0j}^{(a)}|^2 + |a_{1j}^{(a)}|^2 = 1, \tag{2.224}$$

ensuring $\langle\boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}^{(a)}\rangle \equiv \langle\boldsymbol{\zeta}_1^{(a)}|\boldsymbol{\zeta}_1^{(a)}\rangle = 1$. However, a more general case can be considered where the wave function may not necessarily have a fixed number of Zombie operators which gives the requirement

$$\langle\boldsymbol{\zeta}_j^{(a)}|\boldsymbol{\zeta}_j^{(a)}\rangle = 1 \tag{2.225}$$

for all j, $1 \leq j \leq N$. From Eq. (2.220c) and Eq. (2.221) it follows that

$$\langle\boldsymbol{\zeta}_N^{(a)}|\boldsymbol{\zeta}_N^{(a)}\rangle = 1 = |a_{0N}^{(a)}|^2 + |a_{1N}^{(a)}|^2 \tag{2.226a}$$

$$\langle\boldsymbol{\zeta}_j^{(a)}|\boldsymbol{\zeta}_j^{(a)}\rangle = (|a_{0j}^{(a)}|^2 + |a_{1j}^{(a)}|^2)\langle\boldsymbol{\zeta}_{j+1}^{(a)}|\boldsymbol{\zeta}_{j+1}^{(a)}\rangle \tag{2.226b}$$

and using the fact $\langle\boldsymbol{\zeta}_{N+1}^{(a)}|\boldsymbol{\zeta}_{N+1}^{(a)}\rangle \equiv \langle 0|0\rangle = 1$, for Eq. (2.225) to be true for all j

$$|a_{0j}^{(a)}|^2 + |a_{1j}^{(a)}|^2 = 1 \ \forall j \tag{2.227}$$

This is a stronger criterion than Eq. (2.224) so satisfying Eq. (2.227) will ensure that Eq. (2.224) is satisfied (satisfying Eq. (2.224) does not mean Eq. (2.227) is). This normalisation is equivalent to the normalisation used when introducing quantum superposition sampling to the Multiconfigurational Ehrenfest method presented in Ref.[33]. Following this work a simple choice to satisfy Eq. (2.227) is

$$a_{0j}^{(a)} = \cos(\theta_j), \quad a_{1j}^{(a)} = \sin(\theta_j) \tag{2.228}$$

where θ_j is a real number such that $0 \leq \theta_j < 2\pi$. This means that a real, normalised Zombie state can be completely described by N real $\{\theta_j\}$ values. This can be generalized to complex states with

$$a_{0j}^{(a)} = \cos(\theta_j), \quad a_{1j}^{(a)} = \sin(\theta_j)e^{i\phi_j} \tag{2.229}$$

where ϕ_j is drawn from the interval $0 \leq \phi_j < 2\pi$ [33].

2.4.5 The Zombie Wave Function

Elements of the second quantized electronic structure Hamiltonian, can now be found for Zombie states, $\langle \boldsymbol{\zeta}^{(a)}|\hat{H}|\boldsymbol{\zeta}^{(b)}\rangle$. Sequential application of the creation and annihilation operators to $|\boldsymbol{\zeta}^{(b)}\rangle$ and overlapping the result with $\langle \boldsymbol{\zeta}^{(a)}|$ using Eq. (2.223)

$$H_{ab} = \langle \boldsymbol{\zeta}^{(a)}|\hat{H}|\boldsymbol{\zeta}^{(b)}\rangle = \sum_{ij} h_{ij}\langle \boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}_{ij}^{(b)}\rangle + \frac{1}{2}\sum_{klij} W_{klji}\langle \boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}_{klij}^{(b)}\rangle, \tag{2.230}$$

where $|\boldsymbol{\zeta}_{klij}^{(b)}\rangle = \hat{b}_k^\dagger \hat{b}_l^\dagger \hat{b}_i \hat{b}_j |\boldsymbol{\zeta}^{(b)}\rangle$ and $|\boldsymbol{\zeta}_{ij}^{(b)}\rangle = b_i^\dagger \hat{b}_j |\boldsymbol{\zeta}_{ij}^{(b)}\rangle$ [9]. An electronic wave function can be represented as a superposition of N_{bf} basis Zombie states

$$|\Psi\rangle = \sum_a^{N_{bf}} d_a |\boldsymbol{\zeta}^{(a)}\rangle. \tag{2.231}$$

and the matrix elements described above now allow usage of the ansatz Eq. (2.231) for propagation or finding quantum states and their energies. Unlike the other fermionic coherent state constructions, the Zombie states are not eigenstates of the number operator since they are a superposition of all states of any number. Therefore, a wave function of the form in Eq. (2.231) requires a sufficiently large N_{bf} number of Zombie states with coefficients d_a to be optimised such that contributions from unwanted numbers of electrons cancel out. This is a key departure from standard fermionic coherent states and Hartree Fock methods where each basis function has a fixed number of electrons. However, this could be advantageous when investigating molecular dynamics when the process is driven by electrons being added or removed from the system.

2.4.6 General Coherent Zombie states

In Eq. (2.157) it was shown how a single $SU(2)$ coherent state could be generated using the language of generalised coherent states. For a single fermion this can be explicitly defined as

$$|\xi\rangle = \exp\left(\xi b^\dagger - \xi^* b\right)|0\rangle = sin(\theta/2)e^{-i\varphi}|1\rangle + cos(\theta/2)|0\rangle \tag{2.232}$$

where $\xi = (\theta/2)e^{-i\phi}$ and $|0\rangle = |\frac{1}{2}\frac{-1}{2}\rangle$, $|1\rangle = |\frac{1}{2}\frac{1}{2}\rangle$. This general definition takes the same form as the single fermion displacement operator used to construct Grassmann type coherent states, Eq. (2.175). Thus, an N-particle Zombie state can be constructed with a Zombie displacement operator,

$$\begin{aligned} D(\boldsymbol{\zeta}) &= \exp\left(\sum_i b_i^\dagger \zeta_i - \zeta_i^* b_i\right) \\ &= \prod_i \exp\left(b_i^\dagger \zeta_i - \zeta_i^* b_i\right) \quad \boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_N) \end{aligned} \tag{2.233}$$

where $\zeta_j = (\theta_j/2)e^{-i\phi_j}$. A Zombie state can then be defined by the displacement operator acting on the vacuum state $|\mathbf{0}\rangle$,

$$D(\boldsymbol{\zeta})|\mathbf{0}\rangle = \prod_i \exp\left(b_i^\dagger \zeta_i - \zeta_i^* b_i\right)|\mathbf{0}\rangle = \prod_i sin(\theta_i/2)e^{-i\varphi_i}|1_i\rangle + cos(\theta_i/2)|0_i\rangle \tag{2.234}$$

where $|1_i\rangle$ is the occupied state for orbital i and $|0_i\rangle$ is the corresponding unoccupied state. The vacuum state is defined as,

$$|\mathbf{0}\rangle = \prod_i |0_i\rangle \tag{2.235}$$

Thus, Zombie states can be constructed using a method derived from the general method of constructing coherent states. The action of the displacement operator on the vacuum state satisfying Glauber's definition of a coherent state. Using this definition the normalisation condition, Eq. (2.227), is satisfied by construction.

When originally formulated the antisymmetric property was ensured by creating a Slater determinant of one-electron Zombie states and using a sign change rule with the creation and annihilation operators. The fermionic coherent states generated from the $SO(2n)$ Lie group have a basis that equipped with an exterior algebra that ensures anticommutation relation is preserved. Further, the Grassmann generators also anticommute with each other and also the creation and annihilation operators. The Zombie states defined using Eq. (2.234) naturally produce the sign change behaviour when operated on by a creation or annihilation operator. An arbitrary

Zombie state is defined by an ordered set of coefficients $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_i, \zeta_j, \ldots, \zeta_N)$,

$$|\boldsymbol{\zeta}\rangle = D(\boldsymbol{\zeta})|\mathbf{0}\rangle = \left(\prod_{k=1}^{i} \exp\left(b_k^\dagger \zeta_k - \zeta_k^* b_k\right)|\mathbf{0}\rangle\right) \cdot \left(\exp\left(b_j^\dagger \zeta_j - \zeta_j^* b_j\right)|\mathbf{0}\rangle\right) \cdot \left(\prod_{k=j+1}^{N} \exp\left(b_k^\dagger \zeta_k - \zeta_k^* b_k\right)|\mathbf{0}\rangle\right). \tag{2.236}$$

The ZS produced is then $|\boldsymbol{\zeta}\rangle = |\zeta_1, \ldots, \zeta_i, \zeta_j, \ldots, \zeta_N\rangle$. To operate on orbital j it must be the left most within the ket and so the Zombie state can be rewritten,

$$a_j|\boldsymbol{\zeta}\rangle = b_j\left(\exp\left(b_j^\dagger \zeta_j - \zeta_j^* b_j\right)|\mathbf{0}\rangle\right) \cdot \left(\prod_{k=1}^{i} \exp\left(-b_k^\dagger \zeta_k - (-\zeta_k^* b_k)\right)|\mathbf{0}\rangle\right) \cdot \left(\prod_{k=j+1}^{N} \exp\left(b_k^\dagger \zeta_k - \zeta_k^* b_k\right)|\mathbf{0}\rangle\right). \tag{2.237}$$

By swapping the order of the operators for orbitals $< j$ the negative sign is introduced and such an orbital is then defined,

$$\begin{aligned} |\zeta_i\rangle &= D(-\zeta_i)|\mathbf{0}\rangle = \exp\left(-b_i^\dagger \zeta_i - (-\zeta_i^* b_i)\right)|\mathbf{0}\rangle \\ &= sin(-\theta_i/2)e^{-i\varphi_i}|1_i\rangle + cos(-\theta_i/2)|0_i\rangle \\ &= -sin(\theta_i/2)e^{-i\varphi_i}|1_i\rangle + cos(\theta_i/2)|0_i\rangle \end{aligned} \tag{2.238}$$

using the fact sine and cosine are odd and even functions respectively. Thus, Zombie states constructed using the general coherent state definition are by construction normalised and antisymmetrised with creation and annihilation operations that produce the expected behaviour.

2.4.7 Comparison to Standard Fermionic Coherent State Constructions

It is now possible to compare both standard fermionic coherent state constructions and Zombie states. To do this a system with dimension $N = 2$ will be used which corresponds to two spin orbitals. If both orbitals are occupied then a Slater determinant can be constructed as

$$\Psi_{12}(\mathbf{x}_1, \mathbf{x}_2) = 2^{-1/2}(\chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2) - \chi_i(\mathbf{x}_2)\chi_j(\mathbf{x}_1)) \tag{2.239}$$

which can be written succinctly as $|1, 1\rangle$. Coherent states can now be defined to describe this state. Using the Lie group $SO(2 \cdot 2)$ and the general coherent state

definition will give an element of the coset space as,

$$\Omega = \exp\left(\eta_{12} a_1^\dagger a_2^\dagger - \eta_{12}^* a_2 a_1 + \eta_{21} a_2^\dagger a_1^\dagger - \eta_{21}^* a_1 a_2\right). \tag{2.240}$$

Which gives a coherent state by operating Ω on the reference state $|ref\rangle = |0\rangle$,or $|1\rangle$,

$$|z_{SO(2\cdot2)}\rangle = \Omega|ref\rangle = \exp\left(\eta_{12} a_1^\dagger a_2^\dagger - \eta_{12}^* a_2 a_1\right) \exp\left(\eta_{21} a_2^\dagger a_1^\dagger - \eta_{21}^* a_1 a_2\right)|ref\rangle \tag{2.241}$$

note the phase factor created by the stability group has been omitted. A coherent state can be defined by a set of four Grassmann generators $\boldsymbol{\xi} = (\xi_1, \xi_2, \xi_1^*, \xi_2^*)$. Then using Eq. (2.180) the fermionic coherent states can be defined,

$$\begin{aligned} |z_{\mathfrak{G}_2}\rangle = D(\boldsymbol{\xi})|0\rangle &= \exp\left(a_1^\dagger \xi_1 - \frac{1}{2}\xi_1^* \xi_1 + a_2^\dagger \xi_2 - \frac{1}{2}\xi_2^* \xi_2\right)|0\rangle \\ &= \exp\left(a_1^\dagger \xi_1 - \frac{1}{2}\xi_1^* \xi_1\right) \exp\left(a_2^\dagger \xi_2 - \frac{1}{2}\xi_2^* \xi_2\right)|0\rangle \end{aligned} \tag{2.242}$$

Finally, a Zombie state can be defined by $\boldsymbol{\zeta} = (\zeta_1, \zeta_2)$,

$$\begin{aligned} |\boldsymbol{\zeta}\rangle = D(\boldsymbol{\zeta})|\mathbf{0}\rangle &= \exp\left(b_1^\dagger \zeta_1 - \zeta_1^* b_1 + a_2^\dagger \zeta_2 - \zeta_2^* b_2\right)|\mathbf{0}\rangle \\ &= \exp\left(b_1^\dagger \zeta_1 - \zeta_1^* b_1\right)|0_1\rangle \exp\left(b_2^\dagger \zeta_2 - \zeta_2^* b_2\right)|0_2\rangle. \end{aligned} \tag{2.243}$$

When written out explicitly the difference between the two standard fermionic coherent state constructions and Zombie states is clear. $|z_{SO(2\cdot2)}\rangle$ and $|z_{\mathfrak{G}_2}\rangle$ are constructed from the different spin states of each orbital. A system state can then be described in terms of the fermionic coherent state. For two spin orbitals there are three possible states of occupancy, $|1,0\rangle, |0,1\rangle, |1,1\rangle$ and the vacuum state. To describe the doubly occupied Ψ_{12} using $|z_{SO(2\cdot2)}\rangle$ the reference state would be taken to be $|0\rangle$ but the single occupancy states would require a different reference state to account for the unpaired spin. Whereas, the Zombie states are a superposition of all possible configurations. The standard fermionic coherent states are a method for describing a specific configuration in terms of its spin states requiring the state to remain an eigenstate of the number operator. Zombie states can describe a system as a superposition of all configurations. Thus, Zombie states offer a way of describing the linear combination of configurations used in a CI wave function.

2.5 Concluding Remarks

This section uses a detailed description of the Hartree Fock method to motivate understanding of the key concepts in electronic structure theory. It is shown how the physical reality of electrons in a molecule can be represented as an antisymmetrised Slater determinant; then the associated algebra necessary to construct a Hamiltonian matrix. The concept of second quantisation is introduced which moves the antisymmetric property to the action of the creation and annihilation operators. This is followed by discussion of the self-consistent field method. These elementary building blocks serve as a basis for understanding the Zombie states method. A Slater determinant exactly represents a single configuration of electrons in the available spin orbitals. The full-CI wave function consists of all possible electron configurations described using Slater determinants. Using this linear combination of configurations exactly describes the system because it does not continually exist in a single configuration. Generally, coherent states give a means to describe a system as a superposition of another appropriate set of sets such as an electromagnetic field using states of the harmonic-oscillator [11, 12, 99]. Fermionic coherent states provide a method for describing a configuration as superposition of its spin states whereas Zombie states can describe a system as a superposition of all configurations.

The Perelomov and Gilmore general definition constructs coherent states from action of the displacement operator on a reference state [21, 106]. The displacement operator is found by using the appropriate dynamical group and defining its coset or quotient space by use of a stability group. Using this method the coherent states of the Harmonic oscillator, used in the Coupled Coherent States method, and the $SU(2)$ coherent states, additionally used in the Multiconfigurational Ehrenfest method, are derived. The general method results in coherent states of exactly the same form used in CCS and MCE literature [13, 16, 109]. However, significantly the general definition naturally produces $SU(2)$ coherent states defined in the same way as states initiated using Quantum Superposition sampling [33]. By couching the CCS family of methods in terms of general coherent states establishes a simple framework for constructing coherent states and also swamping the type of coherent states used in a simulation.

Both the Grassmann algebra and $SO(2n)$ Lie group coherent state constructions ensure the correct antisymmetric properties are maintained. These standard fermionic coherent states provide a method for describing a specific fermionic configuration in terms of its spin states. In both constructions the fermionic coherent states are eigenstates of the number operator which is strictly conserved; states defined using the $SO(2n)$ Lie group require different reference states depending on whether the number of fermions is odd or even. For the construction, that uses Grassmann algebra, to remain physical it is not permitted to have transitions from

an even to odd number of fermions or between different odd numbers of fermions [18]. Further, Grassmann algebra does not lend itself to numerical computation – there is little literature on the topic.

On the other hand, Zombie states are not eigenstates of the number operator. Each Zombie orbital is a superposition of "dead" and "alive" coefficients the $SU(2)$ coherent state representing the occupied and unoccupied state. Therefore, the Zombie state contains all configurations and so treats odd and even numbers of fermions in the same way. If only binary coefficients are permitted when defining a Zombie state then the state becomes equivalent to a Slater determinant with the same set of occupied spin orbitals. However, when allowing fractional Zombie coefficients, the Zombie state is a superposition of all configurations. So, a sufficiently sized Zombie wave function with appropriately constructed Zombie states should be able to exactly describe a system with the same energies as the full-CI wave function. It has previously been shown that Zombie states can be constructed from a vacuum state, and this maintains the sign change rule caused by creation and annihilation operations [9, 23]. Here the general coherent state definition is applied to the Zombie states for the first time which shows they can be constructed in a manner analogous to the multi-dimensional harmonic oscillator coherent states used in CCS. Using the action of the displacement operator on the vacuum state to construct a Zombie state ensures the normalisation condition for the "dead" and "alive" coefficients is automatically satisfied. Further, this method of construction naturally gives rise to the sign rule following creation and annihilation operations – changing the sign of "alive" coefficients but not "dead" ones.

Zombie states are placed on a similar footing to Slater determinants in the second-quantisation representation. This allows much of the well established and understood algebraic formalism from standard Hartree Fock theory to be translated to the Zombie states method. The hierarchy of electronic basis sets gives the Hartree Fock method systematic quantitative accuracy. Conveniently, this property translates to the Zombie states method and means the set of one- and two-electron integrals, generated by any electronic structure package for a SCF calculation, can be used to construct the Zombie state Hamiltonian matrix. Thus, chemical systems can be set up easily using existing and optimised electronic structure packages while allowing direct comparison between Hartree Fock and ZS results. The work detailed in this thesis focuses on finding the ground state energy of chemical systems with the aim of reaching energies equivalent to the full-CI energy. Both FCIQMC and MCCI aim to find full-CI energies without using the entire set of possible configurations hence, comparison to the ZS method is made where appropriate.

Chapter 3
Finding the Ground State Energy

3.1 Introduction

Quantum chemistry is built around finding solutions to the Schrödinger equation which is a form of the eigenvalue problem. For the TISE the Hamiltonian matrix can be diagonalised to form a new matrix $\boldsymbol{D}$ by $\boldsymbol{P}$ which is a non-singular square matrix which in general terms is

$$\boldsymbol{P}^{-1}\boldsymbol{H}\boldsymbol{P} = \boldsymbol{D} \Longrightarrow \boldsymbol{H}\boldsymbol{P} = \boldsymbol{P}\boldsymbol{D} \tag{3.1}$$

where each column of $\boldsymbol{P}$ is an eigenvector of $\boldsymbol{H}$ and the diagonal elements of $\boldsymbol{D}$ are the eigenvalues – the lowest being the ground state energy. However, diagonalising large matrices is computationally expensive meaning alternative methods are considered. As previously seen for configuration interaction, variational minimisation of the ground state energy is used to find the optimal set of linear coefficients to give the ground state energy. In this chapter two methods for finding ground state energies are presented. Firstly, the method used in the original presentation of Zombie states, a long-time propagation followed by a Fourier transform and the second is imaginary time propagation [9]. Previous work has established that a small basis of randomly selected, trajectory-guided Canonical Coherent States of the harmonic oscillator are an effective way to describe the quantum dynamics of a system with multiple particles and many degrees of freedom. The Coupled Coherent States (CCS) method has been used to simulate the dynamics of nuclei in chemical reaction dynamics [116, 131]. CCS has also been used to study the dynamics of electrons in a strong laser field and quantum decoherence [16, 132]. Therefore, equations, analogous to the CCS method, to propagate Zombie states were developed. These allowed a system of Zombie states to be propagated and the energy of states to be recovered.

However, the long-time propagation was not a particularly efficient way to retrieve energies. Computational efficiency is a key part of Pople's framework for method development and so an alternative approach was needed. The method,

which has now been adopted as the standard procedure, employs an imaginary time propagation to recover ground state values. This is a key component of the well established FCIQMC method validating its choice. Imaginary time propagation for Zombie states is verified by comparison to exact energies obtained by diagonalisation of a Hamiltonian constructed using a full-CI basis set.

3.2 Long Time Propagation and Fourier Transformation

Throughout this work the TISE is used but when first presented a time dependent basis was used to propagate the Zombie states. A 1D Canonical Coherent State is a Gaussian wave packet

$$\langle x|z\rangle = \langle x|q,p\rangle = \sqrt{\frac{\gamma}{2}} exp(-\frac{\gamma}{2}(x-q)^2 = ip(x-q) + i\frac{qp}{2}) \tag{3.2}$$

with $z = \dfrac{\gamma^{1/2}q + p\left(\gamma^{1/2}\hbar\right)^{-1}}{2^{1/2}}$ and $\gamma^{1/2}$ being the width of the wave packet in the vicinity of the phase space point q,p. Ideas developed for the Coupled Coherent Sates method can be applied to fermionic Zombie states. The Zombie wave function is made time dependent by making both the basis functions, $|\boldsymbol{\zeta}^{(a)}(t)\rangle$, and their coefficients, $d_a(t)$, time dependent. The time evolution is a system of linear equations

$$\begin{aligned} \sum_a^N \langle \boldsymbol{\zeta}^{(b)}|\boldsymbol{\zeta}^{(a)}\rangle \frac{dd_a}{dt} &= -\sum_a^N \langle \boldsymbol{\zeta}^{(b)}|\frac{\boldsymbol{\zeta}^{(a)}}{dt}\rangle d_a \\ &= -i\sum_a^N \langle \boldsymbol{\zeta}^{(b)}|\hat{H}|\boldsymbol{\zeta}^{(a)}\rangle d_a \end{aligned} \tag{3.3}$$

As in CCS the analogue of the classical Hamilton's equation is

$$\frac{d\boldsymbol{\zeta}^{(a)}}{dt} = -i\frac{\partial\langle \boldsymbol{\zeta}^{(b)}|\hat{H}|\boldsymbol{\zeta}^{(a)}\rangle}{\partial \boldsymbol{\zeta}^{*(a)}} \tag{3.4}$$

which is used to find the trajectory of $|\boldsymbol{\zeta}^{(k)}(t)\rangle$ [109, 131]. This produces a set of trajectories based on a multiconfigurational wave function which can give an exact value in that it can converge to the exact result [9]. This is significantly different to single configuration methods like time-dependent Hartree-Fock Bogoliubov which only uses a single Slater Determinant.

The autocorrelation function was defined as

$$\langle \Psi(0)|\Psi(t)\rangle = \sum_a^N d_a(t)\langle \Psi(0)|\boldsymbol{\zeta}^{(a)}(t)\rangle. \tag{3.5}$$

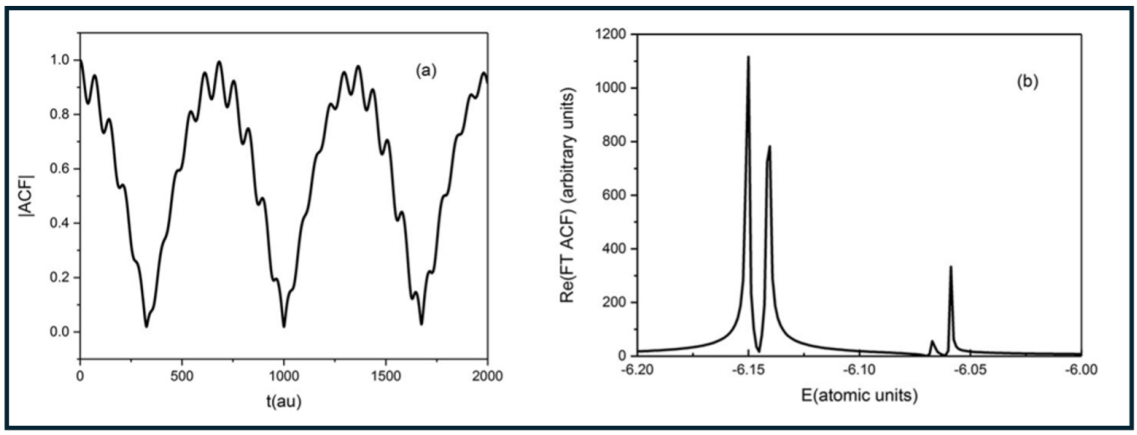


Figure 3.1: Results for LiH in the 6-31G** basis with four electrons in six spin orbitals. All one- and two- electron integrals were calculated using MOLPRO and comparisons made to the full-CI values the program produced. Frame (a) shows the rapid oscillations of the autocorrelation function, $\langle\Psi(0)|\Psi(t)\rangle$. This was calculated by propagating a single Hartree Fock configuration on the Zombie state basis. Frame (b) is the real part of the autocorrelation Fourier transform. The peaks exactly match the eigenvalues of the electronic Hamiltonian. From Ref. [9].

LiH and Li_2 were used to show that Zombie states could produce identical autocorrelation functions to exact benchmark values obtained using the electronic propagator in the full Fock space. Fourier transforms of the function then also gave identical peak positions for states of LiH which had the greatest configuration contributions in the full-CI space this result is illustrated in Fig. 3.1 [9].

3.3 Imaginary Time Propagation

3.3.1 Theory

First a wave function, $|\Psi\rangle = \sum_a d_a|\psi^{(a)}\rangle$, is constructed from an orthonormal set of states, $\{|\psi^{(a)}\rangle\}, a = 1, \ldots, N_{bf}$. This wave function can then be substituted into the time-dependent Schrödinger equation which gives

$$i\hbar\frac{d}{dt}|\Psi\rangle = i\hbar\sum_a \dot{d}_a|\psi^{(a)}\rangle = \sum_a d^{(a)}\hat{H}|\psi^{(a)}\rangle = \hat{H}|\Psi\rangle. \tag{3.6}$$

Assuming the states have no intrinsic time dependency an equation for $\dot{d}_b$ can be found

$$\dot{d}_b = \frac{dd_b}{dt} = -\frac{i}{\hbar}\sum_a \langle\psi^{(b)}|\hat{H}|\psi^{(a)}\rangle d_a \tag{3.7}$$

which can be used to time evolve the states. In the standard way time evolution of an initial state is found by $|\Psi(t)\rangle = e^{-i\hat{H}t/\hbar}|\Psi\rangle$ which can be expanded in terms of

its eigenstates $|\phi_n\rangle$ with energies E_n so

$$|\Psi(t)\rangle = e^{-i\hat{H}t/\hbar}|\Psi\rangle = \sum_n |\phi_n\rangle e^{-iE_n t/\hbar}\langle\phi_n|\Psi\rangle. \tag{3.8}$$

Comparison of the Boltzmann operator $e^{-\beta\hat{H}}$ and the time evolution operator $e^{-i\hat{H}t/\hbar}$ shows that $\beta = it/\hbar$ which can be substituted into this into Eq. (3.6) which yields,

$$-\frac{d}{d\beta}|\Psi\rangle = \hat{H}|\Psi\rangle \tag{3.9}$$

In a similar way to the TDSE the imaginary time evolution of $|\Psi\rangle = \sum_a d_a|\psi^{(a)}\rangle$ can be found by,

$$\frac{dd_b}{d\beta} = -\sum_a \langle\psi^{(b)}|\hat{H}|\psi^{(a)}\rangle d_a. \tag{3.10}$$

The wave function can be expanded again in terms of its eigenfunctions

$$e^{-\beta\hat{H}}|\Psi\rangle = \sum_n e^{-\beta E_n}|\phi_n\rangle\langle\phi_n|\Psi\rangle. \tag{3.11}$$

Assuming that the ground state, $|\phi_0\rangle$, and its eigenvalue, E_0, are non-degenerate, then $E_n > E_0\ \forall n > 0$. Therefore,

$$\lim_{\beta\to\infty} e^{\beta E_0}e^{-\beta\hat{H}}|\Psi\rangle = \lim_{\beta\to\infty}\sum_n e^{-\beta(E_n-E_0)}|\phi_n\rangle\langle\phi_n|\Psi\rangle \tag{3.12}$$

$$= |\phi_0\rangle\langle\phi_0|\Psi\rangle = p|\phi_0\rangle. \tag{3.13}$$

$\langle\phi_0|\Psi\rangle = p$ is a numerical constant and provided $p \neq 0$ then continued evolution in imaginary time can be used to find the ground state. The Zombie basis is nonorthogonal therefore to apply imaginary time propagation, using the same reasoning as in Ref. [133], the identity matrix is needed

$$\boldsymbol{I} = \sum_{a,b} |\boldsymbol{\zeta}^{(a)}\rangle \Omega^{-1}_{ab} \langle\boldsymbol{\zeta}^{(b)}|. \tag{3.14}$$

A wave function $|\Psi\rangle$ that can be expanded in the basis of N_{bf} Zombie states $|\boldsymbol{\zeta}^{(a)}\rangle$, where $N_{bf} \leq 2^{N_{el}}$, which are normalised but not necessarily orthogonal,

$$|\Psi\rangle = \sum_a^{N_{bf}} d_a|\boldsymbol{\zeta}^{(a)}\rangle. \tag{3.15}$$

Using the reasoning in Ref. [109] that $|\Psi\rangle = \boldsymbol{I}|\Psi\rangle$ so Eq. (3.10) can be transformed

$$-\frac{d\langle\boldsymbol{\zeta}^{(c)}|\Psi\rangle}{d\beta} = -\sum_{a,b}\langle\boldsymbol{\zeta}^{(c)}|\hat{H}|\boldsymbol{\zeta}^{(a)}\rangle\Omega^{-1}_{ab}\langle\boldsymbol{\zeta}^{(a)}|\Psi\rangle \tag{3.16}$$

$$\frac{dd_c}{d\beta} = -\sum_{a,b}^{N_{bf}} d_a \langle \boldsymbol{\zeta}^{(c)} | \hat{H} | \boldsymbol{\zeta}^{(b)} \rangle \Omega_{ab}^{-1} \tag{3.17}$$

which can be put into matrix notation as

$$\dot{\mathbf{d}} = -\boldsymbol{\Omega}^{-1}\mathbf{H}\mathbf{d}. \tag{3.18}$$

A single imaginary time step can then be defined as

$$d_a(\beta + \Delta\beta) = d_a(\beta) + \dot{d}_a(\beta)\Delta\beta. \tag{3.19}$$

3.3.2 Application of Imaginary Time Propagation to Li_2

As in the first presentation of the Zombie states method Li_2 will be used as an example system. It will be shown that imaginary time propagation is effective and efficient when finding ground state energies. PyScf is used to calculate the one- and two-electron integrals in the 6-31G** basis and the system is constructed with five spacial (meaning ten spin) molecular orbitals [134]. The number of spin orbitals has been truncated to ten to remove additional computational cost not required to validate the use of imaginary time propagation. The energies obtained are compared to a full-CI basis consisting of 1024 Slater determinants. This number of Slater determinants is chosen as it contains all possible configurations for all possible numbers of electrons, $2^{10} = 1024$. All calculations are carried out in atomic units, so energies are in Hartrees. Trivially a complete basis set of 1024 Slater determinant,$|\varphi_{m_e}^{(j)}\rangle$ Zombie states is considered. Using the six-electron restricted Hartree Fock determinant as a starting point it can be seen in Fig. 3.2 that imaginary time propagation yields the neutral ground state energy of Li_2. Next a basis of $2^{10} = 1024$ randomly generated basis functions $|\boldsymbol{\zeta}^{(k)}\rangle$ was generated. The dead and alive coefficients for each Zombie state were calculated $a_{1j} = \cos(\theta_j)$ and $a_{0j} = \sin(\theta_j)$ respectively using a random number $0 \leq \theta_j < 2\pi$. Rather than the binary Slater determinant basis each random Zombie state consists of superposition of "dead" and "alive" electrons. The initial vector of Zombie coefficients is a superposition of random Zombie states chosen to be equal to the RHF determinant (although this is not a requirement). Imaginary time propagation for the random Zombie basis, the solid line in Fig. 3.2, matches the result obtained by using the Slater Determinant basis. The final energy of each state obtained through imaginary time propagation is verified by comparison to the eigenvalues found by diagonalizing the complete Slater determinant Hamiltonian.

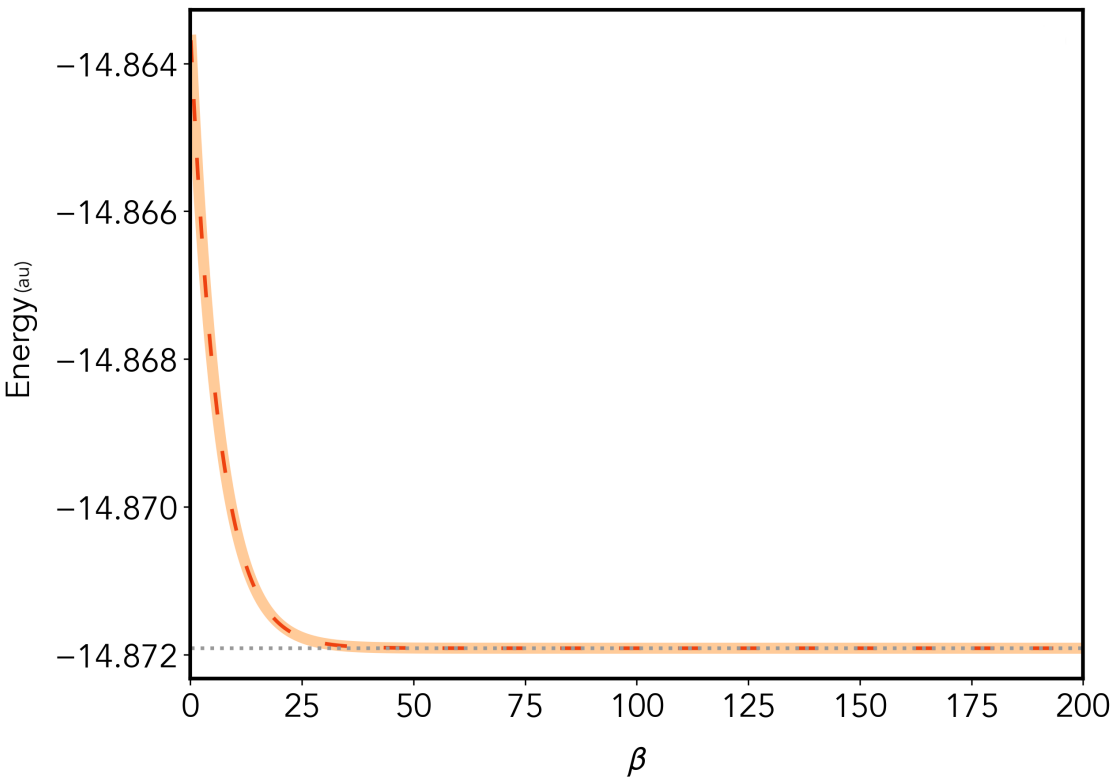


Figure 3.2: Imaginary time propagation starting at the 6-electron restricted Hartree Fock Slater determinant energy for a complete Slater determinant basis (dashed, red line) and a complete random basis (solid, orange line) of Zombie States. Both results tend to the neutral ground sate energy of Li_2 found by diagonalizing the Slater determinant Hamiltonian, shown as a grey dotted line.

3.4 Conclusions

The implementation of imaginary time propagation to the Zombie state method is an important step towards a practical electronic structure method. The lowest-lying state of a system can be found using imaginary time propagation. For a complete basis of randomly generated Zombie states there is exact agreement with the Slater determinant basis for Li_2. Although the system and ground state energy are trivial it is significant that the need for long-time evolution and Fourier transforms is removed. This makes the method more practical to use due to lower computational expense and time which has enabled further development. It is now possible to investigate larger more complicated systems which is integral when verifying the applications of the method. Further, computationally expensive additions to the method, like the gradient descent method described in the next chapter, are now possible since significant time is not required to find ground state energies. The adoption of imaginary time propagation has also led to using Zombie states to find energies of excited states which is discussed in Chapter 5.

Chapter 4
Reducing the Basis Set Size

4.1 Introduction

The introduction of imaginary time propagation to find ground state energies advances the Zombie state method to be computationally inexpensive which is an essential part of Pople's framework for method development. However, thus far all numerical results found using ZSs can be easily recovered when using wave functions of Slater Determinants. For the method to be more than an academic endeavour and eventually offer something beyond current electronic structure techniques it is necessary to use less than complete sized basis sets and still recover exact or near to exact energies. Using the same 6-31G** truncated basis of ten spin orbitals for Li_2 a random set of 200 Zombie states is constructed. In Fig. 4.1, the basis set is no longer capable of reproducing the ground state energy. The energy obtained via imaginary time propagation, in this smaller basis, no longer matches the value found by diagonalising the complete Slater determinant basis.

For the Lithium dimer system used here a basis set of 200 functions, containing basis functions with the incorrect number of electrons, is approximately 20% the size of the complete basis. Moreover, when considering a basis of Slater determinants containing only configurations with the correct number of electrons ($10!/6!(10-6)! = 210$) then 200 basis functions is 95% of set size. Clearly using an incomplete set number of randomly generated Zombie states is not a practical approach; new methods to generate generate Zombie states are required. In this section, I will first outline the initial work that indicated a smaller set of Zombie states could be used while still obtaining reasonable ground state energies. This took the form of two methods named biasing and cleaning. Biasing being used to generate better amplitudes in the Zombie states and cleaning being used to improve the ground state energy achieved post imaginary time propagation.

However, neither method was appropriate for the long-term development of the ZS method. Hence, Gradient Descent (GD) is introduced to the Zombie state method. This forms a significant portion of the results of this thesis. It will be shown

how GD can systematically optimise the set of Zombie state coefficients without any significant *a priori* knowledge of the system. The implementation of Gradient Descent to the Zombie states method is validated by a selection of systems considerably more challenging than the truncated Li_2.

4.2 Cleaning

The imaginary time propagation outlined in Chapter 3.3 relaxes the wave function to its lowest energy state so long as the wave function contains contributions from wave functions of the same symmetry and number of electrons as the lowest-energy state. As Zombie state "dead" and "alive" coefficients give fractional spin-orbital occupations ZSs are not restricted to a particular number of electrons. However, they can be projected onto a Fock Space with a given number of electrons. This can be used to improve the ground state energy obtained and also give insight into why a wave function is not capable of recovering the exact ground state energy.

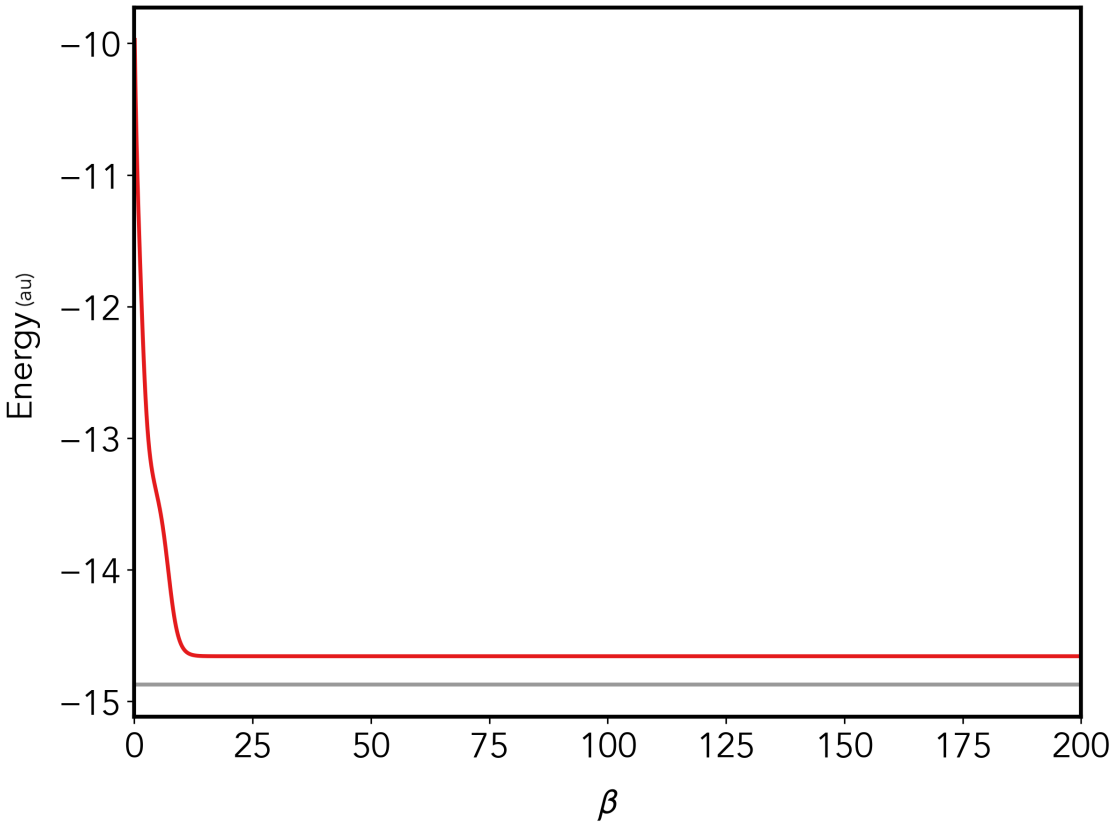


Figure 4.1: Imaginary time propagation starting at the 6 electron restricted Hartree Fock Slater determinant energy and using a random basis of 200 Zombie sates. The neutral ground sate energy of Li_2 found by diagonalising the complete Slater determinant Hamiltonian is also shown in grey. The one- and two- electron integrals were generated using the PyScf program [134].

The identity operator, $\boldsymbol{I}$, can be written as a sum of m_e electron identities $\boldsymbol{I}_{m_e}$:

$$\boldsymbol{I} = \sum_{m_e=0,N_{fci}} \hat{I}_{m_e}, \tag{4.1}$$

meaning the identity covering the Fock Space with m_e electrons is

$$\boldsymbol{I}_{m_e} = \sum_a |\varphi_{m_e}^{(a)}\rangle\langle\varphi_{m_e}^{(a)}| \tag{4.2}$$

where the sum is over all Fock configurations with the correct number of electrons i.e. $\langle\varphi_{m_e}^{(a)}|\hat{N}|\varphi_{m_e}^{(a)}\rangle = m_e$. These Zombie states have "binary" amplitudes 1 and 0 and m_e unit amplitudes of alive electrons on m_e occupied spin-orbitals and so are equivalent to one member configuration in a full-CI. The superposition of Zombie states in Eq. (2.231) or Eq. (3.15) can then be projected onto a Fock Space of m_e electrons as

$$|\Psi_{m_e}\rangle = \boldsymbol{I}_{m_e}|\Psi\rangle = \sum_a c_a|\varphi_{m_e}^{(a)}\rangle \tag{4.3}$$

where

$$c_a = \sum_{b=1,N_{zs}} d_b\langle\varphi^{(a)}|\boldsymbol{\zeta}^{(b)}\rangle. \tag{4.4}$$

The portion of the total energy from the m_e configurations and its norm can then be calculated,

$$E_{m_e} = \sum_{ba} c_a^* c_b \langle\varphi_{m_e}^{(a)}|\hat{H}|\varphi_{m_e}^{(b)}\rangle, \tag{4.5}$$

$$N_{m_e} = \sum_b c_b^* c_b, \tag{4.6}$$

respectively. The energy contribution for each m_e together will sum to the total energy of the whole Zombie wave function Eq. (2.231) and all m_e norms add up to 1:

$$\langle\Psi|\hat{H}|\Psi\rangle = \sum_{m_e} E_{m_e} \tag{4.7}$$

$$\sum_{m_e} N_{m_e} = 1. \tag{4.8}$$

For a completely converged Zombie wave function the norm for the correct number of electrons m_e will be one (the other configuration norms being zero) and so the total energy will only be made up by contributions from configurations with the correct number of electrons. But if convergence is incomplete the energy of the

converged m_e wave function $|\Psi_c\rangle$ can be estimated as

$$\langle\Psi_c|\hat{H}|\Psi_c\rangle \approx E_{m_e}/N_{m_e}. \tag{4.9}$$

by division of the m_e energy by the m_e norm.

Using the same 200 Zombie state basis set used in Fig. 4.1 for the lithium dimer system the clean procedure can be run for each number of electrons the results of which are shown in Fig. 4.2. The largest contribution to the energy (and so also norm) is from the seven electron configurations followed by those with eight electrons not from configurations containing six electrons that describe the ground state. The total energy being dominated by configurations that give higher energy states explains why the wave function is not able to converge to the ground state energy as the complete basis could. Moreover, since the six electron norm is small $E(6e)/N(6e)$ does not yield a more accurate value for the ground state energy than the system total. Therefore, it is clearly necessary to construct basis sets that contain a much greater number of superpositions of configurations with the correct number of electrons i.e. for Li_2 minimise the norm for all numbers of electrons except six.

4.3 Biasing

Pople's framework calls for methods to be applied generally so they can act as a black box without the need for prior knowledge and input. However, this does not mean that reasonable scientific intuition cannot be built into the method. As previously established the restricted Hartree Fock determinant is a very reasonable approximation for the true ground state energy [135]. However, a far larger basis consisting of all possible configurations is necessary to guarantee the exact ground state energy. This was also achieved by a basis of randomly selected 2^M ZSs $|\boldsymbol{\zeta}^{(k)}\rangle$ which is also complete and is capable of yielding correct quantum energies and wave functions. The random ZS basis functions are superpositions of all possible configurations of electrons which together cancel out the configurations with the incorrect number of electrons to give the exact energy. So if a smaller basis is used the ZSs must be constructed to ensure proper cancellation of states with the wrong number of electrons to also find the exact or reasonably close to the ground state energy. This should also show a norm near to one for the correct number of electrons when the cleaning procedure is run on the wave function.

In CASSCF calculations electrons and orbitals are split into three groups: inactive, active and virtual [136]. From this we have designed a biasing method to set up a ZS basis to better ensure proper cancellation of incorrect configurations. Inactive, or core electrons, are low lying and are always (at least in a CASSCF calculation) to be occupied; the active orbitals can either be occupied or unoccupied and virtual

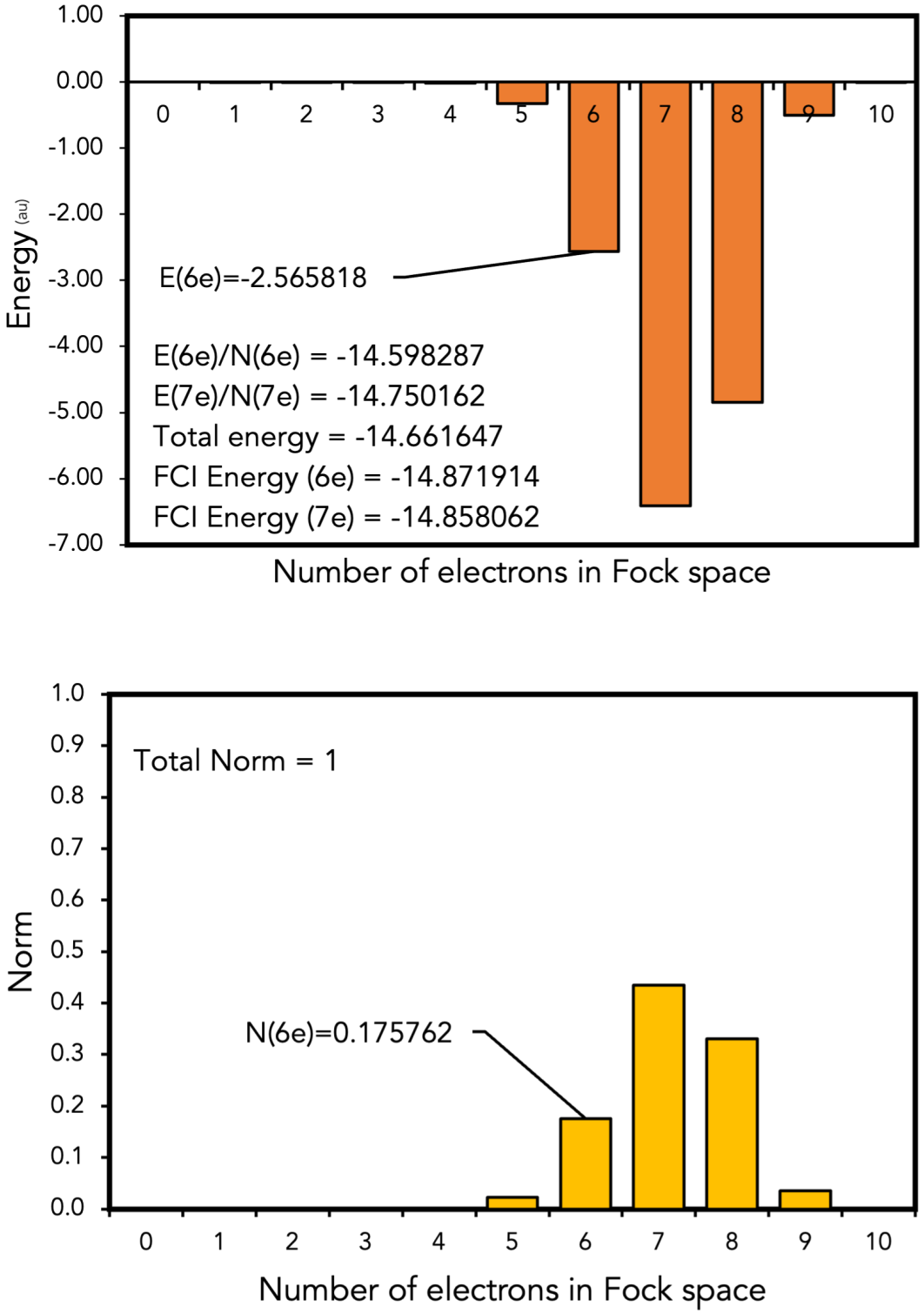


Figure 4.2: Energy (top) and norm (bottom) distribution for each number of electrons for a basis set of 200 randomly generated Zombie states. This is the same basis set used in Fig. 4.1. Notice that less than 20% of the final wave function belongs to the Fock space on $n_e = 6$ electrons.

orbitals are always empty. The RHF determinant fits into the CASSCF system with the lowest set of orbitals being occupied and the virtual orbitals empty. This is a good approximation of the ground state energy because lower energy orbitals are far more likely to be occupied that higher energy ones. Thus, the ZS amplitudes within each Zombie state should reflect this. Rather than assigning each amplitude randomly a biasing regime can be implemented so the amplitudes of each Zombie state are more likely to closely resemble the restricted Hartree Fock determinant. In practise this means, generally, lower energy spin orbitals will have larger "alive" coefficients and high energy orbitals will have larger "dead" amplitudes which is made possible by the fractional occupation of a Zombie state. Like in CASSCF core orbitals are set to be completely "alive" while orbitals in the active space can be biased to be to be more or less "dead" or "alive". In CASSCF virtual orbitals are unoccupied which in the Zombie state representation would mean "dead" coefficients set to 1. In practise there were no completely "dead" orbitals but instead a strong bias towards "dead" coefficients of one for the higher energy orbitals.

"Dead" and "alive" amplitudes for Zombie states are randomly generated from a normal distribution centred around either completely "alive" or completely "dead" values with the width of the Gaussians used decided by the activity level of the orbital. In practise, the low-lying core electrons have "alive" amplitudes set to one, although a very narrow Gaussian could also be employed, which corresponds to a normal distribution width of zero giving a δ function with $a_{1j} = 1$ and $a_{0j} = 0$. Next the active space orbitals have normal distributions that can be centred to favour either completely alive or dead amplitudes. The place where centring changes from "alive" to "dead" being system dependent. Thinner distributions are used for orbitals at either end of the active electron/orbital group and wider distributions for orbitals where the chance, or not, of occupation is relatively equal. Normalisation stipulated in Eq. (2.227) was satisfied by only randomly generating one amplitude and the other being set using Eq. (2.228).

4.3.1 Results

Here the biasing method described above is implemented for the the Li_2 system with ten spin orbitals and six electrons with one- and two-electron integrals calculated using the 6-31G** basis set in the PyScf program [134]. A biasing regime was implemented using sensible chemical intuition of the electronic structure of Li_2. To ensure "dead" and "alive" coefficients were normalised Eq. (2.229) was used with each θ_j generated randomly. Biasing towards "alive" electrons in the RHF determinant $[j = 1 \ldots 6]$ centred the normal distribution at $\frac{1}{2}\pi$; "dead" orbital distributions were centred around 0. The width, σ, of the normal distributions for each orbital are summarised in Table. 4.1.

Table 4.1: Description of the normal distributions, centre μ and width σ, used to generate θ_j for the j-th electron of each Zombie state for Li_2.

j-th Electron	Normal Distributions	
	$\mu/2\pi$	$\sigma/2\pi$
1, 2, 3, 4	0.25	0
5, 6	0.25	0.175
7, 8	0	0.351
9, 10	0	0.120

The first four core orbitals are set to be always "alive" with the remaining orbitals allowed to vary. This biasing regime is shown schematically in Fig. 4.3. The first four spin orbitals always being occupied while the final six having occupations that overlap but on average are biased towards higher orbitals being a smaller fractional occupancy.

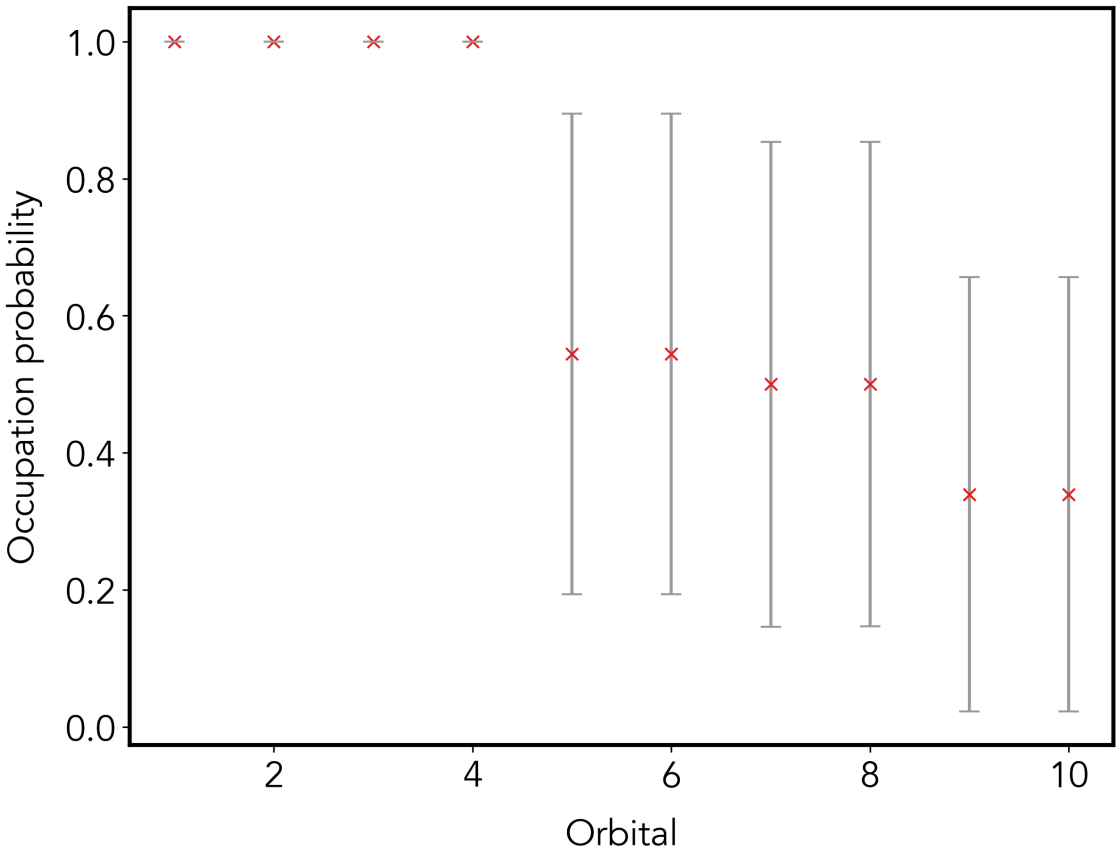


Figure 4.3: Occupational probability plotted against orbital number for ten orbitals. The first four core spin orbitals are to be completely occupied, the remaining orbitals are set using the normal distributions described in Table. 4.1. The average probability is marked by a cross and one standard deviation either side is the bar.

Two wave functions were set up with 63 biased Zombie states and and an additional Zombie state set to either the six electron RHF determinant or the seven electron open-shell restricted Hartree Fock determinant. Depending on the choice of the additional basis function it can be seen in Fig. 4.4 that each basis set is now

recovering the exact full-CI energy for either the ground state or the ground state of the Li_2^- anion. Smaller biased basis sets (of 10, 30 and 50 ZSs) were then tested with the first function in all wave functions set as the six electron RHF determinant. Fig. 4.5 shows the smaller basis sets again not being able to recover the exact ground state which was possible with the 64 ZS wave function nonetheless these results are more accurate than the 200 random ZS wave function. The cleaning procedure does show an improvement for all biased basis set sizes. The complete set of results for the cleaning procedure for the 30 ZS wave function are shown in Fig. 4.6. The biasing method has created a wave function that has a norm almost entirely in the six electron Fock space which improves the energy bringing it closer to the full-CI result.

4.3.2 Conclusion

The biasing method clearly improves upon randomly setting Zombie state amplitudes. However, the scheme sets the lowest four orbitals as occupied leaving two electrons to be distributed across 6 orbitals which gives $2^6 = 64$ Slater determinants in a complete basis and increasing the number of functions creates singularities in the overlap matrix due to linear dependencies. Thus, the basis of 64 functions is again a complete set and reductions in the basis set size move away from the exact ground state energy albeit giving results far closer to the true value than if a random set of ZSs is used. 64 basis functions is less than the 210 possible configurations found by placing six electrons in ten spin orbitals. (210 possible configurations are found using $\frac{N!}{K!(N-K)!}$, $\frac{10!}{6!(10-6)!} = \frac{10!}{6!4!} = 210$.) But it is much larger than the 15 configurations found by placing two electrons in six orbitals caused by fixing the occupancy of the first four spin orbitals. Here the use of ZSs has not given any improvement in the number basis functions needed to recover an exact energy. For the small active space the deviation from states with the correct number of electrons is not advantageous. Furthermore, though cleaning does improve upon the ground state energies, obtained while using the biased basis sets, it still requires the construction of a Hamiltonian representing the entire Fock space. Constructing Hamiltonians for trivially small Fock spaces is computationally inexpensive but for larger spaces it of course becomes problematic without reasonable truncations requiring prior knowledge of the important states. So the application of the cleaning method is possibly limited as either the expense of using a complete set of configurations is immediately required or there is reliance on another method such as FCIQMC to find the most important subset of the Fock space rendering the Zombie states method superfluous. Nonetheless, the biasing method results suggest that finding an optimal set of ZS coefficients could give accurate results when using a small basis set.

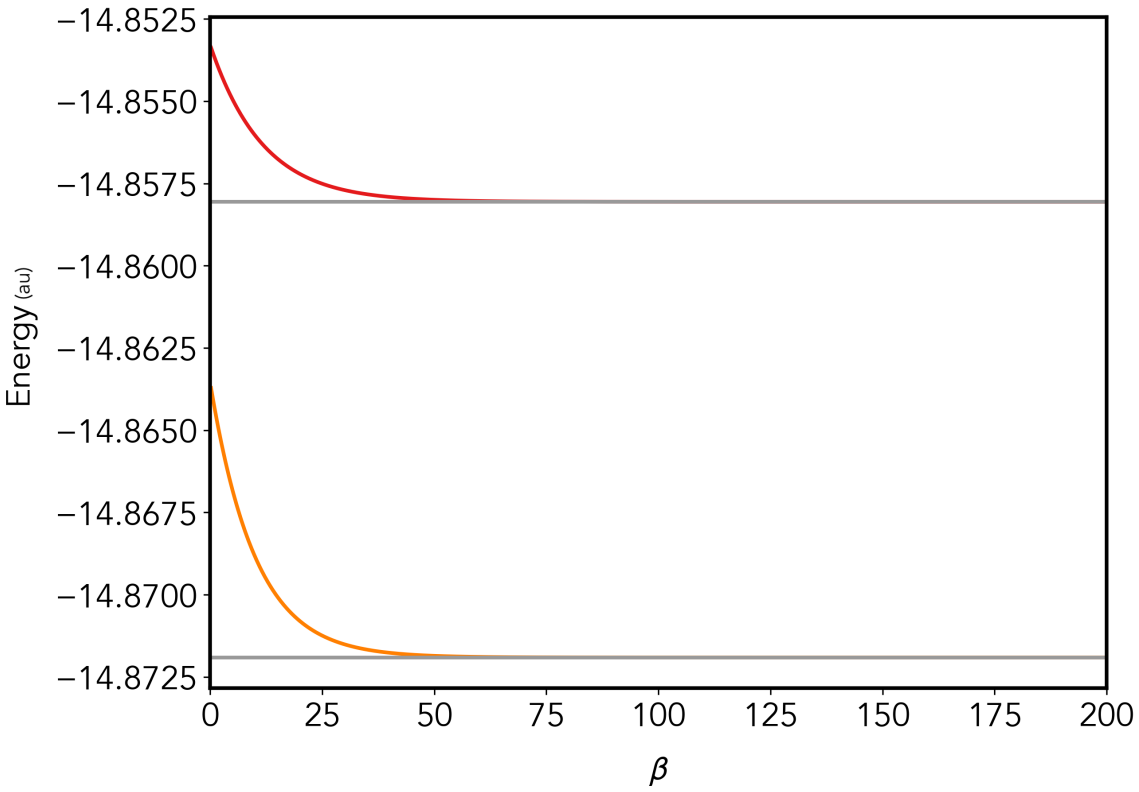


Figure 4.4: Two bases of 63 biased Zombie state functions and an additional state either the 6 electron RHF determinant (orange bottom line) or the 7 electron open-shell RHF determinant (red top line). Imaginary time propagation evolves both cases to exact values (horizontal lines) the six electron basis to the Li_2 neutral ground state; the seven electron basis to the higher Li_2^- anion ground state.

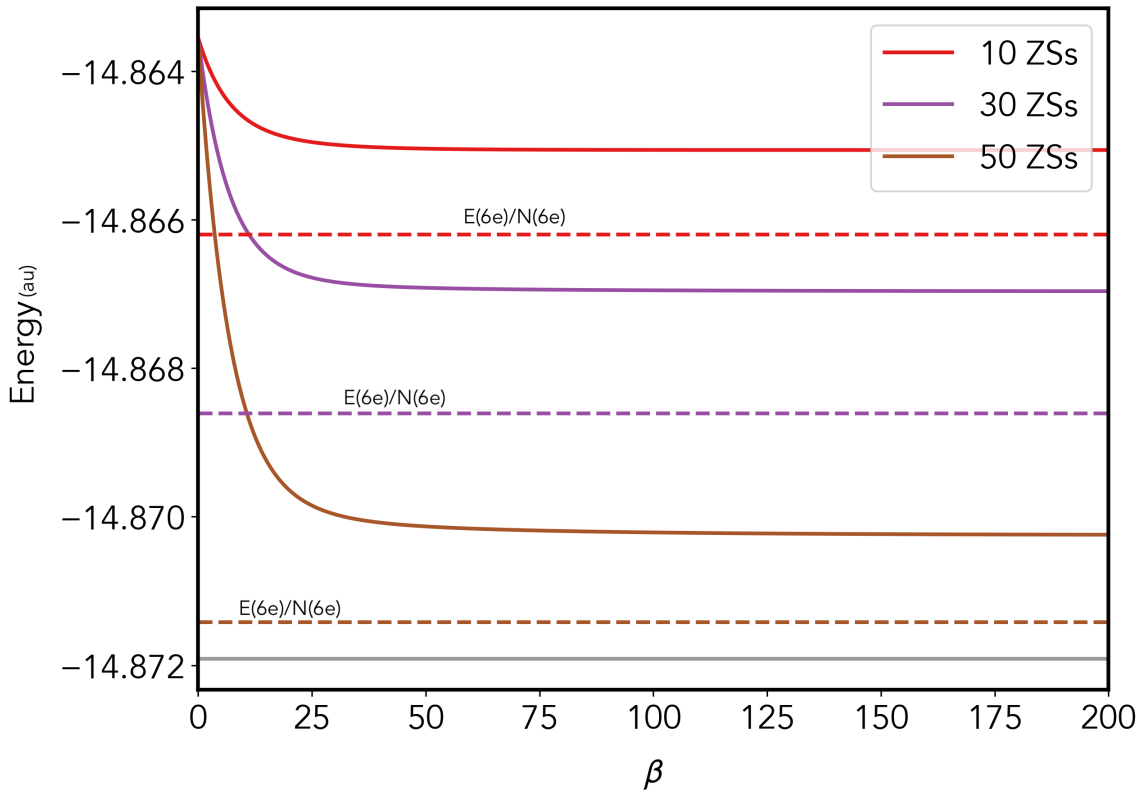


Figure 4.5: The first function in each set is the six electron RHF determinant and the remaining ZSs are biased by the method in Table. 4.1. Imaginary time evolution is shown for basis sets of 50, 30 and 10 functions. The exact value (solid horizontal line) for the Li_2 ground state for comparison. The cleaned energies, $E(6e)/N(6e)$ are dashed lines.

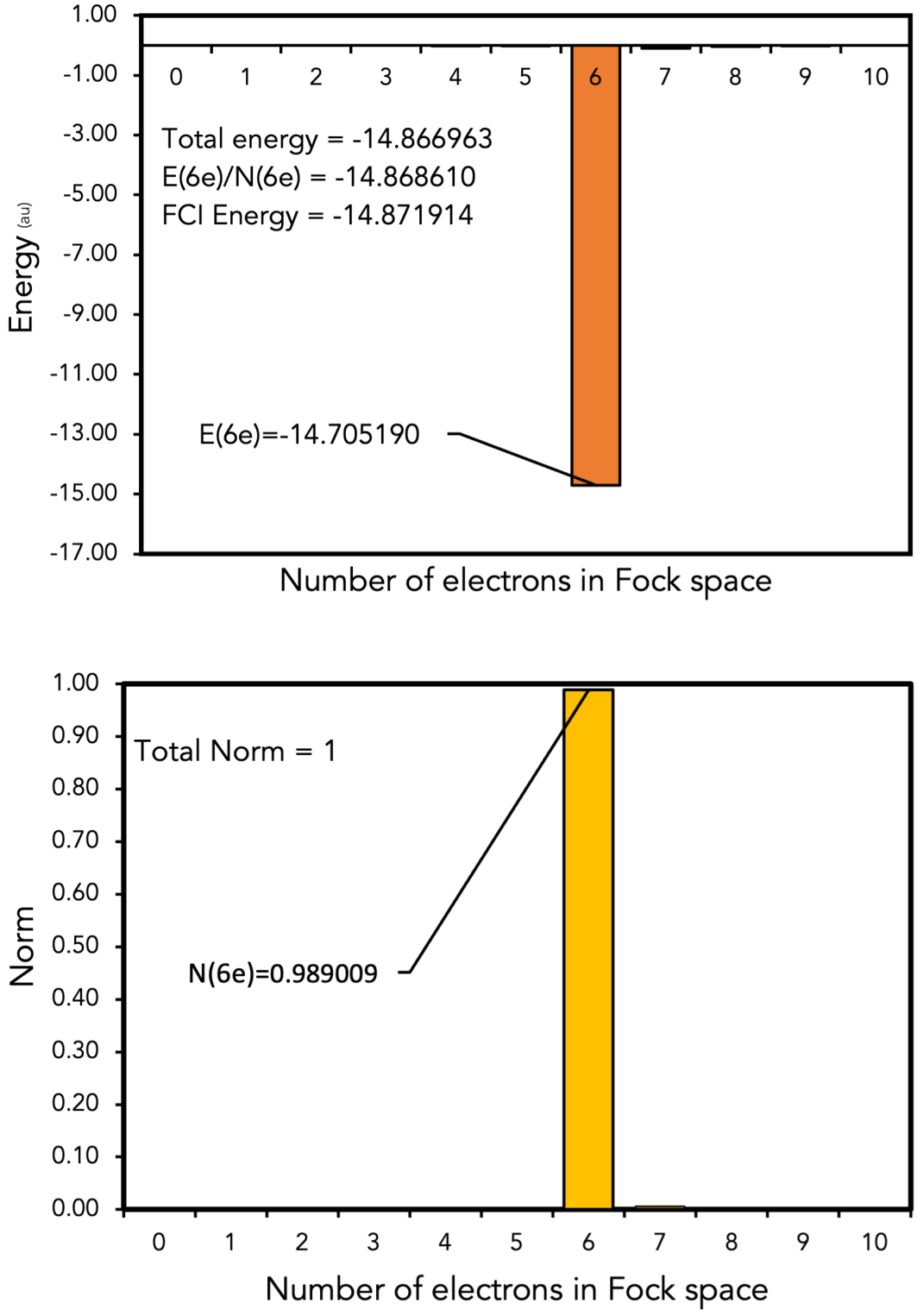


Figure 4.6: Energy (top) and norm (bottom) distribution for each number of electrons for a basis containing 30 biased Zombie states the first ZS is exactly the six electron restricted Hartree Fock determinant. 99% of the final wave function belongs to the Fock space on $n_e = 6$ electrons.

4.4 Gradient Descent

In MCCI and FCIQMC a random walk is used to explore the Fock space to find the set of the most important configurations [79–81, 87, 88, 88, 137]. This means there are two considerations to be made for each configuration in the Fock space: is it included in the wave function and if so what weighting is it given within the set? In FCIQMC the weighting question is handled by the number of walkers and the decision of inclusion is controlled at the pruning step [79]. Whilst this ensures all functions are eigenstates of the number operator a large number of walkers have to be generated whilst also considering a large number of states for inclusion. Zombie states are by construction non-binary meaning individual amplitudes can be altered within a state changing the superposition of states the ZS encompasses rather than removing or adding a specific configuration. With the weighting of individual Zombie states controlled by imaginary time propagation constructing the ZS basis set can be seen as an optimisation problem to find the set of "dead" and "alive" amplitudes that give the minimal ground state energy. This intuitively leads to the Gradient Descent (GD) process which is (in its simplest form) a first-order iterative optimization algorithm used to find the local minimum of a differentiable function. A set of variables X are the input to a differentiable function $f(X)$ for which there exists some set of values that give its minimum which can be found iteratively by,

$$X_{n+1} = X_n - \gamma \nabla f(X_n). \tag{4.10}$$

The learning rate γ is used to set the size of the step, in the negative gradient direction, $-\gamma \nabla f(X_n)$. This alters X_n to produce a new set of parameters X_{n+1}, the process is repeated using the new set of X values. For the Zombie state method the function to be minimized is the energy of the ground state

$$f(X) = E = \sum_{ab}^{N_{bf}} d_a^* d_b \langle \zeta^{(a)} | \hat{H} | \zeta^{(b)} \rangle \tag{4.11}$$

after imaginary time propagation. On first inspection one might expect X to be the set of Zombie states, however, the X is in fact the set of values used to construct each Zombie state. A Zombie state can be constructed using its displacement operator,

$$D(\boldsymbol{\zeta}) |\mathbf{0}\rangle \tag{4.12}$$

where $\boldsymbol{\zeta} = (\zeta_1, \ldots \zeta_{N_{orb}})$ and $\zeta_i = (\theta_i'/2) e^{-i\phi_j}$. Since, only Zombie states with real coefficients will be considered in this work the exponential term can be omitted so

$\zeta_i = \theta_i$ with $\theta_i = 2\theta_i'$. Clearly, each Zombie state is defined by a unique set of values,

$$|\boldsymbol{\zeta}^{(a)}\rangle = D(\boldsymbol{\zeta}^{(a)})|\mathbf{0}\rangle \tag{4.13}$$

with $\boldsymbol{\zeta}^{(a)}$ being the unique set of values used to define $|\boldsymbol{\zeta}^{(a)}\rangle$ this set is given the notation,

$$\boldsymbol{\zeta}^{(a)} = (\zeta_{\theta_1}^{(a)}, \zeta_{\theta_2}^{(a)}, \ldots, \zeta_{\theta_{N_{orb}}}^{(a)}) \tag{4.14}$$

to make clear which Zombie state and orbital each value corresponds to. Thus for a basis set with N_{bf} Zombie states with N_{orb} spin orbitals the set of values to be optimised are,

$$X = \{\zeta_{\theta_1}^{(1)}, \zeta_{\theta_2}^{(1)}, \ldots, \zeta_{\theta_{N_{orb}}}^{(1)}, \ldots, \zeta_{\theta_1}^{(N_{bf})}, \ldots, \zeta_{\theta_{N_{orb}}}^{(N_{bf})}\}. \tag{4.15}$$

The same set is also found by using the normalisation criterion defined in Eq. (2.228). Therefore, the gradient vector is,

$$\nabla f(X) = \nabla f\left(\zeta_{\theta_1}^{(1)}, \ldots, \zeta_{\theta_{N_{orb}}}^{(N_{bf})}\right) = \left(\frac{\partial f}{\partial \zeta_{\theta_1}^{(1)}}, \ldots, \frac{\partial f}{\partial \zeta_{\theta_{N_{orb}}}^{(N_{bf})}}\right) \tag{4.16}$$

where each $\frac{\partial f}{\partial \zeta_{\theta_i}^{(a)}}$ is the partial derivative of the energy function with respect to θ_i used to generate the "dead" and "alive" coefficients in the i-th orbital in Zombie state "a", $\zeta^{(a)}$. Therefore, a single element of the gradient vectors is given using,

$$\begin{aligned}\frac{\partial E}{\partial \zeta_{\theta_i}^{(a)}} = \sum_{ab}^{N_{bf}} \frac{\partial (d_a^*)}{\partial \zeta_{\theta_i}^{(m)}} d_b \langle \zeta^{(a)}|\hat{H}|\zeta^{(b)}\rangle + \sum_{ab}^{N_{bf}} d_a^* \frac{\partial (d_b)}{\partial \zeta_{\theta_i}^{(m)}} \langle \zeta^{(a)}|\hat{H}|\zeta^{(b)}\rangle \\ + \sum_{ab}^{N_{bf}} d_a^* d_b \frac{\partial \langle \zeta^{(a)}|\hat{H}|\zeta^{(b)}\rangle}{\partial \zeta_{\theta_i}^{(a)}}.\end{aligned} \tag{4.17}$$

For Gradient Descent to be applicable Eq. (4.11) must be continually differentiable and a full derivation of this proof can be found in Appendix C. The first two summations account for the Zombie coefficients becoming dependent on X through imaginary time propagation, Eq. (3.19).

4.4.1 Alternative Gradient Calculation

To calculate the gradient using Eq. (4.17) directly is a computationally expensive task that dominates the program execution but a simple algebraic re-framing reduces computational complexity. The energy equation in vector notation, $\boldsymbol{d}^*\boldsymbol{H}\boldsymbol{d}$, is differentiated with respect to the the $\boldsymbol{d}$ vector

$$\frac{\partial}{\partial \boldsymbol{d}}\boldsymbol{d}^*\boldsymbol{H}\boldsymbol{d} = 2\boldsymbol{H}\boldsymbol{d} = \frac{\partial E}{\partial \boldsymbol{d}}. \tag{4.18}$$

After each imaginary time step a new Zombie state coefficient vector is found denoted, $\boldsymbol{c}$, which is then normalised to produce $\boldsymbol{d}$. The normalisation occurs by,

$$\boldsymbol{d} = \frac{\boldsymbol{c}}{\sqrt{|\boldsymbol{c}^*\Omega\boldsymbol{c}|}} \tag{4.19}$$

where $\boldsymbol{c}$ is the unnormalised Zombie coefficient vector and Ω is the overlap matrix. This can then be differentiated with respect to an orbital in a Zombie state $\frac{\partial}{\partial \zeta_{\theta_i}^{(a)}}$

$$\frac{\partial}{\partial \zeta_{\theta_i}^{(a)}}\frac{\boldsymbol{c}}{\sqrt{|\boldsymbol{c}^*\Omega\boldsymbol{c}|}} = \frac{-\boldsymbol{c}}{2(\sqrt{\boldsymbol{c}^*\Omega\boldsymbol{c}})^3}\frac{\boldsymbol{c}^*\Omega\boldsymbol{c}}{|\boldsymbol{c}^*\Omega\boldsymbol{c}|}\frac{\boldsymbol{c}^*\partial\Omega\boldsymbol{c}}{\partial \zeta_{\theta_i}^{(a)}} = \frac{\partial \boldsymbol{d}}{\partial \zeta_{\theta_i}^{(a)}}. \tag{4.20}$$

Then using the chain rule an expression for $\frac{\partial E}{\partial \zeta_{\theta_i}^{(a)}}$ can be found by combining Eq. (4.18) and Eq. (4.20),

$$\frac{\partial E}{\partial \zeta_{\theta_i}^{(a)}} = \frac{\partial E}{\partial \boldsymbol{d}}\cdot\frac{\partial \boldsymbol{d}}{\partial \zeta_{\theta_i}^{(a)}} = \frac{-\boldsymbol{c}\boldsymbol{H}\boldsymbol{d}}{(\sqrt{|\boldsymbol{c}^*\Omega\boldsymbol{c}|})^3}\frac{\boldsymbol{c}^*\Omega\boldsymbol{c}}{|\boldsymbol{c}^*\Omega\boldsymbol{c}|}\frac{\boldsymbol{c}^*\partial\Omega\boldsymbol{c}}{\partial \zeta_{\theta_i}^{(a)}}. \tag{4.21}$$

The first two terms are the invariant regardless of the choice of derivative, and require very little extra calculation than is already required during imaginary time propagation. $\frac{\boldsymbol{c}^*\Omega\boldsymbol{c}}{|\boldsymbol{c}^*\Omega\boldsymbol{c}|} = \pm 1$; $(\sqrt{|\boldsymbol{c}^*\Omega\boldsymbol{c}|})$ is already required for normalisation and $\boldsymbol{c}\boldsymbol{H}\boldsymbol{d}$ easily calculated. The only derivative that needs to be calculated is of the overlap matrix which can be done very simply with modification of the choice of amplitudes in the overlap equation. The derivative of an overlap matrix element with respect to a coefficient not in that overlap will be zero as will any derivatives of diagonal elements of Ω. This removes the need to directly calculate derivatives of the Hamiltonian matrix which is computationally expensive.

4.4.2 Algorithmic Specifications

This development process can be split into two distinct categories: program design and program implementation. The program design for Zombie Gradient Descent has focused on deciding which ZS amplitudes are altered in a single step; the learning rate, γ, used at each step and choices to ensure a good rate of convergence. The implementation of the program requires the design of two key algorithms to produce Hamiltonian elements and gradients which need to be made as efficient as possible as well as the various decisions to be made about code base such as parallelisation and library routines. Despite the distinct categorisation made between parts of the development process both had to happen concurrently. In particular speed-ups in the gradient calculation algorithm led to changes in the program design. In this section a description of the program design will be given to understand the overall process of the program. Whereas a full description of the specific algorithmic choices implemented in the Zombie states program is given in Appendix B where the algorithmic development is also detailed.

A single epoch is completed when the entire data set (in this case the ZS basis set) is passed through the gradient descent algorithm. Generally, reducing the number of epochs, required to reach the function's minimum, will mean a faster program but the length of time each epoch takes to complete is also an important consideration. There are three main variations in the type of gradient descent algorithm that differ in how a single epoch is completed. In batch GD the gradient is calculated for the entire data set and all points updated concurrently. Conversely in stochastic GD the gradient is calculated for a single instance which is used to estimate the gradient of the entire data set; each epoch is made up N iterations as the process is repeated for each individual data point. Mini-batch GD uses a subset of data points to estimate the gradient before updating the input values which requires fewer iterations per epoch than stochastic GD [138]. However, these methods have predominately been developed for machine learning and neural network applications where input data is used to train a system which is slightly different to the optimization problem for a set of ZS basis functions. Within a basis set each ZS can be very different to others as collectively they describe the electronic structure of the system so it would not be appropriate to alter all ZSs based on the gradient calculated from either a single or a small group of other ZS basis functions. The batch GD method was not appropriate due to the back tracing condition duly discussed. Thus, a coordinate gradient descent algorithm is employed: on every epoch each coefficient within each ZS basis function is individually attempted to be altered [139]. So the gradient with respect to each "dead" and "alive" coefficient within a single ZS is calculated and

that amplitude is then altered,

$$\zeta_{\theta_i}^{'(a)} = \zeta_{\theta_i}^{(a)} - \gamma \cdot \frac{\partial E_p}{\partial \zeta_{\theta_i}^{(a)}} \tag{4.22}$$

this altered ZS amplitude means an altered wave function, denoted $|\Psi'\rangle$. With any change in the wave function a new Hamiltonian can be calculated and using imaginary time propagation a new ground state energy can be found $E' = \langle\Psi'|\hat{H}|\Psi'\rangle$.

However, rather than accepting every changed amplitude, a modified version of the Armijo rule is used [140]. The energy from the previous step is denoted E_p which gives the back tracing condition,

$$E' \leq E_p - \alpha \tag{4.23}$$

which if met then the new "dead" and "alive" coefficients are accepted and the system ground state energy is updated, $E_{p+1} = E'$. The parameter α can be set to control the minimum step down in the ground state energy. In practice α is set to ensure any reduction in the energy is within the numerical precision of the computer system. The back tracing condition was integral for convergence without it ensuring lower ground state energies the process just oscillates around a point. This is illustrated in Fig. 4.7 which shows, for the truncated Li_2 system, the gradient descent algorithm getting stuck in a local minima rather than converging towards the ground state. This justifies the "wasted" cost of calculating a new Hamiltonian element that are not then used in a subsequent step. It is still important to try and minimise the amount of wasted computational cost which is why the batch GD method was not appropriate. If all amplitudes are altered concurrently then an entirely new Hamiltonian needs to be calculated and then possibly rejected. In fact this becomes more likely the more amplitudes are changed at once. Whereas, attempting to change amplitudes in a single Zombie state means it is only necessary to calculate a single new row of Hamiltonian matrix elements. A completely new Hamiltonian would also need to be calculated at each gradient descent step if amplitudes for the same orbital across all Zombie states were also concurrently altered. The computational cost of calculating a new Hamiltonian element is the same if a single orbital or all orbitals in the same Zombie state have been changed. Thus, a basis set of N_{bf} ZSs with N_{orb} spin orbitals would take N_{bf} steps to complete an epoch if all amplitudes in a single state were altered concurrently compared to $N_{bf} \times N_{orb}$ steps if altering each amplitude individually. However, as established by the biasing method and the principles of CASSCF calculations higher energy orbitals can be treated differently to lower energy orbitals. Low energy core orbitals are likely to be occupied and so have "alive" amplitudes close to 1. Once the amplitudes are within this region they are likely to only need small corrections achieved with a small learning rate. The same argument can be made for higher energy orbitals and "dead" amplitudes

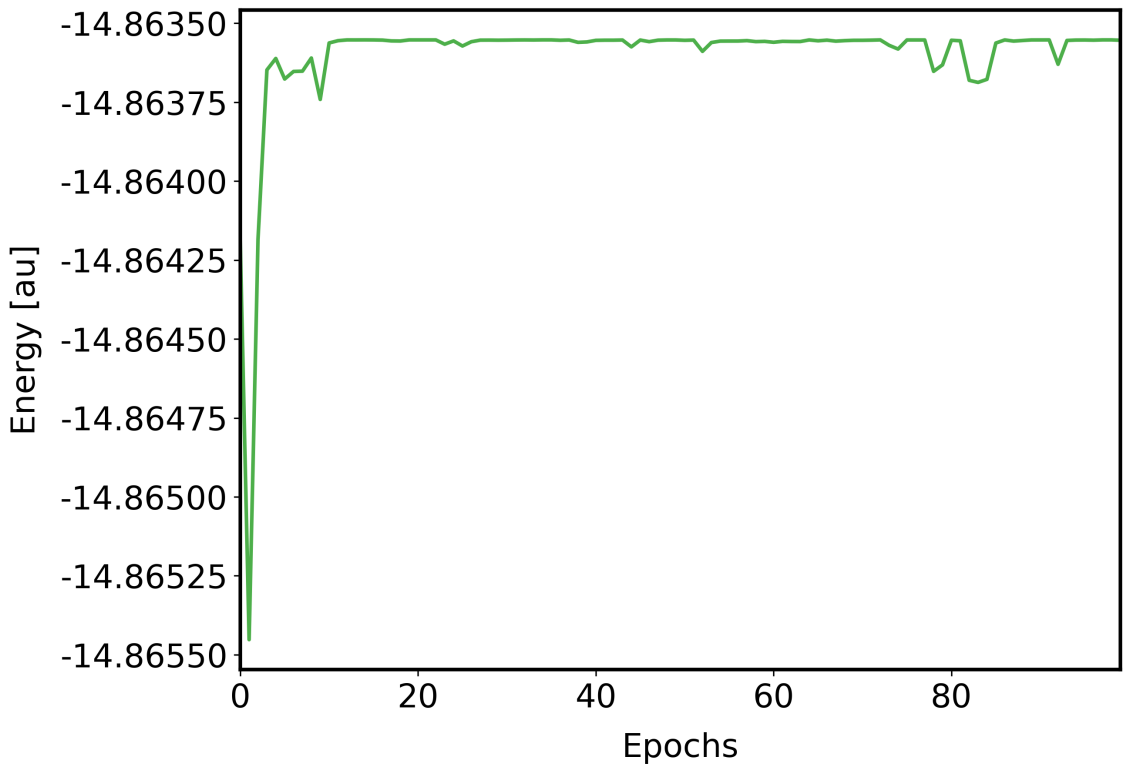


Figure 4.7: The ground state energy over 100 epochs is shown when the back tracing condition in Eq. (4.23) is not used. Accepting all alterations to the Zombie state amplitudes causes the gradient descent process to oscillate around a local minima rather than converge towards the ground state energy. A basis set of ten ZSs was used for the same truncated Li_2 system with ten spin orbitals in the $6-31G^{**}$ basis.

being also being near to one. Orbitals in the active space have amplitudes that can fluctuate much more greatly and can be changed using larger learning rates. So attempting to concurrently alter amplitudes in a Zombie state using the same learning rate could result in changes being rejected that individually would reduce the ground state energy.

The learning rate is controlled according to a schedule: fixed for each epoch then decreased for the next until a minimum is reach at which point the schedule restarts,

$$\gamma_p = \nu^p \gamma \tag{4.24}$$

with γ being the initial learning rate and ν the learning rate reduction parameter. p is increased by 1 at each epoch if $p > p_{max}$ then minimum learning rate has been reached and the cycle is restarted by setting $p = p_{min} = 0$ to reset the learning rate to γ. Larger learning rates mean greater steps and usually larger reductions in the ground state energy at each step while smaller learning rates mean more orbitals are likely to be changed per epoch. By using a decreasing learning rate scheme large steps down in the ground step energy can be achieved while also allowing multiple orbitals to be changed over the course of the schedule. A small change to amplitudes in one ZS can facilitate larger changes in another because the wave

function as a whole needs to contain superpositions of the important Fock states but not necessarily within each individual Zombie state. There is, however, a threshold for the number of ZSs altered in an epoch which if not reached then that learning rate is removed from subsequent schedules. As the minimum is approached the changes to $\zeta_{\theta_i}^{(a)}$ that result in a reduction to the ground state energy become smaller and hence a smaller learning rate is required. By removing learning rates that no longer result in a significant number of amplitude alterations the wasted computational cost, from calculating a new Hamiltonian that is rejected, is minimised. The range of learning rates used is easily controlled by changing the maximum and minimum values p can take. The gradient descent process is outline in Fig. 4.8. The criteria used to terminate the GD process is not shown but if the maximum number of epochs is reached or no further changes are possible the gradient descent process ends.

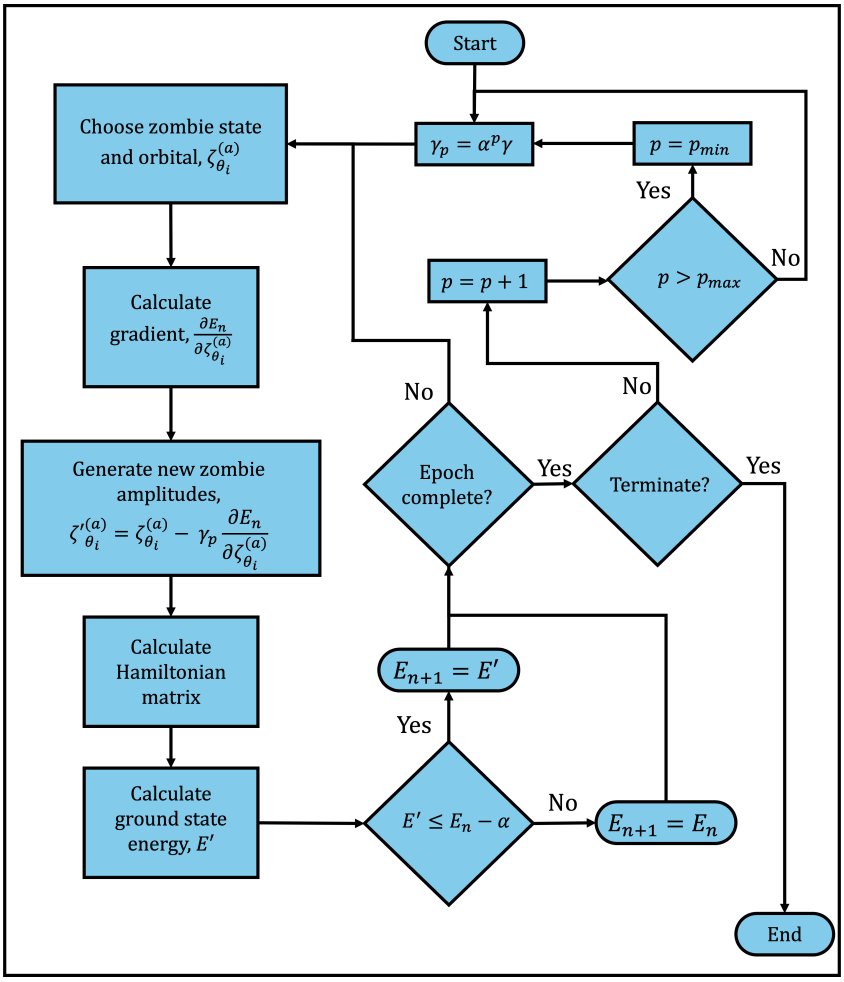


Figure 4.8: Flowchart summarising the gradient descent process used to optimise Zombie state amplitudes. Details of the exact termination criteria have been omitted for clarity.

4.4.2.1 Initiating the Wave Function

The simplest way to initially set the Zombie wave function is to randomly generate numbers $0 \leq \theta < 2\pi$ and then use Eq. (2.228) to assign the "dead" and "alive" coefficients as the sine and cosine of θ. A general biasing regime has been developed based on the regime used for the truncated lithium dimer. The regime generates a set of θ_j for each Zombie state which are then used to set the "dead" and "alive amplitudes using Eq. (2.228). Each θ_j is drawn randomly from a normal distribution, the centring and the width of which are set by the biasing regime for each spatial orbital. To set values the biasing regime uses the RHF configuration as a reference. So for fully occupied spatial orbitals the normal distributions are centred at $\pi/2$ to bias towards larger "alive" amplitudes. For all other orbitals the Gaussian distributions are centred at 0. The width of the distributions are then equally spaced between start and end points for the "dead" and "alive" biased orbitals. The "alive" biasing distributions, centred at $\pi/2$, have widths starting and ending

$$\sigma_{alive} : \epsilon_{a1} \cdot \lceil \frac{10}{N_{orb}} \rceil \rightarrow \epsilon_{a2} \cdot \lceil \frac{10}{N_{orb}} \rceil. \tag{4.25}$$

The distributions centred at 0 that start and end,

$$\sigma_{dead} : \epsilon_{d1} \cdot \lceil \frac{10}{N_{orb}} \rceil \rightarrow \epsilon_{ad2} \cdot \lceil \frac{10}{N_{orb}} \rceil. \tag{4.26}$$

ϵ values for the each set of distributions can then be set in the code. Each Zombie state can then be initialised, the normal distribution for each spatial orbital being used to generate θ_j for both constituent spin orbitals. This regime is refereed to as biasing regime 1.

A second biasing regime was also developed that classifies the spin orbitals into three groups in a similar way to CASSCF. The core orbitals are set to have a fully occupied "alive" coefficient; active space orbitals are randomly generated from the interval $0 \leq \theta < \frac{1}{2}\pi$ and virtual orbitals are all set to $\theta = 1.0 \times 10^{-4}$. Eq. (2.228) is again used to ensure normalised "dead" and alive coefficients. The regime has been summarised in Table. 4.2 showing how it is set up for different numbers of electrons. It should be noted that regardless of the method used to initiate the Zombie states all spin orbitals are treated equally and allowed to be altered by the gradient descent program. Thus, core orbitals initially set to have "dead" and "alive" coefficients of 0 and 1, respectively, can still be altered. A final additional detail is that the first basis function in all basis sets is set to be the RHF determinant and is unchanged by the GD algorithm making the initial wave function energy a good first guess.

Table 4.2: Table showing the biasing regime used to initiate Zombie states used in the gradient descent method[1].

N_{el}	Values of θ for range of spin orbitals, ζ_i		
	Core, $\theta = 1/2\pi$	Active, $0 \leq \theta < 1/2\pi$	Virtual, $\theta = 1.0 \times 10^{-4}$
> 4	$1 \leq \zeta_i \leq 4$	$5 \leq \zeta_i \leq (N_{el} + N_{act})$	$(N_{el} + N_{act}) < \zeta_i$
4	$1 \leq \zeta_i \leq 2$	$3 \leq \zeta_i \leq (N_{el} + N_{act})$	$(N_{el} + N_{act}) < \zeta_i$
< 4		$1 \leq \zeta_i \leq (N_{el} + N_{act})$	$(N_{el} + N_{act}) < \zeta_i$

4.4.2.2 Cloning

A simple modification to the initialisation process when is to incrementally increase the size of the basis set once the existing ZS functions have undergone some optimisation by the gradient descent process. This is inspired by the cloning method implemented in MCE [15]. Rather than starting with a large basis set and its associated computational cost a smaller basis set ZS functions is initialised which can be optimised faster. Then when the cloning conditions are satisfied additional Zombie states are generated and a new wave function is calculated. A maximum basis set size is set as well as the number of ZSs that are added. Cloning is initiated after a certain number of epochs or when the number of Zombie states altered does not reach a predetermined threshold. The cloning step occurs in-between an epoch.

4.4.3 Results

The gradient descent algorithm is demonstrated for a range of different chemical systems and basis sets. All one- and two-electron integrals are calculated using PyScf as are the comparative full-CI energies unless otherwise indicated [134]. PyScf is used as a helper program separate to the main Zombie states code, full details of how this works can be found in D. The type of initial Zombie state and the electronic basis set used is indicated for each molecule. When cloning has been used the additional Zombie states are of the same kind as the initial basis set. Different maximum gaps between cloning events are used but conditional cloning is set so a cloning event occurs when less than a third of the ZSs are altered, during an epoch with the smallest learning rate. The minimum and maximum learning rates are specified for each system but the learning rate reduction parameter, $\nu = 0.2$, is used universally. All energies are given in atomic units accurate to $1.0 \times 10^{-6} au$ which is equivalent to $6.275 \times 10^{-4}\ kcal/mol$, $2.626\ J/mol$ or $2.721 \times 10^{-2}\ meV$ [141].

[1] N_{el} is the number of electrons; $N_{act} = 4$ if N_{el} is even or $N_{act} = 5$ if N_{el} is odd. "Dead" and "alive" coefficients are generated using $cos(\theta)$ and $sin(\theta)$ respectively. The active space orbitals are randomly generated.

4.4.3.1 Li_2 (Truncated Basis Set)

For completeness the gradient descent process is applied to the truncated Li_2 basis with ten spin orbitals in the $6-31G^{**}$ basis used throughout this thesis. Three basis sets consisting of 10, 20 and 30 Zombie states underwent gradient descent. All amplitudes were initiated randomly except the first Zombie state which was kept as the RHF determinant. A learning rate cycle of $\gamma = 2500$ was used with a $p_{min} = 6$ giving a final learning rate of $\gamma = 0.16$. For the 30 ZS basis set the energy over the course of the gradient descent process is shown in Fig. 4.9, with the reduction in the ground state energy being clearly demonstrated in the Fig. 4.10. The results for all three basis sets have been summarised in Table. 4.3. Unsurprisingly, increasing the size of the basis set yields final ground state energies closer to the full-CI energy. The ground state energy for the basis of 30 Zombie states set up using the original biasing method, detailed in Table. 4.1, is also shown for comparison. Basis sets of all sizes, that underwent gradient descent, produce final ground state energies far closer to the true full-CI energy than the biased basis. Significantly, all spin orbitals in the wave functions optimised by gradient descent are treated equally, meaning no assumptions have been made about the electronic configurations. Further, the optimised basis sets of 20 and 30 Zombie states have final ground state energies within chemical accuracy of the full-CI energy, while being significantly smaller than the complete basis set of 1024 configurations or 210 if only the correct number of electrons are considered.

Table 4.3: Table summarising the ground state energies, for the truncated Li_2 system in the $6-31G^{**}$ basis. The 10, 20 and 30 Zombie state basis sets have all undergone the gradient descent process[1].

Basis Set	Ground State Energy $[au]$	Percentage Error with full-CI [%]	CPU Time $[s]$	Error/Time
RHF	-14.863553	5.6×10^{-2}		
10 ZS	-14.871912	1.4×10^{-5}	145	9.7×10^{-8}
20 ZS	-14.871913	4.2×10^{-6}	773	5.4×10^{-9}
30 ZS	-14.871914	1.4×10^{-7}	2976	4.7×10^{-11}
30 biased ZS	-14.866963	3.3×10^{-2}		
full-CI	-14.871914			

[1]The 30 biased Zombie state basis is constructed using the biasing method detailed in Table. 4.1. The full-CI energy was calculated by diagonalising the Hamiltonian matrix for the complete Slater determinant basis. The biased basis gives a ground state energy with an error the same order of magnitude as the RHF energy whereas the basis sets optimised by gradient descent all show significantly improved energies compared to the full-CI energy. The CPU time required to to complete the GD process for each basis set. The final column shows how the error scales with CPU time.

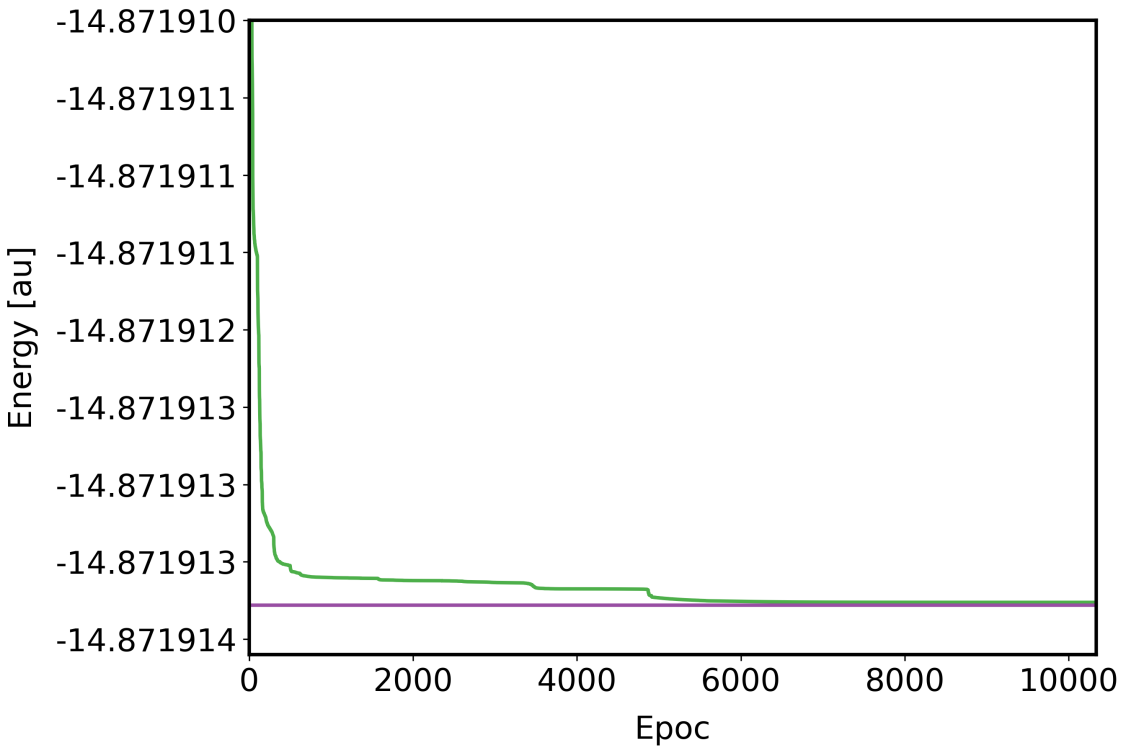


Figure 4.9: Plot showing the gradient descent process for the truncated Li_2 molecule in the $6-31G^{**}$ basis with a basis set of 30 randomly initialised Zombie states. The full-CI energy, found by diagonalising the complete basis is also shown for comparison as the solid (purple) line.

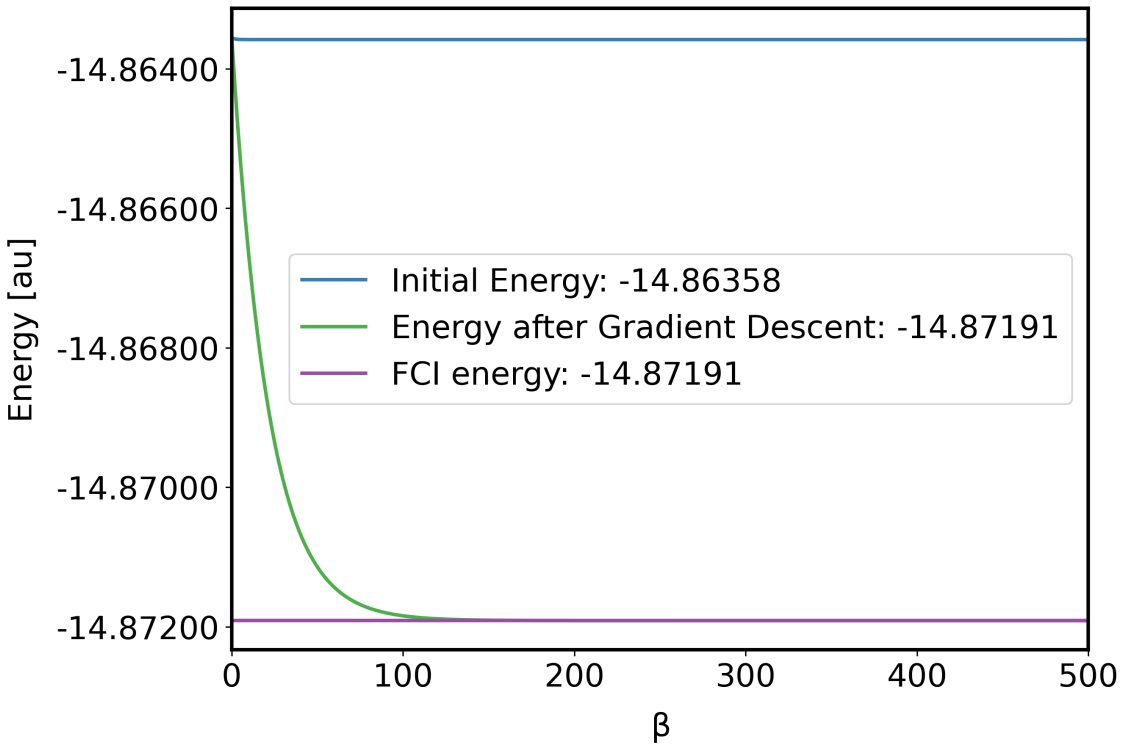


Figure 4.10: Plot comparing the imaginary time propagation for the initial and final wave function to find the ground state energy of the truncated Li_2 molecule in the $6-31G^{**}$ basis. A basis set of 30 randomly initialised Zombie states. The full-CI energy, found by diagonalising the complete basis is also shown for comparison as the solid (purple) line.

The average populations for each orbital, before and after gradient descent are shown in Fig. 4.11. Random initialisation sets each orbital to have a broad range of occupations, similar for each orbital. Looking first at the average populations for the first four spin orbitals the gradient descent process clearly moves occupations closer to full occupancy – as would be expected. The final six orbitals then have significantly lower average populations. However, the spread of population values is also important – darker plots showing overlapping values. The gradient descent process causes populations to cluster at either end of the scale for each orbital. Orbitals nine and ten have a higher average population than orbitals 5-8 but these are clustered at either extreme. Whereas, the lower orbitals show a greater spread of values indicating orbitals with a more variable occupation. This visualises how a Zombie basis, when properly optimised, can effectively describe the ground state of a system. The ZS functions cancel out incorrect configurations, with orbital populations generally reflecting either an occupied or unoccupied state. However, the Zombie states still include contributions from low population configurations that are nonetheless crucial for recovering full-CI energies.

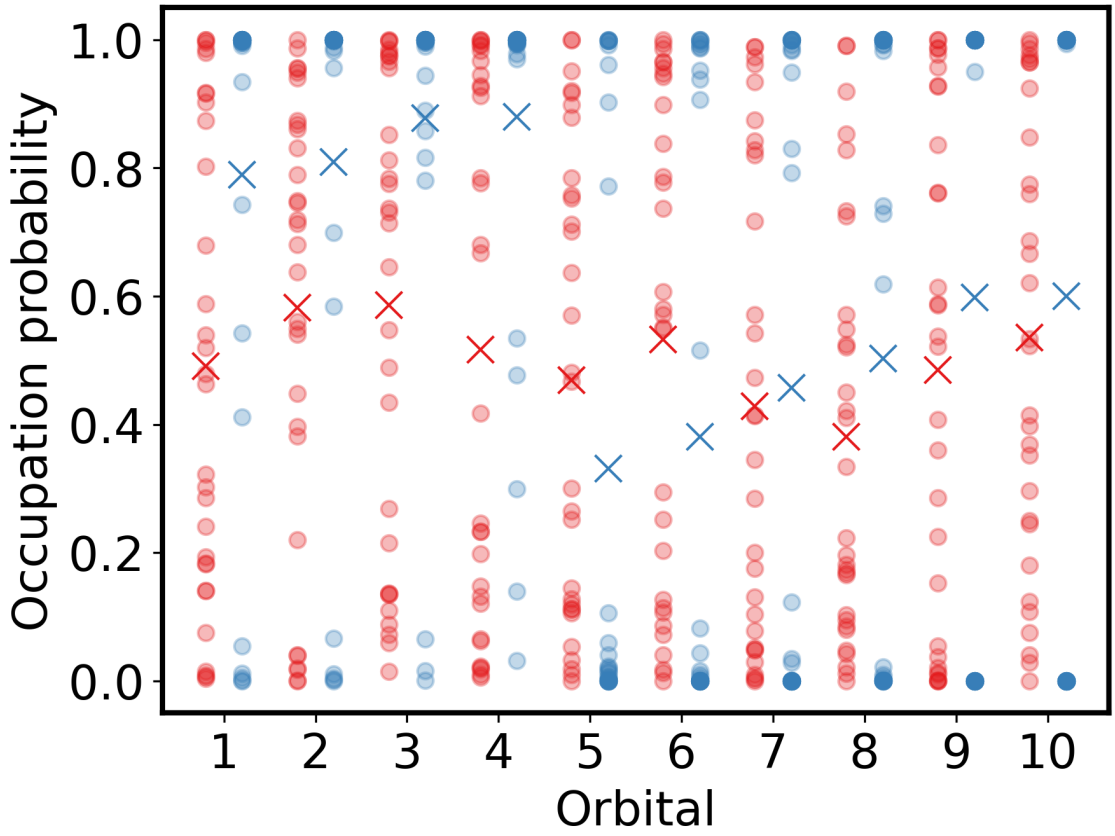
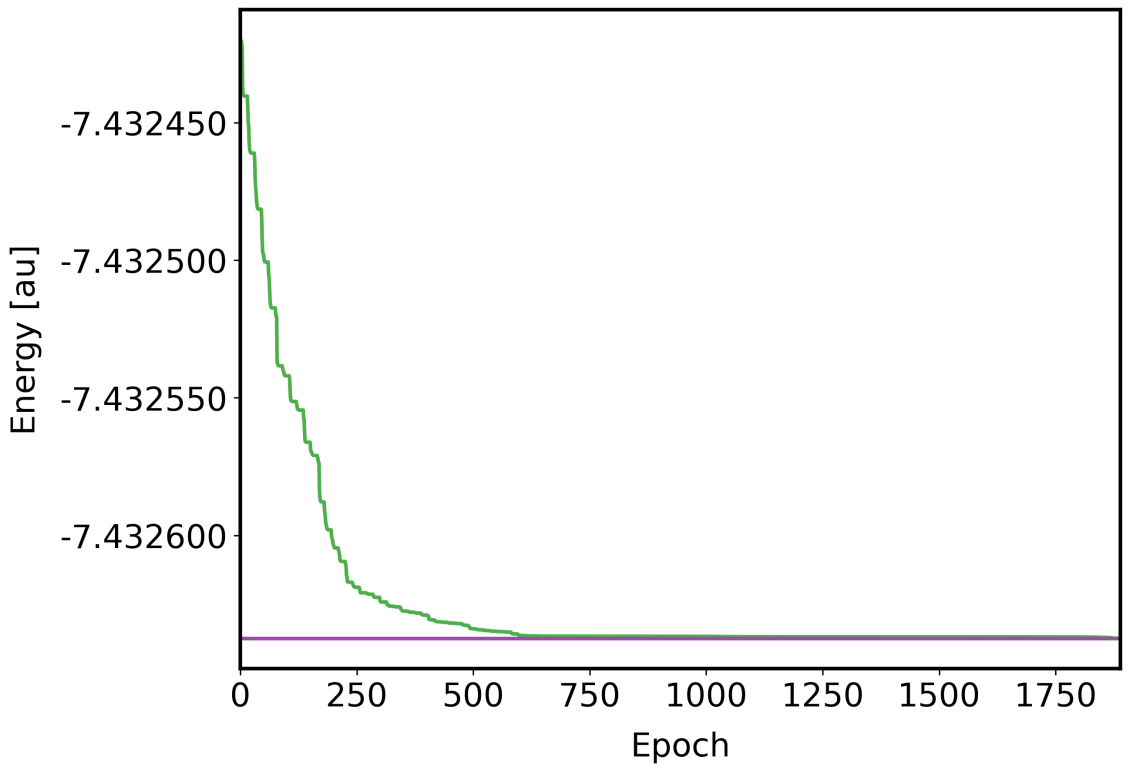


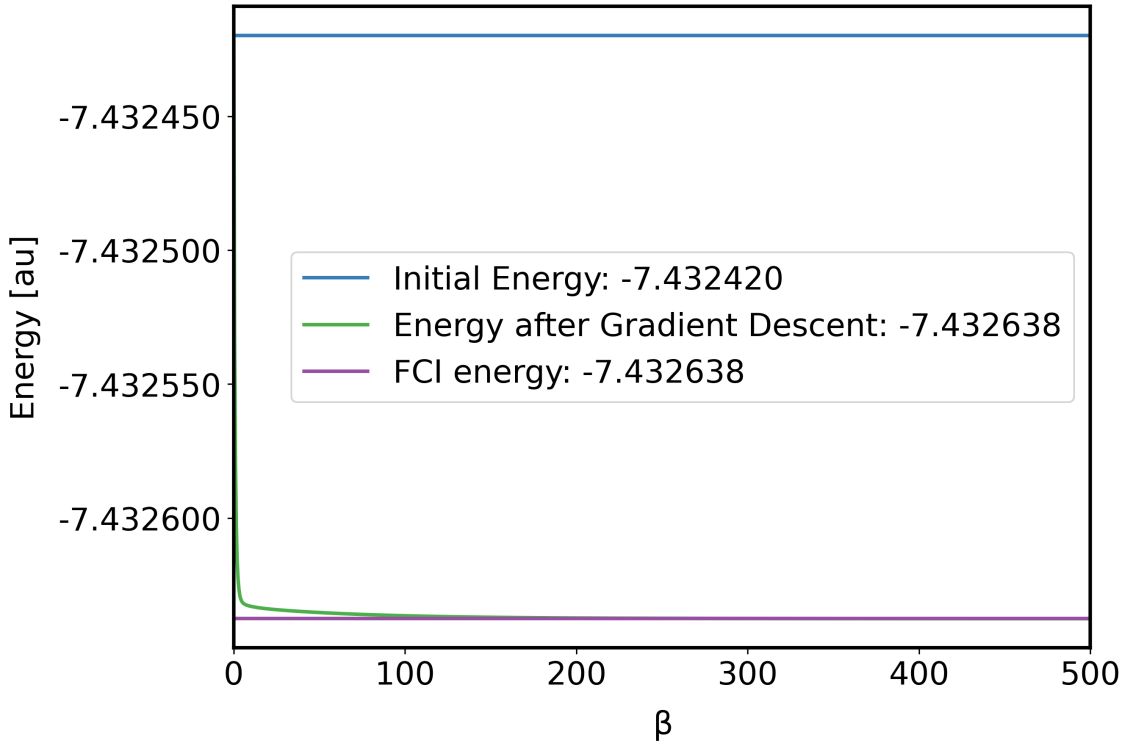
Figure 4.11: Occupational probability plotted against orbital number for ten spin orbitals for the lithium dimer before and after gradient descent. The population of each Zombie state, for each orbital, is plotted with the average occupation marked with a cross. Values in red are the initial randomly generated populations and blue values are the populations after gradient descent.

4.4.3.2 Atomic Systems in the cc-pVDZ Basis

The ground state energies of row one elements are a good set of systems to demonstrate the capabilities of an electronic structure method. The cc-pVDZ basis was used as it offers a higher level of complication than the truncated Li_2 system while also having readily available exact energies for comparison. An initial basis of $N_{bf} = 4$ Zombie states was used for the lithium atom. Cloning events occurred at the end of each learning rate cycle until the maximum basis set size was reached. This was initially set to $N_{bf} = 40$ and then extended to $N_{bf} = 45$ at 977 epochs. The gap between cloning events was doubled when the basis set size was a multiple of ten. An initial learning rate of $\gamma = 62500$ was used with a $p_{min} = 14$ giving a final learning rate of $\gamma = 1.05 \times 10^{-5}$. The gradient descent process can be seen in Fig. 4.12.a. which shows as the exact ground state energy is approached the rate of reduction slows. 315 epochs and 23 ZSs were required to achieve a ground state energy accurate to 5 d.p. but required a further 1574 epochs and 22 additional Zombie states to be within $1.0 \times 10^{-6} au$ of the full-CI energy. Fig. 4.12.b. plots imaginary time propagation for the initial and optimised wave function. The biasing regime used the values $\epsilon_{a1} = 0.0001$, $\epsilon_{a2} = 0.17$, $\epsilon_{d1} = 0.35$ and $\epsilon_{d2} = 0.15$ to set the start and end points for the widths of the normal distributions. The occupational probability for the initial and optimised wave function, for lithium, is shown in Fig. 4.13. In a similar manner to the truncated Li_2 system the gradient descent process optimises amplitudes so the populations are clustered around either being fully "alive" or "dead". The average optimised populations show the first three sin orbitals having higher populations than any other and populations decreasing as orbital number increases. Comparing the initial biased populations in Fig. 4.13 to the randomly generated ones in Fig. 4.11 the biasing regime better captures the trend of decreasing populations for higher orbitals. However, the biasing regime sets average populations for all but the first spin orbitals much higher than their optimised equivalents. There is also a drop in population between spin orbitals three and four, as lithium has three electrons, that is not reflected in the average populations in the biasing regime at all.



a: Plot showing the ground state energy over the course of the gradient descent process.



b: Imaginary time propagation for the initial (blue) and optimised (green) wave functions. The full-CI energy is the (purple) horizontal line.

Figure 4.12: Plots showing the gradient descent process and initial and final ground state energies for Li in the cc-pVDZ basis. An initial basis of $N_{bf} = 4$ ZSs was used, cloning added a single ZS to reach a final basis set size of $N_{bf} = 45$. Learning rates ranged from $\gamma = 62500$ to $\gamma = 1.05 \times 10^{-5}$. The biasing regime, with values $\epsilon_{a1} = 0.0001$, $\epsilon_{a2} = 0.17$, $\epsilon_{d1} = 0.35$ and $\epsilon_{d2} = 0.15$ was used to initialise the ZSs.

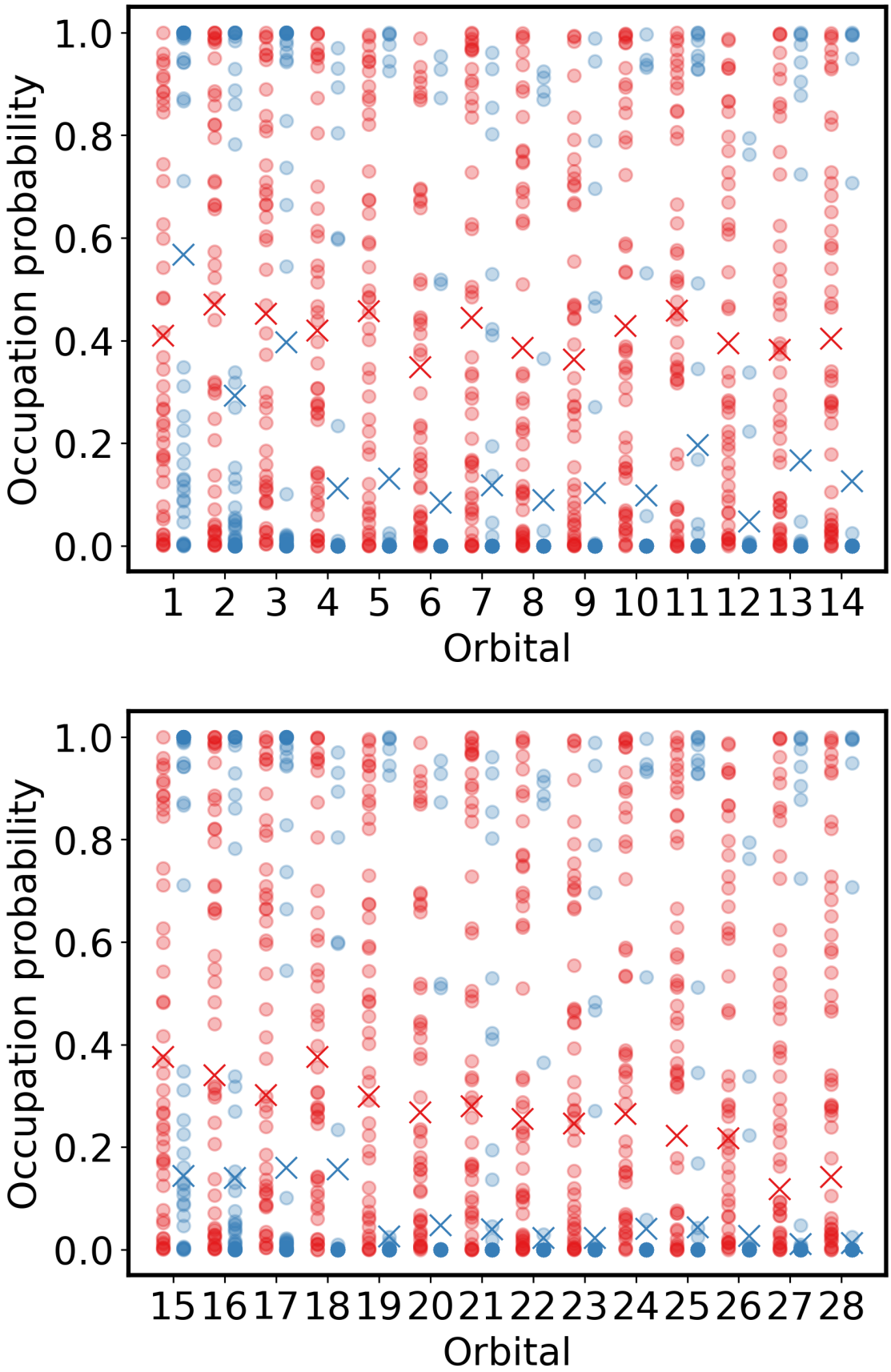


Figure 4.13: Occupational probability plotted against orbital number for all 28 spin orbitals (1-14 top, 15-28 bottom) for the lithium atom before (red) and after gradient descent (blue). The orbital population of each Zombie state is plotted with the average occupation marked with a cross. The biasing regime, with values $\epsilon_{a1} = 0.0001$, $\epsilon_{a2} = 0.17$, $\epsilon_{d1} = 0.35$ and $\epsilon_{d2} = 0.15$ was used to initialise the ZSs.

For the boron atom an initial basis of $N_{bf} = 4$ was used with a maximum of $N_{bf} = 60$ initialised using the biasing regime with the same ϵ values as for the lithium atom. The same learning rate cycle was used as in the lithium atom with the first regime set to have cloning events after one learning rate cycle. The gradient descent process is plotted in Fig. 4.14.a, the initial 500 epochs showing the largest and fastest reduction in energy after which the rate of change slows.

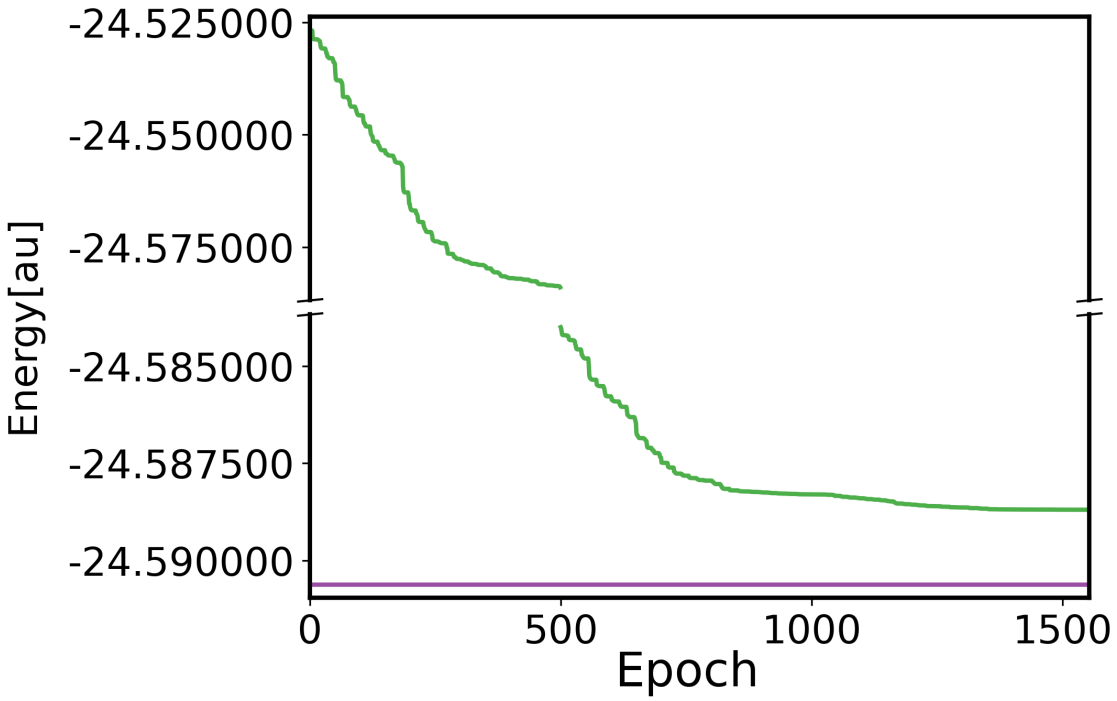
For the beryllium atom the optimised set of Zombie states from the lithium atom were used as the starting point. The only change being the first ZS which was swapping the Li RHF determinant for beryllium. The learning rate cycle was also extended so $p_{min} = 16$ giving a minimum learning rate of $\gamma = 4.10 \times 10^{-7}$. The gradient descent process and comparison of initial and final energies is shown in Fig. 4.15. The gradient descent process has managed to optimise the wave function to give a ground state energy that is significantly more accurate than the initial energy. But the exact energy has not been returned because of the slow rate of convergence combined with each epoch being computationally expensive.

The nitrogen atom was set up using the biasing method from Table. 4.2 with an initial basis of $N_{bf} = 4$ with cloning set after two learning rate cycles until a final size of $N_{bf} = 60$ was achieved. The learning rate cycled from $\gamma = 500$ to $\gamma = 1.05 \times 10^{-5}$ using reduction parameter $\nu = 0.2$. The energy after 1738 epochs is not yet converged as can be see by the energy in Table. 4.4. However, the updated biasing regime shown in Fig. 4.16 is a much better reflection of the expected populations and is the preferred method for initialising the Zombie states.

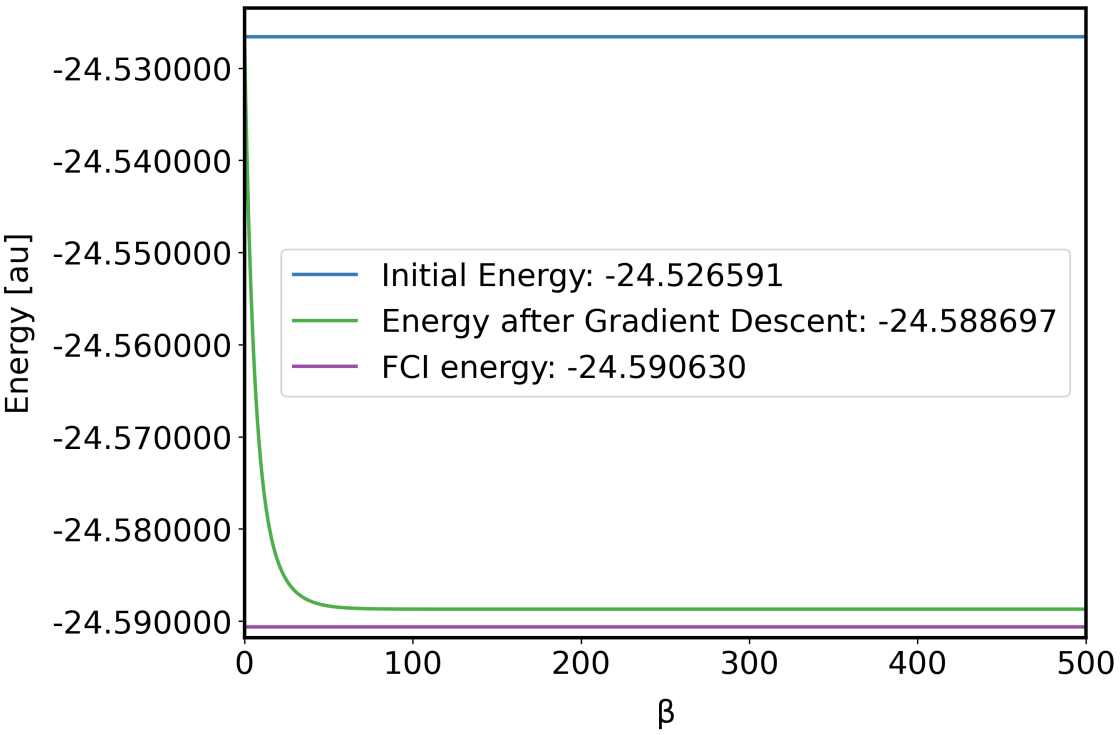
Table 4.4: Table comparing the ground state energies for selected row one elements in the cc-pVDZ basis[1]. Restricted Hartree Fock and full-CI energies are calculated using PyScf which is also used to generate the one- and two-electron integrals used in the Zombie state calculation [134].

	Zombie state		*PySCF*		
Atom	Energy $[au]$	N_{bf}	RHF Energy $[au]$	Full-CI Energy $[au]$	N_{bf}
$\mathrm{Li}^{(a)}$	-7.432638	45	-7.432420	-7.432638	3276
$\mathrm{Be}^{(a)}$	-14.61701	45	-14.572338	-14.617410	20475
B	-24.588697	56	-24.526591	-24.590630	98280
$\mathrm{N}^{(a)}$	-54.435063	60	-54.248220	-54.480115	1184040

[1]For the Zombie states N_{bf} is the final number of functions used whereas for the PyScf calculation N_{bf} is the maximum number of electron configurations with the correct number of electrons i.e $\frac{N_{orb}!}{N_{el}!(N_{orb}-N_{el})!}$. In all cases the Zombie state method requires orders of magnitude fewer functions to achieve chemically equivalent energies. (a) Energies are also equivalent to full-CI results in Ref. [135]. All energies are in atomic units.

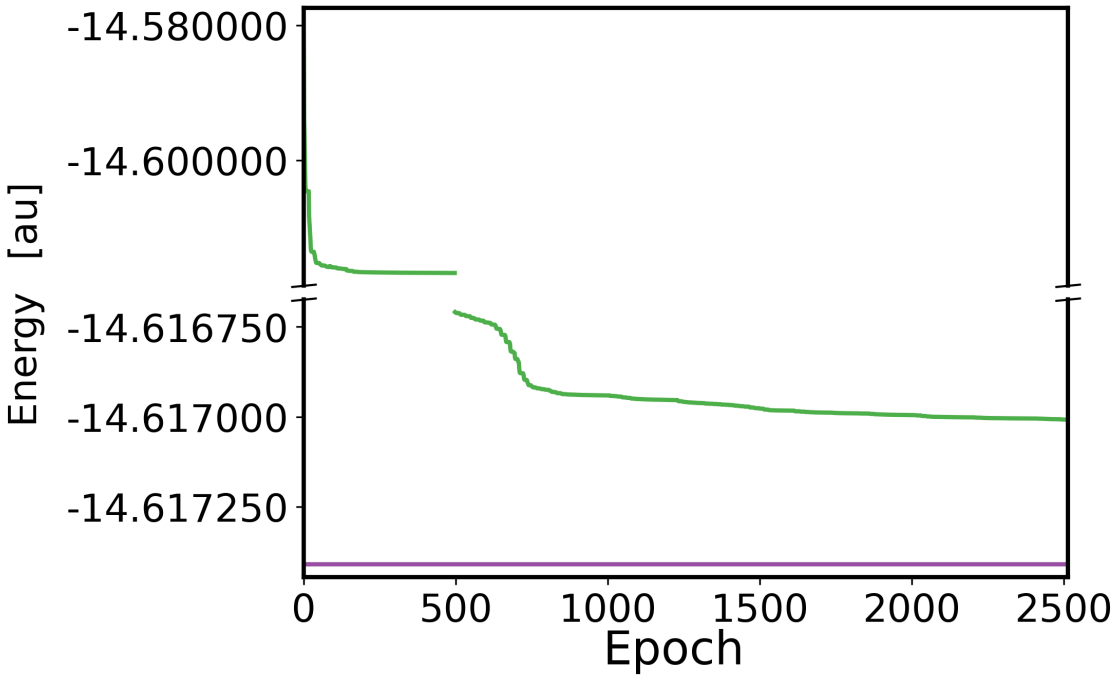


a: Plot showing the ground state energy over the course of the gradient descent process.

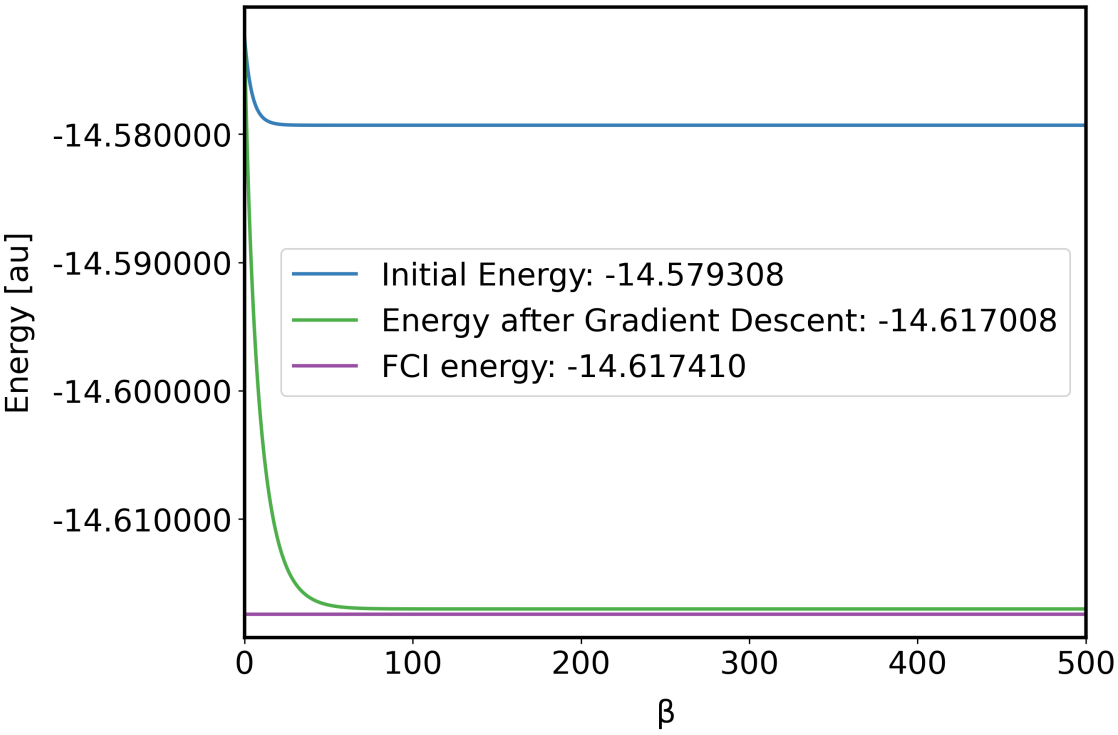


b: Plot showing the imaginary time propagation for the initial wave function and the final wave function optimised by gradient descent. The full-CI energy calculated using PyScf is the (purple) horizontal line [134]

Figure 4.14: Plot showing imaginary time propagation for an initial basis set of $N_{bf} = 4$ ZS and final wave function of $N_{bf} = 60$ ZSs optimised using gradient descent for Boron in the cc-pVDZ basis. Zombie states were initialised using the biasing regime with values $\epsilon_{a1} = 0.0001$, $\epsilon_{a2} = 0.17$, $\epsilon_{d1} = 0.2$ and $\epsilon_{d2} = 0.001$. Learning rates ranged from $\gamma = 62500$ to $\gamma = 1.05 \times 10^{-5}$ with a reduction parameter of $\nu = 0.2$.



a: Plot showing the ground state energy over the course of the gradient descent process.



b: Plot showing the imaginary time propagation for the initial wave function and the final wave function optimised by gradient descent. The full-CI energy calculated using PyScf is the (purple) horizontal line [134]

Figure 4.15: Plots showing the gradient descent process and initial and final ground state energies for Be atom in the cc-pVDZ basis. The optimised Zombie states from the lithium atom, from Fig. 4.12 were used as the initial basis. The learning rates ranged from $\gamma = 62500$ to $\gamma = 1.05 \times 10^{-5}$ with a reduction parameter of $\nu = 0.2$.

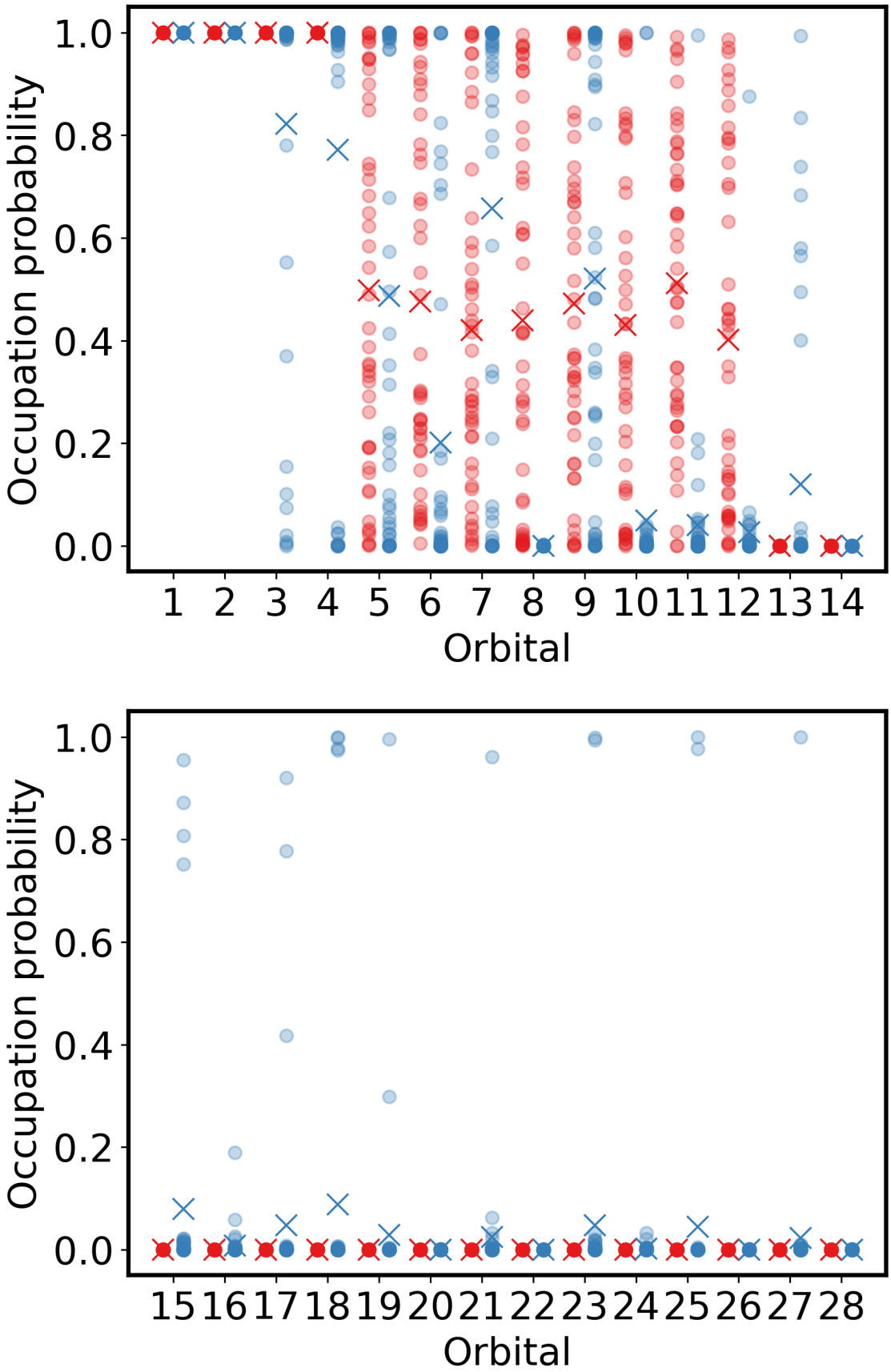


Figure 4.16: Occupational probability plotted against orbital number for the all 28 spin orbitals for the nitrogen atom before (red) and after (blue) gradient descent. The population of each Zombie state, for each orbital, is plotted with the average occupation marked with a cross. The populations were set using the biasing regime described in Table. 4.2.

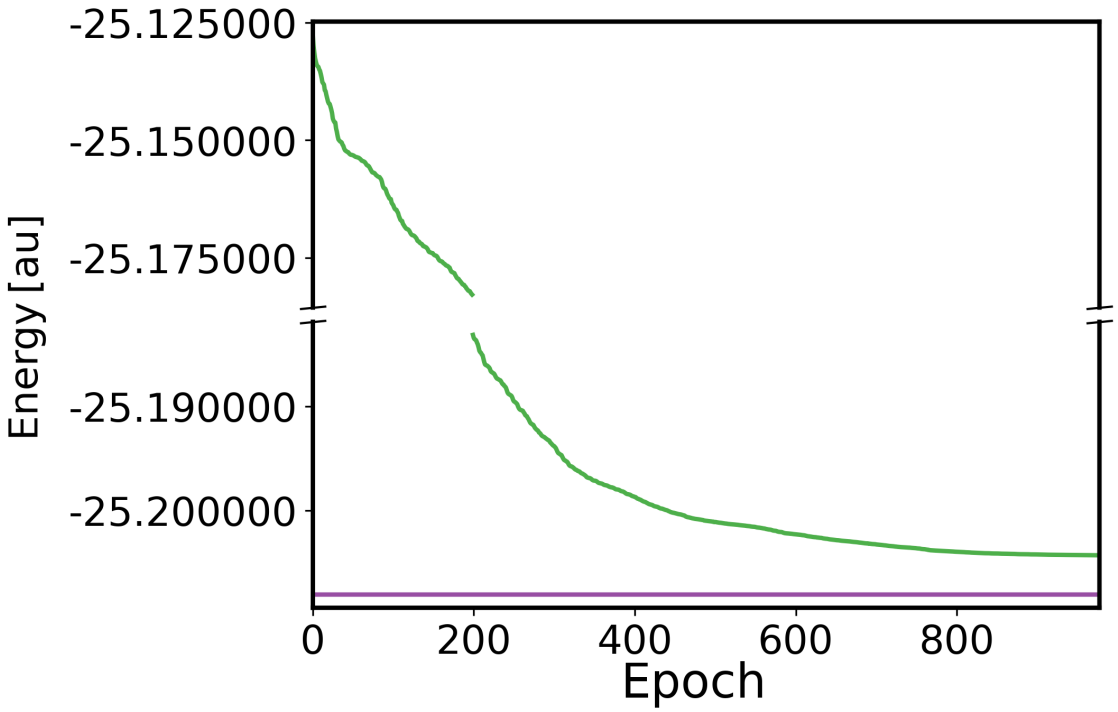
4.4.3.3 BH in the $6-31G^{**}$ Basis

Two systems of for BH with different bond lengths were considered using Zombie states. Firstly BH with a bond length of 1.234 Å was considered with an initial basis set of $N_{bf} = 25$ growing to a final basis set size of $N_{bf} = 300$ all initialised using the biasing method in Table. 4.2. The learning rate cycle which ranged from $\gamma = 2500$ to $\gamma = 2.56 \times 10^{-4}$ at the end of which cloning was used to add five additional Zombie states. A similar system was also investigated with a bond length of 4.0 Å initial basis of $N_{bf} = 20$ which is increased by a five ZSs, at the end of the learning rate cycle, up to a maximum basis set size of $N_{bf} = 245$.

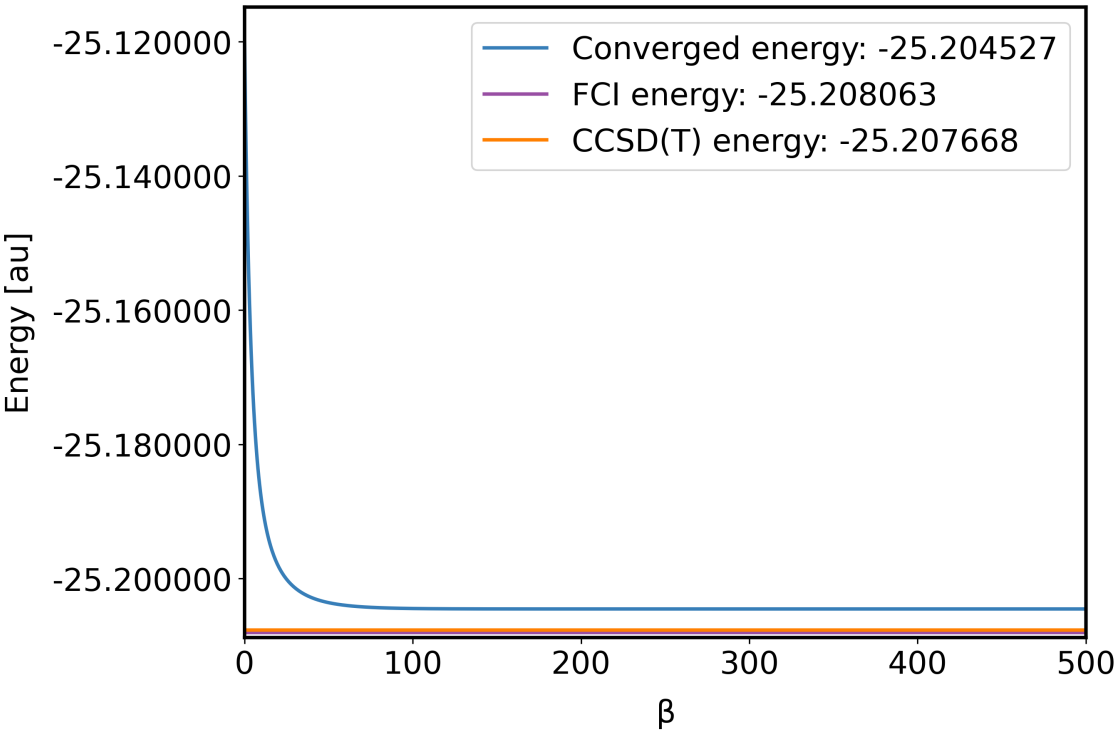
The BH molecule in the $6-31G^{**}$ has 38 spin orbitals and six orbitals which gives a total of $2,760,681$ possible configurations with the correct number of electrons so both systems are using basis sets that are significantly smaller than the complete Fock space. Though not completely converged to the full-CI result the Zombie state method is approaching the the full-CI energy with the stretched configuration approximately 1.40 $kcal/mol$ higher than the exact energy. For the 1.234 Å bond length system the ZS method has a ground state energy that is 2.2 $kcal/mol$ higher than the full-CI energy. However, for the stretched bond length the Zombie state method does give a ground state energy that is more accurate than the CCSD(T) method. In both Fig. 4.17.a and Fig. 4.18.a the gradient descent process has not plateaued and with further epochs the full-CI energy should be achieved.

4.4.3.4 Diatomic Molecules in the cc-pVDZ Basis

Gradient descent was also applied to three larger diatomic molecules using the cc-pVDZ basis. Using the second biasing regime a Li_2 system was set up with ten Zombie states. The learning rate cycled from $\gamma = 62500$ to $\gamma = 0.16$ with a single Zombie state added at the end of the cycle until a maximum of $N_{bf} = 100$ was reached. The gradient descent process is shown in Fig. 4.19 with the full-CI energy for comparison shows the process has not yet fully converged but this should be possible with more epochs.

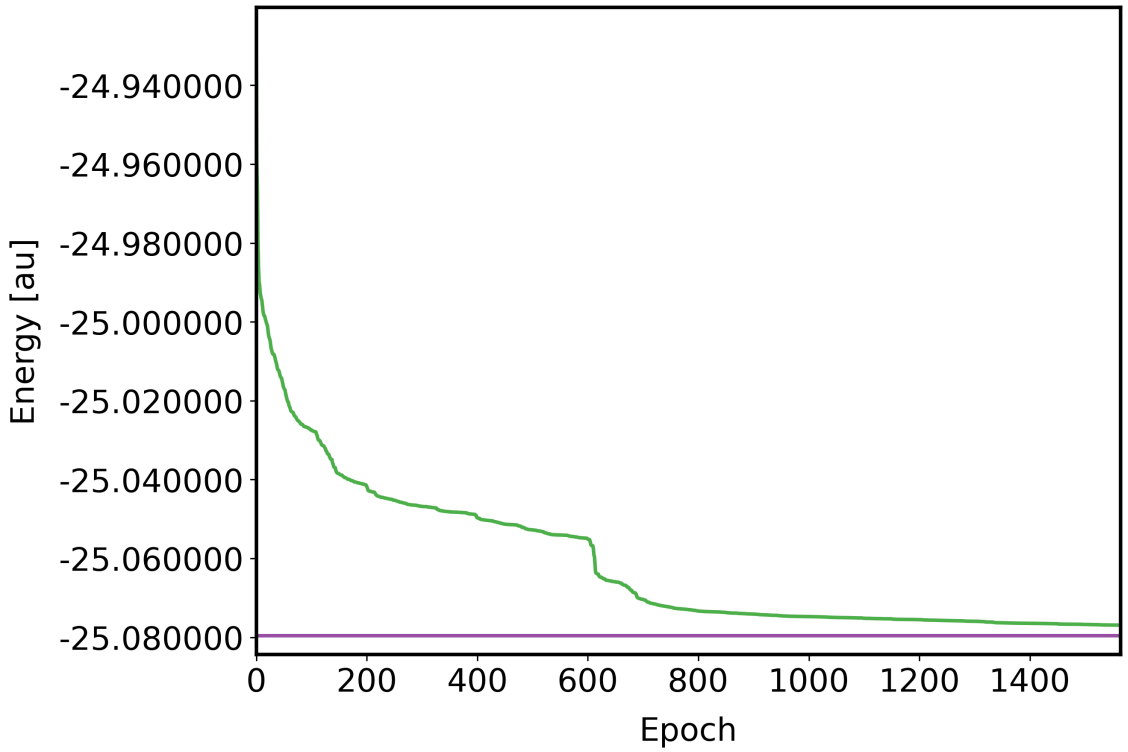


a: Plot showing the ground state energy over the course of the gradient descent process.

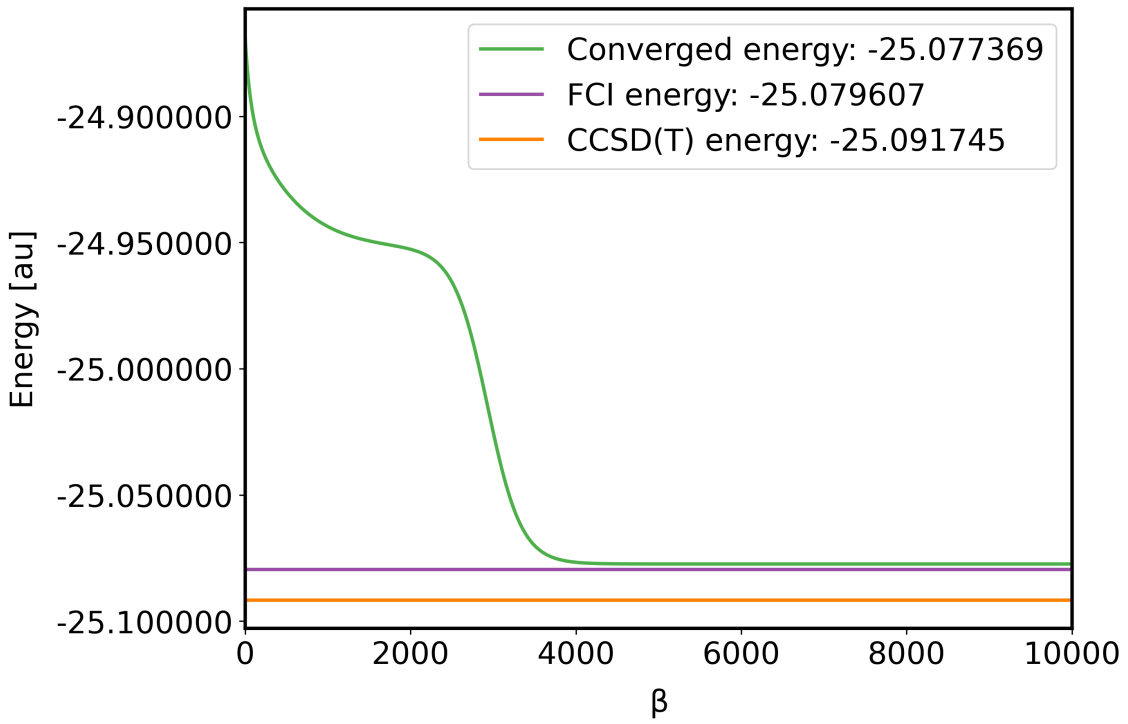


b: Plot showing the imaginary time propagation for the final wave function optimised by gradient descent. The full-CI energy is the purple horizontal line and the CCSD(T) the orange.

Figure 4.17: Plots showing the gradient descent process and final ground state energies for BH in the $6-31G^{**}$ basis with a bond length of 1.234 Å for $N_{bf} = 300$. A set of $N_{bf} = 25$ Zombie states was initialised with biasing regime 2 with 5 additional functions added at the end of a learning rate cycle which ranged from $\gamma = 2500$ to $\gamma = 2.56 \times 10^{-4}$.



a: Plot showing the ground state energy over the course of the gradient descent process



b: Plot showing the imaginary time propagation for the final wave function optimised by gradient descent. The full-CI energy is the purple horizontal line and the CCSD(T) the orange.

Figure 4.18: Plots showing the gradient descent process and final ground state energies for BH in the $6-31G^{**}$ basis with a bond length of 4.0 Å. A set of $N_{bf} = 20$ Zombie states was initialised with biasing regime 2 with 5 additional functions added at the end of a learning rate cycle which ranged from $\gamma = 12500$ to $\gamma = 2.05 \times 10^{-6}$.

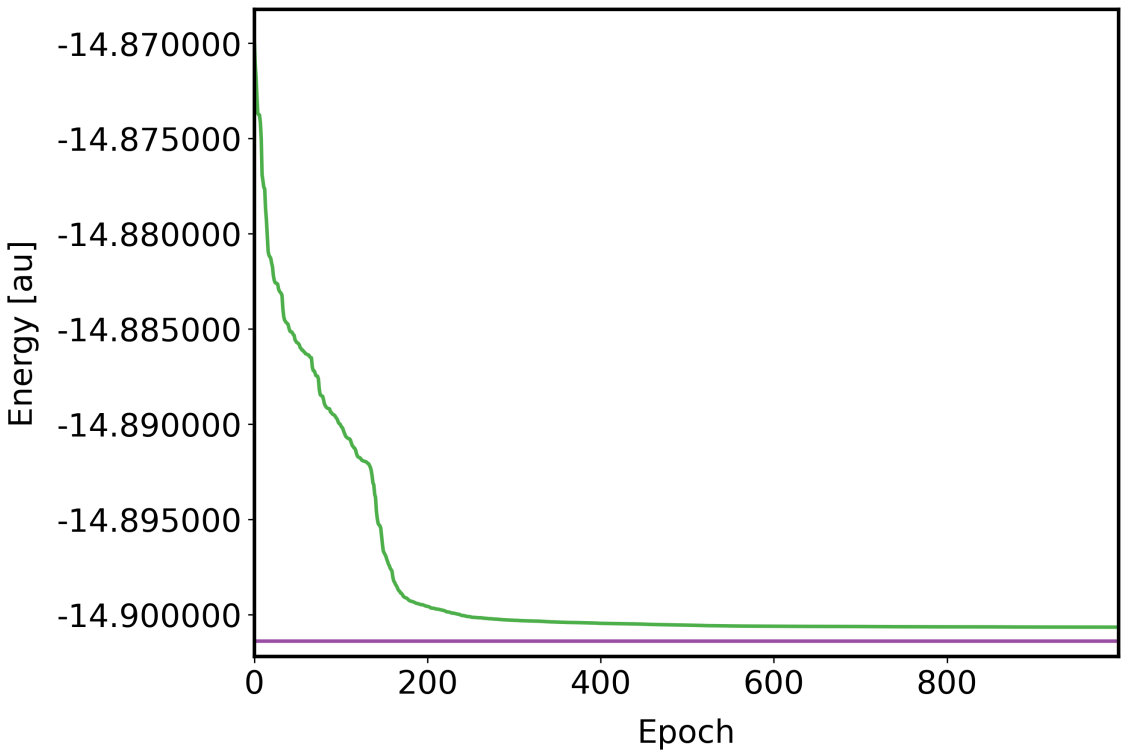


Figure 4.19: Plot showing the gradient descent process for Li_2 in the cc-pVDZ basis. An initial basis of $N_{bf} = 10$ was used, set up using biasing regime 2, with cloning occurring at the end of each learning rate cycle. A single ZS was added up to a maximum of $N_{bf} = 100$. The learning rate cycled from $\gamma = 2500$ to $\gamma = 0.16$. The full-CI energy calculated using PyScf is the (purple) horizontal line [134].

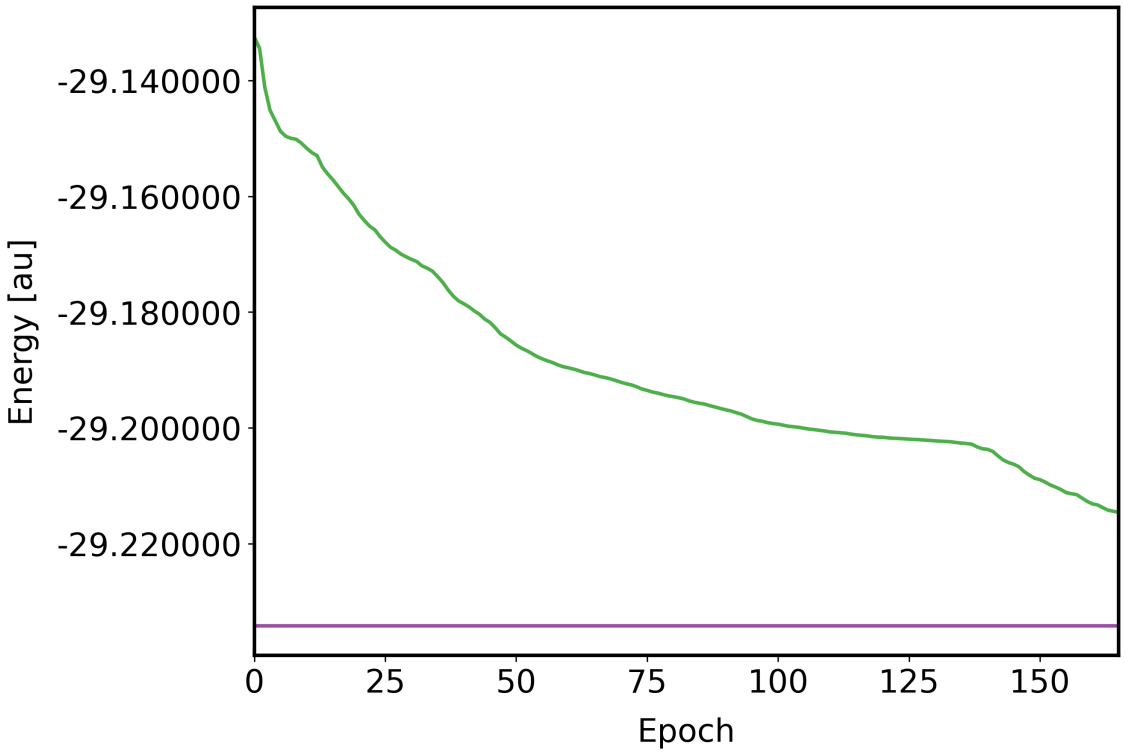


Figure 4.20: Plot showing the gradient descent process for Be_2 in the cc-pVDZ basis. An initial basis of $N_{bf} = 10$ was used, set up using biasing regime 2, with cloning occurring at the end of each learning rate cycle. A single ZS was added up to a maximum of $N_{bf} = 170$. There were three available learning rates $\gamma = 2500, 500, 100$ in the cycle. The CCSD(T) energy calculated using PyScf is the (purple) horizontal line [134].

A Be_2 molecule was then investigated using an initial ZS basis of $N_{bf} = 10$ and cloning used to increase the number of Zombie states, at the end of each learning rate cycle, to a maximum of $N_{bf} = 170$. A learning rate cycle of $\gamma = 2500, 500, 100$ was used to grow the size of the basis set quickly. The nitrogen dimer was set up using the same running conditions reaching a maximum basis set of $N_{bf} = 200$. The gradient descent process in both Fig. 4.20 and Fig. 4.21 is clearly not yet converged. But the ground state energy is still decreasing at a reasonable rate and so needs to be continued with lower learning rates available.

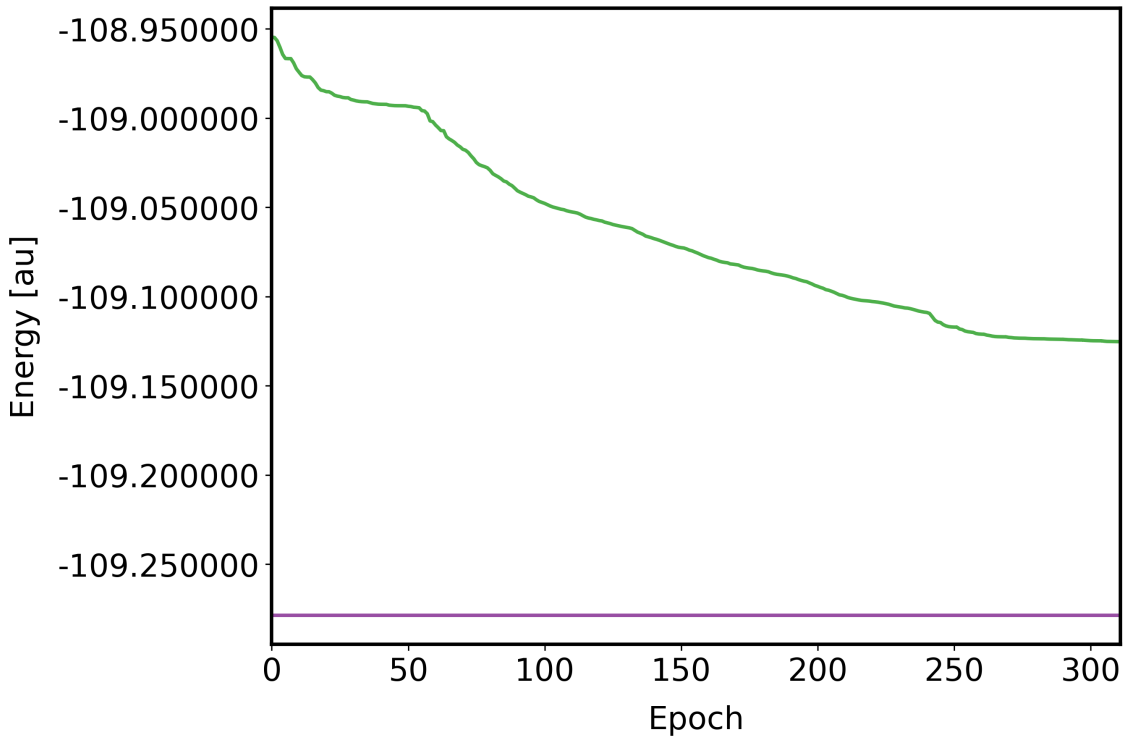


Figure 4.21: Plot showing the gradient descent process for N_2 in the cc-pVDZ basis. An initial basis of $N_{bf} = 10$ was used, set up using biasing regime 2, with cloning occurring at the end of each learning rate cycle. A single ZS was added up to a maximum of $N_{bf} = 200$. There were three available learning rates $\gamma = 2500, 500, 100$ in the cycle. The CCSD(T) energy calculated using PyScf is the (purple) horizontal line [134].

4.5 Conclusions and future work

The use of gradient descent to optimise Zombie state coefficients has significantly increased the complexity of system the method is capable of handling. The truncated lithium dimer system has been used throughout this work as a simple numerical example to verify properties of the Zombie state method. The biasing method indicated that, with proper sampling of ZS amplitudes, it should be possible to recover exact energies while using an incomplete basis set size. However, the biasing method when applied to Li_2 was limited by the need to fix core orbital occupations

to fully "alive" and the method overall lacked universality, as each orbital bias was set using a process of trial and error. The gradient descent algorithm essentially generalises the biasing process optimising "dead" and "alive" Zombie coefficients so the wave function can better describe the ground state energy. For the truncated Li_2 system, a basis set of 30 optimised Zombie states was required to recover the full-CI ground state energy. It should be noted that the number of epochs required for convergence is considerably longer than other results which is due to the relatively high minimum learning rate. This limits the number of possible alterations that are accepted as the exact energy is approached. In fact it can be seen from Fig. 4.10 that the energy is almost converged by 500 epochs and most are spent making very small adjustments. This highlights the importance of a range of learning rates being available to ensure a reasonably timed convergence. However, the energy did still converge with a basis set half the size required for the biased ZSs to achieve the same result. Moreover, the gradient descent process optimises amplitudes with no *a priori* information about electron configurations beyond the RHF determinant. Thus, allowing all orbitals in a system to be treated on an equal footing. This significantly improves the ZS method's ability to be applied universally, a stipulation in Pople's framework [3].

The fourth stage of Pople's framework for method development is the verification of the method compared to known quantities. The results summarised here evidence, for the first time, that the Zombie states method can be applied to larger systems that have not been significantly truncated. Atomic studies were used to verify both FCIQMC and MCCI and a similar set of results has been produced by the ZS method [83, 91]. The cc-pVDZ basis, for the first row elements contains 28 spin orbitals, significantly larger than the 10 in the truncated Li_2 system. For the lithium atom it was possible to optimise the Zombie basis to return the full-CI energy. Table. 4.4 shows that even for the unconverted systems the GD process does make significant improvements to the wave function's ability to describe the ground state, while using a basis set orders of magnitude smaller than the complete Fock space. Nonetheless, to allow direct comparison between the ZS method, FCIQMC and MCCI, further simulation of these atomic systems should be completed using the aug-cc-pVDZ basis. It should then also be possible to start calculating properties of the system beyond just the ground state energy such electron affinities. Applying the gradient descent process to the two BH systems showed promising results, with the energy nearly converging to the exact values. However, the diatomic systems in the cc-pVDZ basis demonstrated that the method is still hindered by its computational expense. For Be_2 and N_2 the steepness of the plots in Fig. 4.21 and Fig. 4.20 indicates that the gradient descent process is nowhere near complete but has been limited by the time required to complete a single epoch with these larger systems.

An important consideration in Pople's framework for developing a new method is

achieving reasonable computational time and cost. Despite the improvements to the Hamiltonian algorithm, outlined in Appendix B.3.1.3 and the indirect method for gradient calculation, the gradient descent process is still slow. For example, PyScf is able to calculate the full-CI ground state energy for any of the atomic systems discussed in this work in less than a minute whereas using gradient descent to optimise ZS coefficients takes multiple hours with multi-threading. Of course, these systems have been chosen precisely because there is a known point of comparison to demonstrate the capabilities of the ZS method rather than its current speed. Nonetheless, remedying the long calculation times is still important and there are number of different approaches that could be considered. The bottleneck in the gradient descent calculation is the recalculation of Hamiltonian matrix elements. The simplest solution would to use more computing power – increasing the number of parallel threads available. This approach is limiting, simulations of larger systems will quickly reach the computational limits again. Although, using multiple CPUs to calculate different matrix elements rather than splitting the calculation over multiple threads on a single CPU could be considered [142]. Alternatively, the computing power of a GPU could be utilised which is effective at completing large numbers of repetitions of the same calculation using different bits of data. For a particular system all Hamiltonian matrix elements are calculated using exactly the same summation of one- and two-electron components which ultimately can be seen as the same set of multiplication and additions simply using different Zombie states. Thus, the same instruction multiple data (SIMD) model used on a GPU would be well suited to a Hamiltonian matrix element calculation.

Further, it should also be possible to reduce the number of times the Hamiltonian recalculation algorithm is called. This can be achieved by reducing the number of epochs, increasing the efficiency of each epoch and finding an alternative to the Hamiltonian algorithm. Firstly, the number of epochs needed for convergence can be greatly reduced by using a basis set with an initial energy as close to the full-CI ground state energy as possible. In the MCE method sampling techniques have been shown to be very important and this is clearly the case with Zombie state amplitudes [15]. The current biasing regime only accounts for the number of electrons and sets all Zombie states in the same way. However, Fig. 4.11 shows that once optimised, although many of the orbital populations follow the expected pattern, there are still a significant number of amplitudes indicating the opposite – populations in higher orbitals and empty lower orbitals. Therefore, the biasing regime should attempt to replicate some of this behaviour while also taking into account system specific properties. The starting point of this work should be a systematic analyses of how the magnitudes, combinations and placement of "dead" and "alive" amplitudes, both within individual Zombie states and the basis set effect the ground state energy.

The current gradient descent algorithm optimises amplitudes by minimising the

ground state energy alone but there are of course parameters that could also be optimised including the number of electrons and the total spin. Simplest of all, each parameter could be optimised sequentially however this could be counter productive if for example the Zombie state is optimised to better satisfy the number operator but in doing so increases the overall ground state energy. Thus, it would likely be beneficial to optimise the ZSs taking all functions into account at the same time which can be done using the weighted sum method [143]. A new objective function, $h(X)$, can be defined,

$$h(X) = \sum_i w_i f_i(X) \tag{4.27}$$

with the weights w_i to be decided. The ZSs amplitudes can then be optimised using gradient descent with the gradient now being the weighted sum of gradients of multiple functions. The total spin, number of electrons and their configurations all contribute to describing the state of a system. Each Zombie state contains superpositions of states containing any number of electrons, with incorrect states hopefully being cancelled out to allow a accurate description of a state. By minimising the ground state energy and other system parameters concurrently the Zombie wave function should converge to the ground state faster. Rather than leaving these other, important, system parameters to be optimised by chance adding them to the gradient descent process ensures their optimisation aiding the cancelling of incorrect states. Moreover, the final wave function should also provide a better description of the ground state.

As described the, GD algorithm considers all ZS amplitudes for alteration in a single epoch. This means with the current back tracing method multiple Hamiltonian matrix elements are calculated and then discarded, which is computationally wasteful and time consuming. Therefore, reducing the number of Hamiltonian matrix elements that need to be calculated in an epoch, should give a faster convergence time. Firstly, if the algorithm can make smarter choices about which amplitudes to attempt to change, the number of rejected Hamiltonian matrix elements can be reduced. Some possible parameters that could be used to indicate when an alteration should be attempted could include the magnitude of the gradient; the resultant difference in "dead" and "alive" coefficients or the frequency of previous alterations. The data from the proposed analyse of "dead" and "alive" amplitudes could also be used to make the ZSs added in a cloning step more useful to the basis set. A further, widely used way to improve the rate of convergence uses the second derivative to direct a gradient descent step. The second derivative is a measure of how the gradient will vary as the input is varied. This information can be used so each gradient descent step can be optimally sized [144].

However, new Hamiltonian matrix elements will still have to be calculated carrying significant computational cost. Therefore, finding a way to predict Hamiltonian

matrix elements without direct calculation could offer even greater speed ups. It should be possible to train an artificially intelligent algorithm to make Hamiltonian matrix element predictions. When the GD process attempts to change a ZS amplitude, the current Hamiltonian matrix is known as well as all the amplitudes used in its calculation. These amplitudes are all the same except for a single orbital. Each Hamiltonian matrix element is calculated using the same set of creation and annihilation operations followed by an overlap calculation. The parts of which will only differ for the altered orbital. Thus, the algorithm can be trained to predict Hamiltonian values based on the value of the Hamiltonian element; the original and altered Zombie amplitudes and the placement of the altered orbital. Moreover, comparing the Hamiltonian calculations between systems, the key difference is just the one- and two-electron integrals, the effect of which can be integrated into the training procedure allowing the algorithm to be applied universally.

Chapter 5
Excited States

5.1 Introduction

The Zombie states method is being developed with the aim of using it in non-adiabatic molecular dynamics simulations. Thus, having the capability to compute low-lying excited states would be beneficial as they are often an import part of a reaction mechanism. Excited states can be found by diagonalising the Hamiltonian of a complete, full-CI, wave function as they are simply the higher energy eigenvalues. This can, of course, be computationally expensive and necessitates using alternative methods. However, most electronic structure methods are designed to find ground states with the excited electronic states found using the ground state as a baseline. The equations-of-motion coupled-cluster (EOM-CC) method uses a coupled cluster reference state operator on a linear excitation operator of the form,

$$\mathscr{R}_n = \frac{1}{n!^2} \sum_{ijk...}^{abc...} a_a^\dagger a_i a_b^\dagger a_j a_c^\dagger a_k \tag{5.1}$$

where a, b, c are unoccupied orbitals and i, j, k are occupied producing excited configurations in reference to the the ground state [145, 146]. This produces an effective Hamiltonian that is formerly exact although in practise reasonable truncations to $\mathscr{R}_n$ are made [147]. An alternative extension to the coupled cluster method is coupled-cluster linear response (CCLR) which introduces a time-dependent perturbation in the form of the linear-response function. This is produced using a time-independent operator, which has a time-dependent expectation value induced by an external field. The excited states occur at the poles of the CCLR function and are the eigenvalues of the CCLR matrix [148–151]. CCLR has been used to calculated excitation energies and dipole transitions [151–153]. Rather than using a single reference state a multi-configurational self-consistent field (MCSCF) approach can be taken, the most well known of which is CASSCF [154–157]. Multi-reference CI (MRCI) can then produce a set of excited determinants generated from the MCSCF space [158–160].

Further, using the FCIQMC method it was shown that, with minimal extra

computational cost, a Gram-Schmidt orthogonalisation (GSO) procedure could be used to find multiple low-energy excited states [24]. The orthogonality of the Slater determinants is exploited by using a different wave function to describe the ground state and each excited state. The regular FCIQMC algorithm is applied to each wave function to find and weight the electron configurations drawn from the complete Fock space. To find an excited state GSO is used between wave functions which has the effect of removing ground state contributions from the excited state wave function. Since configurations for the ground state are removed the excited state becomes the lowest available state that imaginary time propagation will evolve the wave function to. The orthogonalisation step is implemented once the FCIQMC random walk step has completed for each of the wave functions allowing them to be treated completely separately until orthogonalisation which is the reason for the low additional computational cost [24]. Like FCIQMC the Zombie states method has access to every possible configuration and with a shared use of imaginary time propagation it was logical to also implement GSO in the Zombie states method.

5.2 Theory

In general, Gram-Schmidt orthogonalisation, for any set of vectors, is defined

$$\mathbf{u}_n = \mathbf{v}_n - \sum_{m=1}^{n-1} \mathrm{proj}_{\mathbf{u}_m}(\mathbf{v}_n) \tag{5.2}$$

with the projection operator

$$\mathrm{proj}_u(\mathbf{v}) = \frac{\langle \mathbf{u}, \mathbf{v} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle} \mathbf{u}. \tag{5.3}$$

A set of vectors can then be orthogonalised by first setting $\mathbf{u}_1 = \mathbf{v}_1$ and then following the above process for each vector in the set. Gram-Schmidt orthogonalisation can then be applied to two wave functions denoted $|\Psi^{(m)\perp}\rangle$ and $|\Psi^{(n)}\rangle$; the $\perp$ symbol is used to show orthogonalisation. Thus, a projection operator can be defined,

$$\mathrm{proj}_{|\Psi^{(m)\perp}\rangle}(|\Psi^{(n)}\rangle) = \frac{\langle \Psi^{(m)\perp} | \Psi^{(n)} \rangle}{\langle \Psi^{(m)\perp} | \Psi^{(m)\perp} \rangle} |\Psi^{(m)\perp}\rangle. \tag{5.4}$$

Therefore an orthogonalised wave function can be found by

$$|\Psi^{(n)\perp}\rangle = |\Psi^{(n)}\rangle - \sum_{m=1}^{n-1} \frac{\langle \Psi^{(m)\perp} | \Psi^{(n)} \rangle}{\langle \Psi^{(m)\perp} | \Psi^{(m)\perp} \rangle} |\Psi^{(m)\perp}\rangle. \tag{5.5}$$

In FCIQMC a basis of Slater determinants is used which means that the orthogonalisation step is simple, removing components of the lower energy states from

the higher states [24]. A Zombie state wave function however is a superposition of all possible configurations and so the same set of Zombie states is used but with a different Zombie coefficient vector which means Eq. (5.5) can be rewritten as

$$|\Psi^{(n)\perp}\rangle = |\Psi^{(n)}\rangle - \sum_{m=1}^{n-1} \frac{\mathbf{d}^{*(m)\perp}\Omega\mathbf{d}^{(n)}}{\mathbf{d}^{*(m)\perp}\Omega\mathbf{d}^{(m)\perp}}|\Psi^{(m)\perp}\rangle \tag{5.6}$$

where Ω is the overlap matrix between Zombie states, evaluated using Eq. (2.223). Further since $|\boldsymbol{\zeta}^{(m,a)}\rangle = |\boldsymbol{\zeta}^{(n,a)}\rangle \ \forall a$ the orthogonalisation process can be further simplified to

$$\mathbf{d}^{(n)\perp} = \mathbf{d}^{(n)} - \sum_{m=1}^{n-1} \frac{\mathbf{d}^{*(m)\perp}\Omega\mathbf{d}^{(n)}}{\mathbf{d}^{*(m)\perp}\Omega\mathbf{d}^{(m)\perp}}\mathbf{d}^{(m)\perp}. \tag{5.7}$$

The N_{es} lowest electronic states are found using the same number of ZS coefficient vectors are required to be orthogonalised and then propagated in imaginary time as in Eq. (3.19)

$$\dot{\mathbf{d}}^{(n)\perp} = -\Omega^{-1}\mathbf{H}\mathbf{d}^{(n)\perp}. \tag{5.8}$$

This gives a single time step of,

$$\begin{aligned}\mathbf{d}^{(n)\perp}(\beta+\Delta\beta) = \mathbf{d}^{(n)\perp}(\beta) + \dot{\mathbf{d}}^{(n)\perp}(\beta)\Delta\beta \\ - \sum_{m=1}^{n-1} \frac{\mathbf{d}^{(m)\perp}(\beta+\Delta\beta)\cdot\left(\mathbf{d}^{(n)\perp}(\beta) + \dot{\mathbf{d}}^{(n)\perp}(\beta)\Delta\beta\right)}{\mathbf{d}^{(m)\perp}(\beta+\Delta\beta)\cdot\mathbf{d}^{(m)\perp}(\beta+\Delta\beta)}\mathbf{d}^{(m)\perp}(\beta+\Delta\beta).\end{aligned} \tag{5.9}$$

Which gives a final repeating procedure of orthogonalisation, normalisation followed by propagation.

5.3 Results

Using the truncated Li_2 system, that has been used as a proof of concept throughout this work, Gram-Schmidt orthogonalisation is shown to be capable of recovering excited states when a complete basis of random Zombie states is used. In Fig. 5.1 the ground state and first three excited states are exact when compared to the eigenstate energies found by diagonalising the Hamiltonian matrix. A much longer imaginary time propagation is needed to evolve all of the states to the correct energies. In a similar result to the complete random basis, the biased basis of 64 Zombie states (described in Table. 4.1) also evolves to the ground state and excited states.

Reducing the size of the basis set so it is no longer complete as in Fig. 5.3 renders the wave function unable to describe the ground state or any of the excited states.

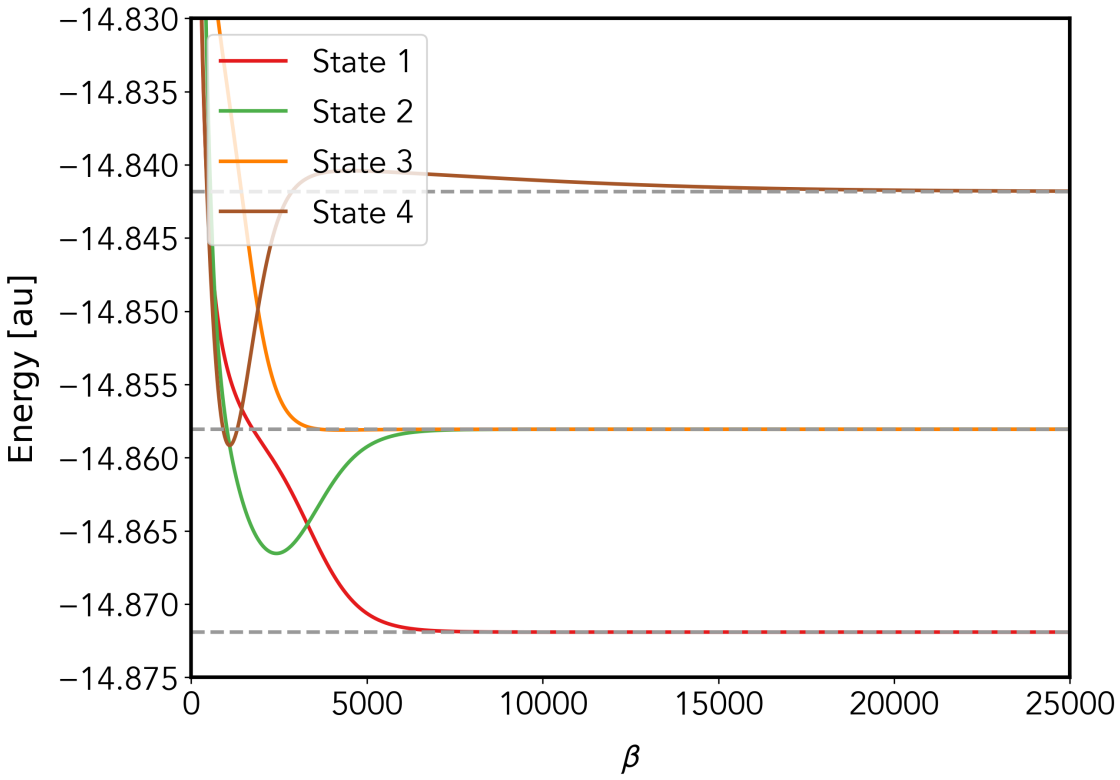


Figure 5.1: Imaginary time propagation with GSO for a complete random basis for truncated Li_2. The final energies correspond to the ground, degenerate anion and first excited states with comparison to the eigenvalues of the complete Slater determinant basis.

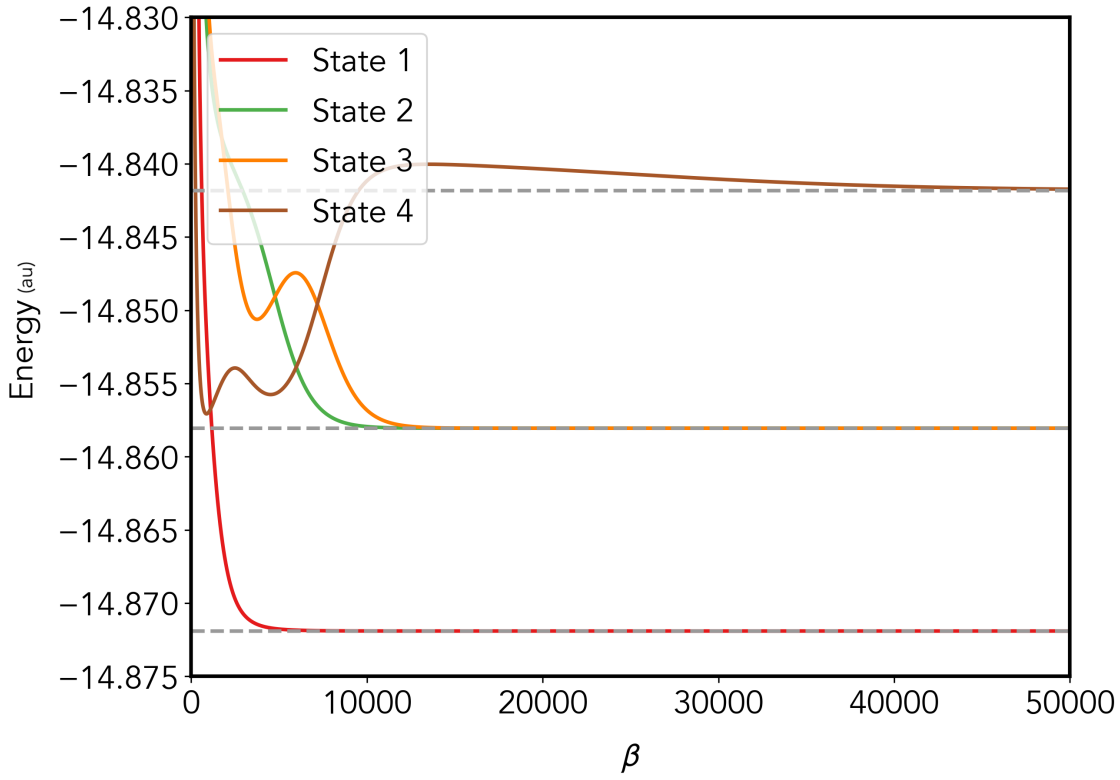


Figure 5.2: Imaginary time propagation using GSO with a biased basis of 64 ZSs for Li_2. The biasing regime is described in Table. 4.1. The exact energies for the ground, degenerate anion and first excited state are shown as dashed lines for comparison.

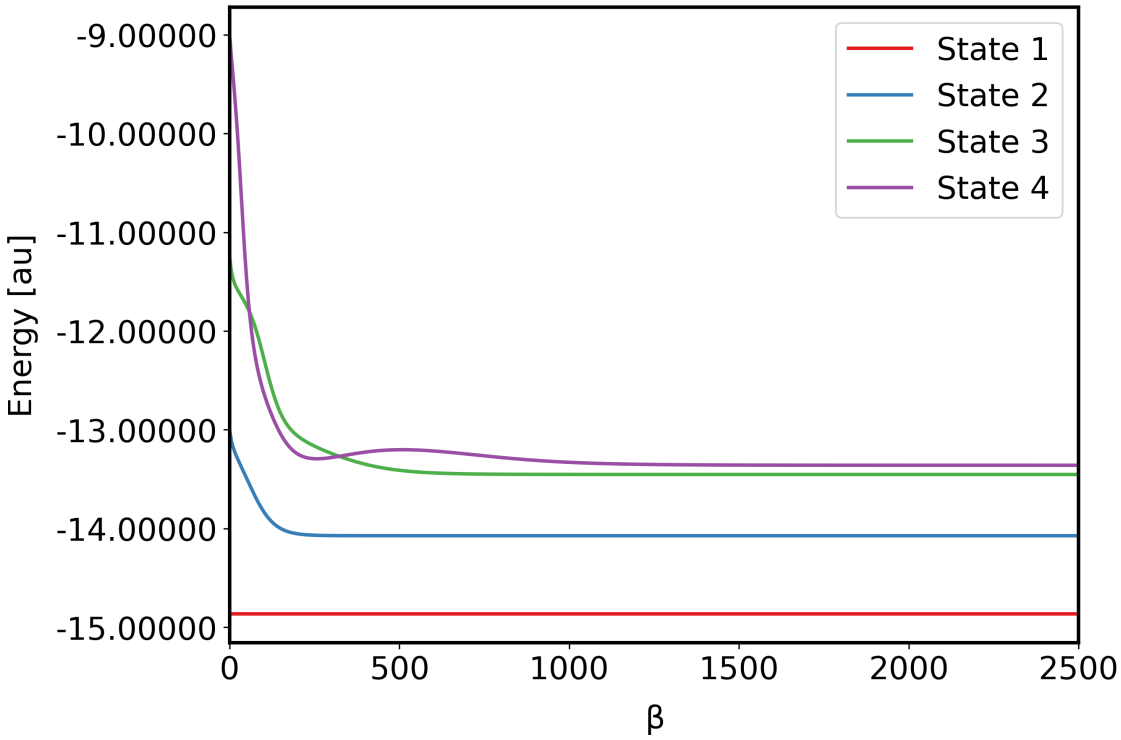


Figure 5.3: Imaginary time propagation with GSO for a basis of 200 random ZSs for the truncated Li_2 system. The final energies of each state correspond, in order, to the neutral ground, degenerate anion and first neutral excited states.

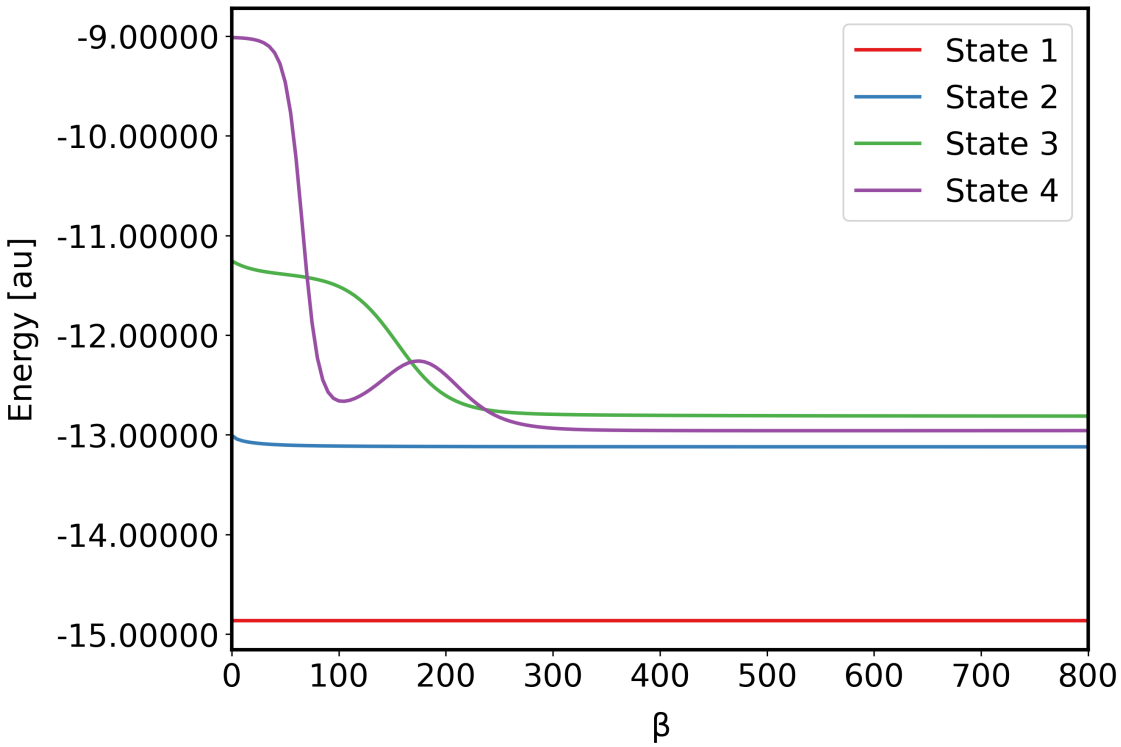


Figure 5.4: Imaginary time propagation with GSO for the optimised wave function from Fig. 4.10. The final energies of each state correspond, in order, to the neutral ground, degenerate anion and first neutral excited states.

Similarly, in Fig. 5.4 when GSO is applied to the final wave function containing 30 Zombie states optimised using gradient descent to find the ground state energy. The optimised wave function is propagated in imaginary time with GSO once the gradient descent process is complete which shows the correct ground state energy but incorrect energies for the excited states. The eigenvalue energies are not shown in either Fig. 5.3 or Fig. 5.4 as the energies for the excited states are too inaccurate to warrant comparison.

5.4 Gradient Descent with Gram-Schmidt Orthogonalisation

As can be seen in Fig. 5.4 the GD process has optimised "dead" and "alive" amplitudes to describe the ground state which gives a poor description of the excited states, similar to the basis set of 200 random ZS as in Fig. 5.3. However, it is also possible to implement GSO during the gradient descent process so the wave function is optimise to describe an excited state. Like in FCIQMC a separate wave function is required for each state each requiring a separate set of Zombie coefficient vectors for the state of interest and all states below it. For example, the second excited state would require three coefficient vectors one for the state of interest and then a further two for the ground and first excited state. So for an excited state, arbitrarily labelled n, the gradient descent process then proceeds as normal with the addition of GSO using the time step outline in Eq. (5.9). Gradients are then calculated using $\mathbf{d}^{(n)\perp}$ and a gradient descent step attempted. This is the only modification required to the standard GD process.

Table 5.1: Table detailing the initial and final energies for the first four states of Li_2 in the $6-31G^{**}$ basis. The gradient descent with Gram-Schmidt orthogonalisation process was used to optimise each wavefunction[1].

State	Initial/finalN_{bf}	Energy $[au]$			Epoch
		Initial	Final	full-CI	
1	30/30	−14.863582	−14.871914	−14.871914	10334
2	2/12	−13.005780	−14.858062	−14.858062	2000
3	3/23	−11.254147	−14.858062	−14.858062	34700
4	4/20	−14.605008	−14.841836	−14.841836	5754

[1]Each wave function was initialised randomly and the first state set as the RHF determinant. State 1 is the ground state and is the result from Fig. 4.10. For states 2 and 3 GSO was used with the gradient descent process and the cloning method was used to grow the number of basis functions. An additional ZS was added to the basis sets after, at most, 280 epochs. This was reduced to 99 epochs for State 4 with 1 ZSs being added and all amplitudes initialised using the biasing regime from Table. 4.2. States 2 and 3 are the degenerate 7 electron anion state and state 4 is the first neutral excited state. The number of epochs required for the energy to converge to the full-CI energy, found by diagonalising the complete basis Hamiltonian.

Using the truncated Li_2 system, in the $6-31G^{**}$ basis[1], gradient descent is completed with Gram-Schmidt orthogonalisation for the first three excited states. Each basis set was initialised randomly and the cloning method was used to increase the number of ZSs adding an extra function after no more than 280 epochs for states 2 and 3. For state 4 a single ZS was added after no more than 99 epochs with the biasing regime from Table. 4.2 used to set the initial amplitudes. The gradient descent process used a maximum learning rate of $\gamma = 2500$ being reduced on each epoch to a minimum of $\gamma = 0.16$, before the maximum number of basis functions was reach, then decreasing further to $\gamma = 5.12 \times 10^{-5}$. The gradient descent process with GSO to find the first excited state is shown in Fig. 5.5. The final energies for each wave function are given in Table. 5.1 along with the number of epochs, required for the energy to converge. The initial and final basis set sizes and the full-CI energy are also given for comparison. The different epoch lengths are exemplify the importance of available Zombie states and learning rates. For the third state, at around 30,000 epochs, the learning rate was allowed to reduce further to $\gamma = 4.10 \times 10^{-7}$. At 34400 epochs the maximum basis set size was extended from 20 to 23 ZSs, which were added using cloning. Whereas, for the fourth state the gradient descent process was set to allow the learning rate to reach $\gamma = 4.10 \times 10^{-7}$ and was initialised with 20 ZSs.

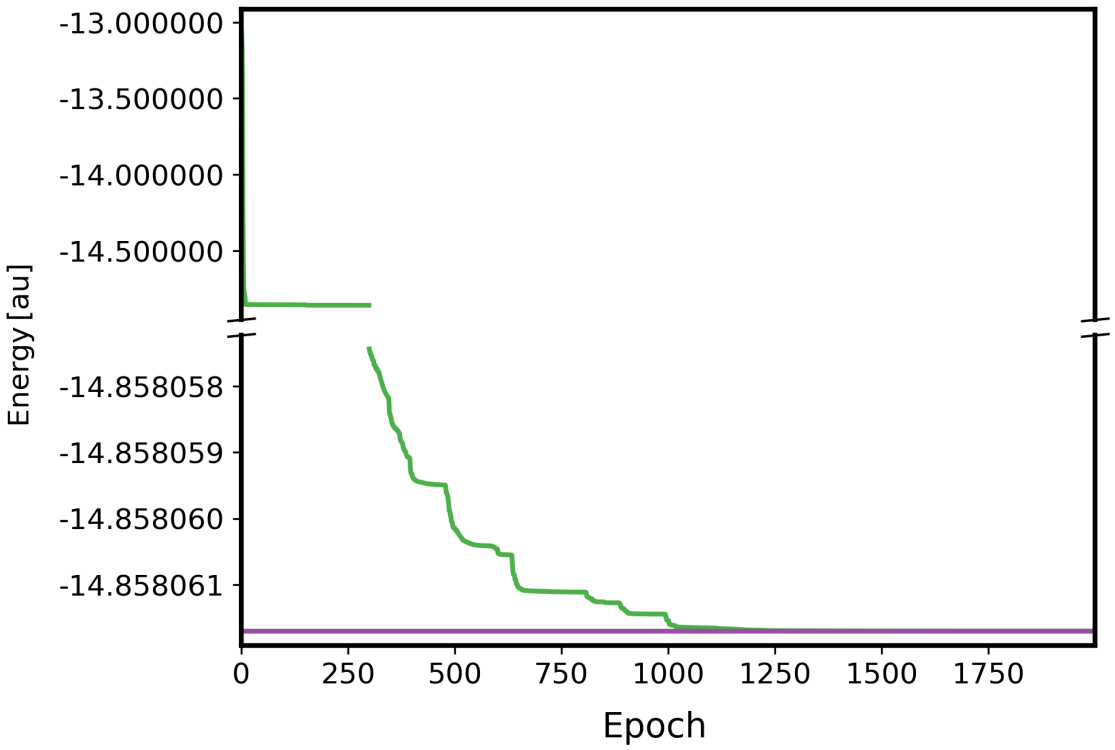


Figure 5.5: Plot of energy during the gradient descent process to find the second excited state truncated Li_2 in the 6-31G** basis. 30 randomly generate ZSs were optimised using gradients calculated following imaginary time propagation using GSO. The energy of the second excited state is shown as the solid (purple) line.

[1]The absence of hydrogen means the $6-31G^{*}$ basis could have been used.

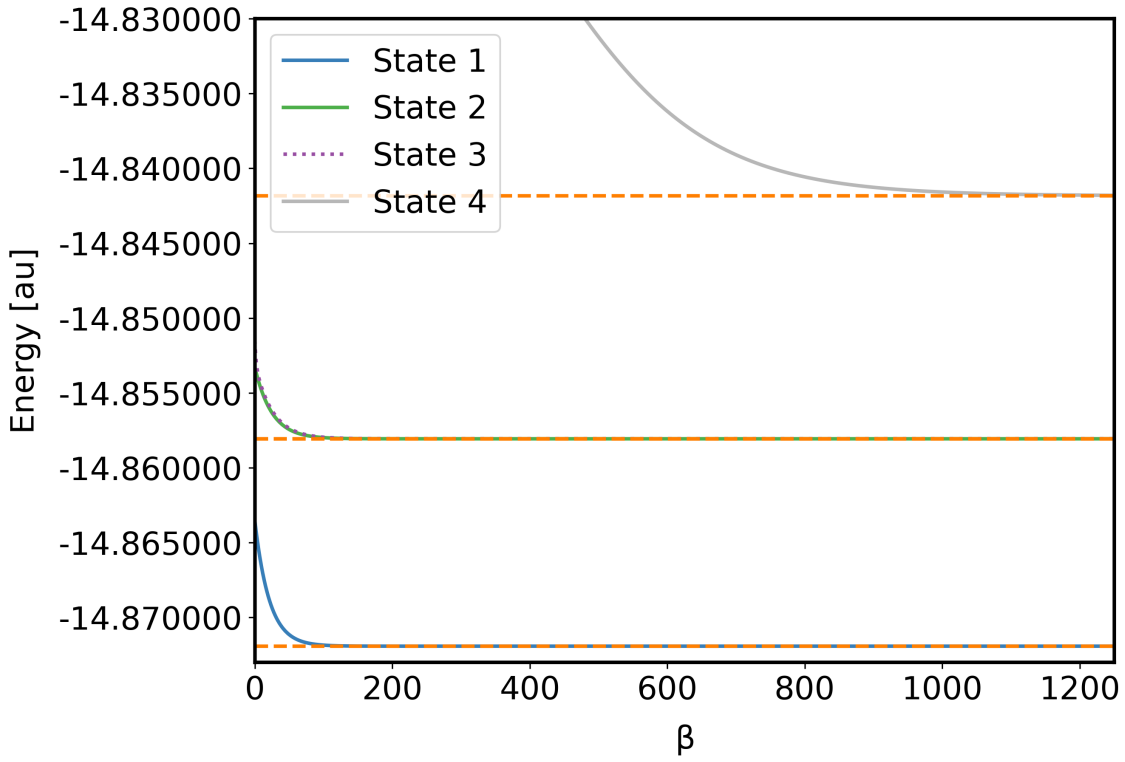


Figure 5.6: Plot of the imaginary time propagation for four separate wave functions optimised using the gradient descent process to describe a specific state for the truncated Li_2. Each wave function is optimised separately. The final energies of each state correspond accordingly: State 1 is the neutral ground state; states 2 and 3 are the anion ground state and state 4 is the first excited state. The exact energies are shown for comparison as dashed lines, found by diagonalising the Slater determinant Hamiltonian.

The final imaginary time propagations for each wave function and the ground state from Fig. 4.10 are all plotted in Fig. 5.6. The final energies for each state are equivalent to the full-CI energies found by diagonalising the Slater determinant Hamiltonian.

5.5 Conclusions

The Zombie states method is capable of finding excited states using a GSO process using a complete basis set. Excited states are orthogonal to each other so by orthogonalising the set of ZS coefficients during imaginary time propagation forces the coefficients to evolve to the the next lowest available state. Unsurprisingly, an incomplete random basis does not evolve to the correct ground state excited state energies. Moreover, when applying the GSO imaginary time propagation process to an incomplete basis that has been optimised to describe the ground state the basis is also not capable of describing the excited states. The gradient descent process optimises the "dead" and "alive" coefficients of each Zombie state to find the ground state energy. This improves the cancellation of contributions from configurations

not in the ground state and stops the wave function from being able to effectively describe the higher energy states.

Significantly, it is possible to find excited states with basis sets that are not of a complete size if GSO is used during the gradient descent optimisation process. By using separate wave functions and the appropriate orthogonalised ZS coefficient vector it is demonstrate in Fig. 5.5 that the gradient descent process can optimise ZS amplitudes so the wave function describes an excited state. The plot of imaginary time, Fig. 5.6, for the first four states of Li_2 requires four separate wave functions containing a total of 85 basis functions. This is significantly fewer than the complete basis set of 1024 functions and is also smaller than the total number of possible configurations containing six and seven electrons, $210+120=330$ (the seven electron configurations are included as the first two excited states of Li_2 are its anion).

As previously discussed most methods for finding excited states require the use of a reference state which is usually the ground state to be calculated first. Even in the FCIQMC utilisation of GSO, the separate wave functions have to be brought together for the orthogonalisation process. This means that a time step for the ground step must be complete before orthogonalisation of the excited state wave function can occur [24]. However, there is no requirement for a resolved reference state or wave function when using the Zombie excited state gradient descent process. Each wave function, describing a different state, can be treated separately at all times. Gradient descent minimises a function to find a local minima and imaginary time propagation finds the lowest state. The addition of Gram-Schmidt orthogonalisation redirects the gradient descent process away from the global minimum to another local minima – an excited state. In FCIQMC wave functions are constructed by finding the best set of configurations drawn from the complete Slater determinant basis; GSO then removes contributions from lower states. This overall creates a process of adding configurations and then removing others from the wave function. For a Zombie wave function all possible configurations are, by construction, present from initialisation with Gram-Schmidt orthogonalisation directing the gradient descent process to optimise the wave function to the excited state rather than adding and removing configurations. The complete separation of the wave functions during the optimisation process means it is possible to find the energy of any excited state without having to consider the construction of any states below it, beyond the inclusion of the additional coefficient vectors. This has great potential for using Zombie states in simulations requiring multiple electronic states due to the possible time and computational cost savings. Firstly, only states necessary to the simulation need to be calculated, so if only a set of excited states are required there is no requirement to find the ground state to reference from. These states can also be calculated concurrently with no need to share information between them; with the availability of multi-processor clusters this has the potential to reduce overall

computation times.

The process is still in its infancy and needs to applied to a variety of systems. There is large variability in the number of epochs, shown in Table. 5.1, required to converge the energies for each state. The third excited state took the greatest number of epochs to converge due to both an insufficient number of ZSs and the minimum learning rate initially being set too high. Whereas, the availability of lower learning rates and the use of the biasing regime meant state 4 require fewer epochs to be resolved. As discussed in the previous chapter there are multiple places the gradient descent process can be improved to aid faster and less expensive convergence. Some of these proposed improvements, such as the faster calculation of matrix elements, will be universally beneficial. But it will also be necessary to tailor certain parameters to finding excited states. The learning rate cycle, initial basis set size and timing of cloning events all need investigation specifically for excited states. Further, for the small lithium dimer the random initialisation was mainly used when finding both the ground and excited states with gradient descent. However, when verifying the GD method for larger systems the biasing regime was required to find ground state energies and even biasing the ZSs to the ground state helped resolve the fourth state of Li_2 faster than with random initialisation. Therefore, it is highly likely that a specifically designed biasing regime will be required when verifying the GD with GSO method for the excited states of these larger systems. Nonetheless, even as just a proof of concept, the addition of Gram-Schmidt orthogonalisation to the Zombie state gradient descent process is an important expansion to the method. The ZS method is capable of recovering both the ground and excited states which is integral to simulations of non-adiabatic dynamics.

Chapter 6
Conclusions and Outlook

The work presented in thesis advances the Zombie states (ZS) method from a theoretically interesting idea to a viable method with various avenues of development. Within the context of Pople's framework, this means improvements to the formulation and implementation of the method, which has been verified by comparison to a number of chemical systems. It has been shown that the ZSs method can recover ground state energies, with accuracy comparable to using full-CI basis. The addition of gradient descent to optimise ZS amplitudes has made it possible to use considerably smaller basis sets while maintaining high accuracy. This is evidenced using a range of chemical systems containing considerably more spin-orbitals than the example systems used as an initial proof of concept [9]. However, despite algorithmic improvements and the implementation of imaginary time propagation to find ground state energies, computation times are still a limiting factor in need of improvement. Although, the potential applications of the method have been significantly increased by the addition of Gram-Schmidt orthogonalisation to find excited states. This simple addition allows the ZSs to describe excited states with minimal computational cost and without the need for any sort of reference state.

In Chapter 2 the Zombie state method is formulated in three different but equivalent ways. Originally, a multi-particle Slater determinant is made up of one-electron Zombie states in a manner that is similar to the standard Slater determinants used in HF theory [9]. It is also demonstrated that Zombie states can be built using a Zombie operator that acts on a vacuum state analogous to the method used in the second quantization approach [23]. The general coherent state definition, outlined by Perelomov and Gilmore, allows the coherent states used in CCS and MCE to be derived using the same method – all starting from a reference, vacuum state [21, 22]. This definition can also be applied to the construction of Zombie states from which their key properties are naturally defined. The Zombie displacement operator construction ensures the correct normalisation of "dead" and "alive" coefficients and gives rise to the sign change rule when undergoing creation or annihilation operations by construction. Further, by defining Zombie states in this manner, it is easy to make comparison between them and standard fermionic coherent state

constructions. Both the Lie group and Grassmann algebra fermionic coherent state constructions take a specific configuration of electrons and describe it as superposition of its spin states, which makes them fixed eigenstates of the number operator [18, 19]. On the other hand, Zombie state orbitals are superpositions of the occupied and unoccupied states and so contain all configurations of electrons. Hence, ZSs are generally not eigenstates of the number operator, which allows them to describe the multiple configurations necessary to give an accurate description of a state but also allows configurations with incorrect numbers of electrons. This gives ZSs the potential to return full-CI energies using small basis sets but also necessitates the work to ensure incorrect configurations cancel each other out.

The implementation of imaginary time propagation, outlined in Chapter 3, has facilitated the other developments detailed in this thesis. Being able to find ground state energies in a practical amount of time, rather than needing a long time propagation, is in isolation progress. However, without this improved practicality, it would not have been possible to implement gradient descent. The method relies on being able to recalculate and check the ground state energy to ensure convergence; without imaginary time propagation this process would not be computationally practical. Further, its addition has led to the adoption of Gram-Schmidt orthogonalisation to find excited states which again would not have been possible without the ease with which energies can now be found. The work in Chapter 4 has demonstrated, for the first time, the full potential of the Zombie states method to describe chemical systems with high accuracy while using significantly contracted basis sets. Gradient descent minimises the ground state energy by optimising the ZS amplitudes. It was demonstrated that Zombie states achieve ground state energies comparable to the full-CI results but with much smaller basis sets. Although, the larger systems are currently limited by the slow convergence of the method. The extension of the method to include excited states, while using a truncated basis set, significantly broadens its applications. Further, the method does not require the excited states to be found using a reference state. This potentially offers large computational savings for simulations requiring various electronic states as each state can be resolved completely independently.

The addition of gradient descent to find ground and excited states has the potential to be a computationally inexpensive tool for describing electronic systems. However, in its current form the process converges very slowly when compared to the heavily optimised code used for full-CI calculations in PyScf. Thus, systematic work is required to find computational savings at all points in the Zombie state program. There are relatively simple improvements that can be made to the code base to ensure all subroutines and functions are as efficient as possible. Significant improvements to the time needed per epoch can be made by offloading the repetitive Hamiltonian recalculation to a GPU, which should provide a significant reduction

to the total gradient descent time. With better initial sampling fewer optimisation steps should be required offering the largest computational saving. The starting point of this process should be an analysis of the effect of the value of "dead" and "alive" amplitudes and the combination of values both within a single Zombie state and a basis set as a whole. This should also influence the cloning technique which currently adds a Zombie state using the same initialisation method, not taking into account the existing Zombie state values or how they have been optimised so far. Similarly, the gradient descent algorithm can be profiled with the aim of making it "smarter" and so more efficient. The current algorithm attempts to alter all Zombie amplitudes with equal frequency with no ability to consider the likelihood of an alteration being successful. Finding relationships between existing or very easily calculated data points and a successful alteration should allow the algorithm to be formulated to be less computationally wasteful. Some possible data points that should be analysed include the success of previous alterations, the magnitude of the gradient or the resultant difference in ZS amplitudes.

However, finding the energy levels of a system is not the envisaged final use case of the Zombie states method. Existing electronic structure methods can be used to ascertain various properties of a system. To bring the ZSs method in-line with well-established methods it needs to be capable of producing a range of data points about a chemical system. A simple addition would be producing the average orbital populations which is possible with CI methods. Further, adding the ability to to calculate the Hessian matrix would allow vibrational frequencies of the molecule to be found. This would also facilitate the optimisation of the molecular geometry. Moreover, CCS was developed for the simulation of the quantum dynamics of systems with many degrees of freedom and so the Zombie states method, as the fermionic analogue to CCS, is ultimately aimed to simulate electron dynamics [32]. The CCS family of methods has often been benchmarked against Multiconfigurational Time-Dependent Hartree (MCTDH) method while incurring lower computational cost [13, 17]. The Multiconfigurational Time-Dependent Hartree-Fock (MCTDH-F) extends MCTDH to treat many-body fermionic systems which allows a unified method to describe both the nuclear and electronic wave packets in a quantum dynamical manner [139, 161–168]. Currently, the CCS family of methods use the time-dependent wave function,

$$|\Psi\rangle = \sum_{a}^{N_{bf}} c_a |\psi^{(a)}\rangle \tag{6.1}$$

$$|\psi^{(a)}\rangle = |\chi^{(a)}\rangle |\boldsymbol{z}^{(a)}\rangle \tag{6.2}$$

where N_{bf} is the number of electronic states ($N = 1$ for CCS and $N > 1$ for MCE); c_a is the basis function coupling; $|\boldsymbol{z}^{(a)}\rangle$ are the coupled coherent states and $|\chi^{(a)}\rangle$ is an electronic state. Using an adiabatic basis, electronic structure calculations

are used to calculate the potential energy surface which then also requires the calculation of the non-adiabatic coupling matrix. Zombie states would replace $|\chi^{(a)}\rangle = \sum_a^{N_{zs}} d_a |\boldsymbol{\zeta}^{(a)}\rangle$, N_{zs} being the number of Zombie states needed to describe a single electronic state. So, the wave function with a single electronic state would have the form,

$$|\Psi\rangle = \sum_a^{N_z s} d_a |\boldsymbol{\zeta}^{(a)}\rangle |\boldsymbol{z}^{(a)}\rangle. \tag{6.3}$$

The quantum dynamics can then be found by using the propagation equations for CCS and Zombie states. As shown, the Zombie states can be constructed using the one- and two-electron integrals from computationally inexpensive Hartree-Fock calculations. The gradient descent method has the potential to reduce the number of Zombie states required to describe an electronic state making the proposed dynamic simulations possible. But this can only be achieved if the computational cost of the gradient descent process is significantly reduced.

Appendix A
Mathematical Concepts for Coherent States

In this Appendix some key mathematical concepts used to define general coherent states and the Grassmann algebra needed for fermionic coherent states are briefly discussed. The reasoning given here assumes an appreciation of some mathematical concepts and notation more akin to a background in physical chemistry rather than rigorous mathematics. Thus, the exterior product is first defined and then used to construct algebra over a field; Lie groups and stability subgroups.

A.1 Groups and Fields

Firstly, a group G is defined as a non-empty set that has a binary operation, the group operator, $\cdot$, defined for all elements of G with the following properties

$$\forall x, y \in G,\ x \cdot y \in G \tag{A.1a}$$
$$\forall x, y, z \in G,\ (x \cdot y) \cdot z = x \cdot (y \cdot z) \tag{A.1b}$$
$$\exists 1 : 1 \cdot x = x \cdot 1 = x, \forall x \in G \tag{A.1c}$$
$$\forall x \in G\, \exists\, x^{-1} : x x^{-1} = x^{-1} x = 1. \tag{A.1d}$$

A field, F, is an extends a group by the addition of a second binary operator, $+$, which has the following properties 1

$$\forall x, y \in F,\ x + y = z \in F \tag{A.2a}$$
$$\forall x, y \in F,\ x + y = y + x \tag{A.2b}$$
$$\forall x, y, z \in F,\ (x + y) + z = x + (y + z) \tag{A.2c}$$
$$\exists 0 : 0 + x = x + 0 = x,\ \forall x \in F \tag{A.2d}$$
$$\forall x \in F\, \exists - x : x + (-x) = -x + x = 0 \tag{A.2e}$$
$$\forall x, y, z \in F,\ x \cdot (y + z) = (x \cdot y) + (x \cdot z). \tag{A.2f}$$

A.2 The Exterior Product

First the exterior product of two vectors gives a bivector which has the fundamental anti-symmetric quality

$$\hat{\mathbf{x}} \wedge \hat{\mathbf{y}} = -\hat{\mathbf{y}} \wedge \hat{\mathbf{x}} \tag{A.3}$$

It is easy to see how then taking the exterior product of a vector with itself is zero

$$\hat{\mathbf{x}} \wedge \hat{\mathbf{x}} = 0 \tag{A.4}$$

This can be visualised by taking two vectors $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ in three dimensions, that are defined using the basis e_1, e_2 and e_3,

$$\begin{aligned} \hat{\mathbf{x}} &= a_1 e_1 + a_2 e_2 + a_3 e_3 \\ \hat{\mathbf{y}} &= b_1 e_1 + b_2 e_2 + b_3 e_3 \end{aligned} \tag{A.5}$$

The exterior product of $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$,

$$\begin{aligned} \hat{\mathbf{x}} \wedge \hat{\mathbf{y}} &= (a_1 e_1 + a_2 e_2 + a_3 e_3) \wedge (b_1 e_1 + b_2 e_2 + b_3 e_3) \\ &= (a_1 b_1) e_1 \wedge e_1 + (a_1 b_2) e_1 \wedge e_2 + (a_1 b_3) e_1 \wedge e_3 \\ &+ (a_2 b_1) e_2 \wedge e_1 + (a_2 b_2) e_2 \wedge e_2 + (a_2 b_3) e_2 \wedge e_3 \\ &+ (a_3 b_1) e_3 \wedge e_1 + (a_3 b_2) e_3 \wedge e_2 + (a_3 b_3) e_3 \wedge e_3 \end{aligned} \tag{A.6}$$

This can be simplified using the fact $e_i \wedge e_i = 0$ and the antisymmetric characteristic to give,

$$\hat{\mathbf{x}} \wedge \hat{\mathbf{y}} = (a_2 b_3 - a_3 b_2) e_2 \wedge e_3 + (a_3 b_1 - a_1 b_3) e_3 \wedge e_1 + (a_1 b_2 - a_2 b_1) e_1 \wedge e_2. \tag{A.7}$$

This bivector can be visualised as the area created by the exterior product of the two vectors which is shown in two dimensions in Fig. A.1.

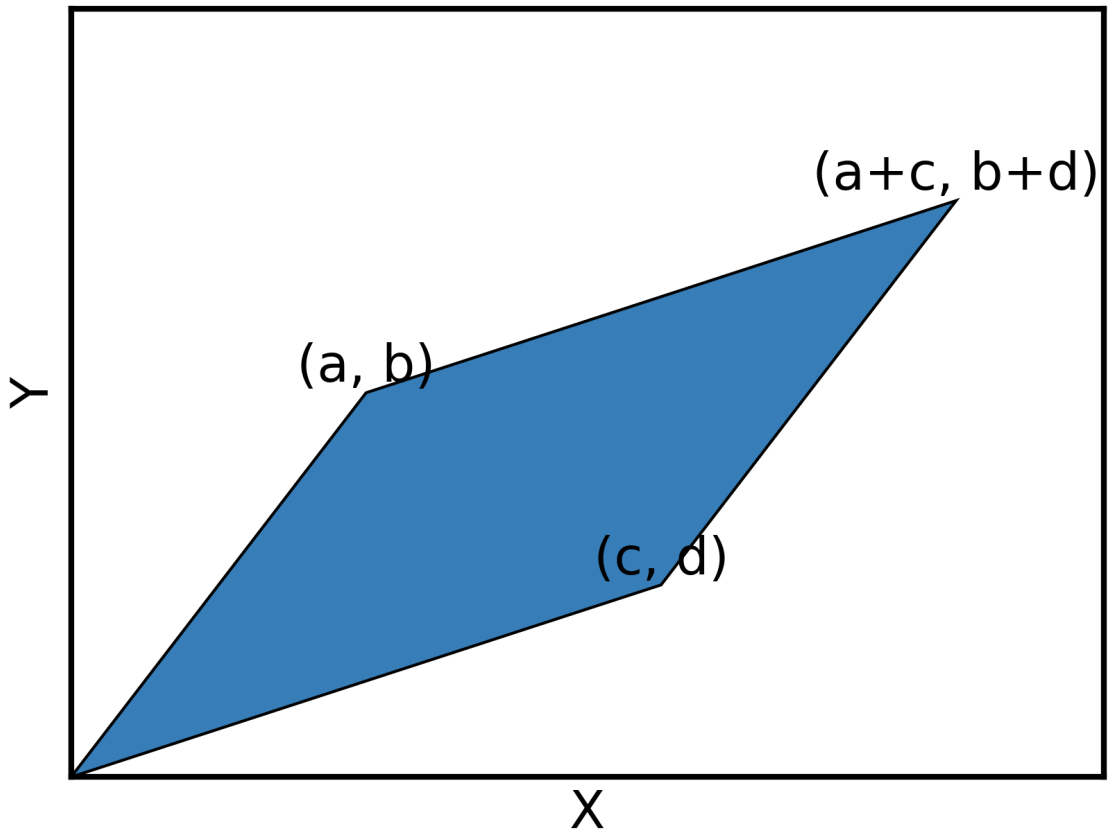


Figure A.1: Plot showing the exterior product between two vectors $\hat{\mathbf{x}} = ae_1 + be_2$ and $\hat{\mathbf{y}} = ce_1 + de_2$. The exterior product $\hat{\mathbf{x}} \wedge \hat{\mathbf{y}} = (ad - bc)e_1 \wedge e_2$ which is the area of the parallelogram formed by the two vectors.

A.3 Algebra Over A Field

The concept of algebra over a field will be formalised using real vectors in 3-dimensions as the motivating example. As previously seen, the vectors e_1, e_2 and e_3 form a basis for 3-dimensional Euclidean space and take the form $(1,0,0), (0,1,0)$ and $(0,0,1)$ respectively. With these three vectors it is possible to generate any other vector of the form $\hat{\mathbf{x}} = (x_1, x_2, x_3)$ using some linear combination of e_1, e_2 and e_3, $\hat{\mathbf{x}} = x_1e_1 + x_2e_2 + x_3e_3$. This is perhaps obvious but in more mathematically rigorous terms these vectors are the generating set for the vector space over the field of real numbers $F = \mathbb{R}$ in $N = 3$ dimensions, $\mathbb{R}^3$. Further, as the vectors are all linearly independent the set $B = \{e_1, e_2, e_3\}$ can also be described as a basis set. Two arbitrary vectors can be defined, $\hat{\mathbf{x}} = (a_1e_1 + a_2e_2 + a_3e_3)$ and $\hat{\mathbf{y}} = (b_1e_1 + b_2e_2 + b_3e_3)$ which both belong to the vector space $\mathbb{R}^3$. Each vector will be placed in its own vector space so $\hat{\mathbf{x}} \in X$ and $\hat{\mathbf{y}} \in Y$. The cross product of these two vectors can be

found using the following rules as

$$\begin{aligned} e_1 \times e_2 &= e_3 \\ e_2 \times e_3 &= e_1 \\ e_3 \times e_1 &= e_2 \end{aligned} \tag{A.8}$$

$$\hat{\mathbf{x}} \times \hat{\mathbf{y}} = (a_2 b_3 - a_3 b_2) e_1 + (a_3 b_1 - a_1 b_3) e_2 + (a_1 b_2 - a_2 b_1) e_3 = \hat{\mathbf{z}} \tag{A.9}$$

Notice that the coefficients are the same as in the exterior product, Eq. (A.7). So, the cross product forms a vector perpendicular to $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ and the exterior product a bivector with an area of the same magnitude as the perpendicular cross product vector. This new vector $\hat{\mathbf{z}}$ can also belong to a new vector space, $\hat{\mathbf{z}} \in Z$. This means the cross product is a bilinear map which is defined as a function that takes three vector spaces, X, Y and Z over the same field and combines any pair of elements in X and Y to produce an element of Z,

$$B : X \times Y \to Z \tag{A.10}$$

Explicitly for the vector space $\mathbb{R}^3$ over the field of real numbers, $\mathbb{R}$, the cross product is the bilinear map defined

$$\mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3. \tag{A.11}$$

A vector space over a field with a bilinear map is called algebra over a field. Different vector spaces have different bilinear operators to create an algebra over a field for example, a vector space of $\mathbb{R}^{n \times n}$ square matrices uses matrix multiplication as its bilinear map and as will be shown Grassmann generators over the field of complex numbers use the exterior product as the bilinear map.

A.3.1 Structure Coefficients

Each vector space is constructed from a basis set, the multiplication of which controls the bilinear map. For the $\mathbb{R}^3$ vector space the basis vector multiplication rules were shown in Eq. (A.8). These rules and the associated, scalar coefficients can be summarised as such,

$$e_i e_j = \sum_{k=1}^{N} c_{i,j}^k e_k \tag{A.12}$$

where for $\mathbb{R}^3$ $N = 3$ for example if $i = 1, j = 2$ then $k = 3$

$$e_1 e_2 = c_{1,2}^3 e_3 = a_1 b_2 e_3, \tag{A.13}$$

$c_{i,j}^k$ is a scalar quantity corresponding to the coefficient in Eq. (A.9) for e_3 written using Einstein notation. The structure coefficients are constructed to ensure the bilinear map functions correctly for all basis functions.

A.4 Lie Groups and Lie Algebra

A Lie group is defined as a mathematical group, G, that has a manifold structure where all group operations are smooth. A manifold, M, is a topological space that for any point within M the neighbourhood around the point resembles an open set in Euclidean space. For example, in Fig. A.2 a unit sphere with the equation $x^2 + y^2 + z^2 - 1 = 0$ is shown to have a two-dimension manifold as it can be split into 6 different regions, hemispheres either side of each axis, which only depend on two coordinates which map perfectly onto two-dimensional disks. The manifold for the regions $z > 0$ and $z < 0$ are described by disks with the equation $x^2 + y^2 < 1$. These regions resemble a part of $\mathbb{R}^2$ Euclidean space. Further, the disks are said to be an open set of $\mathbb{R}^2$ because at any point, p, on the disk there exists a positive value, ϵ, that for any other point, p_ϵ, less than the distance, ϵ, away from p is also on the disk. The smooth condition means that the manifold is locally similar enough to a vector space allowing the application of calculus. In the same way a Lie group is a group with additional properties a Lie algebra is specific type of algebra over a field. A Lie algebra is defined for a vector space $\mathfrak{g}$ which has the bilinear map, or Lie bracket, $[\cdot, \cdot] : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$ which satisfies the antisymmetric property,

$$[x, y] = -[y, x], \ \forall x, y \in \mathfrak{g} \tag{A.14}$$

and the Jacobi identity

$$[x, [y, z]] + [y, [z, x]] + [z, [x, y]] = 0, \forall x, y, z \in \mathfrak{g} \tag{A.15}$$

These two concepts are naturally related a Lie Group has an accompanying Lie algebra.

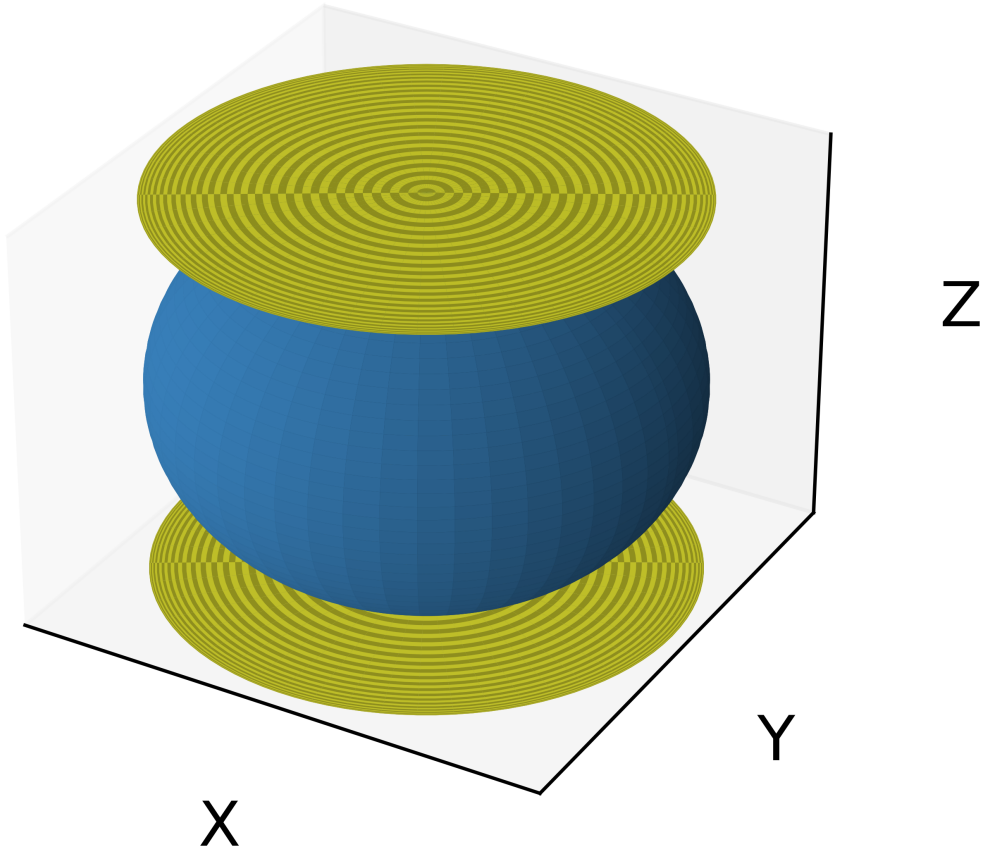


Figure A.2: Plot showing the unit sphere with equation $x^2+y^2+z^2-1=0$. The manifolds for regions $z>0$ and $z<0$ are the two disks with equation $x^2+y^2<1$.

A.4.1 Closed Subgroups

An arbitrary Lie group, G, is defined to have Lie algebra, $\mathfrak{g}$ then another group, H, is said to be a subset of G, denoted $H \subset G$, if every element of H is found in G. The complement, $G \setminus H$ is the set of elements in G not contained in H, if this complement is an open set then H is a closed subgroup. If H is a closed subgroup of Lie group G then the conditions for Cartan's theorem are met meaning that H is an embedded Lie subgroup of G [169]. The Lie algebra of H is then defined as

$$\mathfrak{h} = \{X \in \mathfrak{g} | e^{tX} \forall\, t \in \mathbb{R}\}. \tag{A.16}$$

The left translation is a map, L_g, for any fixed $g \in G$ is defined

$$L_g : G \to G \quad L_g(a) = ga \;\forall a \in G. \tag{A.17}$$

L_g is a continuously differentiable map on G which is also a diffeomorphism of G, $L_g^{-1} = L_{g^{-1}}$. A quotient or coset space G/H is a decomposition of all the elements of G into unique subsets of equal size that when multiplied by an element of H an

element of G is returned so

$$G/H \times H \to G \quad \Omega h = g \tag{A.18}$$

where $g \in G$, $h \in H$, $\Omega \in G/H$ which means that the left cosets of H in G are found by multiplying every element in G by a fixed element in G

$$G \times H \to G/H \quad \Omega = gh,\ g \in G,\ \forall h \in H \tag{A.19}$$

which can alternatively be written as gH, element g left multiplying all elements in H. If $H \subset G$ is a closed Lie subgroup then the coset space G/H is a unique real-analytic manifold. The coset space is homogeneous to G space if there is a continually differentiable mapping

$$\pi : G \times G/H \to G/H \quad g(aH) = (ga)H, \forall g, a \in G \tag{A.20}$$

and if $L_g(x) \in G/H \forall x \in H, g \in G$. Which gives the mapping

$$\tau_g : G/H \to G/H \quad \forall\, x, y \in G/H\, \exists\, g \in G :\ y = gx \tag{A.21}$$

which is the left translation by g [170]. A further condition of H to be a maximal compact subgroup can also be applied which means H is the maximum sized subgroup of G that is both closed and bounded.

A.4.2 Classical Lie Groups

$GL(n, F)$ is the group of $n \times n$ invertible matrices with elements of form F where $F = \mathbb{R}, \mathbb{C}$. Then using Cartan's theorem any closed subgroup of $GL(n, F)$ is a Lie group. Weyl called these subgroups over the real, $\mathbb{R}$, and complex, $\mathbb{C}$, numbers and quaternions, $\mathbb{H}$, the Classical groups[171]. These groups can be used to construct coherent states, some of the most common are summarised in table A.1. These special Lie groups have a determinant of one so are a subgroup of the group containing matrices of the same construction (but with determinants not necessarily equal to one.) These are in turn a subgroup of $GL(n, F)$ for the appropriate field. For example,

$$SU(n) \subset U(n) \subset GL(n, \mathbb{C}) \tag{A.22}$$

the special unitary group is a subgroup of the unitary group which is a subgroup of the general linear group over the complex field.

Table A.1: Summary of a selection of Classical Lie groups used in constructing coherent states. All groups consist of $n \times n$ invertible matrices, U with elements from the specified field. All matrices which have a determinant of one, $det(U) = 1$. Additionally, $n = p + q$ when needed

Name	Group	Field	Construction	Maximal compact subgroup
Special linear	$SL(n,\mathbb{R})$	$\mathbb{R}$		$SO(n)$
Complex special linear	$SL(n,\mathbb{C})$	$\mathbb{C}$		$SU(n)$
Special orthogonal	$SO(n)$	$\mathbb{R}$	$U^T = U^{-1}$	$S(O(p) \times O(q))$
Special unitary	$SU(n)$	$\mathbb{C}$	$U^* = U^{-1}$	$S(U(p) \times U(q))$

A.5 Grassmann Algebra

In this section the concept of Grassmann algebra is outlined including some useful results for manipulating them. Equations notation has been adapted with particular reference to Refs. [18, 20, 121, 122].

A.5.1 Properties of Grassmann Generators

The concepts of algebra over a field and the exterior product are combined. V is a set of N Grassmann generators $V = \{\xi_1, \xi_2, \ldots, \xi_N\}$ which form a complex basis for vector space of dimension N. The Grassmann generators by definition anticommute with each other so

$$\xi_i \xi_j = -\xi_j \xi_i \tag{A.23}$$

and so, their squares vanish $(\xi_i)^2 = 0$ and they commute with ordinary complex numbers $\xi_i x = x \xi_i$. The Grassmann generators form an algebra over the field of complex numbers, $\mathbb{C}$, with the exterior product being the bilinear operator. This is called the exterior algebra and is formally defined as

$$\Lambda(V) = \mathbb{C} \oplus V \oplus (V \wedge V) \oplus \ldots \oplus \underbrace{(V \wedge V \wedge \ldots \wedge V)}_{N} \equiv \mathbb{C} \oplus \Lambda^1 V \oplus \Lambda^2 V \oplus \ldots \oplus \Lambda^N V \tag{A.24}$$

This exterior algebra or Grassmann algebra can be given the symbol $\mathfrak{G}_N$. A general Grassmann number within $\mathfrak{G}_N$ is an arbitrary product of at most N Grassmann

generators which is given as

$$g = c_0(g) + \sum_{k\geq 1}^{N} \sum_{i_1,i_2,\ldots,i_k} c_{i_1,i_2,\ldots,i_k}(g)\xi_{i_1}\xi_{i_2}\ldots\xi_{i_k} \tag{A.25}$$

where $c_0(g)$ and $c_{i_1,i_2,\ldots,i_k}(g)$ are complex numbers. This definition becomes unique if it is stipulated that $i_1 < i_2 < \ldots < i_k$ are strictly increasing. Therefore, a basis is formed, $\mathfrak{G}_N = \{\xi_{i_1}\ldots\xi_{i_k}, i_1 < i_2 < \ldots < i_k\}$, with 2^N distinct possible products, including the empty product which has a value of one.

For example, $\mathfrak{G}_2$ has elements that can be uniquely written as

$$g = g_0 + g_1\xi_1 + g_2\xi_2 + g_3\xi_1\xi_2, \tag{A.26}$$

where $g_i = c_i(g)$ and for neatness of notation $\xi_1 \wedge \xi_2 = \xi_1\xi_2$ but the exterior product is still present. The Grassmann numbers form a vector space that is isomorphic to $\mathbb{C}^{2^N}$ which for the case $\mathfrak{G}_2$ is

$$1 \mapsto \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \xi_1 \mapsto \begin{pmatrix}0\\1\\0\\0\end{pmatrix}, \xi_2 \mapsto \begin{pmatrix}0\\0\\1\\0\end{pmatrix}, \xi_1\xi_2 \mapsto \begin{pmatrix}0\\0\\0\\1\end{pmatrix} \Rightarrow g \mapsto \begin{pmatrix}g_0\\g_1\\g_2\\g_3\end{pmatrix} \tag{A.27}$$

The left multiplication by a Grassmann number is linear operation on $\mathfrak{G}_2$ which can be represented

$$\begin{gathered} 1 \mapsto \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}, \xi_1 \mapsto \begin{pmatrix}0&0&0&0\\1&0&0&0\\0&0&0&0\\0&0&1&0\end{pmatrix}, \xi_2 \mapsto \begin{pmatrix}0&0&0&0\\0&0&0&0\\1&0&0&0\\0&-1&0&0\end{pmatrix}, \\ \xi_1\xi_2 \mapsto \begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\1&0&0&0\end{pmatrix} \Rightarrow g \mapsto \begin{pmatrix}g_1&0&0&0\\g_2&g_1&0&0\\g_3&0&g_1&0\\g_4&-g_3&g_2&g_1\end{pmatrix} \end{gathered} \tag{A.28}$$

$\mathfrak{G}_N$ is a Grassmann algebra with N generators which allows for an index set, $\mathfrak{E}[\mathbf{N}] = \{0,1\}^N$ to be introduced. Hence, if $\epsilon \in \mathfrak{E}[\mathbf{N}]$, then $\epsilon = (\epsilon_1, \ldots, \epsilon_N)$ with $\epsilon_j = 0, 1$. So, a set of N Grassmann generators can be denoted $\xi^\epsilon = \xi_1^{\epsilon_1}\ldots\ldots\xi_N^{\epsilon_N}$. Note $|\epsilon| = \epsilon_1 + \ldots + \epsilon_N$ is the number of occupied fermionic states. So Eq. (A.26) can be written so a Grassmann number $g \in \mathfrak{G}_N$ is

$$g = \sum_{\epsilon\in\mathfrak{E}[\mathbf{N}]} c_\epsilon(g)\xi^\epsilon. \tag{A.29}$$

g is invertible only if $g_0(g) \neq 0$. Odd and even functions are be defined for a Grassmann number $\psi \in \mathfrak{G}_N$ as such

$$f(-\psi) = f(\psi) \quad \text{Even function} \tag{A.30a}$$
$$f(-\psi) = -f(\psi) \quad \text{Odd function.} \tag{A.30b}$$

If a function is smooth around zero a Taylor formula can be defined

$$f(\psi) = \sum_{k\in\mathbb{N}} \frac{f^{(k)}(0)}{k!}\psi^k \tag{A.31}$$

which is finite because for $\psi^k = 0\ \forall k > N$. $2N$ Grassmann generators can be defined by $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_N)$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_N)$ then $f(\boldsymbol{\xi}, \boldsymbol{\gamma}) = e^{\boldsymbol{\xi}\cdot\boldsymbol{\gamma}}$ is well defined for $\boldsymbol{\xi}\cdot\boldsymbol{\gamma} = \sum_{1\leq k\leq N} \xi_k\gamma_k$,

$$e^{\boldsymbol{\xi}\cdot\boldsymbol{\gamma}} = 1 + \boldsymbol{\xi}\cdot\boldsymbol{\gamma} + \sum_{\epsilon\in\mathfrak{E}[\mathbf{N}],|\epsilon|\geq 2} (-1)^{\mu(\epsilon)}\boldsymbol{\xi}^{\epsilon}\boldsymbol{\gamma}^{\epsilon} \tag{A.32}$$

$\mu(\epsilon)$ is defined[1] by,

$$\mu(\epsilon) = \begin{cases} 1 & \text{if } 2,3 \equiv |\epsilon| mod(4) \\ 0 & \text{if } 0,1 \equiv |\epsilon| mod(4) \end{cases} \tag{A.33}$$

where $|\epsilon| mod(4)$ is the remainder of $|\epsilon|/4$.

A.5.2 Derivatives and Integrals of Grassmann Algebra

Derivatives with respect to a Grassmann generator,ξ, can be defined as such if $g = g_0 + \xi_i g_1$ and $g \in \mathfrak{G}_N$ and both g_0 and g_1 are independent of ξ_i then

$$\frac{\partial}{\partial\xi_i} g = g_1. \tag{A.34}$$

The order of the differential operator is important if $i \neq j$

$$\frac{\partial}{\partial\xi_i}\xi_j(a + \xi_i b + \xi_j c + \xi_i\xi_j d) = -b\xi_j \tag{A.35}$$
$$\xi_j\frac{\partial}{\partial\xi_i}(a + \xi_i b + \xi_j c + \xi_i\xi_j d) = b\xi_j \tag{A.36}$$

[1] This result is derived in lemma 72 in Ref [20].

else if $i = j$

$$\frac{\partial}{\partial \xi_i}\xi_i(a+\xi_i b) = a \tag{A.37}$$

$$\xi_i\frac{\partial}{\partial \xi_i}(a+\xi_i b) = \xi_i b. \tag{A.38}$$

The two objects clearly commute and so

$$\{\frac{\partial}{\partial \xi_i}, \xi_j\} = \delta_{ij} = \{a_i, a_j^\dagger\} \tag{A.39}$$

is the anticommutator relation which is equal to Eq. (2.29c) and means creation and annihilation operators can be defined as ξ_j and $\frac{\partial}{\partial \xi_i}$ respectively. Let $\psi \in \mathfrak{G}_N$ and $\mathfrak{G}_N = \{\xi_1 \dots \xi_N, 1 \leq i_1 < i_2 < \dots < i_k \leq N\}$ then

$$\int \psi d\xi_{i_1} \dots d\xi_{ik} \in \mathfrak{G}_N \tag{A.40}$$

is the Grassmann integration a linear map $\mathfrak{G}_N \mapsto \mathfrak{G}_N$which has the following properties

$$\int d\xi_j = 0, \forall j = 1 \dots N \tag{A.41a}$$

$$\int d\xi_j \xi_j = 1, \forall j = 1 \dots N \tag{A.41b}$$

$$\int \xi_j d\xi_k = \delta_{jk}, \forall j,k \tag{A.41c}$$

$$\int d\xi_j d\xi_k = 0, \forall j,k \tag{A.41d}$$

$$\int\int \psi(\xi_j)\varphi(\xi_k)d\xi_j d\xi_k = \int \psi(\xi_j)d\xi_j \int \varphi(\xi_k)d\xi_k, \forall j,k. \tag{A.41e}$$

Thus, integration coincides with differentiation so

$$\int \psi d\xi_{i_1} \dots d\xi_{i_k} = \frac{\partial}{\partial \xi_{i_k}} \dots \frac{\partial}{\partial \xi_{i_1}} \psi(\xi_1, \dots, \xi_N). \tag{A.42}$$

It is useful to construct a complex structure for the Grassmann algebra. $\mathfrak{G}_{2N}$ is generated by $\xi_1, \dots \xi_N; \xi_1^*, \dots \xi_N^*$ with the following conditions

$$(z\alpha)^* = \bar{z}\alpha^* \text{ if } z \in \mathbb{C} \text{ and } \alpha \in \mathfrak{G}_{2N} \tag{A.43a}$$

$$(\alpha\beta)^* = \beta^*\alpha^*, \forall \alpha, \beta \in \mathfrak{G}_{2N} \tag{A.43b}$$

which give rise to the following anticommutation relations,

$$\{\xi_i, \xi_j\} = 0 \tag{A.44a}$$

$$\{\xi_i^*, \xi_j\} = 0 \tag{A.44b}$$

$$\{\xi_i^*, \xi_j^*\} = 0. \tag{A.44c}$$

This complex (and conjugate) structure can be denoted $\mathfrak{G}_N^C$. Integration of the Grassmann algebra, with this specified complex structure, is defined

$$\int \psi d\boldsymbol{\xi} d\boldsymbol{\xi}^* = \int \psi d\xi_1 d\xi_1^* \dots d\xi_N d\xi_N^* \tag{A.45}$$

the notation $d\boldsymbol{\xi} d\boldsymbol{\xi}^*$ to show the integral is over all values of $\boldsymbol{\xi}$ and its conjugate. Note that $d\xi_i d\xi_i^* = -d\xi_i^* d\xi_i$. It is noted that the integral $\int \psi d\xi d\xi^*$ is invariant under unitary change of variable, $\xi = U\zeta, U \in SU(n)$

$$\int \psi d\xi d\xi^* = \int \psi(\bar{U}\zeta, U\zeta) d\zeta d\zeta^*. \tag{A.46}$$

as $\xi_N \dots \xi_1 = (\det U)\zeta_N \dots \zeta_1$ then

$$\xi_N^* \xi_N \dots \xi_1^* \xi_1 = (\zeta_N^* \zeta_N \dots \zeta_1^* \zeta_1)\overline{\det U} \det U = \zeta_N^* \zeta_N \dots \zeta_1^* \zeta_1. \tag{A.47}$$

Further, if B is an $N \times N$ Hermitian matrix then the diagonal matrix of B is defined by $UBU^\dagger = b\boldsymbol{I}_N$ where $b = (b_1, \dots, b_N)$, then using $\eta = U\xi$

$$\begin{aligned} \int e^{-\xi^* B \xi} d\xi^* d\xi = \int e^{-\eta^* b \boldsymbol{I}_N \eta} d\eta^* d\eta &= \prod_{1 \le j \le N} \int e^{-b_j \eta_j^* \eta_j} d\eta_j^* d\eta_j \\ &= \det B. \end{aligned} \tag{A.48}$$

Appendix B
Algorithmic and Programming Details

B.1 Program Overview

B.1.1 Program Design

The Zombie states program was designed to fulfil the following requirements

- The program must be highly accurate.
- The program must be efficient with minimal redundant or superfluous code.
- The program must make use of parallelisation when possible while also considering future possible hand-off to GPUs.
- The program must be designed to run for any type of molecule or atom and allow for a variety of different one- and two-electron sources.

While most considerations are self-explanatory the final point and GPUs are worth further discussion. There are multiple well used programs used to generate one- and two-electron integrals such as Q-Chem, MOLPRO and the open source PyScf [134, 172]. As will be subsequently detailed the Zombie state program has been designed to use PyScf to generate one- and two-electron integrals however this integration is independent of the main program and other sources could be very easily used with very minimal additional code needing to be written. This allows the program to be used more universally giving users freedom to use their electronic structure package of choice or for personal code to be used. The current program is written for CPU execution only making use of parallelisation opportunities where possible but the possibility of using GPUs for execution of simple but numerous calculations has been considered in the design of the algorithms. GPUs can run many simple concurrent tasks on parallel threads and their adoption in machine learning has driven recent advancements in the field. By designing algorithms that consist of

repeated multiplications and additions large future speed-ups should be more easily achieved by moving these parts of the algorithm to a GPU kernel for concurrent calculation.

The program has been designed in a modular way which allows for easier program development allowing modules to be restructured without major revisions to the code base being required elsewhere. Further, by using a modular structure future integration with other programs such as *ab initio* MCE is considerably simpler. The program is organised into the following main sections:

- **Main Program**, which determines the overall flow,
- **Electron integral module** contains subroutines to read in the one- and two-electron integrals and carry out the pre-processing for the Hamiltonian calculation,
- **Zombie state generation module**, set-ups the initial Zombie state functions,
- **Hamiltonian generation module**, generates Hamiltonian and overlap matrix elements
- **Imaginary time propagation module**, contains code for the imaginary time evolution process.
- **Cleaning module**, contains code to run the Zombie state cleaning routine.
- **Gradient descent module**, carries out the gradient descent process to optimise Zombie state "dead" and "alive" coefficients.
- **Input module**, reads in run parameters.
- **Output module**, produces data outputs for the simulations.

A basic summary of the program is illustrated in Fig. B.1. The detail of how a gradient descent epoch occurs has been omitted as it has been detailed in Fig. 4.8. Note the program uses some library functions from the BLAS library.

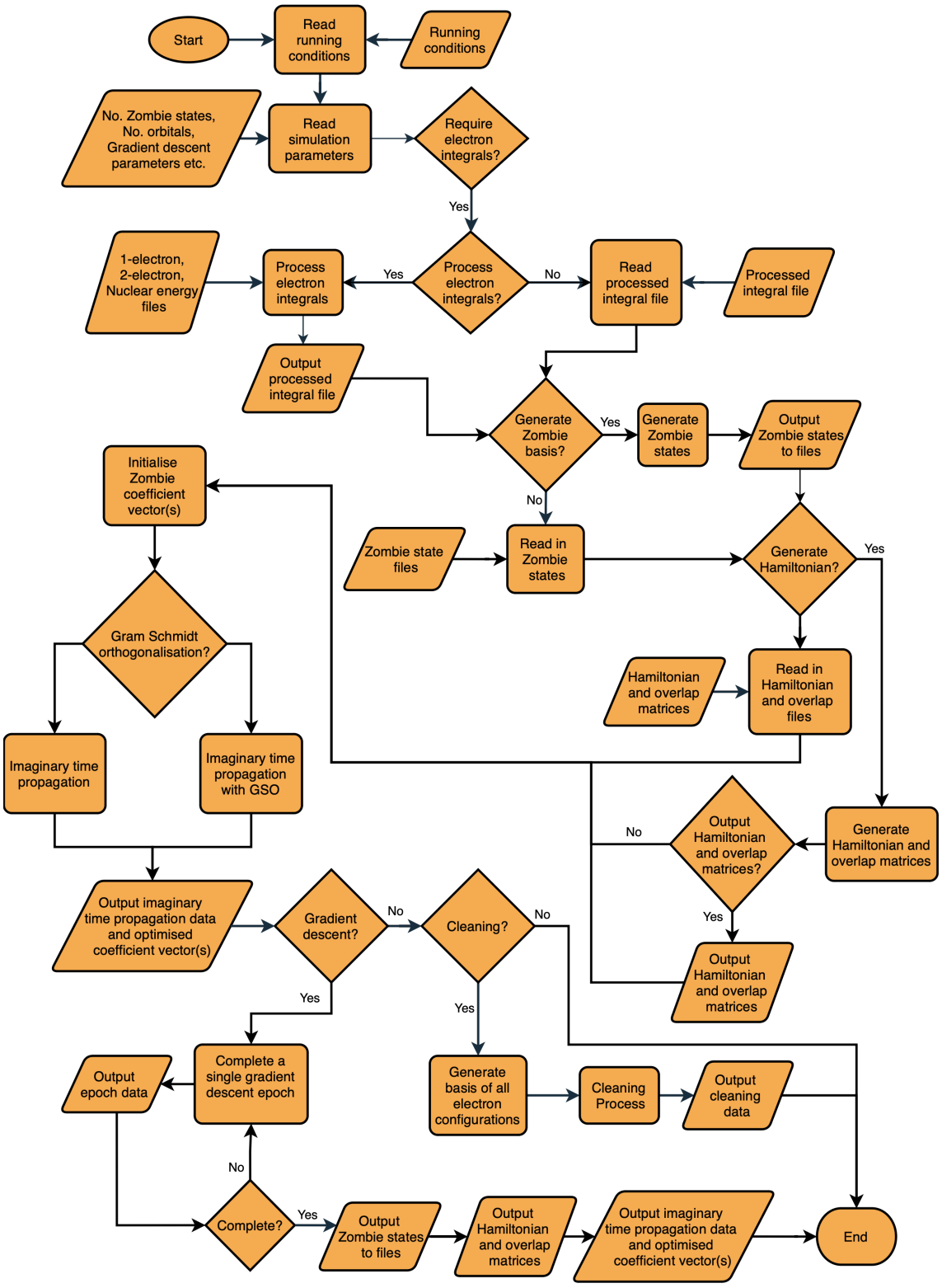


Figure B.1: Flow chart showing the layout of the Zombie states program.

B.1.2 Program Implementation

The main program has been written in a Fortran95 separated into modules containing related subroutines and functions. These modules are listed in Table. B.1 which also details their respective dependencies. However, it should be noted that the program does currently use the *iso_fortran_env* intrinsic module to define the precision of numerical parameters and standard outputs. This module was introduced in Fortran 2003 and updated in Fortran 2008 so should not be an issue if a relatively modern compiler is being used [173]. But to ensure portability the *iso_fortran_env* module is imported into the *mod_types* module and its values assigned to program specific parameters – see Listing. B.1. If the compiler is not compatible with *iso_fortran_env* then this module can be simply edited to redefine the standard outputs and variable precision. The parameter "*wp*" is used throughout the program as the working precision which is set to be double precision in the *GlobVars* module. Note the *rangen* module used for random number generation also uses *iso_fortran_env* which would require more work to make compatible with an older compiler and it would probably be easier to change the method of random number generation.

```
1  module mod_types
2      use iso_fortran_env, only:  int8, int16, int32, int64,
       real32, real64, input_unit, output_unit, error_unit
3      implicit none
4      integer, parameter :: sp        = real32
5      integer, parameter :: dp        = real64
6      integer, parameter :: int8_t    = int8
7      integer, parameter :: int16_t   = int16
8      integer, parameter :: int32_t   = int32
9      integer, parameter :: stdin     = input_unit
10     integer, parameter :: stdout    = output_unit
11     integer, parameter :: stderr    = error_unit
12 end module
```

Listing B.1: Mod_types module which allows program wide variable precision and standard outputs to be set to the *iso_fortran_env* standard and easily altered if necessary.

Table B.1: Table of modules, their purpose and dependencies for the Zombie states program.

Module Name	*Module Purpose*	*Module Dependencies*
rangen	Generate random numbers	iso_fortran_env
mod_types	Set iso_fortran	iso_fortran_env
GlobVars	Store global variables	mod_types
AlArrays	Allocate and deallocation of arrays	mod_types GlobVars
Electrons	Process one- and two-electron integrals	GlobVars AlArrays
Zom	Initiate Zombie functions	mod_types GlobVars
Ham	Calculate Hamiltonian and overlap matrices	mod_types GlobVars AlArrays
Imgtp	Perform imaginary time propagation	mod_types GlobVars AlArrays
ReadPars	Read in the simulation parameters	mod_types GlobVars
Outputs	Generate data output files	mod_types GlobVars
Clean	Perform cleaning routine	mod_types GlobVars AlArrays Zom Outputs ReadPars Ham
Gradient_descent	Contain gradient descent routine and gradient calculation routines	mod_types GlobVars AlArray Zom Outputs Ham Imgtp
MainZombie	Controls program flow	mod_types rangen GlobVars AlArrays Electrons Zom Imgtp ReadPars Outputs Clean Gradient_descent

All processes in the Zombie state program are centred round a set of ZS functions and so a defined type is used to contain this information Listing. B.2. The wave function is then constructed as an array of the Zombie state data type equal to the number of basis functions.

```
type Zombiest
  real(wp),dimension(:),allocatable::phi
  real(wp),dimension(:),allocatable::val
    integer::gram_num
end type Zombiest
```

Listing B.2: Definition of the "Zombiest" data type.

Within each Zombie state there are two arrays and an integer the variable *phi* is allocated to the number of orbitals and contains the set of values used to generate the "dead" and "alive" coefficients. The array *val* is set to be indexed from 0 to double the number of orbitals. The "alive" coefficients are placed in indexes $1 \rightarrow N_{orb}$ and the "dead" coefficients are stored in indexes $N_{orb}+1 \rightarrow 2N_{orb}$, the value at index zero is set to also be zero which is used when calculating Hamiltonian matrix elements. The *gram_num* variable is used to keep track of which wave function a Zombie state is in when Gram Schmidt orthogonalisation is being used. A Hamiltonian data type is defined to contain the Hamiltonian, overlap and inverse overlap matrix as well as the product of the Hamiltonian and inverse overlap matrix the structure of which is shown in Listing. B.3 with all arrays allocated to be of size $N_{bf} \times N_{bf}$.

```
type hamiltonian
  real(wp), dimension(:,:), allocatable::hjk
  real(wp), dimension(:,:), allocatable::ovrlp
    real(wp), dimension(:,:), allocatable::inv
    real(wp), dimension(:,:), allocatable::kinvh
end type hamiltonian
```

Listing B.3: Definition of the "Hamiltonian" data type.

Error checking has been built into the program to handle fatal errors caused by incorrect inputs or a problem with the machine. If an error is caught an error message is returned to the standard error output including the error code; the global variable *errorflag* is then set to 1. Each subroutine and function starts with the line of error check code shown in, Listing. B.4 which has the effect of skipping all further processes forcing the program to its end. Rather than simply terminating the program at the point of error, the skipping behaviour makes it easier to find the location of unexpected errors. The program contains a non-standard method to check for infinite or undefined numerical values which usually occur when there

is unexpected division by zero. Due to some of the aggressive optimisation flags being used (e.g. *-Ofast* in gfortran) the standard methods to check for these errors sometimes did not work. Thus, an alternative method written by Zaikun Zhang is used which is contained in three files *inf.f90*, *infnan_constants.f90* and *infnan.f90* [174]. The function *is_nan(x)* takes a numerical input and then checks if it has the value infinity or NaN which then returns a Boolean.

```
if (errorflag .ne. 0) return
```

Listing B.4: Check for prior errors at the beginning of each subroutine or function.

B.2 Zombie state Creation

Zombie states are initialised with either random values, biased "dead" and "alive" coefficients or integers so they are equivalent to a HF Slater determinant. The random and biased initialisation routines require random number generation. The Multi-configurational Ehrenfest method program utilises the ZBQL* family of subroutines to generate random numbers and since there is the possibility of future integration with the MCE program it was sensible to continue using these functions [175]. The set of random number generation routines used in MCE were written by Richard Chandler and Paul Northrop with subsequent modification by Stuart Reed [176]. The library is written as a set of external subroutines in Fortran77 style and is not thread safe when used in parallel programming. The Zombie states program uses a modified version of the original library that contains all the subroutines within a Fortran module and is written in an updated Fortran95 style. The original common block is shown in Listing. B.5.

```
BLOCK DATA ZBQLBD01
      COMMON /ZBQL0001/ ZBQLIX,B,C
      DOUBLE PRECISION ZBQLIX(43),B,C
      INTEGER I
      DATA (ZBQLIX(I),I=1,43) /8.001441D7,5.5321801D8,
        ...
      +2.63576576D8/
      DATA B / 4.294967291D9 /
      DATA C / 0.0D0 /
END
```

Listing B.5: Common block structure used in rangen.f external subroutine [176]. For neatness some of the numerical values in the variable ZBQLIX have been omitted.

The module variables used in the modified version are shown in Listing. B.6 using the same variable names as in the original code. The variable attribute, *save*, is used to ensure that when re-accessing the module all necessary values are retrained to continue generating random numbers. Clearly there are additional variables present in Listing. B.6 that are not present in Listing. B.5 this is because certain subroutines and functions also have variables that need to be accessed again on subsequent calls.

```
MODULE randgen
use iso_fortran_env, only:  int8, int16, int32, int64,
   real32, real64, input_unit, output_unit, error_unit
implicit none
real(real64), private, save::B,C
real(real64), private, save, dimension(43)::ZBQLIX
integer, private, save::CURPOS,ID22,ID43
integer, private, save::INIT
integer, private, save::STATUS
real(real64), save::SPARE
real(real64), parameter::PI = 4.0D0*DATAN(1.0D0)
real(real64), parameter::RLN2P = 0.5D0*DLOG(2.0D0*PI)
real(real64), private, parameter, dimension(0:6)::zbqllg_c
   =[1.000000000190015D0, ... ,-0.5395239384953D-5]

contains

  subroutine ZBQLBD01()
        implicit none

        ZBQLIX=[8.001441D7,5.5321801D8,
            ...
            3.20429173D8,2.63576576D8]
        B = 4.294967291D9
        C = 0.0D0
        CURPOS = 1
        ID22 = 22
        ID43 = 43
        STATUS = -1

    end subroutine ZBQLBD01
```

Listing B.6: Random generation module variables and initialisation routine modified from original rangen.f library [176]. For neatness some numerical values in the variables ZBQLIX and zbqllg_c have been omitted.

The other type of Zombie state are those analogous to a Slater determinant. Defining the Zombie states is simple as the "dead" and "alive" coefficients are binary. A single

Slater determinant with a specific set of occupied orbitals can be encoded manually. To generate a complete set of Slater determinants containing all possible electron configurations adapted from Ref. [177] is used. The algorithm takes the number of spin orbitals N_{orb} and the number of electrons, N_{el}, as inputs and produces a list of all possible ways N_{el} electrons can be distributed across the available orbitals. Using the list of configurations the corresponding Zombie states are then be generated.

B.3 Operator Algorithms

B.3.1 Hamiltonian Matrix Algorithm

The calculation of the Hamiltonian matrix is the most computationally expensive part of the entire Zombie states program and so there has been much work to construct more efficient algorithms to complete the task. Initial work focused on just the two-electron part of the Hamiltonian as this was the greatest source of the bottleneck. From equation Eq. (2.230) each two-electron part of the Hamiltonian matrix element,

$$\hat{H}_2 = \frac{1}{2} \sum_{i,j,k,l} \langle ij|kl\rangle \hat{b}_i^\dagger \hat{b}_j^\dagger \hat{b}_l \hat{b}_k. \tag{B.1}$$

can, naïvely, be calculated by summing the result of sequential applications of creation and annihilation operators followed by calculation of an overlap

$$\langle \boldsymbol{\zeta}^{(a)}|\hat{H}_2|\boldsymbol{\zeta}^{(b)}\rangle = \frac{1}{2} \sum_{ijkl}^{N_{orb}} \langle ij|kl\rangle \langle \boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}^{(b)}_{ijlk}\rangle \tag{B.2}$$

where $|\boldsymbol{\zeta}^{(b)}_{ijlk}\rangle = b_i^\dagger \hat{b}_j^\dagger \hat{b}_l \hat{b}_k |\boldsymbol{\zeta}^{(b)}\rangle$ [9]. This requires ${N_{orb}}^4$ terms to be evaluated because each term has four creation/annihilation operations each of which is $\sim \mathcal{O}(M)$ due to the sign changing rule followed by an overlap calculation which is $\mathcal{O}(M)$ as

$$\Omega_{ab} = \langle \boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}^{(b)}\rangle = \prod_{m=1}^{M} \sum_{n_m=0,1} a^{(a)*}_{n_m m} a^{(b)}_{n_m m}. \tag{B.3}$$

Therefore, naïve evaluation of Eq. (2.4.5) overall has $\mathcal{O}(M^5)$ scaling. Therefore, by the same reasoning the one-electron part requires M^2 creation/annihilation operations and is $\mathcal{O}(M^3)$ overall. Improvements to the two-electron algorithm are made by firstly reducing the prefactor and then the overall scaling of the algorithm. Finally, the current Hamiltonian algorithm that brings together the one- and two-electron parts into a single algorithm using the concepts introduced for the two-electron part.

B.3.1.1 Reduced Prefactor Hamiltonian Algorithms

The prefactor of the two-electron algorithm is reduced by removing parts of the summation already known to be zero. This is done by using spin symmetry

$$\langle ij|kl\rangle = \delta_{\sigma(i)\sigma(k)}\delta_{\sigma(j)\sigma(l)} \tag{B.4}$$

where $\sigma(i)$ is the spin of spin orbital i. Thus, it is only necessary to evaluate $|\boldsymbol{\zeta}^{(b)}_{ijlk}\rangle = b_i^\dagger \hat{b}_j^\dagger \hat{b}_l \hat{b}_k |\boldsymbol{\zeta}^{(b)}\rangle$ if $\langle ij|kl\rangle \neq 0$ on spin symmetry grounds. As shown in Table. B.2 when the code is modified to ignore the zero values there is nearly a four-fold speed up when calculating a matrix element between two ten-orbital ZSs. This is as expected as three quarters of the two-electron integrals are zero from spin symmetry.

Table B.2: Time (in seconds) to compute a two-electron Hamiltonian matrix element between two Zombie states with either 10 or 50 orbitals for N_2 in the $6-31G^{**}$ basis. The naïve algorithm loops through all indices the second algorithm only applies creation and annihilation operators when $\langle ij|kl\rangle \neq 0$. This gives a good speed improvement.

Number of orbitals N_{orb}	Naïve	Ignore zeros	Speed up
10	0.2466	0.0634	3.89
50	463.9408	29.2356	15.87

Further, albeit modest, improvements can be made if only combinations of $\{ijkl\}$ already known to be non-zero are included in the loop. So, loops are constructed to only include orbitals which satisfy, $|\boldsymbol{\zeta}^{(b)}_{ijlk}\rangle$, see Table. B.3.

Table B.3: Time (in seconds) to compute a two-electron Hamiltonian matrix element between two Zombie states with either 10 or 50 orbitals for N_2 in the $6-31G^{**}$ basis. The naïve algorithm loops over all indices the second algorithm considers $ijkl$ with appropriate spin symmetry and values known to be non-zero before calculating the action of creation and annihilation operators. This gives a good improvement on the naïve algorithm and slight improvement on the algorithm above that does not have improved looping.

Number of orbitals N_{orb}	Naïve	Ignore zeros and better loops	Speed up
10	0.2466	0.0630	3.91
50	463.9408	27.9327	16.61

Alternatively, the number of expensive creation and annihilation operations can be reduced. Eq. (B.1) and Eq. (B.2) can be rearranged to give

$$\langle \boldsymbol{\zeta}^{(a)}|\hat{H}_2|\boldsymbol{\zeta}^{(b)}\rangle = \frac{1}{2}\sum_{ijkl}^{N_{orb}} \langle ij|kl\rangle \langle \boldsymbol{\zeta}^{(a)}_{ij}|\boldsymbol{\zeta}^{(b)}_{lk}\rangle \tag{B.5}$$

where

$$\langle \boldsymbol{\zeta}_{ij}^{(a)}| = \langle \boldsymbol{\zeta}^{(a)}|\hat{b}_i^\dagger \hat{b}_j^\dagger = \left(\hat{b}_j \hat{b}_i |\boldsymbol{\zeta}^{(a)}\rangle\right)^\dagger \tag{B.6}$$

$$|\boldsymbol{\zeta}_{lk}^{(b)}\rangle = \hat{b}_l \hat{b}_k |\boldsymbol{\zeta}^{(b)}\rangle \tag{B.7}$$

$\{|\boldsymbol{\zeta}_{ij}^{(a)}\rangle\}\forall i,j$ can then be calculated and $\{|\boldsymbol{\zeta}_{lk}^{(b)}\rangle\}\forall l,k$, requiring N_{orb}^2 creation and annihilation operations while still overall being $\mathcal{O}(M^5)$. With $\{|\boldsymbol{\zeta}_{ij}^{(a)}\rangle\}$ and $\{|\boldsymbol{\zeta}_{lk}^{(b)}\rangle\}$ precomputed Eq. (B.5) can then be computed. Further, if both the "dead" and "alive" amplitudes for the same orbital are zero from the definition of the overlap in Eq. (2.223) then its overlap with any other ZS will be zero. So, for the jth spin orbital

$$|\boldsymbol{\zeta}^{(c)}\rangle = \begin{bmatrix} a_{11}^{(c)} & a_{12}^{(c)} & \dots & a_{1(m-1)}^{(c)} & 0 & a_{1(j+1)}^{(c)} & \dots & a_{1N_{orb}}^{(c)} \\ a_{01}^{(b)} & a_{02}^{(c)} & \dots & a_{0(m-1)}^{(c)} & 0 & a_{1(j+1)}^{(c)} & \dots & a_{0N_{orb}}^{(c)} \end{bmatrix} \tag{B.8}$$

Furthermore, as the creation and annihilation operators only move amplitudes within an orbital the overlap with $|\boldsymbol{\zeta}^{(c)}\rangle$ will continue to be zero regardless of any creation or annihilation operation. So, if a state with zero "dead" and "alive" amplitudes is generated by creation and annihilation operation it does not need to be considered in in further calculations. As previously established it is impossible to annihilate an electron that does not exist and so if $a_1 = 0$ then

$$\hat{b}|\zeta\rangle \equiv \hat{b}\begin{bmatrix} a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} 0 \\ a_1 \end{bmatrix}. \tag{B.9}$$

returns a vanishing state with no overlap. Hence, if an annihilation operation on orbital i returns a vanishing state when looping over orbital indices in Eq. (B.5) any further calculations for any other index j,l,k can be disregarded and the next index i considered. So, if any $a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)}$ terms are zero in

$$\Omega_{ab} = \langle \boldsymbol{\zeta}^{(a)}|\boldsymbol{\zeta}^{(b)}\rangle = \prod_{j=1}^{N_{orb}} a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)} \tag{B.10}$$

then $\Omega_{ab} = 0$ and the calculation can be terminated. Implementation of these improvements gave a similar time improvement to the spin symmetry changes when considering with ten orbitals, but only a half as good time improvement when states have 50 orbitals as detailed in Table. B.4.

Table B.4: Time (in seconds) to compute a two-electron Hamiltonian matrix element between two Zombie states with either 10 or 50 orbitals for N_2 in the $6-31G^{**}$ basis. The naïve algorithm loops through all indices the second algorithm precomputes $\{|\zeta_{ij}^{(a)}\rangle\}$ and $\{|\zeta_{lk}^{(b)}\rangle\}$ and applies a rule that if any $a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)}$ terms are zero, then $\Omega_{ab} = 0$. This gives a good improvement on the naïve algorithm for 10 orbitals but the time improvement is not as good for 50 orbitals as the previous algorithms.

Number of orbitals N_{orb}	Naïve	Precomputation	Speed up
10	0.2466	0.0668	3.69
50	463.9408	49.1610	9.44

B.3.1.2 Lower-scaling Hamiltonian Algorithm

It is also possible to reduce the scaling of the two-electron Hamiltonian algorithm from $\mathcal{O}(N_{orb}^5)$ to $\mathcal{O}(N_{orb}^4)$. Writing the two-electron Hamiltonian as

$$\langle \boldsymbol{\zeta}^{(a)}|\hat{H}_2|\boldsymbol{\zeta}^{(b)}\rangle = \frac{1}{2}\sum_{ijkl}^{N_{orb}} \langle ij|kl\rangle \langle \boldsymbol{\zeta}_{ij}^{(a)}|\hat{b}_l|\boldsymbol{\zeta}_k^{(b)}\rangle. \tag{B.11}$$

For a specific set of i,j,k it is possible to calculate $\langle \boldsymbol{\zeta}_{ij}^{(a)}|\hat{b}_l|\boldsymbol{\zeta}_k^{(b)}\rangle$ with $\mathcal{O}(N_{orb}^3)$ scaling and then summing over all values of l must be done in $\mathcal{O}(N_{orb})$ operations to give an overall scaling of $\mathcal{O}(N_{orb}^4)$. $\langle \boldsymbol{\zeta}^{(a)}|\hat{b}_l|\boldsymbol{\zeta}^{(b)}\rangle$ can be written as

$$\langle \boldsymbol{\zeta}^{(a)}|\hat{b}_l|\boldsymbol{\zeta}^{(b)}\rangle = \left(\prod_{i=1}^{l-1} a_{0i}^{(a)*}a_{0i}^{(b)} - a_{1i}^{(a)*}a_{1i}^{(b)}\right)\cdot a_{0l}^{(a)*}a_{1l}^{(b)}\cdot\left(\prod_{j=l+1}^{N_{orb}} a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)}\right) \tag{B.12}$$

and then define

$$s_i = a_{0i}^{(a)*}a_{1i}^{(b)}, \tag{B.13a}$$

$$e_i = a_{0i}^{(a)*}a_{0i}^{(b)} - a_{1i}^{(a)*}a_{1i}^{(b)}, \tag{B.13b}$$

$$f_i = a_{0i}^{(a)*}a_{0i}^{(b)} + a_{1i}^{(a)*}a_{1i}^{(b)}. \tag{B.13c}$$

which can be trivially calculated in $\mathcal{O}(N_{orb})$ steps. As such the following can be defined,

$$g_l = \prod_{i=1}^{l} e_i, \tag{B.14a}$$

$$h_l = \prod_{i=l}^{M} f_i \tag{B.14b}$$

which can also be calculated recursively in $\mathcal{O}(N_{orb})$ steps. Thus

$$\langle \boldsymbol{\zeta}^{(a)}|\hat{b}_l|\boldsymbol{\zeta}^{(b)}\rangle = g_{l-1}s_l h_{l+1} \tag{B.15}$$

which requires $\mathcal{O}(N_{orb})$ steps. Comparing the reduced scaling algorithm to the naïve algorithm shows and improvement in the time needed to calculate the two-electron matrix elements of over 20 times for 10 orbitals and over 100 times for 50 orbitals.

Number of orbitals N_{orb}	Naïve	Scaled	Speed up
10	0.2466	0.0116	21.31
50	463.9408	4.5004	103.09

Table B.5: Time (in seconds) to compute a two-electron Hamiltonian matrix element between two Zombie states with either 10 or 50 orbitals for N_2 in the $6-31G^{**}$ basis. The naïve algorithm loops through all indices and scales $\mathcal{O}(N_{orb}^5)$ the scaled algorithm is $\mathcal{O}(N_{orb}^4)$.

B.3.1.3 Combined Hamiltonian Matrix Element Equation

The introduction of the Gradient descent method meant that Hamiltonian matrix elements were being recalculated throughout the program and despite the improvements to the two-electron Hamiltonian algorithm it continued to be the most time intensive part of the program. Therefore, to reduce the matrix element calculation time it is useful to carry out as much pre-processing as possible once at the beginning of the program minimising the total number of operations at the point of matrix element calculation. The new algorithm takes aspects of the reduced scaling and reduced prefactor algorithms used for the two-electron Hamiltonian and applies them to the entire Hamiltonian matrix element calculation. Each part of the Hamiltonian matrix element is the sum of overlap calculations, between two Zombie states acted on by some combination of creation and annihilation operations, multiplied by a one- or two-electron integral. Each Hamiltonian matrix element is calculated using the same order and set of operations. So, to maximise computational efficiency, by requiring fewer processor decisions to be made during execution, the set of creation and annihilation operations can be stored along with the integral they correspond to. So, the one- and two-electrons parts can be combined into a single summation calculated in any desired order with operations being chosen from a stored list.

Using the rules established for the two-electron algorithm it is possible to remove all zero values from the stored list of creation and annihilation operations. Next it is possible to explicitly qualify the different types of values that can occur in an overlap calculation according to the combinations of creation and annihilation operations. These values are summarised in Table. B.6 for an arbitrary orbital, j, note that the negative values occur due to the sign change rule when there are creation and annihilation operations on higher orbitals and the number of operations must be odd for this effect not to be cancelled out.

Table B.6: Summary of all possible values that could occur in an overlap calculation between $\langle\zeta^{(a)}|$ and $|\zeta^{(b)}\rangle$ for orbital j. Negative values are caused by an odd number of creation and/or annihilation operations on higher orbitals. The terms are classified so they can be processed accordingly. No operator terms are the same as in the overlap matrix; Sign change terms only have a negative sign in front of the alive coefficient and the rest are classified according to the operator(s) acting on that orbital.

Term in Overlap	Operators	Classification
$a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)}$		No operator
$a_{0j}^{(a)*}a_{0j}^{(b)} - a_{1j}^{(a)*}a_{1j}^{(b)}$		Sign change
$a_{0j}^{(a)*}\cdot 0 + a_{1j}^{(a)*}a_{0j}^{(b)} = a_{1j}^{(a)*}a_{0j}^{(b)}$	$\hat{b}_j^\dagger$	Creation
$a_{0j}^{(a)*}\cdot 0 - a_{1j}^{(a)*}a_{0j}^{(b)} = -a_{1j}^{(a)*}a_{0j}^{(b)}$	$\hat{b}_j^\dagger$	Creation
$a_{0j}^{(a)*}a_{1j}^{(b)} + a_{1j}^{(a)*}\cdot 0 = a_{0j}^{(a)*}a_{1j}^{(b)}$	$\hat{b}_j$	Annihilation
$-a_{0j}^{(a)*}a_{1j}^{(b)} + a_{1j}^{(a)*}\cdot 0 = -a_{0j}^{(a)*}a_{1j}^{(b)}$	$\hat{b}_j$	Annihilation
$a_{0j}^{(a)*}\cdot 0 + a_{1j}^{(a)*}a_{1j}^{(b)} = a_{1j}^{(a)*}a_{1j}^{(b)}$	$\hat{b}_j^\dagger\hat{b}_j$	Both
$a_{0j}^{(a)*}\cdot 0 - a_{1j}^{(a)*}a_{1j}^{(b)} = -a_{1j}^{(a)*}a_{1j}^{(b)}$	$\hat{b}_j^\dagger\hat{b}_j$	Both

Like in the lower scaling algorithm it is possible to reduce the number of required operations by allowing values to be shared between overlap calculations. For example, $\langle\zeta^{(a)}|\hat{b}_l^\dagger\hat{b}_j|\zeta^{(b)}\rangle$ has the same set of multiplicands in the overlap calculation as $\langle\zeta^{(a)}|\hat{b}_l^\dagger\hat{b}_{j+1}|\zeta^{(b)}\rangle$ up to orbital $j-1$. Hence, values up to a specific orbital can be calculated once and then used repeatedly in other calculations which is similar to the idea of the lower scaling algorithm but now also including the one-electron parts. Further, the first term in Table. B.6 is identical to that found in the calculation of an element of overlap matrix Ω_{ab} and within the Hamiltonian matrix element calculation these are the most common multiplicand across the entire summation. Since, it is necessary to calculate the overlap matrix Ω anyway the number of processes can be reduced by calculating the overlap once and then multiplying that value by an appropriate set of ω_j defined as

$$\omega_j = \frac{f(\zeta^{(a)},\zeta^{(b)})}{a_{0j}^{(a)*}a_{0j}^{(b)} + a_{1j}^{(a)*}a_{1j}^{(b)}} \tag{B.16}$$

where

$$f(\zeta^{(a)},\zeta^{(b)}) = \begin{cases} a_{0j}^{(a)*}a_{0j}^{(b)} - a_{1j}^{(a)*}a_{1j}^{(b)} \\ \pm a_{1j}^{(a)*}a_{0j}^{(b)} \\ \pm a_{0j}^{(a)*}a_{1j}^{(b)} \\ \pm a_{1j}^{(a)*}a_{1j}^{(b)} \end{cases} \tag{B.17}$$

Using the values in Table. B.6 each term in every overlap calculation needed in the Hamiltonian matrix summation can be classified and this information stored. So, the Hamiltonian can be calculated by summing together the overlap matrix element

multiplied by a specific set of values defined by Eq. (B.16) and Eq. (B.17).

The pre-processing starts by removing zero valued overlaps and then classifying all parts of the overlap calculation for each set of creation and annihilation operations. Next each overlap is checked to see if any sign changes for orbitals with creation, annihilation or both operations cancel over the entire multiplication if they do not the corresponding electron integral is multiplied by -1. This allows orbitals that are operated on to be considered the same in the calculation. The largest category of multiplicand for each spin orbital is found which is then used to decide the order the calculation for each orbital is carried out in. The order of the integral summation is decided by first grouping integrals with the same classification for the first orbital together. The pre-processing then moves across all of the orbitals reordering the summation so the maximum number of calculations can be reused. So, the first orbital set the summation as *No operator, Creation, Annihilation, Sign change, Both* then at the next orbital the summation would be reordered for each previous classification – the set of integrals within the *No operator* classification for the first orbital could then be reordered *No operator, Annihilation, Creation, Sign change, Both* for the second orbital. This process continues for all orbitals. The indices where *No operator* are stored to allow the algorithm to skip over these as they are no different to the overlap multiplicand. This allows the longest possible blocks of the term to be created and used in multiple overlap calculations. The behaviour of the pre-processing algorithm is shown in Fig. B.2. Each column is a different part of the Hamiltonian summation and each of the orbital classifications are given a different colour. The first 50 parts of the sum are shown before and after processing. The algorithm completes all of the operations in the first row before starting on the next orbital, adjacent squares of the same colour have the same value and so recalculation is not necessary. The top panel shows the summation before processing and the bottom the order of summation after processing. Fig. B.2 makes it easy to visualise how the processing reduces the recalculation of intermediate values.

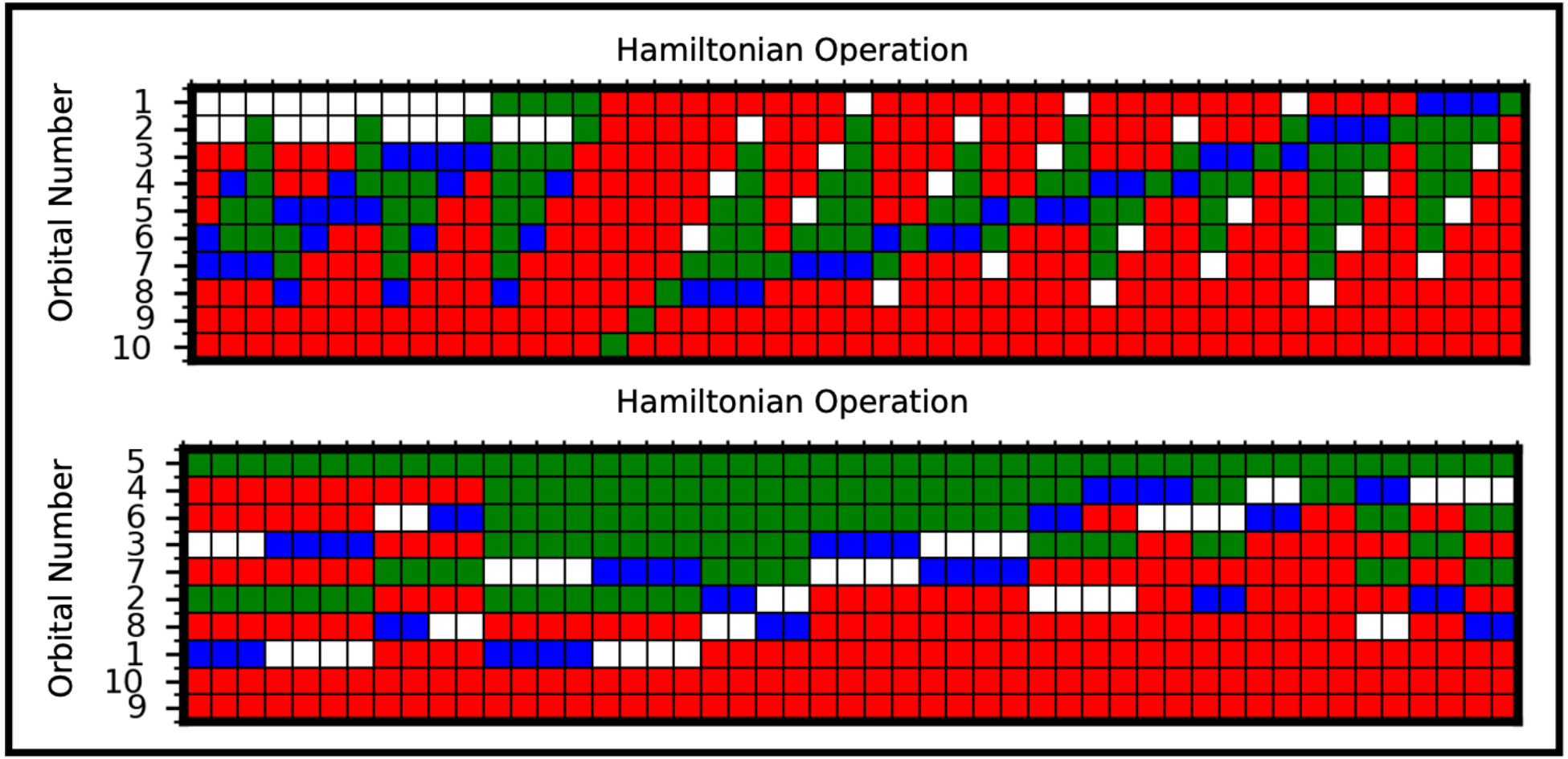


Figure B.2: Visualisation of Hamiltonian matrix element pre-processing algorithm showing the number of intermediate values needed to be calculated is reduced. The first 50 summations in the Hamiltonian matrix equation are shown for the unprocessed (top) and processed (bottom) using Li_2 containing 10 spin orbitals. The algorithm starts in the top left-hand corner and works along the row before starting on the next orbital. Adjacent colours have the same value and so do not need recalculating. Before processing each orbital contribution is calculated in order of index whereas this is optimised along with the order of summation to ensure a greater number of repeated intermediate values. Unchanged values are red, annihilation operations are blue, creation operations are white and a negative alive amplitudes are green.

146

Table B.7: Time (in seconds) to compute a two-electron Hamiltonian matrix element between two Zombie states with either 10 or 50 orbitals for N_2 in the $6-31G^{**}$ basis. The pre-processed algorithm is significantly faster than the naïve algorithm that loops through all indices.

Number of orbitals N_{orb}	Naïve	Pre-processed	Speed up
10	0.2466	1.8291×10^{-5}	13482
50	463.9408	2.7561×10^{-2}	16833

At the point of execution ω_j for all possible $f(\zeta^{(a)},\zeta^{(b)})$ for every orbital j is calculated sequentially as $a_{0j}^{(a)*}a_{0j}^{(b)}+a_{1j}^{(a)*}a_{1j}^{(b)}$ is calculated to find the overlap matrix. The algorithm then loops over all orbitals using the applicable ω_j to alter terms in each summand. Using the list of indices stored in the pre-processing step if terms up to the current orbital but in different summands are the same the value is shared rather than recalculated. Finally, each summand value can be multiplied by its respective W_{klji} or h_{ij} value which now hold any sign change information followed by the final summation.

Comparing the pre-processed algorithm, considering just the two-electron integrals, to the naïve algorithm Table. B.7 shows over 13000 and 16000 times speed-ups for 10 and 50 spin orbitals respectively. The pre-processed algorithm also compares well with the scaled algorithm. When calculating the 2-electron Hamiltonian elements the pre-processed algorithm gives a 630 times speed up for 10 orbitals and 160 times speed up for 50 orbitals for N_2 in the $6-31G^{**}$ basis. Also shown in Table. B.8 is the time the algorithm requires to calculate the complete Hamiltonian matrix element and overlap matrix element. Despite the overall sum being larger the overall time is lower than when just calculating the two-electron part. The inclusion of the one-electron integrals reduces the number of intermediate values that have to be computed directly.

Table B.8: Time (in seconds) to compute a two-electron Hamiltonian matrix element between two Zombie states with either 10 or 50 orbitals for N_2 in the $6-31G^{**}$ basis. The pre-processed algorithm is faster than the scaled algorithm and is marginally faster when calculating the full Hamiltonian and the overlap matrix. Thus, overall is a better choice for calculating the entire Hamiltonian matrix.

Number of orbitals N_{orb}	Scaled	Pre-processed	Speed up	Full algorithm
10	1.1571×10^{-2}	1.8291×10^{-5}	632.61	1.2746×10^{-5}
50	4.5004×10^{0}	2.7561×10^{-2}	163.28	2.7200×10^{-2}

To store the values necessary to use the pre-processed algorithm a derived type is used which is shown in Listing. B.7. The variable *num* stores the total number of non-zero one- and two-electron integrals, N_{Int} which is the length of the array

integrals and *hnuc* is the nucleus energy. The array *orbital_choice3* stores the order of orbitals used in the Hamiltonian and *orbital_choice* stores the multiplicand classifications making it a $N_{orb} \times N_{Int}$ array. The array *orbital_choice2* stores the indices to be looped over for each orbital, for each orbital pairs of start and end indices for each classification to be calculated are stored, which allows the algorithm to skip over any unchanged multiplicands. The first dimension of the array is allocated $0 \rightarrow N_{orb}$ which allows the program to store the total number of index pairs for each orbital and the second dimension is then the largest number of index pairs.

```
type elecintrgl
        integer::num
        real(wp), dimension(:), allocatable::integrals
        integer,dimension(:,:),allocatable::orbital_choice
        integer,dimension(:,:),allocatable::orbital_choice2
        integer,dimension(:),allocatable::orbital_choice3
        real(wp) :: hnuc
    end type elecintrgl
```

Listing B.7: Definition of the "elecintrgl" data type.

A single matrix element is then calculated using the code in Listing. B.8 which starts by calculating the overlap matrix element and the different orbital multiplicands, storing them in the $4 \times N_{orb}$ array *perts* for Zombie states *z1d* and *z2d*. The nucleus contribution is added to the final Hamiltonian total and the 1-D array *ovrlp_vec* length N_{Int} is set to the overlap value. The algorithm then loops through each orbital, index k, altering the values in *ovrlp_vec* by the appropriate value in *perts*, index l and copying the value to other parts of the sum if it is the same, index j. Finally, the values can be summed together to give the final Hamiltonian matrix element.

```
ovrlp=1
do j=1,norb
  ! Alive coefficients multiplied together
  aa=z1d(j)*z2d(j)
  ! Dead coefficients multiplied together
  dd=z1d(j+norb)*z2d(norb+j)
  ! Alive coefficient multiplied with dead coefficient
  ad=z1d(j)*z2d(norb+j)
  ! Dead coefficient multiplied with alive coefficient
  da=z1d(j+norb)*z2d(j)
  ! Contribution to overlap matrix for orbital j
  div(j)=aa+dd
  ! Creation operator multiplicand for orbital j
  perts(1,j)=da/div(j)
  ! Annihilation operator multiplicand for orbital j
  perts(2,j)=ad/div(j)
  ! Both operators multiplicand for orbital j
  perts(3,j)=aa/div(j)
  ! Negative alive coefficient multiplicand for orbital j
  perts(4,j)=(-aa+dd)/div(j)
  ! Overlap matrix element calculation
  ovrlp=ovrlp*div(j)
end do
! Nucleus contribution to Hamiltonian matrix element
ham_tot=ovrlp*elecs%hnuc
! Set all parts of summation to the overlap value
ovrlp_vec=ovrlp

do k=1,norb   ! Loop over the rows in elecs%orbital_choice
  do l=1,elecs%orbital_choice2(0,k)
    ov=ovrlp_vec(elecs%orbital_choice2(k,(l*2)-1))*perts(
   elecs%orbital_choice(k,elecs%orbital_choice2(k,(l*2)-1)),
   elecs%orbital_choice3(k))

    ! elecs%orbital_choice3(k) selects the orbital
    ! elecs%orbital_choice(k,elecs%orbital_choice2(k,(l*2)
   -1)) is the category
    ! elecs%orbital_choice2(k,(l*2)-1) starting the index
    ! and elecs%orbital_choice2(k,l*2) the end index

    do j=elecs%orbital_choice2(k,(l*2)-1),elecs%
   orbital_choice2(k,l*2)
      ovrlp_vec(j)=ov
    end do
  end do
end do
! Final sum of all values to give Hamiltonian matrix element
do j=1,elecs%num
  ham_tot=ham_tot+(ovrlp_vec(j)*elecs%integrals(j))
end do
```

Listing B.8: Code used to calculate Hamiltonian matrix element.

B.3.2 Other Operators

As established, the observable properties of a Zombie state can be computed by iterative application of creation and annihilation operators [9]. Here similar algorithms to the lower scaling two-electron Hamiltonian algorithm are considered for commonly used properties such as the number of electrons and spin properties. These algorithms have been implemented in an additional Fortran module contained in a file *operators.f90* available with the other source code files.

B.3.2.1 Number Operator

The number operator is

$$\hat{N} = \sum_{i=1}^{N_{orb}} \hat{n}_i = \sum_{i=1}^{N_{orb}} \hat{b}_i^\dagger \hat{b}_i \tag{B.18}$$

where $\hat{b}_i$ is the annihilation and $\hat{b}_i^\dagger$ the creation operator for Zombie state i. So,

$$\hat{n}_i|\boldsymbol{\zeta}^{(b)}\rangle = \begin{bmatrix} a_{11}^{(b)} & a_{12}^{(b)} & \dots & a_{1(i-1)}^{(b)} & a_{1i}^{(b)} & \dots & a_{1N_{orb}}^{(b)} \\ a_{01}^{(b)} & a_{02}^{(b)} & \dots & a_{0(i-1)}^{(b)} & 0 & \dots & a_{0N_{orb}}^{(b)} \end{bmatrix} \tag{B.19}$$

$\hat{n}_i$ effectively "deletes" the coefficient $a_{0i}^{(b)}$ with the sign change being cancelled out. This means the number operator, $\hat{N}$ is $\mathcal{O}(N_{orb})$ because the summation is over N_{orb} terms. So overall computing the number of electrons represented by a Zombie state, $\langle\boldsymbol{\zeta}^{(a)}|\hat{N}|\boldsymbol{\zeta}^{(b)}\rangle$ would be $\mathcal{O}(N_{orb}^2)$ due to the number operator and then an overlap calculation which is $\mathcal{O}(N_{orb})$.

However, this scaling can be reduced to $\mathcal{O}(N_{orb})$ by adapting an algorithm from Ref. [178]. Explicitly the action of the number operator is.

$$\begin{aligned} \langle\boldsymbol{\zeta}^{(a)}|\hat{N}|\boldsymbol{\zeta}^{(b)}\rangle &= \sum_{l=1}^{N_{orb}} \langle\boldsymbol{\zeta}^{(a)}|\hat{n}_l|\boldsymbol{\zeta}^{(b)}\rangle \\ &= \sum_{l=1}^{N_{orb}} \left[\prod_{j=1}^{l-1} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)}\right] a_{1l}^{(a)*} a_{1l}^{(b)} \left[\prod_{j=l+1}^{N_{orb}} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)}\right]. \end{aligned} \tag{B.20}$$

Defining,

$$d_i = a_{1i}^{(a)*} a_{1i}^{(b)} \tag{B.21}$$

and then using Eq. (B.13c)

$$g_i = \prod_{i=1}^{m} f_i. \tag{B.22}$$

This can be used along with Eq. (B.14b) to give the recursion relations:

$$g_1 = f_1, \tag{B.23a}$$
$$g_l = g_{l-1} f_l, \quad l = 2, 3, \ldots N_{orb} \tag{B.23b}$$
$$h_{N_{orb}} = f_{N_{orb}}, \tag{B.23c}$$
$$h_m = h_{l+1} f_l, \quad l = N_{orb} - 1, N_{orb} - 2, \ldots, 1. \tag{B.23d}$$

Computation of Eq. (B.21), Eq. (B.13c) and $\{g_l\}$ and $\{h_m\}$, using the recursion relations, are all $\mathcal{O}(N_{orb})$. Putting these into Eq. (B.20),

$$\langle \boldsymbol{\zeta}^{(a)} | \hat{N} | \boldsymbol{\zeta}^{(b)} \rangle = \sum_{l=1}^{N_{orb}} g_{l-1} d_l h_{l+1} \tag{B.24}$$

which is a summation computed in $\mathcal{O}(N_{orb})$ steps. Application of the scaled algorithm shows a large improvement on the naïve algorithm. In Table. B.9 it is demonstrated that when increasing the number of orbitals by a factor of 10, the new algorithm is approximately a further 10 times faster than the naïve one which is to be expected.

Zombie states are usually not eigenstates of the number operator, unlike most Slater determinants, so the uncertainty in the in number of electrons can be defined as a standard deviation

$$\sigma_N = \sqrt{\langle \hat{N}^2 \rangle - \langle \hat{N} \rangle} \tag{B.25}$$

For $\hat{N}^2$,

$$\langle \zeta^{(a)} | \hat{N}^2 | \zeta^{(b)} \rangle = \sum_{k=1}^{N_{orb}} \langle \boldsymbol{\zeta}^{(a)} | \hat{N} \hat{n}_k | \boldsymbol{\zeta}^{(b)} \rangle. \tag{B.26}$$

$\hat{n}_k | \boldsymbol{\zeta}^{(b)} \rangle$ is then simply delete the "dead" coefficient for orbital k. So $| \boldsymbol{\zeta}_k^{(b)} \rangle = \hat{n}_k | \boldsymbol{\zeta}^{(b)} \rangle$ can be defined and computed

$$\langle \boldsymbol{\zeta}^{(a)} | \hat{N}^2 | \boldsymbol{\zeta}^{(b)} \rangle = \sum_{k=1}^{N_{orb}} \langle \boldsymbol{\zeta}^{(a)} | \hat{N} | \boldsymbol{\zeta}_k^{(b)} \rangle \tag{B.27}$$

using the scaled algorithm for the number operator, which means the overall $\hat{N}^2$ is $\mathcal{O}(N_{orb}^2)$ rather than $\mathcal{O}(N_{orb}^3)$.

A "Ghost" operator that counts the "dead" Zombie states rather than the "alive"

can be defined,

$$\hat{G} = \sum_{l=1}^{N_{orb}} \hat{s}_l := \sum_{l=1}^{N_{orb}} \hat{b}_l \hat{b}_l^\dagger. \tag{B.28}$$

The same scaling arguments as for $\hat{N}$ can then be made for $\hat{G}$.

Table B.9: Time (in seconds) to compute orbital occupancy of a Zombie state populated randomly generated "dead" and "alive" coefficients. The Naïve algorithm is significantly slower in absolute terms, with worse scaling.

Number of orbitals N_{orb}	Naïve algorithm	New Algorithm	Speedup
100	0.028711	0.00056004	51.27
1000	3.0294	0.0051531	587.87

B.3.2.2 Spin Operators

B.3.2.2.1 $\hat{S}_z$ operator

It is assumed that the Zombie states are only used in restricted calculations such that for all orbitals with spin $|\alpha\rangle$ (spin up, $m_s = +1/2$) there exists an orbital with the same spatial wave function but opposite spin component $|\beta\rangle$ (spin down, $m_s = -1/2$). Thus, for a system with N_{orb} spin orbitals, there will consequently be N_{spa} spatial orbitals where $N_{spa} = N_{orb}/2, N_{spa} \in \mathbb{N}$. Further it is stipulated that all up-spin orbitals have an odd index $i = 1, 3, \ldots N_{orb} - 1$ and so all spin orbitals with down spin are even indexed $i = 2, 4, \ldots, N_{orb}$. So, for the spatial orbitals indexed $j = 1, 2, \ldots, N_{spa}$ the $|\alpha\rangle$ spin orbital is $i = 2j - 1$ and the $|\beta\rangle$ spin orbital is $i = 2j$ for the jth spatial orbital. Using this numbering convention, in second quantization notation

$$\begin{aligned} \hat{S}_z =& \frac{1}{2} \sum_{j=1}^{N_{spa}} \hat{n}_{2j-1} - \hat{n}_{2j} \\ =& \frac{1}{2} \sum_{i=1}^{N_{orb}} (-1)^{i-1} \hat{n}_i. \end{aligned} \tag{B.29}$$

The optimised number operator can then be adapted by introducing a sign change rule which finds S_z in $\mathcal{O}(N_{orb})$ steps. Comparing the scaled algorithm to the naïve algorithm, which uses sequential creation and annihilation operations, the improved calculation time is clear – see Table. B.10.

Table B.10: Time (in seconds) to compute S_z for randomly generated Zombie states using the slow naïve and faster scaled $\hat{S}_z$ algorithm. The new algorithm clearly demonstrates better scaling and is faster in absolute terms.

Number of orbitals N_{orb}	Slow $\hat{S}_z$	Fast $\hat{S}_z$	Speedup
100	0.029760	0.00062376	47.71
1000	3.7123	0.0063864	581.28

B.3.2.2.2 Faster $\hat{S}_z^2$ computation

Calculation of $\hat{S}_z^2$ would be $\mathcal{O}(N_{orb}^3)$ if Eq. (B.29) is used with no modification,

$$\hat{S}_z^2 = \frac{1}{4}\sum_{k}^{N_{orb}/2}\sum_{j}^{N_{orb}/2}(\hat{n}_{2k-1}-\hat{n}_{2k})(\hat{n}_{2j-1}-\hat{n}_{2j}) = \frac{1}{4}\sum_{k}^{N_{orb}}\sum_{j}^{N_{orb}}\hat{n}_k\hat{n}_j(-1)^{j+k} \tag{B.30}$$

there are $\mathcal{O}(N_{orb})$ steps for the summation over k and the same for the summation over j, and then $\mathcal{O}(N_{orb})$ to evaluate the overlap. However, the scaled number operator algorithm can be used for $\hat{S}_z^2$,

$$\langle\boldsymbol{\zeta}^{(a)}|\hat{S}_z^2|\boldsymbol{\zeta}^{(b)}\rangle = \frac{1}{2}\sum_{i}^{N_{orb}}\langle\boldsymbol{\zeta}^{(a)}|\hat{S}_z\hat{n}_i|\boldsymbol{\zeta}^{(b)}\rangle(-1)^i. \tag{B.31}$$

$\hat{n}_i|\boldsymbol{\zeta}^{(b)}\rangle$ gives another Zombie state denoted $|\boldsymbol{\zeta}^{(b,i)}\rangle$ so,

$$\langle\boldsymbol{\zeta}^{(a)}|\hat{S}_z^2|\boldsymbol{\zeta}^{(b)}\rangle = \frac{1}{2}\sum_{i}^{N_{orb}}\langle\boldsymbol{\zeta}^{(a)}|\hat{S}_z|\boldsymbol{\zeta}^{(b,i)}\rangle(-1)^i. \tag{B.32}$$

The bra-ket due to the operations can be evaluated in $\mathcal{O}(N_{orb})$ followed by an overlap making the $\hat{S}_z^2$ operator calculation $\mathcal{O}(N_{orb}^2)$.

Table B.11: Time (in seconds) to compute S_z^2 of random Zombie states using the unscaled and faster scaled $\hat{S}_z^2$ algorithm. The new algorithm clearly demonstrates better scaling and is faster in absolute terms.

Number of orbitals N_{orb}	Slow $\hat{S}_z^2$	Fast $\hat{S}_z^2$	Speedup
100	6.6990	0.23865	28.07
1000	3554.8	6.8003	522.74

B.3.2.2.3 Total spin

Often it is useful to calculate the total spin [36],

$$\begin{aligned}\hat{S}^2 =& \hat{S}_x^2+\hat{S}_y^2+\hat{S}_z^2\\ =& \hat{S}_+\hat{S}_- - \hat{S}_z+\hat{S}_z^2\end{aligned} \tag{B.33}$$

The faster $\hat{S}_z$ and $\hat{S}_z^2$ algorithms as detailed above. $\hat{S}_+$ and $\hat{S}_-$ are raising and lowering operators,

$$\hat{S}_+ = \sum_{k=1}^{N_{spa}}\hat{s}_{+,k} \tag{B.34a}$$

$$\hat{S}_- = \sum_{k=1}^{N_{spa}}\hat{s}_{-,k} \tag{B.34b}$$

$$\hat{s}_{+,k} = \hat{b}_{2k-1}^\dagger\hat{b}_{2k} \tag{B.34c}$$

$$\hat{s}_{-,k} = \hat{b}_{2k}^\dagger\hat{b}_{2k-1} \tag{B.34d}$$

where N_{spa} is the number of spatial orbitals with the kth spatial orbital containing $|\alpha\rangle$ spin orbital number $2k-1$ and $|\beta\rangle$ spin orbital number $2k$. $\hat{s}_{+,k}$ and $\hat{s}_{-,k}$ acting on a Zombie state (where $j=2k$)

$$\begin{aligned}\hat{s}_{+,k}|\boldsymbol{\zeta}^{(b)}\rangle &= \hat{b}^{\dagger}_{2k-1}\hat{b}_{2k}\begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & a^{(b)}_{1(j-1)} & a^{(b)}_{1j} & a^{(b)}_{1(j+1)} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{0(j-1)} & a^{(b)}_{0j} & a^{(b)}_{0(j+1)} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix} \\ &= \hat{b}^{\dagger}_{2k-1}\begin{bmatrix} -a^{(b)}_{11} & -a^{(b)}_{12} & \dots & -a^{(b)}_{1(j-1)} & 0 & a^{(b)}_{1(j+1)} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{0(j-1)} & a^{(b)}_{1j} & a^{(b)}_{0(j+1)} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix} \\ &= \begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & a^{(b)}_{0(j-1)} & 0 & a^{(b)}_{1(j+1)} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & 0 & a^{(b)}_{1j} & a^{(b)}_{0(j+1)} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix}\end{aligned} \tag{B.35}$$

and

$$\begin{aligned}\hat{s}_{-,k}|\boldsymbol{\zeta}^{(b)}\rangle &= \hat{b}^{\dagger}_{2k}\hat{b}_{2k-1}\begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & a^{(b)}_{1(j-1)} & a^{(b)}_{1j} & a^{(b)}_{1(j+1)} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{0(j-1)} & a^{(b)}_{0j} & a^{(b)}_{0(j+1)} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix} \\ &= \hat{b}^{\dagger}_{2k}\begin{bmatrix} -a^{(b)}_{11} & -a^{(b)}_{12} & \dots & 0 & a^{(b)}_{1j} & a^{(b)}_{1(j+1)} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{1(j-1)} & a^{(b)}_{0j} & a^{(b)}_{0(j+1)} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix} \\ &= \begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & 0 & a^{(b)}_{0j} & a^{(b)}_{1(j+1)} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{1(j-1)} & 0 & a^{(b)}_{0(j+1)} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix}\end{aligned} \tag{B.36}$$

The sign change in both cases exactly cancels so can be ignored when evaluating $\hat{S}_+$ or $\hat{S}_-$. The scaling of the algorithm can also be reduced by first defining $\hat{s}_{-,k}|\boldsymbol{\zeta}^{(b)}\rangle = |\boldsymbol{\zeta}^{(c)}_k\rangle$ which has $\mathcal{O}(N_{orb})$ scaling. So,

$$\begin{aligned}\langle\boldsymbol{\zeta}^{(a)}|\hat{S}_+\hat{S}_-|\boldsymbol{\zeta}^{(b)}\rangle &= \sum_{k=1}^{N_{spa}}\langle\boldsymbol{\zeta}^{(a)}|\hat{S}_+|\boldsymbol{\zeta}^{(c)}_k\rangle = \sum_{k=1}^{N_{spa}}\sum_{l=1}^{K}\langle\boldsymbol{\zeta}^{(a)}|\hat{s}_{+,m}|\boldsymbol{\zeta}^{(c)}_k\rangle \\ &= \sum_{k=1}^{N_{spa}}\left[\sum_{l=1}^{K}\left[\prod_{j=1}^{l-2} a^{(a)*}_{1j}a^{(c)}_{1j} + a^{(a)*}_{0j}a^{(c)}_{0j}\right]\cdot a^{(a)*}_{0(l-1)}a^{(c)}_{1(l-1)}\cdot a^{(a)*}_{1l}a^{(c)}_{0l}\cdot \right. \\ &\quad \left.\left[\prod_{j=l+1}^{N_{orb}} a^{(a)*}_{1j}a^{(c)}_{1j} + a^{(a)*}_{0j}a^{(c)}_{0j}\right]\right]\end{aligned} \tag{B.37}$$

Using a similar reasoning to the scaled Hamiltonian with, Eq. (B.13c) and the number operator with Eq. (B.22) and Eq. (B.21) the following functions can be defined,

$$f_i = (a^{(a)*}_{1i-1} a^{(c)}_{1i-1} + a^{(a)*}_{0i-1} a^{(c)}_{0i-1}) \cdot (a^{(a)*}_{1i} a^{(c)}_{1i} + a^{(a)*}_{0i} a^{(c)}_{0i}) \tag{B.38a}$$

$$g_l = \prod_{i=1}^{l} f_i \tag{B.38b}$$

$$h_l = \prod_{i=l}^{K} f_i \tag{B.38c}$$

$$d_i = a^{(a)*}_{0(i-1)} a^{(c)}_{1(i-1)} a^{(a)*}_{1i} a^{(c)}_{0i} \tag{B.38d}$$

Which gives

$$\langle \boldsymbol{\zeta}^{(a)} | \hat{s}_{+l} | \boldsymbol{\zeta}^{(c)}_k \rangle = g_{l-2} d_l h_{l+1} \tag{B.39}$$

Thus overall, the scaling is reduced from $\mathcal{O}(N^3_{orb})$ to $\mathcal{O}(N^2_{orb})$. Application of this new algorithm compared to the naïve algorithm which uses the no sign change rule are shown in Table. B.12.

However, it is possible to construct an algorithm for $\langle \boldsymbol{\zeta}^{(a)} | \hat{s}_+ \hat{s}_- | \boldsymbol{\zeta}^{(b)} \rangle$ eliminating the need to calculate $|\boldsymbol{\zeta}^{(c)}\rangle$. By setting $m = 2l$ and $n = 2k$,

$$\hat{s}_{+,l} \hat{s}_{-,k} | \boldsymbol{\zeta}^{(b)} \rangle = \hat{b}^\dagger_{2l-1} \hat{b}_{2l} \hat{b}^\dagger_{2k} \hat{b}_{2k-1} \begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & a^{(b)}_{1m} & \dots & a^{(b)}_{1n} & \dots & a^{(b)}_{1M} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{0m} & \dots & a^{(b)}_{0n} & \dots & a^{(b)}_{0M} \end{bmatrix} \tag{B.40}$$

which can have the following outcomes

$$\hat{s}_{+,l} \hat{s}_{-,k} | \boldsymbol{\zeta}^{(b)} \rangle = \begin{cases} \begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & 0 & a^{(b)}_{0n} & \dots & a^{(b)}_{0(m-1)} & 0 & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{1(n-1)} & 0 & \dots & 0 & a^{(b)}_{1m} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix} & 2k < 2l \\ \begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & a^{(b)}_{0(m-1)} & 0 & \dots & 0 & a^{(b)}_{0n} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & 0 & a^{(b)}_{1m} & \dots & a^{(b)}_{1(n-1)} & 0 & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix} & 2k > 2l \\ \begin{bmatrix} a^{(b)}_{11} & a^{(b)}_{12} & \dots & a^{(b)}_{1(m-2)} & a_{1(m-1)} & 0 & a^{(b)}_{1(m+1)} & \dots & a^{(b)}_{1N_{orb}} \\ a^{(b)}_{01} & a^{(b)}_{02} & \dots & a^{(b)}_{0(m-2)} & 0 & a_{0m} & a^{(b)}_{0(m+1)} & \dots & a^{(b)}_{0N_{orb}} \end{bmatrix} & 2k = 2l \end{cases} \tag{B.41}$$

which gives

$$\langle \boldsymbol{\zeta}^{(a)} | \hat{s}_{+,l} \hat{s}_{-,k} | \boldsymbol{\zeta}^{(b)} \rangle = \begin{cases} \left[\prod_{j=1}^{k-2} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right] a_{0(k-1)}^{(a)*} a_{1(k-1)}^{(b)} a_{1k}^{(a)*} a_{0k}^{(b)} \cdot \\ \left[\prod_{j=k+1}^{l-2} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right] a_{1(l-1)}^{(a)*} a_{0(l-1)}^{(b)} a_{0l}^{(a)*} a_{1l}^{(b)} \cdot \\ \left[\prod_{j=l+1}^{M} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right], \quad 2k < 2l \\ \\ \left[\prod_{j=1}^{l-2} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right] a_{1(l-1)}^{(a)*} a_{0(l-1)}^{(b)} a_{0l}^{(a)*} a_{1l}^{(b)} \cdot \\ \left[\prod_{j=l+1}^{k-2} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right] a_{0(k-1)}^{(a)*} a_{1(k-1)}^{(b)} a_{1k}^{(a)*} a_{0k}^{(b)} \cdot \\ \left[\prod_{j=k+1}^{M} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right], \quad 2k > 2l \\ \\ \left[\prod_{j=1}^{l-2} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right] a_{1(l-1)}^{(a)*} a_{1(l-1)}^{(b)} a_{0l}^{(a)*} a_{0l}^{(b)} \cdot \\ \left[\prod_{j=l+1}^{M} a_{1j}^{(a)*} a_{1j}^{(b)} + a_{0j}^{(a)*} a_{0j}^{(b)} \right], \quad 2k = 2l \end{cases} \tag{B.42}$$

The recursion relations are then defined,

$$f_i = (a_{1(i-1)}^{(a)*} a_{1(i-1)}^{(b)} + a_{0(i-1)}^{(a)*} a_{0(i-1)}^{(b)}) \cdot (a_{1(i)}^{(a)*} a_{1(i)}^{(b)} + a_{0(i)}^{(a)*} a_{0(i)}^{(b)}) \tag{B.43a}$$

$$c_i = a_{0(i-1)}^{(a)*} a_{1(i-1)}^{(b)} \cdot a_{1i}^{(a)*} a_{0i}^{(b)} \tag{B.43b}$$

$$d_i = a_{1(i-1)}^{(a)*} a_{0(i-1)}^{(b)} \cdot a_{0i}^{(a)*} a_{1i}^{(b)} \tag{B.43c}$$

$$s_i = a_{1(i-1)}^{(a)*} a_{1(i-1)}^{(b)} a_{0i}^{(a)*} a_{0i}^{(b)} \tag{B.43d}$$

which are calculated in $\mathcal{O}(N_{orb})$ steps. Then the following functions are defined,

$$g_l = \prod_{i=1}^{l} f_i \tag{B.44a}$$

$$h_l = \prod_{i=l}^{K} f_i \tag{B.44b}$$

$$t_{(l,p)} = \prod_{i=l}^{p} f_i \tag{B.44c}$$

Eq. (B.44a) and Eq. (B.44b) are then calculated recursively in $\mathcal{O}(N_{orb})$ steps and

Eq. (B.44c) in $\mathcal{O}(N_{orb}^2)$. Therefore,

$$\langle \boldsymbol{\zeta}^{(a)} | \hat{S}_+ \hat{S}_- | \boldsymbol{\zeta}^{(b)} \rangle = \sum_{k=1}^{N_{spa}-1} \sum_{l=k+1}^{N_{spa}} g_{k-1} c_k t_{(k+1,l-1)} d_l h_{l+1} + \sum_{l=1}^{N_{spa}-1} \sum_{k=l+1}^{N_{spa}} g_{l-1} d_l t_{(l+1,k-1)} c_k h_{k+1} + \sum_{l=1}^{N_{spa}} g_{l-1} s_l h_{l+1} \tag{B.45}$$

The speed up this algorithm provides compared to the naïve algorithm can be seen in the final column of Table. B.12.

Table B.12: Time (in seconds) to compute $\langle \zeta^{(a)} | \hat{S}_+ \hat{S}_- | \zeta^{(b)} \rangle$ of random populated Zombie states. The naïve algorithm uses the no sign change rule for $\hat{S}_+$ or $\hat{S}_-$. The scale reduced algorithm requires the calculation of $|\zeta^{(c)}\rangle$. The prefactor reduction directly computes $\hat{S}_+$ and $\hat{S}_-$ acting on $|\zeta^{(b)}\rangle$. The speedups are between the scaled algorithm when compared to the naïve algorithm. Both scaled algorithms show a real time improvement but by reducing the prefactor as well as the scaling doubles the improvement on just reducing the scaling.

Number of orbitals N_{orb}	Naïve	Scale reduction	Prefactor reduction	Speedups Naïve with Scale reduction	Speedups Naïve with Prefactor reduction
100	0.85121	0.055180	0.025146	15.43	33.85
1000	792.20	5.2828	2.3451	149.96	337.81

So, the naïve total spin algorithm is made up of three separate algorithms $\hat{S}_+\hat{S}_-$, $\hat{S}_z$, $\hat{S}_z^2$ that scaled $\mathcal{O}(N_{orb}^3)$, $\mathcal{O}(N_{orb}^2)$ and $\mathcal{O}(N_{orb}^3)$ respectively. The reduced scaling algorithms by contrast have $\mathcal{O}(N_{orb}^2)$, $\mathcal{O}(N_{orb})$ and $\mathcal{O}(N_{orb}^2)$ scaling respectively. Comparing the two sets of algorithms, in Table. B.13, demonstrates that $\hat{S}^2$ calculation for 1000 orbitals is over 460 times faster with the scaled algorithms compared to the naïve algorithm.

Table B.13: Time (in seconds) to compute total spin of randomly generated Zombie states. The naïve algorithm uses none of the scaled algorithms whereas the new algorithm uses the scaled $\hat{S}_z^2$ and the fastest algorithm to calculate $\langle \zeta^{(a)} | \hat{S}_+ \hat{S}_- | \zeta^{(b)} \rangle$. The new algorithm is considerably faster than the original implementation and shows much better scaling.

Number of orbitals N_{orb}	Naïve	Scaled	Speedup
100	6.4861	0.092060	70.45
1000	4106.4	8.77818	467.60

B.4 Gradient Descent Algorithm

B.4.1 Derivatives of the Overlap Matrix

As shown in section 4.4.1 it is possible to calculate the gradient of the energy function indirectly. The only derivative that needs to be calculated is that of the overlap matrix which is not computationally expensive. The "dead" and "alive" coefficients of a Zombie state for orbital j are $cos(\theta_j)$ and $sin(\theta_j)$ respectively. First it is trivially noted that $\frac{\partial \Omega_{ab}}{\partial \zeta^{(c)}_{\theta_j}} \neq 0$ only if $c = a$, or b. As the derivative is being calculated with respect to a specific spin orbital only that orbital needs to be considered and the rest of the overlap can be considered a scalar which can be denoted Ω^j_{ab}. So, if $\zeta^{(c)} = \zeta^{(a)} = \zeta^{(b)}$ then the derivative of the overlap for orbital j is,

$$\frac{\partial \Omega_{cc}}{\partial \zeta^{(c)}_{\theta_j}} = \Omega^j_{cc} \frac{(sin(\theta^{(c)}_j)sin(\theta^{(c)}_j) + cos(\theta^{(c)}_j)cos(\theta^{(c)}_j))}{\partial \zeta^{(c)}_{\theta_j}} = \Omega^j_{ab} \cdot (sin(2\theta^{(c)}_j) - sin(2\theta^{(c)}_j) = 0. \tag{B.46}$$

Therefore, the only non-zero derivative is calculated,

$$\begin{aligned} \frac{\partial \Omega_{ab}}{\partial \zeta^{(c)}_{\theta_j}} &= \Omega^j_{ab} \frac{(sin(\theta^{(a)}_j)sin(\theta^{(b)}_j) + cos(\theta^{(a)}_j)cos(\theta^{(b)}_j))}{\partial \zeta^{(c)}_{\theta_j}} \\ &= \Omega^j_{cb} \cdot (cos(\theta^{(c)}_j)sin(\theta^{(b)}_j) - sin(\theta^{(c)}_j)cos(\theta^{(b)}_j)) \quad \zeta^{(c)} = \zeta^{(a)}, \zeta^{(a)} \neq \zeta^{(b)}. \end{aligned} \tag{B.47}$$

These values can the calculated in a similar way to the Hamiltonian matrix by dividing the overlap by the contribution to the multiplication for orbital j and then multiplying by the orbital derivative. These results are summarised,

$$\frac{\partial \Omega_{ab}}{\partial \zeta^{(c)}_{\theta_j}} = \begin{cases} 0 & \zeta^{(c)} \neq \zeta^{(a)}, \zeta^{(b)} \\ 0 & \zeta^{(c)} = \zeta^{(a)} = \zeta^{(b)} \\ \Omega_{ab} \cdot \frac{cos(\theta^{(c)}_j)sin(\theta^{(b)}_j) - sin(\theta^{(c)}_j)cos(\theta^{(b)}_j)}{sin(\theta^{(c)}_j)sin(\theta^{(b)}_j) + cos(\theta^{(c)}_j)cos(\theta^{(b)}_j)} & \zeta^{(c)} = \zeta^{(a)} \neq \zeta^{(b)} \end{cases} \tag{B.48}$$

B.4.2 Gradient Calculation Code

To carry out the Gradient Descent process, a type is defined to contain all of the relevant variables which is shown Listing. B.9. The variable *prev_erg* stores the energy from the previous gradient step which can be compared to the newly calculated energy and updated if the change is accepted. *vars* and *grad_avlb* are $N_{orb} \times N_{bf}$ sized arrays ordered to allow faster access to gradients within a Zombie state. *vars* store the value of the gradient for each orbital in each Zombie state, *grad_avlb* takes

either 1 if the gradient has already been calculated or a 0 if it needs calculating. The availability array stops the program wasting time recalculating values already in memory.

```
type grad
  real(wp):: prev_erg
  real(wp),dimension(:,:), allocatable::vars
  integer,dimension(:,:),allocatable::grad_avlb
  real(wp),dimension(:,:,:),allocatable::ovrlp_grad
  integer,dimension(:,:,:),allocatable::ovrlp_grad_avlb
end type grad
```

Listing B.9: Definition of the "grad" data type.

ovrlp_grad and *ovrlp_grad_avlb* are $N_{orb} \times N_{bf} \times N_{bf}$ sized arrays used to store the values and availability of the partial derivatives of the overlap matrix. The variables are set up, so the first index gives the orbital, $theta_j$, the partial derivative is found with respect to; the second index is the other ZS in the overlap calculation and the final index is the Zombie state containing the orbital the derivative is being calculated for, $\zeta^{(c)}_{\theta_j}$. Thus $\frac{\partial \Omega_{cb}}{\zeta^{(c)}_{\theta_j}}$ is stored,

```
ovrlp_grad(j,b,c)
```

Listing B.10: Demonstration of indexing convention in *ovrlp_grad* and *ovrlp_grad_avlb*. Shown is the index for the partial derivative with respect to orbital j in Zombie state $\zeta^{(c)}$ of the overlap between ZSs b and c, $\langle \boldsymbol{\zeta}^{(c)} | \boldsymbol{\zeta}^{(b)} \rangle$.

To calculate the gradient the equation Eq. (4.21) is used which is

$$\frac{\partial E}{\partial \zeta^{(c)}_{\theta_j}} = \frac{-\boldsymbol{cHd}}{(\sqrt{|\boldsymbol{c}^*\boldsymbol{\Omega c}|})^3} \frac{\boldsymbol{c}^*\Omega\boldsymbol{c}}{|\boldsymbol{c}^*\boldsymbol{\Omega c}|} \frac{\boldsymbol{c}^*\partial(\boldsymbol{\Omega})\boldsymbol{c}}{\partial \zeta^{(c)}_{\theta_j}} = sgn(-\boldsymbol{c}^*\boldsymbol{\Omega c}) \cdot \frac{\boldsymbol{cHd}}{||\boldsymbol{d}||} \frac{\boldsymbol{c}^*\partial(\boldsymbol{\Omega})\boldsymbol{c}}{\partial \zeta^{(c)}_{\theta_j}}. \tag{B.49}$$

where $\boldsymbol{c}$ is the unnormalised ZS coefficient vector which is denoted $\boldsymbol{d}$ once normalised and $||\boldsymbol{d}||$ is its norm. The program using the derived type to contain the ZS coefficient vector values d is $\boldsymbol{d}$; *d_1* is the unnormalised vector, $\boldsymbol{c}$; *norm* is $\sqrt{|\boldsymbol{c}^*\Omega\boldsymbol{c}|}$) and *d_o_d* is either 1 or -1 depending on $\boldsymbol{c}^*\Omega\boldsymbol{c}$ being positive or negative.

```
type dvector
  real(wp), dimension(:), allocatable::d
  real(wp), dimension(:), allocatable::d_1
  real(wp), dimension(:,:),allocatable::d_gs
  real(wp):: norm
  integer(int8):: d_o_d
end type dvector
```

Listing B.11: Definition of the "dvector" data type.

The process to calculate a partial derivative is shown in Listing. B.12 for a single spin orbital. The term in Eq. (B.49) containing the Hamiltonian is independent of the partial derivative being calculated and so only needs to be calculated once. At the end of the imaginary time process the program stores all values in the instance of the *dvector* type, *dvec*. The Hamiltonian term in Eq. (B.49) is calculated using the library function *DGEMV* and stored in the variable *ham_c_d*. The partial derivatives are then calculated using Eq. (B.48). The derivative of the overlap matrix will be zero except in row and column corresponding to Zombie state $|\boldsymbol{\zeta}^{(c)}\rangle$. So, when calculating $\partial(\boldsymbol{\Omega})\boldsymbol{c}$ the resultant vector has elements $\Omega_{ca}\cdot\boldsymbol{c}^{(a)}$ except for element c which is $\sum_a \Omega_{ac}\cdot\boldsymbol{c}^{(a)}$ hence the need for line 13 in Listing. B.12

```
1  if(grad_fin%grad_avlb(orb,pick)==0)then
2    call DGEMV("N",ndet,ndet,1.d0,haml%hjk,ndet,dvecs%d,1,0.d0
     ,temp,1)
3    ham_c_d=(dvecs%d_o_d/(dvecs%norm*dvecs%norm*dvecs%norm))*
     dot_product(temp,dvecs%d_1)
4    do j=1,ndet
5      if(grad_fin%ovrlp_grad_avlb(orb,j,pick).eq.0)then
6        grad_fin%ovrlp_grad(orb,j,pick)=haml%ovrlp(j,pick)*&
7          (zstore(j)%val(orb)*zstore(pick)%val(orb+norb)-
     zstore(j)%val(orb+norb)*zstore(pick)%val(orb))/&
8          (zstore(j)%val(orb)*zstore(pick)%val(orb)+zstore(j)%
     val(orb+norb)*zstore(pick)%val(orb+norb))
9        grad_fin%ovrlp_grad_avlb(orb,j,pick)=1
10     end if
11   end do
12   temp=grad_fin%ovrlp_grad(orb,:,pick)*dvecs%d_1
13   temp(pick)=dot_product(grad_fin%ovrlp_grad(orb,:,pick),
     dvecs%d_1)
14   grad_fin%vars(pick,orb)=dot_product(temp,dvecs%d_1)*
     ham_c_d
15   grad_fin%grad_avlb(orb,pick)=1
16 else
```

Listing B.12: Code used to find the gradient of the energy function with respect to a ZS orbital. The chosen Zombie state is held in variable "pick" and the chosen orbital is stored in variable "orb".

B.4.3 The Algorithm

The algorithm implemented in the code follows the process laid out in Fig. 4.8. The process starts by generating a random order for the Zombie states which is stored in a variable called *picker*. The learning rate for the epoch is then set using the code in Listing. B.13.

```
1  t=lr*(lr_alpha**lralt_zs)
```

Listing B.13: Code used to set the learning rate, *t*. The variable *lr* is the initial learning rate, *lr_alpha* is the learning rate reduction parameter and *lralt_zs* learning rate scheduling parameter.

The algorithm then iterates through each orbital within each Zombie state calculating the gradient, using the code in Listing. B.12, to calculate a new set of "dead" and "alive" amplitudes; recalculating the Hamiltonian and finding a ground state energy. The updated ZS amplitudes are accepted if the ground state energy is reduced. Once the epoch is complete the code in Listing. B.14 is used to advance the learning rate and check if a sufficiently large number of ZSs have been altered at that learning rate. The variable *acpt_cnt_2* is used to store the number of Zombie states altered in that epoch. *lralt_extra* is initially set to 0 and is used to remove larger learning rates from the schedule when an insufficient numbers of Zombie states are altered in an epoch. The variables *tracker* and *extra_flag* are used to ensure the learning rates are removed in size order and this can only happen once per learning rate cycle. The second *if* statement resets the learning rate at the end of a cycle which also sets *extra_flag* back to zero making it possible for a learning rate to be removed again. A new random order of ZSs is also generated, updating the variable *picker* using the function *scramble*.

The variable *tracker* is used to control increases to the learning rate schedule and conditional cloning events. *chng_chng* is used to instigate cloning or changes to the learning rate schedule after a fixed number of epochs have occurred. If cloning is possible the Zombie basis set is increased by a predefined number of ZSs. If the maximum basis set size has been reached or cloning has not been specified the program attempts to allow a lower learning rate by increasing the value of *lr_loop_max*. If the minimum learning rate has already been reached the cycle is reset making all learning rates available again. The variable *tracker* is then set to -1 which ensures all Zombie states are attempted to be altered at all possible learning rates at least twice. The algorithm then returns to the beginning of the loop setting a new learning rate and proceeding as before. The loop stops when the maximum number of epochs is reached or the number of consecutive epochs with no change to the energy is more than value of $50 \times N_{bf}$.

```
if((acpt_cnt_2.lt.(0.15*ndet)).and.(extra_flag.eq.0).and.(
   lralt_zs.eq.lralt_extra).and.(tracker.gt.-1))then
  lralt_extra=lralt_extra+1
  extra_flag=1
end if

lralt_zs=lralt_zs+1
chng_chng=chng_chng-1
if(lralt_zs.gt.lr_loop_max)then
  picker=scramble(ndet-1)
  lralt_zs=lralt_extra
  extra_flag=0
  if((acpt_cnt_2.lt.((ndet)/3)).or.((ndet.gt.5).and.(
   acpt_cnt_2.lt.3)).or.(tracker.lt.0))then
    tracker=tracker+1
  end if
end if

if((((tracker.ge.1).and.(lralt_extra.gt.lr_loop_max-2)).or.(
   chng_chng.le.0)))then
  if(ndet.lt.ndet_max)then
    ! Cloning code
  else if(lr_loop_max.lt.min_clone_lr)then
    ! Increase learning rate schedule code
  else
    ! Reset learning rate
  end if
end if
```

Listing B.14: Code used to advance the learning rate and check if a sufficient number of Zombie states have been altered.

```
1  dvecs%d_1=0.0d0
2  dvecs%d_1(1)=1.0d0
3  dvecs%d_gs=0.0d0
4  do k=1, gramnum
5    dvecs%d_gs(k,k+1)=1.0d0
6  end do
7  call gs_dvector(dvecs,haml%ovrlp)
8  db=beta/timesteps
9
10 do k=1,timesteps+1
11   call DGEMV("N",size,size,1.d0,haml%ovrlp,size,dvecs%d_1
       ,1,0.d0,temp,1)
12   norm=dot_product(temp,dvecs%d_1)
13   dvecs%norm = sqrt(abs(norm))
14   dvecs%d=dvecs%d_1/dvecs%norm
15   do g=1, gramnum
16     call DGEMV("N",size,size,1.d0,haml%ovrlp,size,dvecs%d_gs
      (g,:),1,0.d0,temp,1)
17     norm=dot_product(temp,dvecs%d_gs(g,:))
18     dvecs%d_gs(g,:)=dvecs%d_gs(g,:)/sqrt(abs(norm))
19   end do
20
21   call DGEMV("N",size,size,1.d0,haml%hjk,size,dvecs%d,1,0.d0
      ,temp,1)
22   erg(1,k)=dot_product(temp,dvecs%d)
23   do g=1, gramnum
24     call DGEMV("N",size,size,1.d0,haml%hjk,size,dvecs%d_gs(g
      ,:),1,0.d0,temp,1)
25     erg(g+1,k)=dot_product(temp,dvecs%d_gs(g,:))
26   end do
27
28   call DGEMV("N",size,size,db,haml%kinvh,size,dvecs%d,1,0.d0
      ,ddot,1)
29   dvecs%d_1=dvecs%d-ddot
30   do g=1, gramnum
31     call DGEMV("N",size,size,db,haml%kinvh,size,dvecs%d_gs(g
      ,:),1,0.d0,ddot,1)
32     dvecs%d_gs(g,:)=dvecs%d_gs(g,:)-ddot
33   end do
34   call gs_dvector(dvecs,haml%ovrlp)
35 end do
36
37 call DGEMV("N",size,size,1.d0,haml%ovrlp,size,dvecs%d_1,1,0.
     d0,temp,1)
38 norm=dot_product(temp,dvecs%d_1)
39 dvecs%d_o_d=sign_d_o_d(norm)
40 dvecs%norm = sqrt(abs(norm))
41 dvecs%d=dvecs%d_1/dvecs%norm
```

Listing B.15: Code for imaginary time propagation which makes uses of the BLAS routine DGEMV to complete some of the matrix vector multiplications. Gram-Schmidt orthogonalisation is also shown but can be omitted by removing lines 3-7, 15-19, 23-26 and 30-34.

B.5 Imaginary Time Propagation

The algebraic details of imaginary time propagation are given in Chapter 3. The code used for imaginary time propagation is shown in Listing. B.15 including the Gram-Schmidt orthogonalisation process. The BLAS routine *DGEMV* is used to multiply the unnormalised $\boldsymbol{d}$ vector with the overlap matrix and placing it in the variable *temp* before the dot product is used to find the norm. *DGEMV* is also used when calculating the energy using the Hamiltonian matrix. The $\boldsymbol{d}$ vectors for higher energy states are all stored in the variable *dvecs%d_gs* which is an array of size $N_{es} \times N_{bf}$ where N_{es} is the number of excited states. Imaginary time steps and energies are calculated in the same ways for the ground state except with the additional orthogonalisation process. The Zombie states program uses a version of this subroutine that omits the GSO sections, and another used during gradient descent which only calculates the final energy rather than storing the values in the array *erg* so they can be outputted.

B.6 Parallelisation

Completing computing tasks one after another can result in very slow computation times. Parallelisation of code allows certain tasks to be completed concurrently making full use of available computing resources. Modern personal computers using a single processor still have multiple cores to complete tasks on and high-performance cluster computers contain multiple processors each with many cores. To allow simultaneous execution of code there are two approaches defined by the use of either a shared or distributed memory system. A shared memory approach such as OpenMP allows sections of code to run on multiple threads all accessing the same shared memory. Whereas the distributed memory approach, such as MPI splits processes across multiple processors each with their own shared memory.

OpenMP or Open Multi-Processing is a standard approach for adding parallel execution to program without having to make substantial alterations to existing serial code. Where an OpenMP parallel section is created formerly serial code is forked and executed independently before being joined back together, this process is illustrated in Fig. B.3. Since, a shared memory architecture is being used each parallel thread can access the same set of memory addresses without the need for message passing. This allows parallel threads to access the same variables with little additional overhead. On the other hand, Message Passing Interface or MPI uses multiple separate processes to complete tasks concurrently. Usually, a main processor controls the overall process and then moves different sets of tasks to different processors. Each processor completes its set of tasks and returns a result.

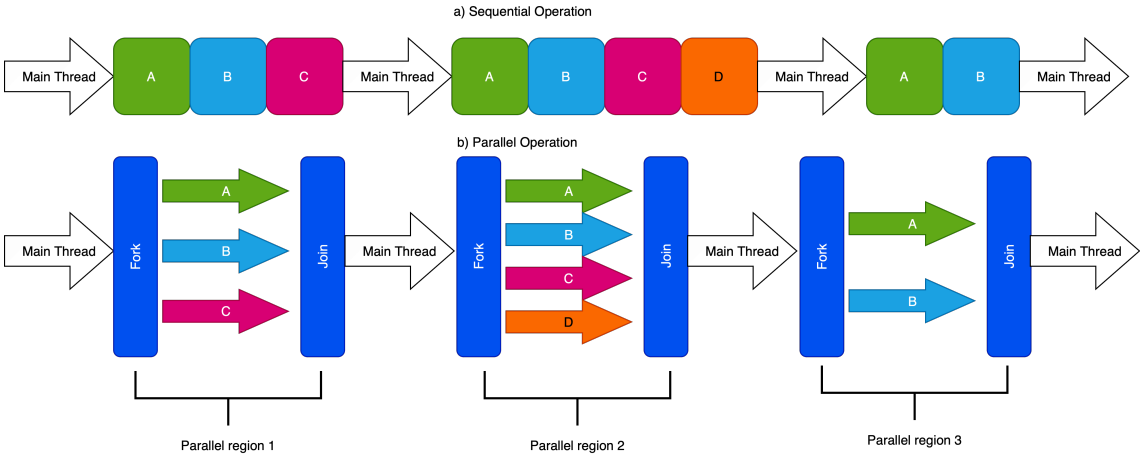


Figure B.3: Illustration of a sequential program (a) and a program that contains parallel regions (b) created by the fork/join function in OpenMP.

This differs from OpenMP as message passing is needed for each processor to be able to communicate with each other. Though there is a cost to move information between processors the memory limit of a single processor's memory is removed, and further parallelisation is then possible on each processor. Thus, MPI is useful for large systems where memory requirements are a concern.

Both OpenMP and MPI are CPU based approaches to adding parallelism to code but increasingly GPUs are being utilised to greatly reduce computation times. GPUs contain a considerably larger number of computational cores than a CPU allowing far more computations to be computed concurrently. GPUs work on a single instruction multiple data SIMD model which means the exact same type of calculation is completed on different data. Further, the type of tasks each thread on a GPU can complete is generally less complicated than on a CPU. A single OpenMP thread could complete multiple parts of a calculation whereas a single GPU thread would be limited to a single operation for example adding two values together. Like with MPI there is an overhead passing information from the CPU cached memory to the CPU. Therefore, using a GPU to complete concurrent tasks is only appropriate in certain circumstances. With a large system that requires multiple, simple, operations of the same type using a GPU can greatly reduce computation times.

B.6.1 Parallel Code in the Zombie states Program

The Zombie states program currently uses OpenMP parallel directives where possible to reduce overall computation times. This approach was chosen as the places where parallelisation is possible will benefit from having access to a shared memory. Further, it is relatively simple to add OpenMP directives to serial code with little rewriting necessary. OpenMP works by using directives that enclose a parallel region

```
!$omp parallel do  default(none) &
!$omp & private(j) &
!$omp & shared(elecs,zstore,temp,row,norb,orb,size)
do j=1,size

  !Code here

end do
!$omp end parallel do
```

Listing B.16: Example of an OpenMP parallel directive to parallelise a do loop. Notice the use of shared and private variables.

instructing the compiler to interpret the following code in a specific way. In Fortran the directives are always preceded by *!$omp* and so a parallel region is always enclosed by *!$omp parallel* and *!$omp end parallel*. If a parallel region only encloses a loop it is possible to combine the usual parallel directive with *!$omp do* –used to split loop iterations onto different threads. In Listing. B.16 an example of how a do loop is parallelised is shown the *!$omp parallel do* opens a parallel environment and tells the compiler to complete each iteration of the loop on a different thread. The variables are then classified as either *private*, so each thread has its own copy of a variable or *shared* meaning each thread accesses the same memory location for these variables. The *!$omp critical* directive is used so only a single thread can access code within the region at a time. The UCL random number generator is not entirely thread safe so when utilised in the MCE program subroutines and functions that required random number generation are enclosed in the *!$omp critical* directive [175]. In the modified version of the code employed in the Zombie states program the parallel directives have been added to the library function. Central to all random number generation is the *ZBQLU01* function which has been modified so only a single thread can access it at a time. The modified function is shown in Listing. B.17, the *critical* directive ensures each thread can access and alter shared values used for random number generation including the functions *ZBQLU01_pointer_check* and *ZBQLU01_X_value()*.

```
1  FUNCTION ZBQLU01(dummy)
2    implicit none
3    real(real64):: ZBQLU01,X,B2,BINV
4    integer::dummy
5
6    !$omp critical
7    B2 = B
8    BINV = 1.0D0/B
9
10   X = ZBQLU01_X_value()
11   ZBQLIX(ID43) = X
12   call ZBQLU01_pointer_check()
13
14   do while(X.LT.BINV)
15     B2 = B2*B
16     X = ZBQLU01_X_value()
17     ZBQLIX(ID43) = X
18     call ZBQLU01_pointer_check()
19   end do
20
21   ZBQLU01 = X/B2
22   !$omp end critical
23
24 END FUNCTION ZBQLU01
```

Listing B.17: Modified ZBQLU01 function that can only be accessed by a single thread at a time. All other functions in the random generation library directly or indirectly use it to generate random numbers.

Appendix C
Supplementary Theory

C.1 Hamiltonian Matrix Elements

The Hamiltonian matrix is constructed by its operation between two Slater determinants (formed of orthonormal orbitals), $\langle\Psi^{(A)}|\hat{H}|\Psi^{(B)}\rangle$. The Hamiltonian in Eq. (2.7) can be rewritten to explicitly have two parts, the first having single electron dependence, $\mathfrak{O}_1$ and the second being dependent on two electrons, $\mathfrak{O}_2$,

$$\hat{H} = \mathfrak{O}_1 + \mathfrak{O}_2 = -(\sum_{i=1}^{N_{el}} \frac{1}{2}\nabla_i^2 + \sum_{A=1}^{N_{nu}} \frac{Z_A}{r_{iA}}) + \sum_{i=1}^{N_{el}} \sum_{j>i}^{N_{el}} \frac{1}{r_{ij}} \tag{C.1}$$

the nuclear-nuclear repulsion has been omitted as it is constant. A set of k spin-orbitals containing N_{el} electrons is defined as $\phi = \chi_i(1),\ \chi_j(2),\ \ldots,\ \chi_k(N)$ with $\chi(l) \equiv \chi(\mathbf{X}_l)$ as defined as in Eq. (2.12). Two arbitrary Slater determinants are constructed from ϕ, $|\Psi^{(A)}\rangle$ and $|\Psi^{(B)}\rangle$. It is useful define a Slater determinant as a function of a general element

$$|\Psi^{(A)}\rangle = \frac{1}{N_{el}!^{1/2}} \sum_{n=1}^{N_{el}!} (-1)^{P_n} \prod_{l=1}^{N} \chi_{\mathfrak{P}_{n_l}}(l) \tag{C.2}$$

$\mathfrak{P}_n$ is an operator that generates the n^{th} permutation of the electron labels and so $\mathfrak{P}_{n_l}$ being the l^{th} element in that sequence and P_n the number of interchanges needed to reach this permutation.

C.1.1 One-electron Operator

Firstly, the one electron part can be calculated

$$\langle \Psi^{(A)} | \mathfrak{O}_1 | \Psi^{(B)} \rangle = \sum_{i=1}^{N_{el}} \langle \phi_A | h(i) | \phi_B \rangle = N \langle \phi_A | h(1) | \phi_B \rangle \tag{C.3}$$

Since the electrons are indistinguishable $h(1) = h(i)$ and each part of the summation is equivalent, $h(1)$ is used by convention.

$$\langle \Psi^{(A)} | \mathfrak{O}_1 | \Psi^{(B)} \rangle = \frac{N_{el}}{N_{el}!} \sum_{i=1}^{N_{el}!} \sum_{j=1}^{N_{el}!} (-1)^{P_i} (-1)^{P_j} \int d\phi [\prod_{l=1}^{N_{el}} \chi^*_{\mathfrak{P}_{i_l}}(l)] h(1) [\prod_{l=1}^{N_{el}} \chi_{\mathfrak{P}_{j_l}}(l)] \tag{C.4}$$

This has three possible solutions the first of which is the case when $|\Psi^{(A)}\rangle = |\Psi^{(B)}\rangle$

$$\begin{aligned}
\langle \Psi^{(A)} | \mathfrak{O}_1 | \Psi^{(A)} \rangle =& \frac{N_{el}}{N_{el}!} \sum_{i=1}^{N_{el}!} \sum_{i=1}^{N_{el}!} (-1)^{P_i} (-1)^{P_j} \int d\phi [\prod_{l=1}^{N} \chi^*_{\mathfrak{P}_{i_l}}(l)] h(1) [\prod_{l=1}^{N_{el}} \chi_{\mathfrak{P}_{j_l}}(l)] = \\
& \frac{1}{(N_{el}-1)!} \sum_{i=1}^{N_{el}!} \int d\phi [\prod_{l=1}^{N_{el}} \chi^*_{\mathfrak{P}_{i_l}}(l)] h(1) [\prod_{l=1}^{N_{el}} \chi_{\mathfrak{P}_{i_l}}(l)] = \\
& \frac{(N_{el}-1)!}{(N_{el}-1)!} \sum_{i}^{N_{el}} \int d\mathbf{X}_1 \chi^*_i(1) h(1) \chi_i(1) = \\
& \sum_{i}^{N_{el}} \langle i | h | i \rangle
\end{aligned} \tag{C.5}$$

The double summation is simplified to a single sum over $N_{el}!$ as the integral evaluates to zero for permutations when $i \neq j$ because different permutations mean a different spin orbital occupation and as by Eq. (2.9) their orthogonality means the integral is zero. Note $N_{el}/N_{el}! = N_{el}/((N_{el}-1)! \times N_{el}) = ((N_{el}-1)!)^{-1}$. This is further simplified by noticing that when placing the first electron in orbital χ_i there are $(N-1)!$ possible permutations when allocating the remaining electrons to the $N_{el}-1$ available spin orbitals. Since $i = j$ the integral over electrons $2, 3, \ldots, N_{el}$ will evaluate to one as the spin orbitals are normalised.

The second case is when $|\Psi^{(A)}\rangle = |\chi_i(1)\chi_j(2)\ldots\rangle$ and $|\Psi^{(B)}\rangle = |\chi_k(1)\chi_j(2)\ldots\rangle$ i.e. there is a single spin orbital different between the two Slater determinants.

$$\begin{aligned}\langle\Psi^{(A)}|\mathfrak{O}_1|\Psi^{(B)}\rangle =& \frac{N_{el}}{N_{el}!}\sum_{i=1}^{N_{el}!}\sum_{i=1}^{N_{el}!}(-1)^{P_i}(-1)^{P_j}\int d\phi[\prod_{l=1}^{N_{el}}\chi^*_{\mathfrak{P}_{i_l}}(l)]h(1)[\prod_{l=1}^{N_{el}}\chi_{\mathfrak{P}_{j_l}}(l)] = \\ & \frac{1}{(N_{el}-1)!}\sum_{i=1}^{N_{el}!}\int d\phi[\prod_{l=1}^{N_{el}}\chi^*_{\mathfrak{P}_{i_l}}(l)]h(1)[\prod_{l=1}^{N_{el}}\chi_{\mathfrak{P}_{j_l}}(l)] = \\ & \frac{(N_{el}-1)!}{(N_{el}-1)!}\int d\mathbf{X}_1\chi_i^*(1)h(1)\chi_j(1) = \\ & \sum_{i}^{N_{el}}\langle i|h|j\rangle\end{aligned} \tag{C.6}$$

Using the same reasoning as in the first case the double sum is reduced to a single one. Then using the orthogonality of different spin orbitals both χ_i and χ_j must be occupied by the same electron which is electron 1 to associate it with the operator $h(1)$. With this condition there are $(N_{el}-1)!$ possible combinations. Finally, if the two Slater determinants vary by more than a single spin orbital the one-electron portion of the Hamiltonian matrix element will be zero. To ensure both orbitals in each Slater determinant are not orthogonal would require them all to be occupied by electron 1 which is impossible as it can only occupy one orbital per Slater determinant.

C.1.2 Two-electron Operator

The same set of cases can then be applied to the two-electron portion of the Hamiltonian like the one-electron part the two-electron operator cannot distinguish between pairs of electrons and so the summation can be reduced to the action of r_{12}^{-1} multiplied by the number of electron pairs

$$\langle\Psi^{(A)}|\mathfrak{O}_2|\Psi^{(B)}\rangle = \sum_{i=1}^{N_{el}}\sum_{j>i}^{N_{el}}\langle\phi_A|\frac{1}{r_{ij}}|\phi_B\rangle = \frac{N_{el}(N_{el}-1)}{2}\langle\phi_A|\frac{1}{r_{12}}|\phi_B\rangle \tag{C.7}$$

In the first case when $|\Psi^{(A)}\rangle = |\Psi^{(B)}\rangle$ the following can be derived

$$\langle\Psi^{(A)}|\mathfrak{O}_2|\Psi^{(A)}\rangle = \frac{N_{el}(N_{el}-1)}{2N_{el}!}\sum_{i=1}^{N_{el}!}\sum_{j=1}^{N_{el}!}(-1)^{P_i}(-1)^{P_j}\int d\phi[\prod_{l=1}^{N_{el}}\chi^*_{\mathfrak{P}_{i_l}}(l)]\frac{1}{r_{12}}[\prod_{l=1}^{N_{el}}\chi_{\mathfrak{P}_{j_l}}(l)] =$$
$$\frac{1}{2(N_{el}-2)!}\sum_{i=1}^{N_{el}!}\int d\phi[\prod_{l=1}^{N_{el}}\chi^*_{\mathfrak{P}_{i_l}}(l)]\frac{1}{r_{12}}[((\chi_{\mathfrak{P}_{i_1}}(1)\chi_{\mathfrak{P}_{i_2}}(2))-(\chi_{\mathfrak{P}_{i_1}}(2)\chi_{\mathfrak{P}_{i_2}}(1)))\prod_{l=3}^{N_{el}}\chi_{\mathfrak{P}_{i_l}}(l)] =$$
$$\frac{(N_{el}-2)!}{2(N_{el}-2)!}\sum_{i}^{N_{el}}\sum_{j\neq i}^{N_{el}}\int d\mathbf{X}_1 d\mathbf{X}_2\chi^*_i(1)\chi^*_j(2)\frac{1}{r_{12}}[\chi_i(1)\chi_j(2)-\chi_i(2)\chi_j(1)] =$$
$$\frac{1}{2}\sum_{i}^{N_{el}}\sum_{j}^{N_{el}}\langle ij|ij\rangle - \langle ij|ji\rangle \tag{C.8}$$

As previously established with the one-electron operator permutations must be the same for both Slater determinants if the integral over these electrons is to evaluate to a non-zero value. As the spin orbitals are the same in both Slater determinants both electron 1 and 2 can be placed in either orbital $\chi_{\mathfrak{P}_{i_1}}$ or $\chi_{\mathfrak{P}_{i_2}}$ resulting in a non-zero integral. So, either ordering of electrons 1 and 2 are permitted and swapping their order gives the other permutation which is the source of the negative sign in the term after the operator. The restriction $j \neq i$ can be dropped because when $i = j$ the integral sums to zero $\langle ii|ii\rangle - \langle ii|ii\rangle$. The second case when the Slater determinants differ by a single spin orbital is now considered for the two-electron operator

$$\langle\Psi^{(A)}|\mathfrak{O}_2|\Psi^{(B)}\rangle = \frac{N_{el}(N_{el}-1)}{2N_{el}!}\sum_{i=1}^{N_{el}!}\sum_{j=1}^{N_{el}!}(-1)^{P_i}(-1)^{P_j}\int d\phi[\prod_{l=1}^{N_{el}}\chi^*_{\mathfrak{P}_{i_l}}(l)]\frac{1}{r_{12}}[\prod_{l=1}^{N_{el}}\chi_{\mathfrak{P}_{j_l}}(l)] =$$
$$\frac{1}{2(N_{el}-2)!}\sum_{i=1}^{N_{el}!}\int d\phi[\prod_{l=1}^{N_{el}}\chi^*_{\mathfrak{P}_{i_l}}(l)]\frac{1}{r_{12}}[((\chi_{\mathfrak{P}_{j_1}}(1)\chi_{\mathfrak{P}_{j_2}}(2))-(\chi_{\mathfrak{P}_{j_1}}(2)\chi_{\mathfrak{P}_{j_2}}(1)))\prod_{l=3}^{N_{el}}\chi_{\mathfrak{P}_{j_l}}(l)] =$$
$$\frac{(N_{el}-2)!}{2(N_{el}-2)!}\sum_{j\neq i}^{N_{el}}\int d\mathbf{X}_1 d\mathbf{X}_2[\chi^*_i(1)\chi^*_j(2)\frac{1}{r_{12}}[\chi_k(1)\chi_j(2)-\chi_k(2)\chi_j(1)]+$$
$$\chi^*_j(1)\chi^*_i(2)\frac{1}{r_{12}}[\chi_j(1)\chi_k(2)-\chi_j(2)\chi_k(1)]] =$$
$$\sum_{j\neq i}^{N_{el}}\int d\mathbf{X}_1 d\mathbf{X}_2\chi^*_i(1)\chi^*_j(2)\frac{1}{r_{12}}[\chi_k(1)\chi_j(2)-\chi_n(1)\chi_p(2)] =$$
$$\sum_{j}^{N_{el}}\langle ij|kj\rangle - \langle ij|jk\rangle \tag{C.9}$$

The simplification used for the second term uses the fact $r_{12}^{-1} = r_{21}^{-1}$

$$\begin{aligned}
&\int d\mathbf{X}_1 d\mathbf{X}_2 \chi_j^*(1)\chi_i^*(2)\frac{1}{r_{12}}[\chi_j(1)\chi_p(2) - \chi_j(2)\chi_k(1)] = \\
&\int d\mathbf{X}_1 d\mathbf{X}_2 \chi_j^*(2)\chi_i^*(1)\frac{1}{r_{21}}[\chi_j(2)\chi_p(1) - \chi_j(1)\chi_k(2)] = \\
&\int d\mathbf{X}_1 d\mathbf{X}_2 \chi_i^*(1)\chi_j^*(2)\frac{1}{r_{12}}[\chi_k(1)\chi_n(2) - \chi_k(2)\chi_j(1)]
\end{aligned} \tag{C.10}$$

As in the one electron case, using the same reasoning, if there more than one different spin orbitals between the Slater determinants the integral is zero.

C.2 Derivation of the Equations for the Gradient of the Energy Function

In this section a full derivation of the Gradient of the energy function is given. These equations were originally encoded into the Zombie states program to calculate derivatives of the energy function. However, a more efficient method requiring far fewer calculations is now employed as shown in section B.4.1. However, they prove that a Gradient Descent method is an applicable choice to optimise the Zombie state coefficients as the energy function is continually differentiable. First it is worth restating that "dead" and "alive" coefficients for each Zombie state orbital are generated as follows

$$a_{0j}^{(a)} = \cos(\theta_j), \quad a_{1j}^{(a)} = \sin(\theta_j). \tag{C.11}$$

So the gradient for a basis set of N_{bf} ZSs with N_{orb} spin orbitals will be vector of length $N_{bf} \cdot N_{orb}$ where each element is the partial derivative with respect to $\zeta_{\theta_j}^{(a)}$ i.e. the θ_j that generates the "dead" and "alive" coefficients for the j^{th} spin orbital in the a^{th} ZS. The energy equation to be differentiated can be written as,

$$f(X) = \sum_{ab}^{N_{bf}} d_a^* d_b \left(\sum_{ij}^{N_{orb}} h_{ij} \langle \zeta^{(a)} | b_i^\dagger \hat{b}_j | \zeta^{(b)} \rangle + \frac{1}{2} \sum_{klij}^{N_{orb}} W_{klji} \langle \zeta^{(a)} | \hat{b}_k^\dagger \hat{b}_l^\dagger \hat{b}_i \hat{b}_j | \zeta^{(b)} \rangle + H_{nuc} \Omega_{ab} \right). \tag{C.12}$$

An element in $\nabla f(X)$ is then $\frac{\partial f(X)}{\partial \zeta_{\theta_j}^{(c)}}$. By differentiating by parts

$$\frac{\partial f(X)}{\partial \zeta_{\theta_j}^{(c)}} = \sum_{ab}^{N_{bf}} \frac{\partial (d_a^*)}{\partial \zeta_{\theta_j}^{(c)}} d_b \langle \zeta^{(a)} | \hat{H} | \zeta^{(b)} \rangle + d_a^* \frac{\partial (d_b)}{\partial \zeta_{\theta_j}^{(c)}} \langle \zeta^{(a)} | \hat{H} | \zeta^{(b)} \rangle + d_a^* d_b \frac{\partial \langle \zeta^{(a)} | \hat{H} | \zeta^{(b)} \rangle}{\partial \zeta_{\theta_j}^{(c)}} \tag{C.13}$$

This derivative can then be split into three parts differentiating d_a^*, d_b and $\langle \zeta^{(a)} | \hat{H} | \zeta^{(b)} \rangle$ separately.

C.2.1 Differentiating the Overlap Matrix

The overlap is found multiplicatively so when differentiating with respect to a single $\zeta_{\theta_j}^{(a)}$ only the j^{th} term will be effected giving two possible outcomes

$$\frac{\partial}{\partial \theta_j^{(a)}} (a_{0j}^{(a)*} a_{0j}^{(b)} + a_{1j}^{(a)*} a_{1j}^{(b)}) = \begin{cases} -\sin(\theta_j^{(a)}) \cos(\theta_j^{(b)}) + \cos(\theta_j^{(a)}) \sin(\theta_j^{(b)}) & \zeta^{(a)} \neq \zeta^{(b)} \\ \sin(2\theta_j^{(a)}) - \sin(2\theta_j^{(b)}) = 0 & \zeta^{(a)} = \zeta^{(b)} \end{cases} \tag{C.14}$$

meaning derivatives of diagonal elements of the overlap matrix will always be zero.

C.2.2 Differentiating the Second Quantization Hamiltonian

Following creation and annihilation operations an overlap is then calculated to find Hamiltonian matrix elements. So, if the derivative is being found for an orbital where no creation or annihilation operations have taken place the result of Eq. (C.14) can be used. For derivatives of orbitals which have creation/annihilation operations the following results are found for operators all enacting to the right on $\langle \zeta_j^{(b)} | \hat{O} | \zeta_j^{(b)} \rangle$ where $\hat{O}$ is a creation or annihilation operator or both,

$$\frac{\partial}{\partial \theta_j^{(a)}} \hat{O} (a_{0j}^{(a)*} a_{0j}^{(b)} + a_{1j}^{(a)*} a_{1j}^{(b)}) = \begin{cases} cos(\theta_j^{(a)}) cos(\theta_j^{(b)}) & \zeta^{(a)} \neq \zeta^{(b)}, \hat{O} = \hat{b}_j^\dagger \\ -sin(\theta_j^{(a)}) sin(\theta_j^{(b)}) & \zeta^{(a)} \neq \zeta^{(b)}, \hat{O} = \hat{b}_j \\ cos(\theta_j^{(a)}) sin(\theta_j^{(b)}) & \zeta^{(a)} \neq \zeta^{(b)}, \hat{O} = \hat{b}_j \hat{b}_j^\dagger \\ sin(2\theta_j^{(a)}) & \zeta^{(a)} = \zeta^{(b)}, \hat{O} = \hat{b}_j^\dagger \\ -sin(2\theta_j^{(a)}) & \zeta^{(a)} = \zeta^{(b)}, \hat{O} = \hat{b}_j^\dagger \\ sin(2\theta_j^{(a)}) & \zeta^{(a)} = \zeta^{(b)}, \hat{O} = \hat{b}_j \hat{b}_j^\dagger \end{cases} \tag{C.15}$$

Using these results, it is possible to differentiate the second quantization Hamiltonian

$$\frac{\partial}{\partial\zeta_{\theta_j}^{(c)}}\langle\zeta^{(a)}|\hat{H}|\zeta^{(b)}\rangle = \left(\sum_{ij}^{N_{orb}} h_{ij}\frac{\partial\langle\zeta^{(a)}|\hat{b}_i^\dagger\hat{b}_j|\zeta^{(b)}\rangle}{\partial\zeta_{\theta_j}^{(c)}} + \frac{1}{2}\sum_{klji}^{N_{orb}} W_{klji}\frac{\partial\langle\zeta^{(a)}|\hat{b}_k^\dagger\hat{b}_l^\dagger\hat{b}_j\hat{b}_i|\zeta^{(b)}\rangle}{\partial\zeta_{\theta_j}^{(c)}} + H_{nuc}\frac{\partial\langle\zeta^{(a)}|\zeta^{(b)}\rangle}{\partial\zeta_{\theta_j}^{(c)}}\right)$$
$$= \left(\sum_{ij}^{N_{orb}} h_{ij}\frac{\partial\langle\zeta^{(a)}|\boldsymbol{\zeta}_{ij}^{(b)}\rangle}{\partial\zeta_{\theta_j}^{(c)}} + \frac{1}{2}\sum_{klji}^{N_{orb}} W_{klji}\frac{\partial\langle\zeta^{(a)}|\boldsymbol{\zeta}_{klji}^{(b)}\rangle}{\partial\zeta_{\theta_j}^{(c)}} + H_{nuc}\frac{\partial\langle\zeta^{(a)}|\boldsymbol{\zeta}(b)\rangle}{\partial\zeta_{\theta_j}^{(c)}}\right). \tag{C.16}$$

Eq. (C.15) and Eq. (C.14) can then be applied as necessary to evaluate the summation.

C.2.3 Differentiating the d Vector

Derivatives of the Hamiltonian and Overlap matrix are used regularly throughout the following section so the notation $\partial H_{ab} = \frac{\partial}{\partial\theta_j^{(a)}}H_{ab}$ and $\partial\Omega = \frac{\partial}{\partial\theta_j^{(a)}}\Omega$ to represent partial derivatives of the Hamiltonian and overlap matrices with the subscript ab used to show a specific matrix element. $\mathbf{d}$ is found by an iterative process using imaginary time propagation starting at a value $\mathbf{d}_0$ which is initially set as either 1 or 0. An imaginary time step in matrix notation is

$$\mathbf{d}_t(\beta) = \mathbf{d}_{t-1}(\beta) + \dot{\mathbf{d}}_t(\beta)\Delta\beta = \mathbf{d}_{t-1} - \boldsymbol{\Omega}^{-1}\mathbf{H}\mathbf{d}_{t-1}\Delta\beta, \tag{C.17}$$

and then the resultant vector is normalised

$$\hat{\mathbf{d}}_t = \frac{\mathbf{d}_t}{\sqrt{|\mathbf{d}_t^*\boldsymbol{\Omega}\mathbf{d}_t|}}. \tag{C.18}$$

Thus, a single equation for $\hat{\mathbf{d}}_t$ is

$$\hat{\mathbf{d}}_t = \frac{\mathbf{d}_{t-1}}{\sqrt{|\mathbf{d}_{t-1}^*\boldsymbol{\Omega}\mathbf{d}_{t-1}|}} - \boldsymbol{\Omega}^{-1}\mathbf{H}\frac{\mathbf{d}_{t-1}}{\sqrt{|\mathbf{d}_{t-1}^*\boldsymbol{\Omega}\mathbf{d}_{t-1}|}}\Delta\beta. \tag{C.19}$$

The norm can be written as $||\mathbf{d}_{t-1}|| = \sqrt{|\sum_{ab}^{N_{bf}} d_{t-1}^{*(a)} d_{t-1}^{(b)} \langle \zeta^{(a)} | \zeta^{(b)} \rangle|}$ Therefore, the partial derivative of $\hat{\mathbf{d}}_t$ is

$$\begin{aligned}\frac{\partial \hat{\mathbf{d}}_t}{\partial \zeta_{\theta_j}^{(c)}} &= \frac{\partial}{\partial \zeta_{\theta_j}^{(c)}} \left(\frac{\mathbf{d}_{t-1}}{\sqrt{|\mathbf{d}_{t-1}^{*} \mathbf{\Omega} \mathbf{d}_{t-1}|}}\right) \\ &- \left[\frac{\partial (\mathbf{\Omega}^{-1})}{\partial \zeta_{\theta_j}^{(c)}} \mathbf{H} \frac{\mathbf{d}_{t-1}}{\sqrt{|\mathbf{d}_{t-1}^{*} \mathbf{\Omega} \mathbf{d}_{t-1}|}} + \mathbf{\Omega}^{-1} \frac{\partial (\mathbf{H})}{\partial \zeta_{\theta_j}^{(c)}} \frac{\mathbf{d}_{t-1}}{\sqrt{|\mathbf{d}_{t-1}^{*} \mathbf{\Omega} \mathbf{d}_{t-1}|}} \right. \\ &\left. + \mathbf{\Omega}^{-1} \mathbf{H} \frac{\partial}{\partial \zeta_{\theta_j}^{(c)}} \left(\frac{\mathbf{d}_{t-1}}{\sqrt{|\mathbf{d}_{t-1}^{*} \mathbf{\Omega} \mathbf{d}_{t-1}|}}\right) \right] \Delta\beta.\end{aligned} \tag{C.20}$$

The notation $\partial(d_{t-1}^{(a)}) = \dfrac{\partial d_{t-1}^{(a)}}{\partial \zeta_{\theta_j}^{(c)}}$ is introduced as shorthand for the partial derivative of element $d_t^{(a)}$ and $\partial(||\mathbf{d}_{t-1}||)$ being the partial derivative of the norm. Hence,

$$\begin{aligned}\frac{\partial d_t^{(c)}}{\partial \zeta_{\theta_j}^{(c)}} &= \frac{||\mathbf{d}_{t-1}|| \partial(d_{t-1}^{(c)}) - d_{t-1}^{(c)} \partial(||\mathbf{d}_{t-1}||)}{||\mathbf{d}_{t-1}||^2} \\ &- \sum_{b}^{N_{bf}} \Omega_{cb}^{-1} \frac{\partial(\Omega_{cb})}{\partial \zeta_{\theta_j}^{(c)}} \Omega_{cb}^{-1} H_{cb} \hat{d}_{t-1}^{(b)} \Delta\beta - \sum_{b}^{N_{bf}} \Omega_{cb}^{-1} \frac{\partial(H_{cb})}{\partial \zeta_{\theta_j}^{(c)}} \hat{d}_{t-1}^{(b)} \Delta\beta \\ &- \sum_{b}^{N_{bf}} \Omega_{cb}^{-1} H_{cb} \frac{||\mathbf{d}_{t-1}|| \partial(d_{t-1}^{(b)}) - d_{t-1}^{(b)} \partial(||\mathbf{d}_{t-1}||)}{||\mathbf{d}_{t-1}||^2} \Delta\beta.\end{aligned} \tag{C.21}$$

The derivative identity $\partial(A^{-1}) = -A^{-1}(\partial A)A^{-1}$ is used to find the second term [179]. The partial derivative of the norm is given by

$$\begin{aligned}\frac{\partial(||\mathbf{d}_{t-1}||)}{\partial \zeta_{\theta_j}^{(c)}} &= \frac{\partial}{\partial \zeta_{\theta_j}^{(c)}} \sqrt{|\sum_{ab}^{N_{bf}} d_{t-1}^{*(a)} d_{t-1}^{(b)} \langle \zeta^{(a)} | \zeta^{(b)} \rangle|} \\ &= \frac{1}{2||d_{t-1}||} \left[\sum_{ab}^{N_{bf}} \frac{\partial(d_{t-1}^{*(a)}) d_{t-1}^{*(b)} |d_{t-1}^{(b)} \langle \zeta^{(a)} | \zeta^{(b)} \rangle|}{|d_{t-1}^{*(a)}|} + \frac{|d_{t-1}^{(a)}| d_{t-1}^{(b)} \partial(d_{t-1}^{(b)}) | \langle \zeta^{(a)} | \zeta^{(b)} \rangle|}{|d_{t-1}^{(b)}|} \right. \\ &\left. + \frac{d_{t-1}^{*(a)} d_{t-1}^{(b)} \langle \zeta^{(a)} | \zeta^{(b)} \rangle \partial(\langle \zeta^{(a)} | \zeta^{(b)} \rangle)}{|\langle \zeta^{(a)} | \zeta^{(b)} \rangle|} \right] \\ &= \frac{1}{2||d_{t-1}||} \left[\sum_{ab}^{N_{bf}} \frac{\partial(d_{t-1}^{*(a)}) d_{t-1}^{*(a)} |d_{t-1}^{(b)} \Omega_{ab}|}{|d_{t-1}^{*(a)}|} + \frac{|d_{t-1}^{(a)}| d_{t-1}^{(b)} \partial(d_{t-1}^{(b)}) | \Omega_{lk}|}{|d_{t-1}^{(b)}|} \right. \\ &\left. + \frac{d_{t-1}^{*(a)} d_{t-1}^{(b)} \Omega_{ab} \partial(\Omega_{ab})}{|\Omega_{ab}|} \right]\end{aligned} \tag{C.22}$$

where $\partial(\Omega_{ab}) = \frac{\partial(\Omega_{ab})}{\partial\zeta_{\theta_j}^{(c)}}$. Finally, the derivative of $\mathbf{d}_0$ needs to be computed. $\mathbf{d}_0$ is set at the beginning of imaginary time propagation before normalisation and so does not have any dependence on $\theta_j^{(c)}$ until it is normalised. So, the derivative can be found by modifying Eq. (C.22),

$$\frac{\partial\hat{\mathbf{d}}_0}{\partial\zeta_{\theta_j}^{(c)}} = \frac{\partial}{\partial\zeta_{\theta_j}^{(c)}}\frac{\hat{\mathbf{d}}_0}{\sqrt{|\sum_{ab}^{N_{bf}} d_0^{*(a)} d_0^{(b)}\langle\zeta^{(a)}|\zeta^{(b)}\rangle|}} = \frac{\hat{\mathbf{d}}_0}{2||\hat{\mathbf{d}}_0||^3}\left[\sum_{ab}^{N_{bf}}\frac{d_0^{*(a)} d_0^{(b)}\Omega_{ab}\partial(\Omega_{ab})}{|\Omega_{ab}|}\right]. \tag{C.23}$$

The gradient with respect to each $\zeta_{\theta_j}^{(c)}$ for $\partial(\boldsymbol{H}), \partial(\boldsymbol{\Omega})$ and $\partial(\boldsymbol{\Omega}^{-1})$ can be calculated using the results from Eq. (C.15) and Eq. (C.14). These can then be used to calculate the gradient of $\hat{\mathbf{d}}_t$ at each imaginary time step. This derivation demonstrates the energy equation is continually differentiable which verifies the ability to use the Gradient Descent process. Clearly, using this direct calculation is computationally expensive due to the large number of intermediate variables needed to be calculated and stored. The indirect method shown in Eq. (4.21) eliminates the need to calculate $\partial(\boldsymbol{H})$ and $\partial(\boldsymbol{\Omega}^{-1})$ and requires a single calculation rather than an iterative process to compute the $\hat{\mathbf{d}}_t$ component of the gradient.

Appendix D
Using the Zombie states Program

The Zombie states program is designed to be used for, in theory, any type of atomic or molecular electronic structure simulation. The program files are sorted into three folders: **run**, **src**, and **build**. The **src** folder contains the source files written in Fortran; the **build** folder contains the Makefiles needed to compile the program on specific systems and any temporary compilation files generated; the **run** folder contains scripts to run and compile the program, input files and scripts to generate results written in Python. The control scripts are the part of the program all users will have to interact with and so Python was chosen as the language of choice due to its popularity and easily comprehensible syntax. Therefore, it should be easier for other users to make changes to the inputs or graphic outputs. The set-up and execution of the Zombie states program is controlled by *run.py* with some additional helper functions kept in *integral_write.py*. All variables to be set by the user are contained in the *inputs.json* file and will be discussed first before explanation of how the *run.py* uses the inputs to set-up and begin the program execution.

D.1 Input Files

The program inputs are set in the *inputs.json* file with similar parameters logically grouped together. The set of input parameters are summarised in Table. D.1 and Table. D.2

Table D.1: Parameters in *inputs.json* file, part 1.

Parameter	Values	Effect	Restrictions
runfolder	string	Runs program in a folder with the name string	Case insensitive
runtime	string	Requested time 'hh:mm:ss' for HPC job	Maximum 48:00:00
nodes	number	No. nodes to request on cluster computer	$1 \leq$ Integer ≤ 100
cores	number	No. threads to run per node	$1 \leq$ Integer ≤ 40
submissions	number	Submits multiple jobs on a cluster computer to run sequentially	
datafolder	string	Name of folder within run folder that contains files to be used to set-up program execution	Case insensitive
elecs	PyScf	Generates electron integrals using PyScf	
	mol	Generates electron integrals from an existing Molpro file	
	integrals	Copies existing *h1ei.csv*, *h2ei.csv* and *hnuc.csv* to execution folder	
	no	Copies existing *elec_integrals.csv* file to execution folder	
pyscf_file	string	Name of the *.json* file with PyScf system information	Case insensitive
ndet	number	Initial number of Zombie state basis functions	Integer
seed	number	Seed of the random number generator	Default 0 takes seed from **/bin/urandom**
zomgen	y/n	Zombie states generated	
hamgen	y/n	Hamiltonian generated	
clean	y/n	Cleaning procedure is carried out	
imagprop	y/n	Zombie states are propagated in imaginary time	
beta	number	Length of imaginary time propagation	Real number
timesteps	number	No. imaginary time steps	Integer
grad	y/n	Use gradient descent	
gram	y/n	Use Gram-Schmidt orthogonalisation	

Table D.2: Parameters in *inputs.json* file, part 2.

Parameter	Values	Effect	Restrictions
norb	number	No. spatial orbitals	Integer
nel	number	No. electrons in the system	Integer
spin	number	Total spin of the system	Integer
zomtyp	HF	HF-like Zombie states	
	ran	Randomly generates ZS amplitudes	
	bb	Uses biasing scheme to set up ZSs	
rhf_1	y/n	Sets first ZS as the RHF Slater determinant	Required for gradient descent
epoc_max	number	Maximum No. epochs	Integer
initial_learning_rate	number	The initial learning rate	Real number
learning_rate_decay	number	Learning rate scaling parameter	$0 <$ Float < 1
decay_ steps	number	No. of learning rates	Integer
clone	y/n	Use the cloning procedure to increase the basis set size	
clone_max	number	Maximum basis set size	Integer
clone_steps	number	Maximum No. epochs between cloning events	Integer
clone_num	number	No. ZSs added during a cloning event	Integer
gramnum	number	No. excited states used in Gram-Schmidt orthogonalisation	Integer
hamfile	string	Name of Hamiltonian file in datafolder	Case insensitive
ovrlfile	string	Name of overlap file in datafolder	Case insensitive
elecfile	string	Name of processed electron integral file in datafolder	Case insensitive

D.2 Running the Program

The Zombie states program can be executed, on any type of operating system, using the command **python run.py** in the **run** folder. The run script starts by checking the input parameters meet the following criteria

- The node and thread parameters are numbers greater than 0
- The number of threads is less than 40 and the number of nodes is less than 100
- The type of Zombie state specified is either random, HF or biased
- The one- and two- electrons are of a type supported by the program
- The Zombie generation flag is either 'y' or 'n'
- The RHF Zombie state flag is either 'y' or 'n'
- The imaginary time propagation flag is either 'y' or 'n'
- The number of imaginary time steps and β are both numbers
- The Hamiltonian generation flag is either 'y' or 'n'
- The cleaning flag is either 'y', 'n' or 'f' (if using an input file)
- The Gram-Schmidt orthogonalisation flag is either 'y' or 'n'
- The gradient descent flag is either 'y' or 'n'
- Previous parameters are compatible with Gradient descent
- The number of learning rates is greater than 1
- The learning rate adjustment parameter is a number between 0 and 1
- The maximum number of epochs is a number greater than 2
- The gradient descent cloning parameter is either 'y' or 'n'
- The maximum basis set size is a number
- The maximum number of epochs between cloning events is a number
- The number of Zombie states added to the basis set when cloning is a number
- Any files to be copied to the execution folder are present

Once the input variables have been checked the execution folder is set up. Within each run folder a separate **data** folder is created to contain the Zombie function files and the Hamiltonian and overlap matrix output files. The run script can copy existing one- and two-electron integrals, if specified in *inputs.json*, to the run folder **integral** folder. Alternatively, the program can generate new integrals using either Molpro or PyScf. If using Molpro the integrals must have already been generated and be available in the run folder so the Zombie run script can read them in and process them to be used by the main ZS program. But if new integrals are to be calculated using PyScf the functionality is integrated into the run script. A *json* file containing information about the system is read in by the run script and example file for the truncated Li_2 in the $6-31G^{**}$ basis is shown in Listing. D.1.

```
pyscf={
    # The units the geometry of the molecule is set up in
    'units':'Bohr',
    # The geometry of the molecule being investigated
    'atoms': 'Li 0 0 0; Li 0 0 6',
    # The type of basis used to generate the 1 and 2
   electron integrals
    'bs' : '6-31g**',
    # How verbose do you want the PyScf output to be in your
    terminal?
    'verbosity' : 4,
    'symmetry' :True,
    'spin':0,
    'charge':0,
    'symmetry_subgroup' : 0,
    # Number of spatial orbitals
    'norb': 5,
    # Number of electrons
    'nel':6
}
```

Listing D.1: Example of a PyScf molecule json file for Li_2. The run script uses this to generate relevant one- and two-electron integrals.

Using the system specific information one- and two-electron integrals are generated using the code in Listing. D.2. A molecule is set-up using the *gto.M()* function which takes parameters specifying the system from the input file. A restricted Hartree Fock calculation is then performed and the molecular orbitals are then extracted. The one- and two-electron integrals are then put into separate variables *h1e* and *eri_full* and the nuclear repulsion *Hnuc* is then written to a file. These are the same variables if values have been read in from Molpro.

The *integral_write.py* script is then called for which puts the integrals into the

form of creation/annihilation operations on each spin orbital. Any zero valued integrals or combinations of creation and annihilation operations that result in a zero valued overlap (see section B.3.1.3 for more details) are removed and these are then written to two files *h1e.csv* and *h2e.csv* which are placed in the **integral** folder.

```
mol = gto.M(
        unit = inputs.pyscf['units'],
        atom = inputs.pyscf['atoms'],
        basis = inputs.pyscf['bs'],
        verbose = inputs.pyscf['verbosity'],
        symmetry = inputs.pyscf['symmetry'],
        spin=inputs.pyscf['spin'],
        charge=inputs.pyscf['charge'],
        # symmetry_subgroup = inputs.pyscf['
    symmetry_subgroup'],
        )
        myhf=scf.RHF(mol)
        myhf.kernel()
        # Extract AO->MO transformation matrix
        c = myhf.mo_coeff
        # Get 1-electron integrals and convert to MO basis
        h1e = reduce(numpy.dot, (c.T, myhf.get_hcore(), c))
        # Get 2-electron integrals and transform them
        eri = ao2mo.kernel(mol, c)
        # Ignore all permutational symmetry, and write as
    four-index tensor, in chemical notation
        eri_full = ao2mo.restore(1, eri, c.shape[1])
        # Scalar nuclear repulsion energy
        Hnuc = myhf.energy_nuc()
```

Listing D.2: Code used to generate one- and two-electron integrals using PyScf. A restricted Hartree Fock calculation is run and the integrals put into spin orbitals.

The main program is then compiled using the appropriate make file to produce the *ZOMBIE* executable. For complete portability the GNU Fortran compiler, gfortran, is used as it is open source. Care has been taken to ensure non-standard Fortran functions are not used and so alternative compilers such as intel could be used with some modification to make files, but this has not been extensively tested. The inputs needed by the main program are then written to a file 'rundata.csv' which can be easily read into the main ZS program at the beginning of its run.

If the program is being run locally on a personal computer the code in Listing. D.3 is used which sets up the number of OpenMP threads and then runs the *ZOMBIE* executable. If the program is being run on a HPC requiring the program to be submitted to a scheduler then a short bash script is created as shown in Listing. D.4. This sets the number of threads; the maximum amount of time the program can

run for; the amount of memory to request per thread and loads the Math Kernel library which contains the BLAS and LAPACK libraries.

```
if(inputs.run['cores']!=1):
      os.environ["OMP_NUM_THREADS"]=str(inputs.run['cores'])
     subprocess.run(["./ZOMBIE"])
```

Listing D.3: Code used to initiate the Zombie states program on a personal computer

The run script then either submits a single job or multiple jobs each to start when the previous one has completed. It is convenient to make multiple job submissions at once as cluster computers, with many users, often have a time limit for each job meaning the program can be aborted before execution is complete. By setting multiple dependent jobs the program can automatically restart from the place the previous job aborted from. To restart a program or to submit further jobs after the original set of submissions are complete the *restart.py* script can be called from within the execution folder. This runs a similar set of error checks as the *run.py* script before restarting the program or submitting more jobs to the scheduler.

```
#$ -cwd -V
#$ -pe smp 16
#$ -l h_rt=48:00:00
#$ -l h_vmem=1G
module add mkl
time ./ZOMBIE
```

Listing D.4: Example run script requesting 16 threads to run for 48 hours, 1 Gigabyte of memory per core and loading the Math Kernel Library to access the BLAS/LAPACK routines.

D.3 Output Files

The output files for the Zombie state program are all of *.csv* type which were chosen because they are easily written and read into the Fortran program allowing the program to continue execution if restarted. Moreover, data can also be extracted using library functions in the python script used to generate plots of the simulation this also makes it simple for users to write custom scripts to extract data or open them using spreadsheet software. As such output files have been designed to make them computer friendly rather than necessarily easily read by a user. The files take either the naming convention *<name>.csv* or *<name>_#.csv* where # is a number.

- *zombie_X#.csv* contains data about the Zombie state where # denotes its number in the basis set. # has up to two leading zeros meaning it takes values $001 \rightarrow 999$. X takes the value 1 if the program is carrying out gradient descent and stores the unconverged ZS, in all other cases including when outputting the converged Zombie state information X=0. The file is arranged in rows of three with N_{orb} columns for each spin orbital. The first row is the set of θ_j used to generate the Zombie state coefficients running across from the first to the final orbital. The next two rows are the "dead" and "alive" coefficients of the ZS which are $\cos(\theta_j)$ and $\sin(\theta_j)$. If the program is undergoing gradient descent every time the ZS coefficients are changed the updated coefficients are written to the *zombie_1#.csv* file.

- *ham.csv* contains the initial Hamiltonian matrix data.

- *ham_final.csv* contains the Hamiltonian matrix data once the gradient descent process has completed.

- *ovrlp.csv* contains the initial overlap matrix data.

- *ovrlp_final.csv* contains the overlap matrix data once the gradient descent process has completed.

- *dvec#.csv* contains the Zombie state coefficient data # is zero if Gram-Schmidt orthogonalisation is not being used or is the number of the state if it is. The first row will contain the initial values after imaginary time propagation is complete if gradient descent is being used the second row will be the ZS coefficient values after gradient descent has complete. Each row will be N_{bf} long.

- *energy.csv* contains the energy at each step during imaginary time propagation. The first column contains the time step and the second the energy at that time. If Gram-Schmidt orthogonalisation is being used the changing energy for each excited state is written to in column three onwards.

- *energy_final.csv* takes exactly the same form as *energy.csv* but for the imaginary time propagation data once gradient descent has been completed.

- *epoc.csv* contains the gradient descent process data. The first column is the epoch number; the second is the energy at the end of the epoch; the third is the learning rate this is followed by the indices of the Zombie states altered on that epoch.

- *elec_integrals.csv* contains the processed one- and two-electron integral data which can be reused on restarted runs. For full details of how the program processes the integral can be found in section B.3.1.3. The first row contains the total number of one- and two- electron integrals, N_{int}. Each subsequent row from $2 - N_{int}$ is then contains the integral value and its corresponding set of creation and annihilation operator classifications for each orbital. Next the maximum number of changes in classification for an orbital is written. The next row is the number of classification changes for each orbital. This followed by the start and end indices for each classification change for each orbital. Next the optimised order of orbital calculation is given and the final row is the nuclear repulsion. This file is unchanged by gradient descent or Gram-Schmidt orthogonalisation.

The ZS, Hamiltonian, overlap and Zombie coefficient files are stored in the **data** folder and the *elec_integrals.csv* is placed in the **integrals** folder. The energy and epoch data files are placed in the execution folder as these are the main program outputs.

D.3.1 Plotting Outputs

Once the main Zombie states program has completed its execution, the results can be plotted using the *graph.py* python script. This script is executed in the run folder and uses the input data to automatically complete set of graphs accordingly. Each plot uses the naming convention *<name>.png* and produces an image 3.37×5.055 inches in size with a dpi=300.

- *energy_state_#.png* plot of energy in atomic units against imaginary time for state number, #, if Gram-Schmidt orthogonalisation used.

- *energy.png* plot of (initial) energy in atomic units against imaginary time which contains all states if Gram-Schmidt orthogonalisation used.

- *energy_final_state_#.png* plot of energy in atomic units against imaginary time after gradient descent has completed for state number, #, if Gram-Schmidt orthogonalisation used.

- *energy_final.png* plot of energy in atomic units against imaginary time after gradient descent has completed.
- *initial_and_final_energy.png* plot of imaginary time propagation for the converged and initial wave function. This puts the data in *energy.png* and *energy_final.png* onto the same plot. If Gram-Schmidt orthogonalisation is used this will contain information about all states.
- *epoc_#.png* plot of ground state energy against the epochs showing how the energy changes over the gradient descent process for state number, # when using gradient descent.
- *epoc.png* plot of ground state energy against the epochs showing how the energy changes over the gradient descent process.

These files are all saved in the execution folder.

Bibliography

[1] R. I. of Great Britain, Proceedings of the royal institution of great britain, in *Proceedings of the Royal Institution of Great Britain*, volume 35, page 251, The Institution, 1951. [Cited on page iii.]

[2] *Energy, structure and reactivity, Proceedings of the 1972 Boulder conference on theoretical chemistry*, 1973. [Cited on page 1.]

[3] J. A. Pople, Rev. Mod. Phys. **71**, 1267 (1999). [Cited on pages 1 and 100.]

[4] D. R. Hartree, Mathematical Proceedings of the Cambridge Philosophical Society **24**, 111 (1928). [Cited on pages 2, 11, and 18.]

[5] J. C. Slater, Phys. Rev. **35**, 210 (1930). [Cited on pages 2 and 12.]

[6] V. Fock, Zeitschrift fur Physik **75**, 622 (1932). [Cited on pages 2 and 14.]

[7] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964). [Cited on page 3.]

[8] W. Kohn and L. J. Sham, Phys. Rev. **140**, A1133 (1965). [Cited on page 3.]

[9] D. V. Shalashilin, The Journal of Chemical Physics **148**, 194109 (2018). [Cited on pages 4, 5, 40, 48, 49, 54, 59, 60, 61, 62, 114, 139, and 150.]

[10] D. V. Shalashilin and M. S. Child, The Journal of Chemical Physics **119**, 1961 (2003). [Cited on pages 4 and 9.]

[11] R. J. Glauber, Phys. Rev. **130**, 2529 (1963). [Cited on pages 4, 9, 31, and 58.]

[12] R. J. Glauber, Phys. Rev. **131**, 2766 (1963). [Cited on pages 4, 31, 44, and 58.]

[13] D. V. Shalashilin, The Journal of Chemical Physics **130**, 244101 (2009). [Cited on pages 4, 9, 58, and 116.]

[14] D. V. Makhov, W. J. Glover, T. J. Martinez, and D. V. Shalashilin, The Journal of Chemical Physics **141**, 054110 (2014). [Cited on pages 4 and 9.]

[15] C. Symonds, J. A. Kattirtzi, and D. V. Shalashilin, The Journal of Chemical Physics **148**, 184113 (2018). [Cited on pages 4, 9, 84, and 101.]

[16] A. Kirrander and D. V. Shalashilin, Phys. Rev. A **84**, 033406 (2011). [Cited on pages 4, 58, and 60.]

[17] M. Ronto and D. V. Shalashilin, The Journal of Physical Chemistry A **117**, 6948 (2013). [Cited on pages 4 and 116.]

[18] K. E. Cahill and R. J. Glauber, Phys. Rev. A **59**, 1538 (1999). [Cited on pages 4, 41, 44, 59, 115, and 126.]

[19] W.-M. Zhang, D. H. Feng, and R. Gilmore, Rev. Mod. Phys. **62**, 867 (1990). [Cited on pages 4, 31, 32, 34, 40, 42, 43, and 115.]

[20] R. D. Monique Combescure, *Coherent States and Applications in Mathematical Physics*, Springer Cham, 2 edition, 2021. [Cited on pages 4, 9, 34, 41, 126, and 128.]

[21] A. M. Perelomov, Communications in Mathematical Physics **26**, 222 (1972). [Cited on pages 4, 31, 33, 58, and 114.]

[22] R. Gilmore, Journal of Mathematical Physics **20**, 891 (1979). [Cited on pages 4, 31, and 114.]

[23] O. A. Bramley, T. J. H. Hele, and D. V. Shalashilin, The Journal of Chemical Physics **156**, 174116 (2022). [Cited on pages 5, 48, 49, 59, and 114.]

[24] N. S. Blunt, S. D. Smart, G. H. Booth, and A. Alavi, The Journal of Chemical Physics **143**, 134117 (2015). [Cited on pages 9, 29, 105, 106, and 111.]

[25] L. Tong, M. Nolan, T. Cheng, and J. Greer, Computer Physics Communications **131**, 142 (2000). [Cited on pages 9 and 29.]

[26] J. R. Klauder, *Coherent states : applications in physics and mathematical physics*, World Scientific, Singapore ;, 1985 - 1985. [Cited on pages 9, 36, and 37.]

[27] D. V. Shalashilin and M. S. Child, The Journal of Chemical Physics **121**, 3563 (2004). [Cited on page 9.]

[28] D. V. Shalashilin, The Journal of Chemical Physics **132**, 244111 (2010). [Cited on page 9.]

[29] K. Saita, M. G. D. Nix, and D. V. Shalashilin, Phys. Chem. Chem. Phys. **15**, 16227 (2013). [Cited on page 9.]

[30] D. V. Makhov, K. Saita, T. J. Martinez, and D. V. Shalashilin, Phys. Chem. Chem. Phys. **17**, 3316 (2015). [Cited on page 9.]

[31] S. Fernandez-Alberti, D. V. Makhov, S. Tretiak, and D. V. Shalashilin, Phys. Chem. Chem. Phys. **18**, 10028 (2016). [Cited on page 9.]

[32] D. V. Shalashilin and M. S. Child, The Journal of Chemical Physics **128**, 054102 (2008). [Cited on pages 9 and 116.]

[33] O. Bramley, C. Symonds, and D. V. Shalashilin, The Journal of Chemical Physics **151**, 064103 (2019). [Cited on pages 9, 41, 54, and 58.]

[34] J. A. Green, Development and application of new numerical extensions to the coupled coherent states family of multidimensional quantum dynamics methods, 2018. [Cited on page 9.]

[35] J. A. Green and D. V. Shalashilin, Phys. Rev. A **100**, 013607 (2019). [Cited on page 9.]

[36] A. Szabo and N. S. Ostlund, *Modern quantum chemistry : introduction to advanced electronic structure theory*, Dover Publications, New York, 1989. [Cited on pages 11 and 153.]

[37] V. Fock, Zeitschrift für Physik **61**, 126 (1930). [Cited on page 12.]

[38] J. C. Slater, Phys. Rev. **34**, 1293 (1929). [Cited on page 12.]

[39] P. A. M. Dirac, *The Quantum Theory of the Emission and Absorption of Radiation*, pages 243–265, Springer Dordrecht, London, 1927. [Cited on page 14.]

[40] P. Jordan and E. Wigner, Zeitschrift für Physik **47**, 631 (1928). [Cited on page 14.]

[41] J. C. Slater, Phys. Rev. **36**, 57 (1930). [Cited on page 16.]

[42] P. M. W. Gill, Advances in Quantum Chemistry **25**, 141 (1994). [Cited on page 16.]

[43] R. J. Mathar, International Journal of Quantum Chemistry **90**, 227 (2002). [Cited on page 16.]

[44] H. B. Schlegel and M. J. Frisch, International Journal of Quantum Chemistry **54**, 83 (1995). [Cited on page 16.]

[45] R. Ditchfield, W. J. Hehre, and J. A. Pople, The Journal of Chemical Physics **54**, 724 (1971). [Cited on page 17.]

[46] D. E. Woon and J. Dunning, Thom H., The Journal of Chemical Physics **98**, 1358 (1993). [Cited on page 18.]

[47] J. Dunning, Thom H., K. A. Peterson, and A. K. Wilson, The Journal of Chemical Physics **114**, 9244 (2001). [Cited on page 18.]

[48] A. K. Wilson, D. E. Woon, K. A. Peterson, and J. Dunning, Thom H., The Journal of Chemical Physics **110**, 7667 (1999). [Cited on page 18.]

[49] B. P. Prascher, D. E. Woon, K. A. Peterson, T. H. Dunning, and A. K. Wilson, Theoretical Chemistry Accounts **128**, 69 (2011). [Cited on page 18.]

[50] J. Dunning, Thom H., The Journal of Chemical Physics **90**, 1007 (1989). [Cited on page 18.]

[51] J. C. Slater, Phys. Rev. **32**, 339 (1928). [Cited on page 18.]

[52] J. A. Gaunt, Mathematical Proceedings of the Cambridge Philosophical Society **24**, 328 (1928). [Cited on page 18.]

[53] H. B. Schlegel, The Journal of Chemical Physics **84**, 4530 (1986). [Cited on page 22.]

[54] J. S. Andrews, D. Jayatilaka, R. G. Bone, N. C. Handy, and R. D. Amos, Chemical Physics Letters **183**, 423 (1991). [Cited on page 22.]

[55] C. C. J. Roothaan, Rev. Mod. Phys. **32**, 179 (1960). [Cited on page 22.]

[56] T. Tsuchimochi and G. E. Scuseria, The Journal of Chemical Physics **133**, 141102 (2010). [Cited on page 22.]

[57] C. C. J. Roothaan, Rev. Mod. Phys. **23**, 69 (1951). [Cited on page 22.]

[58] G. G. Hall, Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences **205**, 541 (1951). [Cited on page 22.]

[59] S. Lehtola, Phys. Rev. A **101**, 012516 (2020). [Cited on page 24.]

[60] S. Lehtola, Journal of Chemical Theory and Computation **15**, 1593 (2019). [Cited on page 24.]

[61] R. Hoffmann, The Journal of Chemical Physics **39**, 1397 (1963). [Cited on page 24.]

[62] C. Møller and M. S. Plesset, Phys. Rev. **46**, 618 (1934). [Cited on page 25.]

[63] O. Sïnanoğlu, The Journal of Chemical Physics **36**, 706 (1962). [Cited on page 26.]

[64] J. Čížek, The Journal of Chemical Physics **45**, 4256 (1966). [Cited on page 26.]

[65] D. Usvyat, K. Sadeghian, L. Maschio, and M. Schütz, Phys. Rev. B **86**, 045412 (2012). [Cited on page 26.]

[66] T. Tsatsoulis, S. Sakong, A. Groß, and A. Grüneis, The Journal of Chemical Physics **149**, 244105 (2018). [Cited on page 26.]

[67] T. Tsatsoulis et al., J Chem Phys **146**, 204108 (2017). [Cited on page 26.]

[68] A. Kubas et al., J Phys Chem Lett **7**, 4207 (2016). [Cited on page 26.]

[69] A. D. Boese and J. Sauer, Journal of Computational Chemistry **37**, 2374 (2016). [Cited on page 26.]

[70] L. Meissner, K. Jankowski, and J. Wasilewski, International Journal of Quantum Chemistry **34**, 535 (1988). [Cited on page 26.]

[71] M. Abe et al., Phys. Rev. A **90**, 022501 (2014). [Cited on page 26.]

[72] H. Sekino and R. J. Bartlett, International Journal of Quantum Chemistry **26**, 255 (1984). [Cited on page 26.]

[73] O. Christiansen, The Journal of Chemical Physics **120**, 2149 (2004). [Cited on page 26.]

[74] S. Perera and R. J. Bartlett, Chemical Physics Letters **314**, 381 (1999). [Cited on page 26.]

[75] J. Cullen, Chemical Physics **202**, 217 (1996). [Cited on page 26.]

[76] M. H. Kalos, Phys. Rev. **128**, 1791 (1962). [Cited on page 28.]

[77] W. M. C. Foulkes, L. Mitas, R. J. Needs, and G. Rajagopal, Rev. Mod. Phys. **73**, 33 (2001). [Cited on page 28.]

[78] J. B. Anderson, The Journal of Chemical Physics **63**, 1499 (1975). [Cited on page 28.]

[79] G. H. Booth, A. J. W. Thom, and A. Alavi, The Journal of Chemical Physics **131**, 054106 (2009). [Cited on pages 28, 29, and 76.]

[80] D. Cleland, G. H. Booth, and A. Alavi, The Journal of Chemical Physics **132**, 041103 (2010). [Cited on pages 28, 29, and 76.]

[81] G. H. Booth, D. Cleland, A. J. W. Thom, and A. Alavi, The Journal of Chemical Physics **135**, 084104 (2011). [Cited on pages 29 and 76.]

[82] G. H. Booth and A. Alavi, The Journal of Chemical Physics **132**, 174104 (2010). [Cited on page 29.]

[83] D. M. Cleland, G. H. Booth, and A. Alavi, The Journal of Chemical Physics **134**, 024112 (2011). [Cited on pages 29 and 100.]

[84] J. Brand, M. Yang, and E. Pahl, Phys. Rev. B **105**, 235144 (2022). [Cited on page 29.]

[85] M. Yang, M. Čufar, E. Pahl, and J. Brand, Condensed Matter **7** (2022). [Cited on page 29.]

[86] R. J. Anderson, C. J. C. Scott, and G. H. Booth, Phys. Rev. B **106**, 155158 (2022). [Cited on page 29.]

[87] J. C. Greer, The Journal of Chemical Physics **103**, 1821 (1995). [Cited on pages 29, 30, and 76.]

[88] J. P. Coe, D. J. Taylor, and M. J. Paterson, The Journal of Chemical Physics **137**, 194111 (2012). [Cited on pages 30 and 76.]

[89] J. P. Coe and M. J. Paterson, The Journal of Chemical Physics **137**, 204108 (2012). [Cited on page 30.]

[90] P.-O. Löwdin, Phys. Rev. **97**, 1474 (1955). [Cited on page 30.]

[91] J. P. Coe, D. J. Taylor, and M. J. Paterson, Journal of Computational Chemistry **34**, 1083 (2013). [Cited on pages 30 and 100.]

[92] J. P. Coe and M. J. Paterson, The Journal of Chemical Physics **139**, 154103 (2013). [Cited on page 30.]

[93] J. P. Coe, Journal of Chemical Theory and Computation **19**, 874 (2023). [Cited on page 30.]

[94] E. Schrödinger, Naturwissenschaften **14**, 664 (1926). [Cited on page 30.]

[95] T. D. Lee, F. E. Low, and D. Pines, Phys. Rev. **90**, 297 (1953). [Cited on page 30.]

[96] P. W. Anderson, Phys. Rev. **110**, 827 (1958). [Cited on page 30.]

[97] J. G. Valatin and D. Butler, Il Nuovo Cimento (1955-1965) **10**, 37 (1958). [Cited on page 30.]

[98] J. Schwinger, Proceedings of the National Academy of Sciences of the United States of America **37**, 452 (1951). [Cited on page 30.]

[99] E. C. G. Sudarshan, Phys. Rev. Lett. **10**, 277 (1963). [Cited on pages 31 and 58.]

[100] J. R. Klauder, Annals of Physics **11**, 123 (1960). [Cited on pages 31 and 41.]

[101] A. O. Barut and L. Girardello, Communications in Mathematical Physics **21**, 41 (1971). [Cited on page 31.]

[102] R. Gilmore, Revista Mexicana de Física **23**, 143 (1974). [Cited on pages 32 and 33.]

[103] C. Aragone, G. Guerri, S. Salamo, and J. Tani, Journal of Physics A: Mathematical, Nuclear and General **7**, L149 (1974). [Cited on page 32.]

[104] M. M. Nieto and L. M. Simmons, Phys. Rev. Lett. **41**, 207 (1978). [Cited on page 32.]

[105] M. M. Nieto and L. M. Simmons, Phys. Rev. D **20**, 1321 (1979). [Cited on page 32.]

[106] R. Gilmore, Annals of Physics **74**, 391 (1972). [Cited on pages 33 and 58.]

[107] A. M. Perelomov, Soviet Physics Uspekhi **20**, 703 (1977). [Cited on pages 33 and 40.]

[108] H. Weyl, Monatshefte für Mathematik und Physik (1928). [Cited on page 35.]

[109] D. V. Shalashilin and M. S. Child, Chemical Physics **304**, 103 (2004), Towards Multidimensional Quantum Reaction Dynamics. [Cited on pages 37, 58, 61, and 63.]

[110] W. Louisell, *Quantum Statistical Properties of Radiation*, A Wiley-Interscience publication, Wiley, 1973. [Cited on page 37.]

[111] A. Perelomov, *Generalized Coherent States and Their Applications*, Springer Berlin Heidelberg, Berlin, Heidelberg, 1 edition, 1986. [Cited on page 37.]

[112] J. J. Sakurai, *Modern quantum mechanics*, Cambridge University Press, Cambridge, third edition. edition, 1994. [Cited on page 37.]

[113] A. Grigolo, T. F. Viscondi, and M. A. M. de Aguiar, The Journal of Chemical Physics **144**, 094106 (2016). [Cited on page 37.]

[114] K. Hepp and E. H. Lieb, Annals of Physics **76**, 360 (1973). [Cited on page 38.]

[115] K. Nemoto, Journal of Physics A: Mathematical and General **33**, 3493 (2000). [Cited on pages 40 and 41.]

[116] D. V. Makhov, C. Symonds, S. Fernandez-Alberti, and D. V. Shalashilin, Chemical Physics **493**, 200 (2017). [Cited on pages 41 and 60.]

[117] M. Mathur and H. S. Mani, Journal of Mathematical Physics **43**, 5351 (2002). [Cited on page 41.]

[118] D. Dahlbom, C. Miles, H. Zhang, C. D. Batista, and K. Barros, Phys. Rev. B **106**, 235154 (2022). [Cited on page 41.]

[119] S.-H. Do et al., npj Quantum Materials **8**, 5 (2023). [Cited on page 41.]

[120] J. M. Radcliffe, Journal of Physics A: General Physics **4**, 313 (1971). [Cited on page 41.]

[121] R. Oeckl, Journal of Physics A: Mathematical and Theoretical **48**, 035203 (2014). [Cited on pages 41 and 126.]

[122] M. Lüscher, Communications in Mathematical Physics **54**, 283 (1977). [Cited on pages 41 and 126.]

[123] R. Balian and E. Brezin, Il Nuovo Cimento B (1965-1970) **64**, 37 (1969). [Cited on page 42.]

[124] S. Helgason, *Differential geometry, Lie groups, and symmetric spaces*, Pure and applied mathematics ; 80, Academic Press, New York ;, 1978. [Cited on page 42.]

[125] F. A. Berezin, Communications in Mathematical Physics **63**, 131 (1978). [Cited on page 42.]

[126] R. Gilmore and D. Hsuan Feng, Progress in Particle and Nuclear Physics **9**, 479 (1983). [Cited on page 42.]

[127] L. keng Hua, Harmonic analysis of functions of several complex variables in the classical domains, 1963. [Cited on page 42.]

[128] R. Oeckl, Journal of Physics A: Mathematical and Theoretical **48**, 035203 (2015). [Cited on page 43.]

[129] R. Delbourgo, P. D. Jarvis, and R. C. Warner, Journal of Mathematical Physics **34**, 3616 (1993). [Cited on page 48.]

[130] A. I. Streltsov, O. E. Alon, and L. S. Cederbaum, Phys. Rev. A **81**, 022124 (2010). [Cited on page 49.]

[131] M. Ben-Nun and T. J. Martínez, *Ab Initio Quantum Molecular Dynamics*, pages 439–512, Advances in Chemical Physics, 2002. [Cited on pages 60 and 61.]

[132] S.-Y. Ye, D. Shalashilin, and A. Serafini, Phys. Rev. A **86**, 032312 (2012). [Cited on page 60.]

[133] D. Huber and E. J. Heller, The Journal of Chemical Physics **89**, 4752 (1988). [Cited on page 63.]

[134] Q. Sun et al., Pyscf: the python-based simulations of chemistry framework, 2017. [Cited on pages 64, 67, 71, 84, 91, 92, 93, 98, 99, and 131.]

[135] W. A. Al-Saidi, S. Zhang, and H. Krakauer, The Journal of Chemical Physics **124**, 224101 (2006). [Cited on pages 69 and 91.]

[136] G. Li Manni, S. D. Smart, and A. Alavi, Journal of Chemical Theory and Computation **12**, 1245 (2016), PMID: 26808894. [Cited on page 69.]

[137] E. Vitale, A. Alavi, and D. Kats, Journal of Chemical Theory and Computation **16**, 5621 (2020). [Cited on page 76.]

[138] S. Ravichandiran, *Hands-on deep learning algorithms with python master deep learning algorithms with extensive math by implementing them using TensorFlow*, Packt, Birmingham, 2019. [Cited on page 79.]

[139] P. Tseng and S. Yun, Mathematical programming **117**, 387 (2009). [Cited on pages 79 and 116.]

[140] L. Armijo, Pacific Journal of Mathematics **16**, 1 (1966). [Cited on page 80.]

[141] N. I. of Standards and Technology, Fundamental physics constants, https://physics.nist.gov/cgi-bin/cuu/Value?hr, 2018, [Accessed: 27.04.2024]. [Cited on page 84.]

[142] A. F. Al-Refaie and J. Tennyson, Computer Physics Communications **221**, 53 (2017). [Cited on page 101.]

[143] R. T. Marler and J. S. Arora, Structural and Multidisciplinary Optimization **41**, 853 (2010). [Cited on page 102.]

[144] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`. [Cited on page 102.]

[145] D. J. ROWE, Rev. Mod. Phys. **40**, 153 (1968). [Cited on page 104.]

[146] H. J. Monkhorst, International Journal of Quantum Chemistry **12**, 421 (1977). [Cited on page 104.]

[147] J. F. Stanton and R. J. Bartlett, The Journal of Chemical Physics **98**, 7029 (1993). [Cited on page 104.]

[148] E. Dalgaard and H. J. Monkhorst, Phys. Rev. A **28**, 1217 (1983). [Cited on page 104.]

[149] H. Koch et al., The Journal of Chemical Physics **92**, 4924 (1990). [Cited on page 104.]

[150] H. Koch and P. Jo/rgensen, The Journal of Chemical Physics **93**, 3333 (1990). [Cited on page 104.]

[151] H. Koch, H. J. A. Jensen, P. Jo/rgensen, and T. Helgaker, The Journal of Chemical Physics **93**, 3345 (1990). [Cited on page 104.]

[152] R. Kobayashi, H. Koch, and P. Jφrgensen, Chemical Physics Letters **219**, 30 (1994). [Cited on page 104.]

[153] H. Koch, R. Kobayashi, A. Sanchez de Merás, and P. Jo/rgensen, The Journal of Chemical Physics **100**, 4393 (1994). [Cited on page 104.]

[154] G. Das and A. C. Wahl, The Journal of Chemical Physics **44**, 87 (1966). [Cited on page 104.]

[155] B. O. Roos, P. R. Taylor, and P. E. Sigbahn, Chemical Physics **48**, 157 (1980). [Cited on page 104.]

[156] B. O. Roos, International Journal of Quantum Chemistry **18**, 175 (1980). [Cited on page 104.]

[157] G. Das, The Journal of Chemical Physics **58**, 5104 (1973). [Cited on page 104.]

[158] S. R. Langhoff and E. R. Davidson, International Journal of Quantum Chemistry **8**, 61 (1974). [Cited on page 104.]

[159] P. G. Szalay, T. Müller, G. Gidofalvi, H. Lischka, and R. Shepard, Chemical Reviews **112**, 108 (2012). [Cited on page 104.]

[160] H. Lischka et al., Chemical Reviews **118**, 7293 (2018). [Cited on page 104.]

[161] J. Caillat et al., Phys. Rev. A **71**, 012712 (2005). [Cited on page 116.]

[162] O. Koch, W. Kreuzer, and A. Scrinzi, Applied Mathematics and Computation **173**, 960 (2006). [Cited on page 116.]

[163] M. Nest, T. Klamroth, and P. Saalfrank, Zeitschrift für Physikalische Chemie **224**, 569 (2010). [Cited on page 116.]

[164] O. E. Alon, A. I. Streltsov, and L. S. Cederbaum, The Journal of Chemical Physics **127**, 154103 (2007). [Cited on page 116.]

[165] O. E. Alon, A. I. Streltsov, and L. S. Cederbaum, Phys. Rev. A **76**, 062501 (2007). [Cited on page 116.]

[166] O. E. Alon et al., Chemical Physics **401**, 2 (2012), Recent advances in electron correlation methods and applications. [Cited on page 116.]

[167] I. S. Ulusoy and M. Nest, The Journal of Chemical Physics **136**, 054112 (2012). [Cited on page 116.]

[168] H. Kono, S. Ohmura, T. Kato, H. Ohmura, and S. Koseki, Journal of Physics: Conference Series **1412**, 042004 (2020). [Cited on page 116.]

[169] E. Cartan, *La théorie des groupes finis et continus et l'analysis situs*, volume 42 of *Mémorial des sciences mathématiques*, Gauthier-Villars, 1930. [Cited on page 124.]

[170] T. Koda, An introduction to the geometry of homogeneous spaces, in *Proceedings of The Thirteenth International Workshop on Differential Geometry*, volume 13, pages 121–144, 2009. [Cited on page 125.]

[171] H. Weyl, *The Classical Groups: Their Invariants and Representations*, Number no. 1, pt. 1 in Princeton Landmarks in Mathematics and Physics, Princeton University Press, 1946. [Cited on page 125.]

[172] H.-J. Werner et al., Molpro, version 2019.2, a package of ab initio programs, 2019, see. [Cited on page 131.]

[173] Free Software Foundation Inc., Iso_fortran_env (the gnu fortran compiler), https://gcc.gnu.org/onlinedocs/gfortran/ISO_005fFORTRAN_005fENV.html, 2022, Accessed: 27.03.2024. [Cited on page 134.]

[174] Z. Zaikun, infnan, https://github.com/equipez/infnan, 2023, Accessed: 6.03.2023. [Cited on page 137.]

[175] C. C. Symonds, Development and applications of new basis set sampling and basis set handling procedures for the coupled coherent states family of methods, 2015. [Cited on pages 137 and 166.]

[176] R. Chandler and P. Northrop, Random number generation, 2003. [Cited on pages 137 and 138.]

[177] R. Code, Combinations — rosetta code, https://rosettacode.org/wiki/Combinations?oldid=348688, 2022, [Accessed: 18.07.2022]. [Cited on page 139.]

[178] T. J. H. Hele, An electronically non-adiabatic generalization of ring polymer molecular dynamics, Master's thesis, Exeter College, University of Oxford, 2011. [Cited on page 150.]

[179] K. B. Petersen and M. S. Pedersen, The matrix cookbook, 2012, Version 20121115. [Cited on page 175.]