MACHINE LEARNING AND COUPLED CLUSTER THEORY APPLIED TO INFINITE MATTER

By

Julie Lynn Butler

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Physics—Doctor of Philosophy

2023

ABSTRACT

An initio many-body methods, such as coupled cluster theory, are able to make accurate predictions on systems, even if no experimental data for the system exists. This makes them an invaluable tool for studying exotic nuclei or systems, such as infinite matter, where experimental results are sparse. However, the performance and accuracy of a coupled cluster calculation suffer from truncations of the cluster operator and basis truncations, both of which are needed to reduce the scale of the problem such that it is computationally feasible. Additionally, when coupled cluster theory is applied to systems of infinite matter (the homogeneous electron gas and infinite nuclear matter), the number of particles in the system must also be truncated to a finite number, this introduces another source of error in the calculations. Finally, when modeling all nuclear systems, including infinite nuclear matter, the choice of nuclear interaction can greatly affect both the results and the computational run time. Simple nuclear interactions, such as the Minnesota potential, have comparatively small run times but lack the accuracy of computationally more complex interactions which are derived from effective field theory.

The goal of this thesis is to improve the accuracy of coupled cluster calculations applied to two infinite matter systems: the homogeneous electron gas and two infinite nuclear matter system (pure neutron matter and symmetric nuclear matter). Coupled cluster calculations at the doubles and triples levels will be compared to determine if the increase in computational time is worth the increase in accuracy. Additionally, two different nuclear interactions will be tested on calculations of pure neutron matter: a toy model called the Minnesota potential, which is computationally simple, and a much more complex set of optimized interactions which are derived from effective field theory, which increase the accuracy but also the computational run time.

Finally, the main part of this thesis is devoted to the development of a simple machine learning algorithm that can accurately extrapolate coupled cluster calculations of infinite systems separately to the complete basis limit and the thermodynamic limit. This algorithm, known as sequential regression extrapolation, combines a Bayesian machine learning algorithm with a unique way of formatting the training data to create a powerful and accurate extrapolate that can be trained on very

little data, does not require hyperparameter tuning, and can automatically produce uncertainties on its predictions. With this method, we are able to accurately predict the coupled cluster correlation energies of infinite matter systems accurately in the complete basis and thermodynamic limits while savings months of computational time in the process.

For Adam, who has never stopped encouraging me to pursue my dreams

ACKNOWLEDGEMENTS

Graduate school has been an amazing journey and, luckily, a journey that I have not had to take alone. All the people listed below have contributed in some way and helped make this dissertation a reality. It has been a dream of mine since I was in high school to get a Ph.D.; there is no way I could have done it without these fantastic people.

There is no other way to start this acknowledgments section than by extending a huge thank you to my advisor, Morten Hjorth-Jensen. Morten is the best teacher and mentor I could have asked for. He has influenced my research, teaching, and outlook on the world more than I could have imagined. Morten, while I may no longer be your student, I will always look to you as an example of what a professor and a research advisor should be.

I want to thank my committee members Scott Bogner, Johannes Pollanen, Danny Caballero, and Sean Liddick for their unending patience and assistance throughout this process. I also must thank Heiko Hergert for introducing me to many-body theory during my first summer at MSU and aiding me throughout this journey. I have also been lucky to share this journey as a graduate student with my group members, Yani Udiani, Jane Kim, Danny Jammooa, and Kang Yu. Being able to experience all of the aspects of being a nuclear theory graduate student with you has been a pleasure.

I would also like to acknowledge my collaborators at Oak Ridge National Laboratory, Gustav Jansen and Justin Leitz, without whom the coupled cluster results presented in this thesis would not be possible. Not only did you make significant contributions to my research, but I also look up to you as scientists and mentors. Both of you made my time living in Oak Ridge much more manageable and enjoyable.

Next, I want to acknowledge Kim Crosslan and Elizabeth Delinski for all of their assistance over the past five years. There is no way I could have navigated all of the different aspects of graduate school without your help.

Graduate school would have been much harder without the encouragement and support of my family. I have to thank my parents, Jimmy and Linda, for their support throughout this process.

My sister, Bre, and my brother-in-law, Graham, have been an unending source of encouragement when things got hard. I must also acknowledge the family I gained through marriage: Chris, Mary Jane, Aaron, Nana, Papa, Nannie, and Pawpaw. There is no way I could have done all I have over the past five years if you were not there to support me.

I want to thank the friends that have been with me since college (Courtney, Connor, Mikhayla, Will) and the friends I have made in graduate school (Eric, Nick, and Ana) for always encouraging me and believing in me even when I did not believe in myself. Graduate school would have been a much more lonely place without all of the company and good times you provide.

I also have to acknowledge to contributions of my dog, Lord Kelvin, to my mental health and sanity throughout all of the virtual classes and work-from-home days I have endured as a graduate student.

Finally, I must acknowledge the person this thesis is dedicated to: my husband, Adam. There is no way I can ever pay you back for all of the assistance you have provided and the sacrifices you have made to help me get this degree, but I dedicate this thesis to you as an acknowledgment of everything you have done.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

A	Number of Particles
BRR	Bayesian Ridge Regression
CC	Coupled Cluster Theory
CCD	Coupled Cluster Doubles
CCDT	Coupled Cluster Doubles and Triples
CCD(T)	Coupled Cluster Doubles with perturbative triples
CCDT-1	Coupled Cluster Doubles and Triples approximation
CCS	Coupled Cluster Singles
CCSD	Coupled Cluster Singles and Doubles
CCSDT	Coupled Cluster Sinlges, Doubles, and Triples
GP	Gaussian Processes
HEG	Homogeneous Electron Gas
HF	Hartree-Fock Theory
ICER	Institute for Cyber-Enabled Computing
INM	Infinite Nuclear Matter
KRR	Kernel Ridge Regression
LR	Linear Regression
M	Number of Single Particle States
MBPT	Many-Body Perturbation Theory
ML	Machine Learning
MSU	Michigan State University
N	Number of Particles
NN	Neural Network
PNM	Pure Neutron Matter

RNN Recurrent Neural Network

RR Ridge Regression

SNM Symmetric Nuclear Matter

TDL Thermodynamic Limit

CHAPTER 1

INTRODUCTION

Describing a system of many-interacting particles is difficult in the context of classical mechanics. It becomes even more complicated when the quantum mechanical properties of the particles are included. The only quantum mechanical many-body problems we can solve exactly are simplified problems, and thus we must develop a framework to approximately solve the many-body problem for large systems of interest, such as atomic nuclei and infinite matter systems [1]. This is the basis of the field of many-body physics, which intersects with nuclear theory, condensed-matter theory, quantum chemistry, and other subfields of chemistry and physics.

Specifically, we will look at *ab initio*, or first principles, many-body methods where the calculations start from a given Hamiltonian and attempt to solve the many-body problem without allowing any uncontrolled approximation. These first principle calculations are essential in nuclear physics and quantum chemistry calculations. It should be noted that while a proper *ab initio* calculation in nuclear physics would begin with the quark and gluon degrees of freedom, in this thesis we use *ab initio* to refer to a group of many-body methods which can which can be improved systematically given the Hamiltonian of the systems and relevant laws of motion.

Though *ab initio* many-body calculations can be expensive, both in terms of computational time and computational resources, they play an important role in many-body physics for a few reasons. First, performing these calculations at a fine-grained and controllable level allows us to probe new and interesting particle interactions and challenge existing computational methods. Second, these first principle methods allow theoretical many-body physicists to guide new experiments and interpret the results from these experiments. Furthermore, since these calculations start from first principles, these *ab initio* many-body methods can make predictions on systems where no experimental data exists [2].

In nuclear physics, the results from performing an *ab initio* calculation on a light nucleus are expected to yield a binding energy that has an error of less than a few percent when compared to experimental results. While this is a good accuracy considering that these methods start only

from the Hamiltonian of the system, the shell model with matrix elements which are fitted to experimental data would achieve a higher level of accuracy. However, *ab initio* methods still have a place in nuclear physics as they allow us to test and expand our knowledge of the various nuclear interaction models. Additionally, they allow us to study nuclei from the nuclear chart in regions where very little data is available for phenomenological models [2]. They are many *ab initio* many-body methods available for nuclear studies, and they differ in their computational costs and the regions of the nuclear chart where they can be applied. They also have differences in flexibility and ability to deal with various Hamiltonians.

The *ab initio* many-body method which will be the focus of this thesis is coupled cluster theory, originally named exp S, which originated in the nuclear physics community over sixty years ago. Coester and Kümmel proposed an exponential ansatz that could be used to describe the correlations within a nucleus. Since its inception in the late 1950s and early 1960s, coupled cluster theory has become an important many-body method, both in its home field of nuclear physics and quantum chemistry.

There are a few reasons why coupled cluster theory is such a popular many-body method. First, coupled cluster theory is fully microscopic and is both size extensive and size consistent [2, 3]. Second, coupled cluster theory can exactly produce the fully correlated many-body wavefunction if all components of its cluster operator are considered. However, since this is usually not possible, coupled cluster theory provides a systematic truncation method that allows for hierarchical improvements. Third, though coupled cluster theory is not a variational method, its energy behaves in most cases as a variational quantity. Finally, the form of the equations in coupled cluster theory is very amenable to parallel implementation, including with MPI and utilizing graphical processing unit (GPU) computations [4].

In the 1960s and 1970s, coupled cluster theory was further developed and was applied to the field of quantum chemistry, where it continues to flourish to this day [5–7]. Though this thesis is in nuclear physics, we will take inspiration from quantum chemistry and use some of the advancements made in that field. Reference [7] provides a good review of coupled cluster theory's importance to the

field of quantum chemistry.

There are several good reviews on coupled cluster theory's history in nuclear physics and developments at that time (see Refs. [2, 8, 9]). However, we will summarize the highlights and mention some recent coupled cluster results in nuclear physics that post-date these reviews. Though coupled cluster theory was initially developed in nuclear physics, it grew slowly in the field and experienced only slow changes and growth during the second half of the 20th century [9]. This was due to a combination of inaccurate models to describe the nuclear interaction and computational limitations, which made describing nuclear systems of interest challenging and inaccurate. Heisenberg and Mihaila were the first to use a high-precision nucleon-nucleon interaction and three-nucleon forces in a coupled cluster calculation of a nucleus, specifically oxygen-16 [10]. Coupled cluster theory experienced a revival in nuclear physics in the early 2000s (see, for example, Reference [4]) and has been successfully and accurately applied to both medium mass nuclei and is now being extended to study high mass nuclei. See Ref. [2] for a review of coupled cluster calculations on medium mass nuclei among other systems and Ref. [11] for a recent result concerning coupled cluster studies of high mass nuclei.

Though coupled cluster theory in nuclear physics does differ from coupled cluster theory in quantum chemistry, an essential advancement in the revival of coupled cluster theory in nuclear physics in the early 2000s occurred when Dean and Hjorth-Jensen took the standard approach to coupled cluster theory from quantum chemistry and translated it to nuclear physics [4]. Additionally, within nuclear physics, several conceptual and theoretical developments involving nuclear interactions increased coupled cluster accuracy when applied to nuclear problems. These advances include the development of soft interactions through the application of effective field theory (EFT) and the inclusion of three-nucleon interactions. These changes, along with the development of more realistic nuclear interactions [12–19], made coupled cluster theory into a more feasible method for application to nuclear studies. Finally, increased computational technology has allowed more computational cycles for faster calculations and the option for coupled cluster calculations to be further accelerated by leveraging modern computational methods such as parallel computing and

utilizing graphical processing units (GPUs) [2, 20]. Recent advances in coupled cluster theory have allowed calculations on nuclei as large as lead-208 ([11]).

One important system which can be analyzed with coupled cluster theory is a system of infinite matter, meaning that the system contains an infinite number of particles and spans an infinite volume in space. The most important infinite matter system in nuclear physics is infinite nuclear matter, where all of the particles are protons and neutrons in various fractions. In this thesis, we will investigate two infinite nuclear matter systems: pure neutron matter, where all the particles are protons, and symmetric nuclear matter, where the particles are evenly divided between protons and neutrons.

Infinite nuclear matter has several essential applications within nuclear physics and astrophysics. Within nuclear physics, calculations of infinite nuclear matter are performed alongside calculations of finite nuclei to support the results (see, for example, [11, 17, 21]). Additionally, calculations of infinite nuclear matter are essential for determining the nuclear equation of state, which can determine the behavior of both finite nuclei and the behavior of matter in astrophysical objects. Finally, some astrophysical objects, such as neutron stars, are challenging to study using traditional observational astronomy. However, we can still conclude their behaviors and properties by performing many-body studies on their particles. The matter that makes up objects, such as neutron stars, can be modeled by various types of infinite nuclear matter approaches [22–33]. There have been several coupled cluster studies of infinite nuclear matter previously [2, 3, 34–36].

Outside of nuclear systems, this thesis will also perform coupled cluster calculations of the homogeneous electron gas (HEG), an infinite matter system containing only electrons with a positive background charge, so there is no net charge [37]. The electron gas is essential in many fields of chemistry, such as density functional theory, where it is used to benchmark calculations, and is used as a model for the electrons in superconducting metals. The HEG has been extensively studied with coupled cluster theory for decades [38–43]. For our purposes, the homogeneous electron gas will be a test case for developing methods that can be applied to infinite nuclear matter. The homogeneous electron gas, made entirely of electrons, is governed by the Coulomb force, a much simpler force

than the nuclear forces. Thus coupled cluster calculations of the homogeneous electron gas are less time-consuming than calculations of infinite nuclear matter, making it a great sandbox to build our methodologies.

While it may be evident that modeling an infinite system in a finite computational space is not feasible, this is a significant hurdle to performing accurate coupled cluster calculations on infinite matter systems. The number of single-particle states and the number of particles in the system must be truncated, introducing basis incompleteness and finite size errors, respectively. The coupled cluster calculation, specifically the coupled cluster correlation operator, must also be truncated due to computational time and resource limitations.

While traditional methods of extrapolation can be used to remove the basis incompleteness and finite size errors with some success ([38, 42, 44–50]), these methods are usually very specific to the exact system and many-body method being studied. It should also be noted that many of these studies focus on extrapolations to the thermodynamic limit and not the complete basis limit. Additionally, performing the calculations at large values of M and N to mitigate these errors is computationally prohibitive due to a significant amount of time and resources, machine learning offers a new and promising avenue for removing these errors. Machine learning is a field at the intersection of artificial intelligence and data science where the algorithms "learn" to perform a task by being given examples of the task they are to perform. Machine learning is becoming a common tool in many-body physics, and there have been many recent studies analyzing its various uses in the field [51–69]. A review is provided in Ref. [70]. The goal of this thesis is to use a class of machine learning algorithms based on Bayesian statistics and a novel method of formatting the data to create an algorithm called sequential regression extrapolation (SRE), which is capable of removing the basis incompleteness and finite size errors from coupled cluster calculations of infinite matter systems, using only complete calculations performed at small values of M and N. Additionally, we wish to develop the SRE method is such a way that it can be applied to remove truncation errors from any system and using any many-body method. Machine learning has been used to perform extrapolations in many-body calculations ([51, 54, 71–75]), and some of these

studies do attempt to use machine learning ro remove the basis incompleteness or the finite size errors, but the SRE method is unique among these studies. We will use the SRE predictions and results calculated at the complete basis limit and the thermodynamic limit to determine the accuracy of the extrapolations. We will also determine the time saved by generating a small training data set at low M and N and using the SRE algorithm to extrapolate versus performing a coupled cluster calculation near the absolute basis limit or the thermodynamic limit.

The remainder of this thesis contains seven content chapters followed by conclusions and future works chapter. Chapter 2 will develop the basics of many-body theory, including second quantization and diagrammatic methods. Chapter 3 develops Hartree-Fock theory, many-body perturbation theory, and coupled cluster theory as the many-body methods of interest in this thesis. Chapter 4 expands on the infinite matter systems studied in this work, including the three-dimensional homogeneous electron gas, pure neutron matter, and symmetric nuclear matter. Chapter 5 introduces machine learning, including neural networks, various regression algorithms, and Bayesian machine learning. Chapter 6 develops the machine learning-based extrapolation algorithm, sequential regression extrapolation (SRE), which will be used to remove the basis incompleteness and finite size errors from coupled cluster calculations of infinite matter. Finally, Chapters 7 and 8 are the results chapter, showing the success of the SRE algorithm in removing the basis incompleteness and finite size errors from infinite matter calculations, first of the electron gas (Chapter 7) and then from infinite nuclear matter (Chapter 8).

CHAPTER 2

MANY-BODY FRAMEWORK

2.1 Introduction to Many-Body Theory

Modeling a system containing many-interacting particles can be a complex problem, even in classical mechanics. This problem becomes even more complicated when the quantum mechanical properties of the particles are taken into account, as this drastically increases the system's complexity. Furthermore, when the particles are protons and neutrons, this problem becomes exceedingly tricky due to the need to include complex nuclear interaction. However, it is still possible to make meaningful studies of nuclear many-body systems using many-body methods [1].

We will start our discussion of many-particle systems by defining the many-body Schrödinger equation used to find the system's energies. The time-dependent Schrödinger equation, shown in Equation (2.1), can be used to find all of the possible wavefunctions of the many-body system. Each wavefunction describes the system in a different configuration, so a many-body system has many wavefunctions that need to be found. This can be represented as:

$$i\hbar\frac{\partial\Phi(\vec{r},t)}{\partial t} = \hat{H}\Phi(\vec{r},t), \tag{2.1}$$

where $\hbar$ is the reduced Planck's constant, Φ is a wavefunction of the system, and $\hat{H}$ is the many-body Hamiltonian, which is unique to the system being studied. In this thesis we will use the common notation where capital Greek letters (Φ and Ψ) represent many-body wavefunctions but lowercase Greek letters (ϕ and ψ) represent single-particle wavefunctions. In many cases, such as in this thesis, we remove the time dependence from the many-body problem to reduce its complexity. This reduces the Schrödinger equation to its time-independent form:

$$\hat{H}\Phi(\vec{r}) = E\Phi(\vec{r}), \tag{2.2}$$

where E is the system's energy if it is in the state represented by the wavefunction Φ. Removing the time dependence reduces the Schrödinger equation to an eigenvalue problem where the eigenvalues are the energies of the many-body system. The eigenvectors are the corresponding wavefunctions (or state vectors).

Next, we need to define the Hamiltonian of our system. The Hamiltonian for a many-particle system can be represented generally in the form shown in Equation (2.3) [35]:

$$\hat{H} = \hat{K} + \hat{V}_1 + \hat{V}_2 + \hat{V}_3 + \dots . \tag{2.3}$$

In the above equation, $\hat{K}$ is the kinetic energy of the system, $\hat{V}_1$ is an external one-body potential, $\hat{V}_2$ represents the two-body interaction and $\hat{V}_3$ represents the three-body interaction. Note that in nuclear physics there will not be an external one-body potential. For a system containing A particles, there are a maximum of A-body interactions, though, in practice, the number of interactions (which corresponds to the number of matrix elements in the Hamiltonian) is usually limited. It is common to limit the many-body Hamiltonian to the 2-body interaction level, sometimes the 3-body interaction level. It is uncommon to include interactions higher than the 3-body interaction level, especially for large systems or nuclear systems with complex interactions, since the complexity of the interactions increases drastically with the number of particles involved. Each sum also includes an increasing number of particles at higher interaction levels. For example, the kinetic energy only has A terms, but the two-body interaction sum already has $\frac{1}{2}$A(A-1) terms [1].

Once the Hamiltonian has been defined, the next step in the many-body problem is to obtain the eigenvalues and the eigenfunctions from $\hat{H}$, which is a matrix. The eigenvalues of $\hat{H}$ are the energies of the system, and the eigenvectors are the wavefunctions of the system and are thus the solutions to the time-independent Schrödinger equation shown in Equation (2.2). However, in practice, Equation (2.2) can only be solved exactly for small, uninteresting systems. Thus we must turn to many-body methods to approximately solve for the energies of a many-body system [1, 76, 77]. In the following two sections, we will develop the notation (Dirac notation) and a way to represent the many-body basis that is more convenient (occupation representation and second quantization). Then we will revisit the many-body Hamiltonian once these tools have been developed before closing the chapter. Finally, we will develop other formalisms, including normal ordering and diagrammatic methods.

2.2 Dirac Bracket Notation for Many-Body Systems

Before developing the different many-body methods, we must develop a framework for representing specific quantities found in many-body theory. This framework work is called Dirac bra-ket notation, and it dramatically simplifies many quantum mechanical expressions. It is commonly used in quantum mechanics and many-body physics because it provides a simple way to represent complex quantum mechanical operations as linear algebra problems.

The base quantity of Dirac notation is a ket. Given a column vector, $\mathbf{x}$, which contains n elements, we can denote the vector in ket notations as:

$$|\mathbf{x}\rangle = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ . \\ . \\ . \\ x_{n-1} \end{bmatrix}, \tag{2.4}$$

where we will begin numbering the elements from 0, leading to the last element having an index of n-1. In general, $\mathrm{x}_i \in \mathbb{C}$.

The corresponding bra vector to the ket is simply the Hermitian conjugate of $|\mathbf{x}\rangle$:

$$\langle \mathbf{x}| = \begin{bmatrix} x_0^* & x_1^* & x_2^* & . & . & . & x_{n-1}^* \end{bmatrix}. \tag{2.5}$$

Note that the bra vector is a row vector instead of a column vector.

In quantum mechanics, different kets represent the system's different states. These kets differ from representing quantum mechanical states in wave mechanics (where the state is represented as a function) and in matrix mechanics (where the states are represented as basis expansions). However, kets encode the same information as these other representations. In addition, kets allow invariance in the state, meaning that the choice of coordinates and basis can be chosen at any point and quickly changed [78]. For now, the bras and kets will only describe a state with one particle; we will develop a formalism to describe many-particle states as bras and kets later in this section.

It is also important to note that since the ket $|\mathbf{x}\rangle$ is a quantum mechanical state, then it belongs to an infinite dimensional Hilbert space, H ($|\mathbf{x}\rangle \in H$). The corresponding bra, $\langle\mathbf{x}|$, also belongs to the same Hilbert space [78].

Hilbert spaces are closed under linear combinations, which gives rise to the superposition of quantum states [78]. Dirac notation also provides an easy way to represent states in a superposition, meaning that they are simultaneous in two or more states, and the same state is determined upon system measurement. Given two states $|\mathbf{x}\rangle$ and $|\mathbf{y}\rangle$, we can say that state $|\mathbf{z}\rangle$ is in a superposition of $|\mathbf{x}\rangle$ and $|\mathbf{y}\rangle$ with the following:

$$|\mathbf{z}\rangle = \alpha|\mathbf{x}\rangle + \beta|\mathbf{y}\rangle, \tag{2.6}$$

where α and β are complex numbers. The bra of the above state $\langle\mathbf{z}|$ can be denoted as:

$$\langle\mathbf{z}| = \alpha^*\langle\mathbf{x} + \beta^*\langle\mathbf{y}|. \tag{2.7}$$

Note that in Equation (2.7), the coefficients of the kets are α^* and β^* (i.e., complex conjugates) instead of just α and β.

The coefficients α and β have a physical interpretation. When the state $|\mathbf{z}\rangle$ is measured and collapses into either $|\mathbf{x}\rangle$ or $|\mathbf{y}\rangle$, there is an $|\alpha|^2$ probability that the state will be found in $|\mathbf{x}\rangle$ and a $|\beta|^2$ chance that it will be found in $|\mathbf{y}\rangle$. Note that the notation $|\alpha|^2$ represents the complex modulus of α. Since both $|\alpha|^2$ and $|\beta|^2$ represent probabilities that the two different outcomes of measuring state $|\mathbf{z}\rangle$ will occur then $|\alpha|^2 + |\beta|^2 = 1$.

Inner Product

For kets $|\mathbf{x}\rangle$ and $|\mathbf{y}\rangle$, the inner product between the two will be defined as:

$$\langle\mathbf{y}|\mathbf{x}\rangle = y_0^* x_0 + y_1^* x_1 + y_2^* x_2 + \ldots + y_{n-1}^* x_{n-1}. \tag{2.8}$$

It is important to note that$\langle\mathbf{y}|\mathbf{x}\rangle = \langle\mathbf{x}|\mathbf{y}\rangle$. Also note that $\langle\mathbf{y}|\mathbf{x}\rangle$ will sometimes be referred to as the overlap between the states $\mathbf{x}$ and $\mathbf{y}$.

Many quantum mechanics bases are orthogonal and normalized and therefore called orthonormal. However, two restrictions are placed on most bases in many-body theory. First, inner products are

used to ensure that a basis is both orthogonal and normalized. In terms of inner products, this means that for each state, **i**, in the basis, the inner product of the state with itself must be one to be normalized:

$$\langle \mathbf{i}|\mathbf{i}\rangle = 1, \ \forall \ \mathbf{i}. \tag{2.9}$$

The second restriction is that the basis is orthogonal. This means that the inner product of a state **i** with another state **j** must be zero:

$$\langle \mathbf{i}|\mathbf{j}\rangle = 0, \ \forall \ \mathbf{i} \neq \mathbf{j}. \tag{2.10}$$

For a basis that is both normalized and orthogonal, the basis is said to be orthonormal, and the conditions can be represented in compact delta notation as:

$$\langle \mathbf{i}|\mathbf{j}\rangle = \delta_{ij}, \ \forall \ \mathbf{i}, \mathbf{j}. \tag{2.11}$$

Outer Product

The outer product between $|\mathbf{x}\rangle$ and $|\mathbf{y}\rangle$ can be computing using the following formula:

$$|\mathbf{x}\rangle\langle\mathbf{y}| = \begin{bmatrix} x_0y_0^* & x_0y_1^* & x_0y_2^* & . & . & . & x_0y_{n-1}^* \\ x_1y_0^* & x_1y_1^* & x_1y_2^* & . & . & . & x_1y_{n-1}^* \\ x_2y_0^* & x_2y_1^* & x_2y_2^* & . & . & . & x_2y_{n-1}^* \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ x_{n-1}y_0^* & x_{n-1}y_1^* & x_{n-1}y_2^* & . & . & . & x_{n-1}y_{n-1}^* \end{bmatrix}. \tag{2.12}$$

Another restriction on a quantum mechanical basis is that it is complete. We can define the completeness relation for a general basis as:

$$\sum_i |i\rangle\langle i| = \mathbf{I}, \tag{2.13}$$

where **I** is the identity matrix in Hilbert space H.

Tensor Product

Tensor products are essential when describing many-particle systems from one-particle state vectors (or wavefunctions). In wave mechanics, we would describe the many-particle wavefunction with N particles as

$$\Psi(\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_N) = \psi_{p_1}(\mathbf{x}_1)\psi_{p_2}(\mathbf{x}_2)...\psi_{p_A}(\mathbf{x}_N), \tag{2.14}$$

where ψ_{p_i} represent one-particle wave functions and $\mathbf{x}_i$ are the single-particle degrees of freedom (usually position and spin) [78]. These are called product states, forming a complete A-body basis [78]. In Dirac notation, we can also represent a many-particle wavefunction similarly, but in this case, we will use tensor products [78]:

$$|\Psi\rangle = |\psi_{p_1}\psi_{p_2}...\psi_{p_A}\rangle = |\psi_{p_1}\rangle \otimes |\psi_{p_2}\rangle \otimes ... \otimes |\psi_{p_A}\rangle. \tag{2.15}$$

In Dirac notation, we can represent the complete N-body basis as:

$$|\Psi\rangle = \sum_{p_1,...p_A} d_{p_1...p_A}|\psi_{p_1}...\psi_{p_A}\rangle, \tag{2.16}$$

where $d_{p_1...p_A} = \langle\psi_{p_1}...\psi_{p_A}|\Psi\rangle$ is the overlap between the states.

Note that the order of the single particle wavefunctions matters in either Equation (2.14) or in Equation (2.15). It is possible to exchange the order of the single particle wavefunctions, which will be discussed in the next section.

Operators, Expectation Values, and Matrix Elements

In Dirac notation, operators are matrices typically represented using hat notation: $\hat{o}$. To find the expectation value of this operator in Dirac notation, it is placed in between a bra and a ket:

$$\langle\psi|\hat{o}|\phi\rangle, \tag{2.17}$$

where ψ and ϕ can be the same or different wavefunction depending on the operator. This combination of bra, operator, and ket is also called a matrix element. Several operators will be defined in the next section once more framework exists to support them, but we can define one operator here. The exchange operator, $\hat{P}$, is defined such that:

$$\hat{P}|\psi_1\psi_2\rangle = \pm|\psi_2\psi_1\rangle, \tag{2.18}$$

where ψ_i are single particle wavefunction [79]. A plus sign on the right side of the equation indicates the particles are bosons, and a minus sign indicates the particles are fermions; in this work, we will only consider particles that are fermions. Now, we can define the following two matrix elements using the exchange operator and the orthonormal rules extended to a many-body basis:

$$\langle\psi_1\psi_2|\hat{P}|\psi_1\psi_2\rangle = -\langle\psi_1\psi_2|\psi_2\psi_1\rangle = 0. \tag{2.19}$$

and

$$\langle\psi_1\psi_2|\hat{P}|\psi_2\psi_1\rangle = -\langle\psi_1\psi_2|\psi_1\psi_2\rangle = -1. \tag{2.20}$$

Note that the exchange operator can work on more than two-particle states. If there are more than two particles in the many-body wavefunction, the particles the exchange operator is acting on are denoted as subscripts:

$$\hat{P}_{12}|\psi_1\psi_2\psi_3...\rangle = -|\psi_2\psi_1\psi_3...\rangle. \tag{2.21}$$

Slater Determinants

The wavefunction for an A-particle system that solves Schrödinger's equation precisely can be expressed as an expansion of Slater determinants which span a complete, usually infinite, basis [76]. However, usually, the single-particle space must be truncated only to contain M single-particle states due to computational limitations, for which an N-particle system leads to $\binom{M}{N}$ determinants [76].

We can represent out many-body wavefunction, Φ, as the following determinant:

$$\Phi = \frac{1}{N}\begin{vmatrix} \psi_1(\vec{x}_1) & \psi_2(\vec{x}_1) & . & . & . & \psi_N(\vec{x}_1) \\ \psi_1(\vec{x}_2) & \psi_2(\vec{x}_2) & . & . & . & \psi_N(\vec{x}_2) \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ \psi_1(\vec{x}_\nu) & \psi_2(\vec{x}_\nu) & . & . & . & \psi_N(\vec{x}_\nu) \end{vmatrix}, \tag{2.22}$$

where Φ is the many-body wavefunction and $\psi_n(\vec{r}_m)$ is the single-particle wavefunction of the n-th particle with spatial and spin coordinates described by $\vec{r}_m$. Each column represents a different single-particle wavefunction, and each row represents a different set of spatial and spin coordinates. Thus represented in the Slater determinant represents every particle being in every configuration.

2.3 Occupation Representation and Second Quantization

Occupation Number Representation

Up to this point, we have used $|\Phi_{p_1 \ldots p_A}\rangle$ to represent a many-fermion wavefunction with A fermions. The single-particle basis in which the single-particle wavefunctions exist is not denoted explicitly, but it contains M single-particle states (where M can be finite or infinite). There is another way to represent a many-fermion wavefunction in a way where the number of single-particle states in the basis is more explicitly shown. This is called occupation number representation. Given an ordered single particle basis, any many-fermion wavefunction can be represented as:

$$|\Phi\rangle = |n_1 n_2 n_3 .... n_M\rangle \quad n_i = 0, 1. \tag{2.23}$$

If the state n_i is occupied (i.e., contains a particle), then n_i is one. If n_i is unoccupied, then it is a zero. As an example, consider the many-fermion wavefunction made up of the single particle wavefunctions ψ_{p_1}, ψ_{p_4}, ψ_{p_7}, and ψ_{p_8}. The many-fermionic wavefunction made from these states can be represented in occupation number representation as:

$$|\Phi_{p_1 p_4 p_7 p_8}\rangle = |010010011000...\rangle. \tag{2.24}$$

In the above equation, the number of trailing zeros can be finite or infinite depending on if M is finite or infinite. The total number of occupied states (states which are one) must equal the number of particles, N, in the system. So in Equation (2.24), there are four particles in the system, so there must be four ones in the many-body wavefunction. Also, note that, as previously, we start indexing the single particle states at zero instead of one.

We can represent a couple of other important wavefunctions in occupation number representation. A single particle wavefunction can be represented in occupation number representation, where only

the index corresponding to the single particle state is non-zero. For example, ψ_2 in occupation number representation is:

$$|\psi_2\rangle = |2\rangle \ = |001000....\rangle, \tag{2.25}$$

where we have introduced the shorthand $|\psi_p\rangle = |p\rangle$ [78]. We can also introduce the true vacuum state in occupation number representation, which is simply the state where all M single-particle states are unoccupied (all the indices are zero) [78]:

$$|0000000000.....\rangle = |0\rangle. \tag{2.26}$$

Occupation number representation will be a more convenient way to represent these many-fermionic wavefunctions moving forward as we develop the rest of the necessary formulas. Therefore, moving forward, a many-fermion wavefunction is assumed to be in occupation number representation unless explicitly stated otherwise.

Creation and Annihilation Operators

The annihilation and creation operators allow us to change from one many-body state to another by removing existing particles and adding new ones. Let $a_p^\dagger$ represent the fermionic creation operator acting on state p and a_p represent the fermionic annihilation operator acting on state p (p = 0, 1, ..., M). The creation and annihilation operators are Hermitian adjoints of each other ($(a_p^\dagger)^\dagger = a_p$ and $(a_p)^\dagger = a_p^\dagger$).

2.3.0.1 Annihilation and Creation Operators on Single-Particle States

When acting on the single particle state $|p\rangle$, the annihilation operator a_p removes the particle at state p, resulting in the true vacuum state. When the creation operator $a_p^\dagger$ acts on state $|p\rangle$, it results in an answer of zero since it cannot create an occupied state at index p if one already exists. Thus we have:

$$a_p|p\rangle = |0\rangle \ and \ a_p^\dagger|p\rangle = 0. \tag{2.27}$$

When the creation operator $a_p^\dagger$ acts on the true vacuum state $|0\rangle$, it creates the single-particle state $|p\rangle$. On the other hand, when the annihilation operator a_p acts on the true vacuum state, it answers

zero since it cannot create an unoccupied state at index p if it is already unoccupied. Then:

$$a_p|0\rangle = 0 \; and \; a_p^\dagger|0\rangle = |p\rangle. \tag{2.28}$$

Annihilation and Creation Operators on Many-Particle States

The annihilation and creation operators are applied to the many-fermion states similarly to the single-particle states. For example, when the creation operator $a_p^\dagger$ is applied to a many-fermion state, it returns the same state with p now occupied (p=1) if p was unoccupied and 0 if state p was already occupied. However, there is now an additional phase factor that needs to be accounted for.

$$a_p^\dagger|...p...\rangle = \begin{cases} (-1)^m|...1...\rangle, & \text{if } p = 0 \\ 0, & \text{if } p = 1 \end{cases} \tag{2.29}$$

In Equation (2.29), m refers to the number of occupied states prior to state p and determines the phase factor. The annihilation operator is applied to a many-fermion state in the same manner:

$$a_p|...p...\rangle = \begin{cases} (-1)^m|...0...\rangle, & \text{if } p = 1 \\ 0, & \text{if } p = 0 \end{cases}. \tag{2.30}$$

Annihilation and creation operators can be applied in chains to a given many-body state. In this case, the rightmost operator acts first, then the next rightmost operator acts on the result, and so on. Below is an example of how chains of operators can act on a given many-body state to create an entirely new state.

$$\begin{aligned} & a_7^\dagger a_6 a_5^\dagger a_3^\dagger a_3|1001101010\rangle \\ &= a_7^\dagger a_6 a_5^\dagger a_3^\dagger|1001101010\rangle \\ &= a_7^\dagger a_6 a_5^\dagger|1000101010\rangle \\ &= a_7^\dagger a_6|1001101010\rangle \\ &= a_7^\dagger|1001111010\rangle \\ &= |1001110110\rangle \end{aligned} \tag{2.31}$$

Every many-fermion state can be created from the true vacuum state with a chain of creation operators.

$$|\Phi_{p_1 \ldots p_A}\rangle = a_1^\dagger \ldots a_A^\dagger |0\rangle \tag{2.32}$$

Using these creation operators, we can create a critical state: the Fermi vacuum state represented as $|\Psi_0\rangle$. For an N particle system, the Fermi vacuum state occupies the N lowest energy states, and the remaining single-particle states are unoccupied. We can create the Fermi vacuum state by applying N creation operators to the actual vacuum state:

$$|\Phi_0\rangle = a_1^\dagger a_2^\dagger \ldots a_N^\dagger |0\rangle. \tag{2.33}$$

By defining the Fermi vacuum state, we also define the Fermi level. The Fermi level is the line that separates the highest energy-occupied single particle state from the lowest energy-unoccupied single particle state in the Fermi vacuum state. We also will denote a new way to label single particle states based on the Fermi vacuum state. States which are occupied in the Fermi vacuum state (i.e., are below the Fermi level) will be labeled with indices i, j, k, l, ... States which are unoccupied in the Fermi vacuum state (i.e., are above the Fermi level) are labeled with the indices a, b, c, d,... . States which are labeled with the indices p, q, r, s, ... can be either above or below the Fermi level. This notation will be vital as we construct the many-body methods in the next chapter.

Anticommutation Relations

All creation and annihilation operators must obey the following two anticommutation rules. Remember that $\{A, B\} = AB + BA = BA + AB = \{B, A\}$. Thus, we have the anticommutation rules:

$$\{a_p^\dagger, a_q\} = \delta_{pq}, \tag{2.34}$$

and

$$\{a_p, a_q\} = \{a_p^\dagger, a_q^\dagger\} = 0. \tag{2.35}$$

Operators in Second Quantization

Number Operators

The first operator we will look at in the second quantization is the number operator. The number operator is defined as:

$$\hat{N}_p = a_p^\dagger a_p. \tag{2.36}$$

When the number operator acts on a many-body state, the result is:

$$\hat{N}_p|...p...\rangle = \begin{cases} p|...p...\rangle, & \text{if } p = 1 \\ 1, & \text{if } p = 0 \end{cases}. \tag{2.37}$$

Since the many-body state is unchanged on both sides of the equation for p=1, the many-body state is an eigenket of the number operator if index p is occupied. Note that $\hat{N}_p$ conserves particle number (because there are an equal number of creation and annihilation operators) and is Hermitian.

The total number operator is defined as the sum of number operators overall single-particle indices:

$$\hat{N} = \sum_p \hat{N}_p. \tag{2.38}$$

When $\hat{N}$ acts on a many-body state, it returns the number of fermions in that state. For example, when the number operator is applied to the A-body state $|\Psi\rangle$:

$$\hat{N}|\Psi\rangle = A|\Psi\rangle. \tag{2.39}$$

Note that $|\Psi\rangle$ is an eigenket of the number operator.

Hamiltonian and k-Body Operators

As in the first section of this chapter, we can define the Hamiltonian for a many-body system as:

$$\hat{H} = \hat{K} + \hat{V}_2 + \hat{V}_3 + ..., \tag{2.40}$$

where $\hat{K}$ is the one-body operator (the kinetic energy), $\hat{V}_2$ is the two-body operator, $\hat{V}_3$ is the three-body operator, and so on. There are, at most, A operators in the Hamiltonian for an N-body system.

Second quantization provides us with two ways to represent these k-body operators: Goldstone and Hugenholtz. In the Goldstone form, the k-body operator can be represented as:

$$\hat{V}_k = (\frac{1}{k!})^2 \sum_{\substack{p_1...p_k \\ q_1...q_k}} \langle p_1...p_k|\hat{v}_k|q_1...q_k\rangle a^\dagger_{p_1}...a^\dagger_{p_k}a_{q_k}...a_{q_1}, \tag{2.41}$$

where the matrix element is defined as:

$$\langle p_1...p_k|\hat{v}_k|q_1...q_k\rangle = \int ... \int \psi^*_{p_1}(x_1)...\psi^*_{p_k}(x_k)\hat{v}_k(x_1,...,x_k)\psi_{q_1}(x_1)...\psi_{q_k}(x_k)dx_1...dx_k \tag{2.42}$$

In Equation (2.42), we again use the single-particle wavefunctions to calculate the many-fermion matrix elements. Again, the k-body interaction in the Hugenholtz form is very similar, except the matrix elements are antisymmetrized:

$$\hat{V}_k = (\frac{1}{k!})^2 \sum_{\substack{p_1...p_k \\ q_1...q_k}} \langle p_1...p_k|\hat{v}_k|q_1...q_k\rangle_A a^\dagger_{p_1}...a^\dagger_{p_k}a_{q_k}...a_{q_1}. \tag{2.43}$$

Using the antisymmetrized matrix elements is often more convenient and compact, so we will primarily use the Hugenholtz form for this thesis.

Due to computation limitations, the maximum k-body interaction included in the calculations in this thesis is the three-body interaction. Therefore, we can explicitly define the one-body, two-body, and three-body interactions in the Hugenholtz form below.

$$\hat{K} = \sum_{pq} \langle p|\hat{k}|q\rangle_A a^\dagger_p a_q \tag{2.44}$$

$$\hat{V}_2 = \frac{1}{2} \sum_{pqrs} \rangle pq|\hat{v}_2|rs\rangle a^\dagger_p a^\dagger_q a_s a_r \tag{2.45}$$

$$\hat{V}_3 = \frac{1}{6} \sum_{pqrstu} \rangle pqr|\hat{v}_3|stu\rangle a^\dagger_p a^\dagger_q a^\dagger_r a_u a_t a_s \tag{2.46}$$

The exact form of $\hat{v}_2$ and $\hat{v}_3$ will depend on the interaction of the system being studied. However, since the systems are defined in detail in Chapter 4, they will be left in their generic forms until then.

2.4 Normal Ordering

We now see how many-body states and operators can be written in second quantization. Though second quantization simplifies many-body calculations, some can still become long and complicated with strings of creation and annihilation operators. Normal ordering simplifies these calculations by arranging chains of creation and annihilation operators such that annihilation operators are all on the right and creation operators are all on the left. We can thus define a typical ordered sequence of operators as:

$$n[\hat{o}_1 ... \hat{o}_m] = (-1)^R a_i^\dagger ... a_j^\dagger a_{j+1} ... a_m, \tag{2.47}$$

where $n[...]$ defines a normal ordered sequence of operators, $\hat{o}_i$ represents either a creation or annihilation operator and R handles the sign change for the permutations—each time two operators exchange location, R increments by one. Normal ordering is essential because it makes determining which expressions and matrix elements will yield a non-zero result easier. For example, consider the following matrix element.

$$\langle 00101 | a_4 a_1^\dagger a_0 | 10001 \rangle \tag{2.48}$$

Written in its current form, it is difficult to tell if this matrix element will be non-zero without evaluating each operator. However, if we normal order the operators, it becomes more apparent.

$$\langle 00101 | n[a_3 a_1^\dagger a_0] | 10010 \rangle = -\langle 00101 | n[a_1^\dagger a_4 a_0] | 10001 \rangle = \langle 00101 | a_1^\dagger a_0 a_4 | 10001 \rangle \tag{2.49}$$

Now it is obvious that the matrix element will yield zero as $a_1^\dagger$ acting on the bra will return zero. All operators that have been derived so far and which will be derived in the coming chapters will be written in a normal ordered form as this will drastically simplify the extensive calculations we will be performing.

A closely related concept to normal ordering is Wick's theorem, which allows for these complex many-body states and operators to be simplified further. However, Wick's theorem will not be used in future derivations, so it will not be derived here. For a thorough explanation and derivation of Wick's theorem, see Reference [79] and [80]. -

2.5 Diagrammatic Methods

Diagrammatic notation for representing Slater determinants and operators originated with Richard Feynman in the field of quantum field theory (QFT). The methods were further developed for applications in many-body physics, see, for example, Paldus and Cizek in 1975 [81]. The following will be a brief description of these diagrams. For a more thorough explanation of these diagrams, please refer to Chapter 4 of Reference [80] and Reference [81].

Diagrammatic methods are practical in studying many-body systems because the equations involved with solving these methods can get out of hand quickly. In addition, diagrammatic methods provide a more compact way to display these equations, which can hint at how specific components of the calculations will cancel or combine with other parts.

First, we will begin by representing the primary states in diagrammatic representation. First, the reference state is represented by nothing; for most systems in this thesis, the reference state will be the Fermi vacuum state ($|\Phi_0\rangle$). A one-particle one-hole excitation, $|\Phi_i^a\rangle$, is represented by two vertical lines. The line pointing upwards represents a state that is unoccupied in the Fermi vacuum state (i.e., indexed by a, b, c, ...), and a line pointing downwards represents a state that is occupied in the Fermi vacuum state (i.e., indexed by i, j, k, ...). Any line which does not have a directional arrow could represent either state (i.e., would be indexed with p, q, r, ...). Note that neither the horizontal arrangements nor spacing are significant in the representation. A two-particle, two-hole excited state, $|\Phi_{ij}^{ab}\rangle$, is represented by four vertical lines, two of which point down and two which point up. Following this pattern, an n-particle n-hole excited state is represented by $2n$ vertical lines, where n points upwards and n points downwards. figure 2.1 shows the diagrammatic representations of the Fermi vacuum state, the one-particle one-hole, and the two-particle two-hole excited states.

Representing the excited states as creation and annihilation operators acting on the Fermi vacuum state is also possible. Remember that the one-particle one-hole excited state can be written as $|\Phi_i^a\rangle = a_a^\dagger a_i|\Phi_0\rangle$ where $a^\dagger$ is a fermionic creation operator and a is the fermionic creation operator. This means that the two-particle two-hole excited state can be written as $|\Phi_{ij}^{ab}\rangle = a_a^\dagger a_b^\dagger a_j a_i|\Phi_0\rangle$. The state $a_a^\dagger a_i|\Phi_0\rangle$ can be represented the same way as $|\Phi_i^a\rangle$, but with two horizontal lines at the

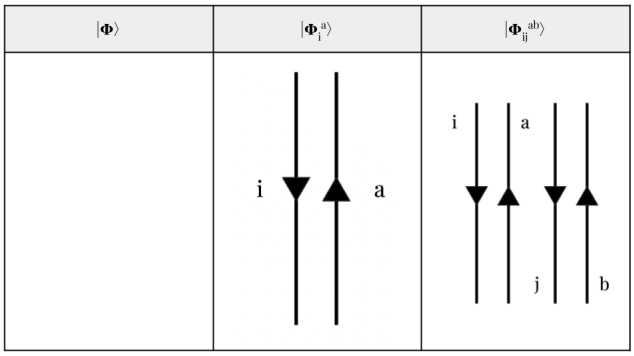


Figure 2.1 Diagrammatic representations for the Fermi vacuum state, the one-particle one-hole excited state, and the two-particle two-hole excited state

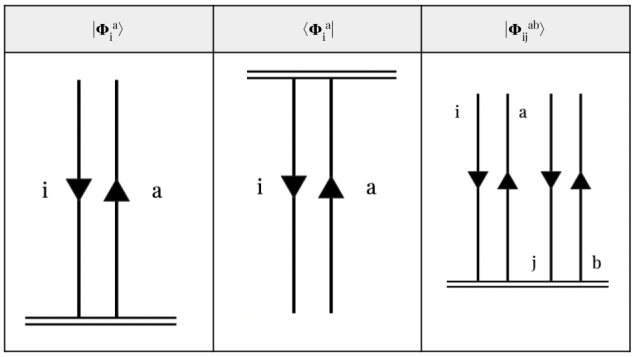


Figure 2.2 Diagrammatic representations for the one-particle one-hole and two-particle two-hole excited states written as annihilation and creation operators applied to the Fermi vacuum state

bottom to represent the Fermi vacuum state. To represent the bra of this state ($\langle\Phi_i^a| = \langle\Phi_0|a_i^\dagger a_a$) is represented in the same way, but the vertical lines representing the Fermi vacuum state are moved to the top. The two-particle two-hole excited state is represented similarly. These diagrams are shown in figure 2.2. Note that there is some ambiguity in the representation of $|\Phi_{ij}^{ab}\rangle$ in this representation, it could also represent $|\Phi_{ij}^{ba}\rangle$.

Now that we have the diagrammatic notation for the various states, we can define a notation for the various operators. Here we will explicitly define the one-body and two-body operators in diagrammatic notation, but the form of the higher-order operators will follow the same patterns.

We can define a generic one-body operator as:

$$\hat{O}_1 = \sum_{pq} \langle p|\hat{o}_1|q\rangle \{a_p^\dagger a_q\}, \tag{2.50}$$

where p and q can be occupied or unoccupied states, this one-body operator can be represented

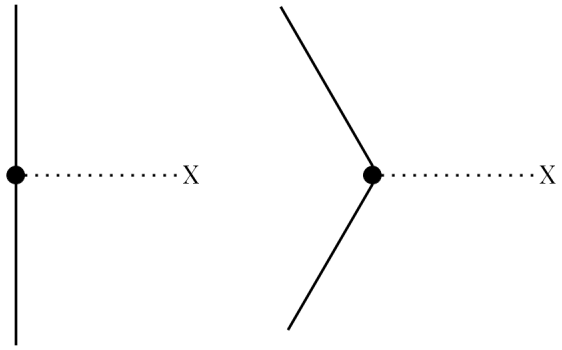


Figure 2.3 A generic representation of the one body operator. The orientation of the lines does not matter, so the diagram can be represented with both lines being entirely vertical or with the lines being slanted. Since these lines do not have arrows, they represent generic single-particle states that could be occupied or unoccupied in the Fermi vacuum state

with either diagram in Figure 2.3. The horizontal orientation of the lines does not matter so the lines can be in the purely vertical orientation, as in the left diagram, or they can be slanted, as in the right diagram. Since the lines in neither diagram of Figure 2.3 have directional arrows, the lines are represented by indices p and q.

When creating the specific diagrams for the one-body operator, there are four options, as shown in Figure 2.4. There is one diagram where both indices are from states that are unoccupied in the Fermi vacuum state, which is represented by the top left diagram and equation form as:

$$\sum_{ab} \langle a|\hat{o}_1|b\rangle \{a_a^\dagger a_b\}. \tag{2.51}$$

One diagram also results from having both states taken from the states occupied in the Fermi vacuum state. This is the diagram shown in the top right of Figure 2.4 and is represented in equation form as:

$$\sum_{ij} \langle i|\hat{o}_1|j\rangle \{a_i^\dagger a_j\}. \tag{2.52}$$

Finally, two diagrams can be created: one state is drawn from the occupied states and one from the unoccupied states. These are shown in the bottom row of Figure 2.4. The bottom left diagram is represented in equation form:

$$\sum_{ia} \langle a|\hat{o}_1|i\rangle \{a_a^\dagger a_i\}, \tag{2.53}$$

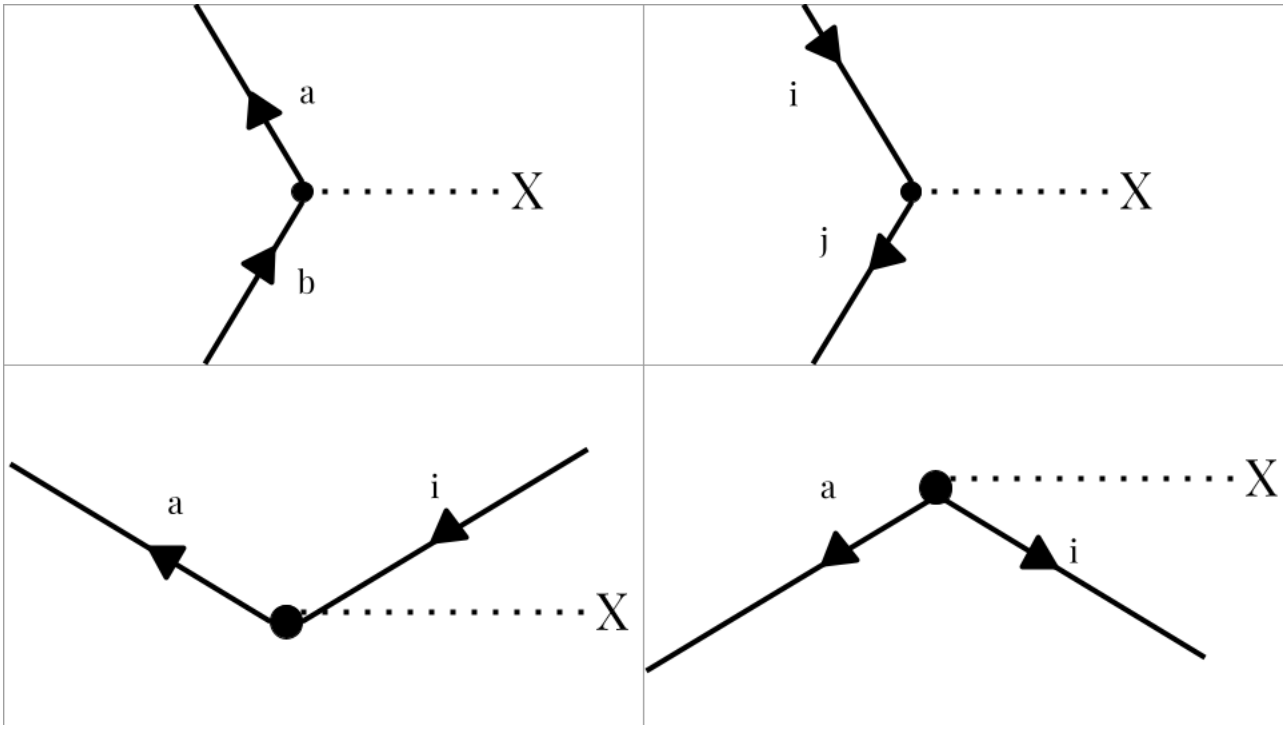


Figure 2.4 The specific diagrams for the one-body operator, with the top left diagram for two unoccupied states, the top right diagram for two unoccupied states, and the bottom two diagrams containing one occupied and one unoccupied state

the bottom right diagram in equation form is:

$$\sum_{ia} \langle i|\hat{o}_1|a\rangle \{a_i^\dagger a_a\}. \tag{2.54}$$

In the diagrams, the summation of the indices is implied using Einstein notation. In general, when creating diagrams from an equation, the index in the bra of the matrix element will exit the interaction vertex, and the index in the ket will be the line that enters the interaction vertex: $\langle left\ line|\hat{o}_1|right\ line\rangle$.

Finally, in this section, we can move onto the diagrammatic representation of a generic two-body operator, which we can represent in equation form as:

$$\hat{O}_2 = \sum_{pqrs} \langle pq|\hat{o}_2|rs\rangle \{a_p^\dagger a_q^\dagger a_s a_r\}. \tag{2.55}$$

The generic two-body diagram can be described in Figure 2.5, and the nine specific two-body diagrams can be found in Figure 2.6.

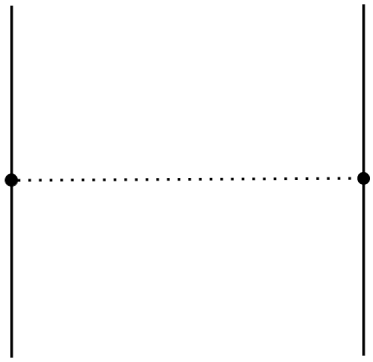


Figure 2.5 The general diagram for a two-body interaction

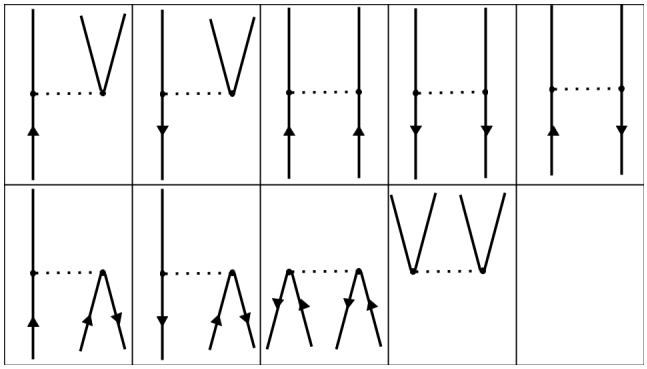


Figure 2.6 The nine specific diagrams for a two-body interaction

From these examples, we can begin to create a set of rules for creating these diagrams and converting them to algebraic expressions:

1. Lines representing occupied states in the Fermi vacuum are labeled with indices like i, j, k, ... and point downwards. Lines representing unoccupied states in the Fermi vacuum are labeled with indices a, b, c, ... and point upwards. Finally, lines that could represent either type of state are labeled with indices p, q, r, ... and have no directional arrows.

2. The horizontal arrangement and spacing of the diagrams do not matter.

3. Lines that represent the bra of the interaction matrix element leave an interaction vertex and are called external lines.

4. Lines that represent the ket of the interaction matrix element enter an interaction vertex and are called internal lines.

5. Every one body interaction vertex represents the factor $\langle out|\hat{o}_1|in\rangle$.

6. Every two-body interaction vertex represents the factor $\langle left-out\ right-out|\hat{o}_2|left-in\ right-in\rangle$.

7. If two lines start and end in the exact location, they are considered equivalent. Each set of equivalent lines in a diagram represents a factor of 1/2.

8. The sign of a diagram is calculated using $(-1)^{(n_{occ}+l)}$, where n_{occ} are the number of lines representing states that are occupied in the Fermi vacuum state and l is the number of loops in the diagram.

9. Each pair of unique external lines (i.e., the bras of the interactions) that are not connected to the same interaction picks up a permutation factor.

2.6 Conclusion

In this chapter, we have developed second quantization and diagrammatic methods, two frameworks that will allow us to represent and solve problems in many-body physics more efficiently. However, many problems of interest are still too large and complicated to be solved with the full Schrödinger's equation approach, even with the advancements made in this chapter. Therefore, in addition to these two frameworks, we need to develop a set of many-body methods, which will provide approximate solutions to the Schrödinger equation, but which will be able to describe much larger systems of interest. In the next chapter, three of these many-body methods will be developed–Hartree-Fock theory, many-body perturbation theory, and coupled cluster theory–with coupled cluster theory being particularly interesting to the remainder of this thesis.

CHAPTER 3

MANY-BODY THEORIES

3.1 Introduction

Ab initio many-body methods aim to solve the many-body problem starting only from the Hamiltonian of the system and a set of known approximations [51]. Although a proper *ab initio* approach in nuclear physics entails dealing with degrees of freedom that include quarks and gluons, we use *ab initio* here to mean a method that can systematically improve upon the many-body framework devised in the last chapter starting from a given Hamiltonian and the respective laws of motion. Many *ab initio* many-body methods are applied to nuclear physics and other fields, but for this thesis, we will limit the scope to just three of these methods. Hartree-Fock theory (HF) is one of the oldest and simplest many-body methods. However, despite its simplicity and small computational requirements, it is one of the least accurate methods. The other two methods investigated in this chapter are post-Hartree Fock methods, which improve the Hartree Fock result. These two methods are many-body perturbation theory (MBPT) and coupled cluster theory (CC).

This chapter will first explore Hartree-Fock theory as the simplest many-body method and the basis of MBPT and CC. Then, after a brief introduction to post-Hartree Fock methods, MBPT will be investigated, followed by a thorough explanation of coupled cluster theory.

3.2 Hartree-Fock Theory

The first many-body method we will explore in this chapter is the Hartree-Fock theory (HF) [82, 83]. Hartree-Fock, one of the earliest many-body methods, is an iterative algorithm that can be used to find the ground state of a system given its Hamiltonian [78]. As an independent-particle model,

Hartree-Fock Theory begins with the Slater determinant, defined in Section 2.2 and rewritten here:

$$\Phi = \frac{1}{N}\begin{vmatrix} \psi_1(\vec{x}_1) & \psi_2(\vec{x}_1) & . & . & . & \psi_N(\vec{x}_1) \\ \psi_1(\vec{x}_2) & \psi_2(\vec{x}_2) & . & . & . & \psi_N(\vec{x}_2) \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ \psi_1(\vec{x}_v) & \psi_2(\vec{x}_v) & . & . & . & \psi_N(\vec{x}_v) \end{vmatrix}, \tag{3.1}$$

where Φ is the many-body wavefunction and $\psi_n(\vec{r}_m)$ is the single-particle wavefunction of the m-th particle with spatial and spin coordinates described by $\vec{r}_m$. Hartree-Fock Theory uses an iterative algorithm to vary the single-particle wavefunctions, ψ such that the energy of the entire many-body wavefunction, represented by the Slater determinant, is minimized [80]. This minimization can be achieved by solving eigenvalue problems for all single-particle wavefunctions. All of the eigenvalue equations are coupled and have the form:

$$\hat{f}\psi_i = \epsilon_i\psi_i, \tag{3.2}$$

where ϵ_i is the single-particle energy and $\hat{f}$ is the single-particle Fock operator. The single-particle Fock operator, which depends on all of the single-particle wavefunctions, has the following matrix elements:

$$\langle p|\hat{f}|q\rangle = \langle p|\hat{t}|q\rangle + \sum_i \langle pi|\hat{v}_2|qi\rangle_A, \tag{3.3}$$

where $\hat{t}$ is the single-particle kinetic energy operator and $\hat{v}_2$ is the interaction between two particles [3]. It is important to note that for nuclear problems, interactions beyond the two-body level contribute significantly to the total interaction. We can also define the many-body Fock operator, $\hat{F}$, as:

$$\hat{F} = \sum_{pq} \langle p|\hat{f}|q\rangle a_p^\dagger a_q. \tag{3.4}$$

For many systems, the solution provided by HF provides an excellent initial approximation for the ground state energy and its corresponding many-body wavefunction. HF can recover approximately

99% of the ground state energy and approximately 95% of the corresponding wavefunction for electronic systems [3, 80]. However, since HF is an independent particle model, the missing energy comes from the interaction between particles, and therefore this energy must be recovered as well. The need to recover the energy from the interactions between particles, known as the correlation energy, has led to the development of more advanced many-body methods. This chapter will develop two of these, many-body perturbation theory and coupled cluster theory.

3.3 Many-Body Perturbation Theory

By this point, it has been well established that finding the energy of a many-body system is done by solving the following eigenvalue problem where $|\Psi\rangle$ is a many-body wavefunction:

$$\hat{H}|\Psi\rangle = E|\Psi\rangle \longrightarrow \langle\hat{H}|\Psi\rangle = E. \tag{3.5}$$

However, this problem can only be solved fully for a few uninteresting systems. Therefore, we have developed our first many-body method, HF theory, capable of finding approximately the energy of a many-body system. However, HF is an independent particle model, and we also wish to include the interactions between the particles in the energy. Thus, we will start by developing many-body perturbation theory (MBPT), a post-Hartree-Fock method, so called because it starts with the Hartree-Fock energy but then adds a correction to account for the interactions between the particles [80, 84–86].

Many-body perturbation theory assumes that the Hamiltonian can be split into two pieces, a non-interacting component, $\hat{H}_0$, and an interacting component, $\hat{H}_I$. The interacting component is a small perturbation away from the non-interacting component, thus the method's name. Thus, we have:

$$\hat{H} = \hat{H}_0 + \hat{H}_I. \tag{3.6}$$

From this definition of the Hamiltonian, we can split the energy of the many-body system into two components: a reference energy, E_0, and a correlation energy, ΔE_{MBPT}, leading to:

$$E = E_{Ref} + \Delta E_{MBPT}. \tag{3.7}$$

The reference energy is defined using the total Hamiltonian and the Fermi vacuum state, $|\Phi_0\rangle$ and the total Hamiltonian as:

$$E_{Ref} = \langle\Phi_0|\hat{H}|\Phi_0\rangle. \tag{3.8}$$

In practice, the reference energy is, in fact, the Hartree-Fock energy, assuming we are using a Fock basis, which we will be using for every calculation in this thesis. Thus the correlation energy is an additional term to the MBPT energy compared to the HF energy.

Now we can define the MBPT correlation energy as the matrix elements formed when the interacting component of the Hamiltonian is applied to a many-body wavefunction as the ket and the Fermi vacuum as the bra, giving:

$$\Delta E_{MBPT} = \langle\Phi_0|\hat{H}_I|\Psi\rangle. \tag{3.9}$$

However, solving equation is no simpler than solving the original eigenvalue problem. Thus we will rephrase the MBPT correlation energy as:

$$\Delta E_{MBPT} = \sum_{i=1}^{\infty} \Delta E^{(i)}, \tag{3.10}$$

where $\Delta E^{(i)}$ is the i-th order correction to the MBPT correlation energy. We can define the form of the first two corrections as follows. The first order correction to the MBPT energy is:

$$\Delta E^{(1)} = \langle\Phi_0|\hat{H}_I|\Psi_0\rangle. \tag{3.11}$$

The second order correction to the MBPT energy is:

$$\Delta E^{(2)} = \frac{1}{4}\sum_{ijab} \frac{\langle ij|\hat{v}_2|ab\rangle\langle ab|\hat{v}_2|ij\rangle}{(\epsilon_i + \epsilon_j) - (\epsilon_a + \epsilon_b)} \tag{3.12}$$

Theoretically, there are infinite corrections to the MBPT correlation energy. In practice, we cannot compute infinite corrections to the MBPT energy, and we must truncate Equation (3.10) to a finite number of terms. However, the form of the correlation energy provides a convenient truncation scheme. The MBPT truncation MBPT1 (many-body perturbation theory to the first order) uses the approximation $\Delta E_{MBPT} \approx \Delta E^{(1)}$, the MBPT truncation scheme MBPT2 (many-body perturbation theory to the second order) uses the approximation $\Delta E_{MBPT} \approx \Delta E^{(1)} + \Delta E^{(2)}$, and so on. This thesis

will primarily focus on MBPT2 as the leading MBPT approximation used in these calculations. Any MBPT results presented here will only include one-body and two-body interactions.

3.4 Coupled Cluster Theory

3.4.1 Introduction to the Method

Coupled cluster theory (CC), initially developed in nuclear physics (for its development, see Reference [87, 88], and for its resurgence in nuclear physics, see Reference [2, 4, 8–10]), has long been the goal standard for quantum chemistry calculations ([7]). Coupled cluster theory provides a method to systematically include complicated interactions beyond the mean field, is non-perturbative, size extensive, non-variational, and is widely used for performing calculations on strongly correlated systems [35]. Being size-extensive is important for CC to be applied to large systems. Computationally, CC scales polynomially with respect to the number of occupied and unoccupied states in the system, making it an efficient many-body method for small to medium-sized systems but relatively slow for large systems. Compared to MBPT, CC is a more accurate method but also incurs more extensive computational run times.

In quantum chemistry and electronic structure, CC is considered the "gold-standard" many-body method but also can be too computationally expensive for some applications, especially for studies of larger molecules [62]. However, quantum chemists have made great strides in accelerating coupled cluster calculations of electronic systems through various methods, including truncations, which will be discussed in a few sections. For interesting applications of coupled cluster theory in quantum chemistry, see, for example, References [6, 7, 40, 42, 54, 62, 89–94].

In CC, we can represent the N-Fermion wavefunction ($|\Psi\rangle$) using the so called exponential ansatz:

$$|\Psi\rangle = e^{\hat{T}}|\Phi_0\rangle. \tag{3.13}$$

In the above equation, $|\Phi_0\rangle$ is the Fermi vacuum state where the N particles in the system occupy the N lowest energy states. $\hat{T}$ is known as the correlation or the cluster operator, and it is the sum of N operators, where N is the number of particles in the system [35, 62], and can be written as:

$$\hat{T} = \sum_{i=1}^{N} \hat{T}_m. \tag{3.14}$$

Each $\hat{T}_m$ operator in Equation (3.14) represents the m-particle m-hole excitation operator, which has the below form [35, 62].

$$\hat{T}_m = (\frac{1}{m!})^2 \sum_{i_1 \ldots i_m a_1 \ldots a_m} t_{i_1 \ldots i_m}^{a_1 \ldots a_m} a_{a_1}^\dagger \ldots a_{a_m}^\dagger a_{i_m} \ldots a_{i_1} \tag{3.15}$$

Single-particle states with labels i_n correspond to states occupied in the Fermi vacuum state, and single-particle states with the labels a_n correspond to states unoccupied in the Fermi vacuum state. The operator $a^\dagger$ is the Fermion creation operator, and the operator a is the Fermion annihilation operator (both are defined in Chapter 2). The coefficients t are called T-amplitudes, and they are determined through a complex set of non-linear equations:

$$\langle \Phi_{i_1, i_2, \ldots, i_k}^{a_1, a_2, \ldots, a_k} | e^{-\hat{T}} \hat{H} e^{\hat{T}} | \Phi_0 \rangle = 0, \tag{3.16}$$

where the index k = 1, 2, ..., A [35]. When k = 1, the one-particle one-hole excitation operator is recovered ($\hat{T}_1$), when k = 2 the two-particle two-hole excitation operator is recovered ($\hat{T}_2$), and so on. Therefore, amplitude equations must be solved for untruncated coupled-cluster theory to fully derive the cluster operator for an N-body system. If coupled cluster equations are solved with the untruncated cluster operator, they arrive as the same result as Schrödinger's equation.

It is important to note that in Equation (3.16), we have used a shorthand notation to refer to the bra vector, which is, in fact, a k-particle k-hole excitation of the Fermi vacuum state [35]. Written out fully in second quantization, we can define the k-particle k-hole excitation of the Fermi vacuum state as:

$$|\Phi_{i_1, i_2, \ldots, i_k}^{a_1, a_2, \ldots, a_k}\rangle = a_{a_1}^\dagger \ldots a_{a_k}^\dagger a_{i_k} \ldots a_{i_1} |\Phi_0\rangle, \tag{3.17}$$

where $a^\dagger$ is the Fermion creation operator, and a is the Fermion annihilation operator.

As a note on notation, the T-amplitudes, scalars that are calculated through determining the m-particle m-hole excitation operators, can be represented equivalently in the following two notations:

$$t_{i_1 \ldots i_m}^{a_1 \ldots a_m} = \langle a_1 \ldots a_m | \hat{t} | i_1 \ldots i_m \rangle. \tag{3.18}$$

As an aside from the form of the CC wavefunction:

$$|\Psi\rangle = e^{\hat{T}} |\Phi_0\rangle, \tag{3.19}$$

one can use a Taylor expansion to expand the exponential function to obtain:

$$|\Psi\rangle = (1 + \hat{T} + \hat{T}^2/2! + \hat{T}^3/3! + ...)|\Phi_0\rangle. \tag{3.20}$$

This expansion explains why some of the later CC approximations we will be looking at contain terms such as $\hat{T}_1$ and $\hat{T}_2$ but also $\hat{T}_1\hat{T}_2$ and $\hat{T}_1^3$.

3.4.2 Energy and Correlation Energy

Given an N-body many-body wavefunction, its energy can be determined by solving the eigenvalue problem that results from applying the Hamiltonian to the wavefunction (i.e., solving the time-independent Schrödinger's equation) using:

$$\hat{H}|\Psi\rangle = E|\Psi\rangle. \tag{3.21}$$

The above equation Equation (3.21) can give just the energy on the right-hand side of the left-hand side turned into a matrix element such that the equation now has the form:

$$\langle\Psi|\hat{H}|\Psi\rangle = E. \tag{3.22}$$

Equation (3.22) has an implicit $\langle\Psi|\Psi\rangle = 1$ on the righthand side of the equation. From here, we can split the Hamiltonian into two pieces, the normal-ordered Hamiltonian and the vacuum expectation value:

$$\hat{H} = \hat{H}_N + E_0. \tag{3.23}$$

where the vacuum expectation value E_0 is defined as:

$$E_0 = \langle\Phi_0|H|\Phi_0\rangle. \tag{3.24}$$

Combining Equations (3.22) and (3.23) yields:

$$\langle\Psi|\hat{H}_N + E_0|\Psi\rangle = E, \tag{3.25}$$

which can be split up into the following two terms:

$$\langle\Psi|\hat{H}_N|\Psi\rangle + \langle\Psi|E_0|\Psi\rangle = E. \tag{3.26}$$

Since $\langle\Psi|E_0|\Psi\rangle = E_0\langle\Psi|\Psi\rangle = E_0$, then we can simplify Equation (3.26) to:

$$\langle\Psi|\hat{H}_N|\Psi\rangle + E_0 = E. \tag{3.27}$$

From here, we can rewrite the coupled cluster exponential equation:

$$e^{\hat{T}}|\Phi_0\rangle = |\Psi\rangle, \tag{3.28}$$

which can be inserted into Equation (3.27) to yield.

$$\langle\Phi_0|e^{-\hat{T}}\hat{H}_N e^{\hat{T}}|\Phi_0\rangle + E_0 = E. \tag{3.29}$$

From here, we can define the similarity transformed normal ordered Hamiltonian to be:

$$\bar{H}_N = e^{-\hat{T}}\hat{H}_N e^{\hat{T}}, \tag{3.30}$$

which yields a final form of the energy equation:

$$\langle\Phi_0|\bar{H}_N|\Phi_0\rangle + E_0 = E. \tag{3.31}$$

In 3.31, E_0 is known as the reference energy and (for this work) it is the Hartree-Fock energy. This makes coupled cluster a post-Hartree-Fock method, similar to MBPT. The term $\langle\Phi_0|\bar{H}_N|\Phi_0\rangle$ is the correlation energy or the coupled cluster correction to the Hartree-Fock energy. As a final definition for this section, the CC correlation energy will be represented as $\Delta E_{CC} = \langle\Phi_0|\bar{H}_N|\Phi_0\rangle$. Much like MBPT, coupled cluster calculations will generally be reported in terms of the correlation energy instead of the total energy. Again, this is due to convention, as the correlation energy is the most important part of the energy calculation. Also note that for some systems (namely the infinite matter systems discussed in the next chapter), CC correlation energies are usually reported as the CC correlation energy per particle by convention such that systems with different numbers of particles can be compared.

The cluster operator $\hat{T}$ is undefined in the above equations. It is obtained by solving a set of coupled cluster amplitude equations set up in the previous section, of which there are N sets of equations for an N particle system. and the number of equations per set depends on the number of single

particle states in the calculation. This system of equations is quite large even for relatively small systems and requires and iterative procedure to be solved. Defining the cluster operator (and, more specifically, the t-amplitudes) is the most computationally extensive step when performing a CC energy calculation.

3.4.3 Computational Methods and Approximations

For almost all CC calculations performed computationally, the cluster operator must be truncated [62]. Systems, where CC calculations can be performed with the full cluster operator, tend to be uninteresting toy models [35]. However, the form of the cluster operator provides a physically motivated method for truncation where all m-particle m-hole excitation operators over a certain level are set to zero. This approximation scheme is referred to as the SUB_n approximation scheme [35]. There is also a convenient naming scheme when the cluster operator is truncated in this way. For example, if $\hat{T} \approx \hat{T}_1$, the approximation is called the coupled cluster single approximation (CCS), and if $\hat{T} \approx \hat{T}_1 + \hat{T}_2$ then we call this the coupled cluster singles and doubles approximation (CCSD), and if $\hat{T} \approx \hat{T}_1 + \hat{T}_2 + \hat{T}_3$ this leads to the coupled cluster singles, doubles, and triples approximation (CCSDT) [51]. Due to computational limitations, approximations over the triples level are rare, but have been performed on a limited set of systems [43]. For the infinite matter systems that are defined in the next chapter, the $\hat{T}_1$ operator gives no contribution to the energy so that we can simplify the above truncations: CCSD becomes CCD (coupled-cluster doubles) where $\hat{T} \approx \hat{T}_2$ and CCSDT becomes CCDT (coupled-cluster doubles and triples) where $\hat{T} \approx \hat{T}_2 + \hat{T}_3$. As the primary goal of this thesis is to use machine learning to extend the range of pre-existing coupled cluster methods instead of developing new methods or improving on existing methods, the explanations below are kept short and generally free of derivations. There have been many great works deriving the various coupled cluster methods in great detail, and we would like to point the readers to References [7, 80, 94], among others, for complete derivations of the methods introduced in this section.

As is expected, the higher the level of approximation, the more accurate the calculations but, the more extended run times. However, if having all N terms in the cluster operator gives 100% of the

energy of an N-particle system, then the CCSD approximation gives approximately 90% of the total energy, and CCSDT gives almost 99% of the total energy [80]. Each additional m-particle m-hole excitation operator does increase the accuracy of the energy calculation, but the improvements progressively decrease in size. Additionally, as expected, the computational time and resource requirements increase drastically as the order of the CC calculation increase. For example, the expected run time for a CCSD calculation is $O(M^6)$, where M is the number of single-particle states. The expected run time for a CCSDT calculation is $O(M^8)$, and considering M is likely to be 1,000 or higher for accurate calculations, this extra factor of M^2 represents a significant increase in run time. However, since we would like to achieve the accuracy of CCSDT, recovering 99% of the total energy, we can instead look at some approximative methods to the CCSDT approximation. These methods will not be as accurate as the CCSDT method, but they will be more accurate than the CCSD method as they contain some components from the $\hat{T}_3$ operator. Additionally, while these approximative triples methods will have a longer run time than a CCSD calculation, they will have much shorter run times than a complete CCSDT calculation, making the triples approximations a good balance of accuracy and run time.

3.4.4 CCSDT Approximations

There are two types of approximative triples calculations: non-iterative perturbative triples approximations, CCSD(T), and iterative CCSDT-n approximations. The methods differ in their complexity and formulations, but give similar results and thus will be compared below.

3.4.4.1 Non-Iterative Triples Approximations

The perturbative triples approximation, CCSD(T), is a non-iterative triples approximation that is the gold standard for quantum chemistry applications [2]. The CCSD(T) method was developed in the late 1980s in the field of quantum chemistry [95]. The computational considerations for CCSD(T) have two components: an iterative component with a cost of $O(M^6)$ (the CCSD equations) and a non-iterative component with a cost of $O(M^7)$, making it faster than complete CCSDT calculations by a factor of M or greater. Since M can be over 1,000, this can represent a significant decrease in the run time for a calculation.

We will start developing the CCSD(T) equations by defining the following two permutation operators:

$$\hat{P}(a/bc) = 1 - \hat{P}_{ab} - \hat{P}_{ac}, \tag{3.32}$$

and:

$$\hat{P}(a/bc|k/ij) = \hat{P}(a/bc)\hat{P}(k/ij), \tag{3.33}$$

where $\hat{P}_{pq}$ is the permutation between states p and q. Next we can define an equation through which the $\hat{T}_3$ amplitudes can be defined from the $\hat{T}_2$ amplitudes:

$$\epsilon_{ijk}^{abc} t_{ijk}^{abc} = \hat{P}(a/bc|k/ij) \sum_d \langle bc|\hat{v}_2|dk\rangle t_{ij}^{ad} = \hat{P}(c/ab|i/jk) \sum_l \rangle lc|\hat{v}_2|jk\rangle t_{il}^{ab}, \tag{3.34}$$

where:

$$\epsilon_{ijk}^{abc} = (\epsilon_i + \epsilon_j \epsilon_k) - (\epsilon_a + \epsilon_b + \epsilon_c). \tag{3.35}$$

Equation (3.34) defines the leading order (i.e., the second-order) terms in the full CCSDT triples equations. This equation will be used here to define the CCSD(T) method and in the next section to define the CCSDT-1a method. Next we can define $E_T^{(4)}$, the triples-excitation contributions to the MBPT4 (MBPT to the fourth order):

$$E_T^{(4)} = \sum_{\substack{i>j>k \\ a>b>c}} t_{ijk}^{abc\,(2)*} t_{ijk}^{abc\,(2)} \epsilon_{ijk}^{abc}. \tag{3.36}$$

Here we are defining the second order $\hat{T}_3$ amplitudes (see Sec. 10.5 of Reference [80]). However, for CCSD(T), we are going to define these as $t_{ijk}^{abc\,[2]}$ instead of $t_{ijk}^{abc\,(2)}$ because we will generate the $\hat{T}_3$ amplitudes using Equation (3.35) and the converged $\hat{T}_2$ amplitudes (the square brackets represent that we are using converged $\hat{T}_2$ amplitudes and the number in the brackets is a generalized order). From here, we are in a place to define the energy that results from a CCSD(T) calculation using the following equation:

$$E_{CCSD(T)} = E_{CCSD} + E_T^{[4]} = E_{CCSD} + E_t^{[4]} + E_{st}^{[4]}, \tag{3.37}$$

where:

$$E_t^{[4]} = \sum_{\substack{i>j>k \\ a>b>c}} t_{ijk}^{abc\,[2]*} t_{ijk}^{abc\,[2]} \epsilon_{ijk}^{abc}, \tag{3.38}$$

and where:

$$E_{st}^{[4]} = (\sum_{ia} t_i^{a\,[2]}) \sum_{\substack{j>k \\ b>c}} \langle bc|\hat{v}_2|jk\rangle t_{ijk}^{abc\,[2]}. \tag{3.39}$$

Note that the only difference between CCSD(T) and another non-iterative CCSDT approximate method, CCSD[T] (see. Reference [96]), is the inclusion of the $E_{st}^{[4]}$ term, which is usually relatively small. We should also mention an improvement to CCSD(T), called Λ-CCSD(T), which has been developed in quantum chemistry and is defined in Reference [89].

3.4.4.2 Iterative Triples Approximations

Another method of approximating the triple contribution is through the iterative CCSDT-*n* methods, where *n* = 1a, 1b, 2, 3, 4, or 5. Though computationally more expensive than their non-iterative counterparts, iterative triples approximations are more accurate [2]. All methods will be briefly described here, but we will focus on CCSDT-1a, which will be used for calculations later in this thesis. For more information on CCSDT-1b - CCSDT-4, please refer to Reference [94] and Reference [96].

The simplest of the CCSDT-n methods is the CCSDT-1a approximation and is created by setting $\hat{T}_1 = \hat{T}_3 = 0$ when being projected against the three-particle three-hole excited state. The CCSDT-1a covers some infinite order terms from the $\hat{T}_3$ operator. This contrasts CCSD, which covers no terms from the $\hat{T}_3$, and CCSDT, which covers all terms. As shown in Table 3.1, the CCSDT-1a method uses the operator $e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$ when projected against the singly excited state, the operator $e^{\hat{T}_1+\hat{T}_2} + \hat{T}_3$ when projected against the doubly excited state, and the operator $1 + \hat{T}_2$ when projected against the three-particle three-hole excited state. These projections are used when constructing the amplitude equations, which are used to derive the T-amplitudes and the energies and are in contrast to a complete CCSDT calculation where the operator $e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$ is used no matter the state [94, 96]. Some successive approximations are made in the CCSDT-1a approximation that will be discussed as the other methods are made.

The CCSDT-1b model uses the complete $e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$ operator for the singly and doubly excited states but, like CCSDT-1a, uses the operator $1 + \hat{T}_2$ for the three-particle three-hole excited state. Because

the full operator is projected against the doubly excited state, the disconnected $\hat{T}_1\hat{T}_3$ clusters are included. Additionally, CCSDT-1b has a similar computational time to CCSDT-1a [96].

The next approximation is the CCSDT-2 approximation, which like CCSDT-1b, applies the complete $e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$ operator to the single and doubly excited states and uses $\hat{T}_1 = \hat{T}_3 = 0$ when applied to the triply excited state. However, it changes the operator projected on the triply excited states to $e^{\hat{T}_2}$ [94, 96]. This is an important approximation since it adds the effects of the disconnected $\hat{T}_2\hat{T}_2$ clusters on the T_3 amplitudes. The CCSDT-3 approximation is conceptually the simplest, where only $\hat{T}_3$ is set to zero, such that the full $e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$ operator is projected onto the singly and doubly excited states but the operator $e^{\hat{T}_1+\hat{T}_2}$ is projected onto the triply excited state [96]. The final approximation, CCSDT-4, projects the full $e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$ operator onto the singly and doubly excited states but uses the operator $e^{\hat{T}_1+\hat{T}_2} + \hat{T}_3$ for the triply excited states. The CCSDT-5 approximation is simply the full CCSDT calculation.

The reason to perform this approximation over the complete CCSDT calculation is computational run times. A total CCSDT calculation is expected to have a run time on the order of $O(M^8)$. The expected run times for the CCSDT-n approximations are in Table 3.2 [94]. CCSDT-1a, CCSDT-1b, CCSDT-2, and CCSDT-3 have computational run times that are shorter than a complete CCSDT calculation by a factor of M (where typically M is greater than 1,000) but are more extended than CCSD calculation by a factor of M. However, they include contributions from $\hat{T}_3$ and the triply excited states that CCSD does not. CCSDT-4 has the exact computational cost as a complete CCSDT calculation, so there are not many cases where it would be preferred over just using a complete CCSDT calculation. Additionally, about 75% of the extra energy from the triples contribution is captured by the CCSDT-1a approximation, which CCSDT-1b, CCSDT-2, and CCSDT-3 not adding any other contributions to the energy from full summations, though they do add some contributions from partial summations [94]. Thus as a good compromise of accuracy and computational run time, we will use the CCSDT-1a approximation.

Method	$\|\Phi_i^a\rangle$	$\|\Phi_{ij}^{ab}\rangle$	$\|\Phi_{ijk}^{abc}\rangle$
CCSD	$e^{\hat{T}_1+\hat{T}_2}$	$e^{\hat{T}_1+\hat{T}_2}$	N/A
CCSDT-1a	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2}+\hat{T}_3$	$1+\hat{T}_2$
CCSDT-1b	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$1+\hat{T}_2$
CCSDT-2	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_2}$
CCSDT-3	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2}$
CCSDT-4	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2}+\hat{T}_3$
CCSDT	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$	$e^{\hat{T}_1+\hat{T}_2+\hat{T}_3}$

Table 3.1 This table summarizes the approximations for $e^{\hat{T}}$ the various CCSDT-n methods use in the amplitude equations, along with the approximations used for CCSD and CCSDT

Method	Computational Cost
CCSD	$O(M^6)$
CCSDT-1a	$O(M^7)$
CCSDT-1b	$O(M^7)$
CCSDT-2	$O(M^7)$
CCSDT-3	$O(M^7)$
CCSDT-4	$O(M^8)$
CCSDT	$O(M^8)$

Table 3.2 The predicted computational run times of various coupled cluster calculations as a function of M, the number of single particle states in the calculation

If we consider the Hamiltonian with at most a two-body force written in normal product form, we get:

$$\bar{H} = \bar{H}_0 + \bar{V}_2 = \sum_{pq}\langle p|f|q\rangle\{a_p^\dagger a_q\} + \sum pqrs\langle pq|\hat{v}_2|rs\rangle\{a_p^\dagger a_q^\dagger a_s a_r\}. \tag{3.40}$$

We can then write the CCSD equations using this Hamiltonian as (following Reference [94]):

$$\langle\Phi_0|\bar{H}(\hat{T}_2+\frac{1}{2}\hat{T}_1^2)|\Phi_0\rangle = \Delta E_{CCSD}, \tag{3.41}$$

and:

$$\langle\Psi_i^a|\bar{H}(\hat{T}_1+\hat{T}_2+\hat{T}_1\hat{T}_2+\frac{1}{2}\hat{T}_1^2+\frac{1}{3!}\hat{T}_1^3)|\Phi_0\rangle = 0, \tag{3.42}$$

and finally:

$$\langle\Phi_{ij}^{ab}|\bar{H}(1+\hat{T}_1+\hat{T}_2+\frac{1}{2}\hat{T}_2^2+\hat{T}_1\hat{T}_2+\frac{1}{2}\hat{T}_1^2\hat{T}_2+\frac{1}{3!}\hat{T}_1^3+\frac{1}{4}\hat{T}_1^4|\Phi_0\rangle = 0. \tag{3.43}$$

Here we have the CCSD energy equation in Equation (3.41), the amplitude equation for the singly excited states in Equation (3.42), and the amplitude for the doubly excited states in Equation (3.43). Using the same system, the CCSDT-1a equations are as follows (also following Reference [94]), with the correlation energy equation being:

$$\langle\Phi_0|\bar{H}(\hat{T}_2+\frac{1}{2}\hat{T}_1^2)|\Phi_0\rangle = \Delta E_{CCSDT-1a}, \tag{3.44}$$

the singly excited amplitude equation being:

$$\langle\Phi_i^a|\bar{H}(\hat{T}_1+\hat{T}_2|\hat{T}_3+\frac{1}{2}\hat{T}_1^2+\hat{T}_1\hat{T}_2|\frac{1}{3!}\hat{T}_1^3|\Phi_0\rangle = 0, \tag{3.45}$$

the doubly excited amplitude equation being:

$$\langle\Phi_{ij}^{ab}|\bar{H}(1+\hat{T}_1+\hat{T}_2|\hat{T}_3|\frac{1}{2}\hat{T}_1^2+\hat{T}_1\hat{T}_2+\frac{1}{2}\hat{T}_2^2|\frac{1}{3!}\hat{T}_1^3|\frac{1}{2}\hat{T}_2\hat{T}_1^2|\frac{1}{4!}\hat{T}_1^4|\Phi_0\rangle = 0, \tag{3.46}$$

and finally the triply excited amplitude equation being:

$$\langle\Phi_{ijk}^{abc}|\bar{H}_0\hat{T}_3+\bar{V}\hat{T}_2|\Phi_0\rangle = 0. \tag{3.47}$$

We can write the T-amplitudes for the $\hat{T}_3$ operator that will result from a CCSDT-1a calculation as (which is the same as Equation (3.35)):

$$\epsilon_{ijk}^{abc}t_{ijk}^{abc} = \hat{P}(a/bc|k/ij)\sum_d\langle bc|\hat{v}_2|dk\rangle t_{ij}^{ad} = \hat{P}(c/ab|i/jk)\sum_l\rangle lc|\hat{v}_2|jk\rangle t_{il}^{ab}. \tag{3.48}$$

For the CCSDT-1a method, we use the unconverged $\hat{T}_2$ amplitudes to calculate the $\hat{T}_3$ amplitudes and converge both sets of amplitudes in an iterative method that requires 10 to 20 steps on average [80]. Another improvement of the CCSDT-1a model over a complete CCSDT calculation is that it avoids the storage of the $\hat{T}_3$ amplitudes. So not only are there reduced computational time requirements, but there is also a reduction in the amount of memory required [80, 94].

Briefly, we can compare CCSD(T) and CCSDT-1a since we will be using both approximations later in this thesis. Timing-wise, CCSD(T) should provide a better cost-performance ratio as its run time is an iterative $O(M^6)$ followed by a non-iterative $O(M^7)$ while CCSDT-1a is a fully iterative $O(M^7)$. Additionally, CCSDT-1a and CCSD(T) should produce similar results, but CCSD(T) has fewer $\hat{T}_3$

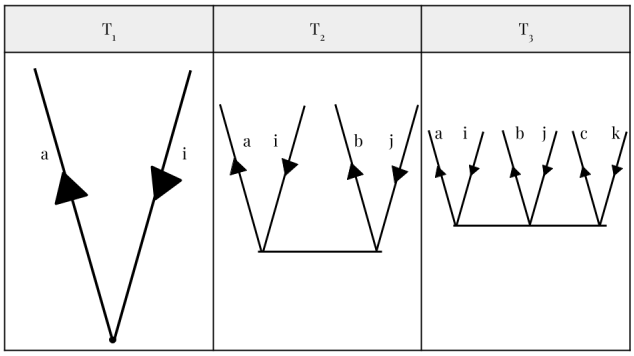


Figure 3.1 Diagrammatic representations for the $\hat{T}_1$, $\hat{T}_2$, and $\hat{T}_3$ operators. Higher order $\hat{T}_m$ operators can be created using the pattern in the operators shown here

terms than CCSDT-1a. However, as both methods have the same number of $\hat{T}_3\hat{T}_3$ coupling terms, CCSD(T) should avoid the exaggeration of the $\hat{T}_3$ effects that can plague CCSDT-1a calculations. Finally, for systems where $\hat{T}_3$ have significant effects, both CCSDT-1a and CCSD(T) provide a reliable method, but for practical applications, the CCSD(T) calculation is preferable as a CCSDT approximation due to run-time considerations and the reduction in exaggerating the effects of the $\hat{T}_3$ operator [94].

3.4.5 Diagrammatic Representations

Finally, we can briefly discuss the diagrammatic representations of the CC approximations up to and including the triples approximation. In this chapter, we will only mention some of the crucial diagrams. For a complete derivation of the coupled cluster diagrams up to and beyond the triples level, see, for example, Reference [80].

To the diagrammatic rules we developed in Chapter 2 of this thesis, we can also add the following diagrammatic rules, which are specific to drawing coupled cluster diagrams:

- Every vertex representing $\hat{T}_m$ picks up a factor of $\langle a_1 ... a_m | \hat{t} | i_1 ... i_m \rangle$ (the T-amplitudes)
- Each pair of $\hat{T}_m$ vertices which are equivalent pick up a factor of 1/2. The vertices are equivalent if they connect to the interaction vertex in the same way.

Representing the $\hat{T}_1$, $\hat{T}_2$, and $\hat{T}_3$ operators in diagrammatic representation is quite simple, and these are shown in Figure 3.1.

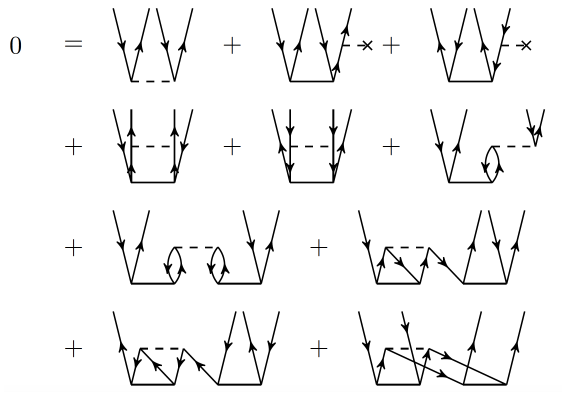


Figure 3.2 The diagrammatic formulation for the CCD amplitude equation

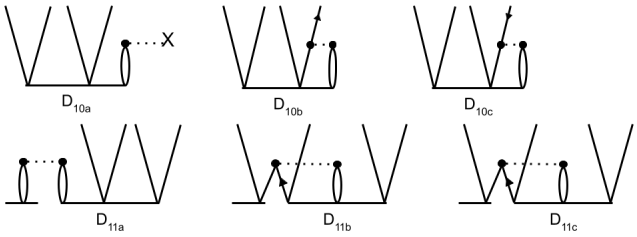


Figure 3.3 Antisymmetrized Goldstone diagrams representing the $\hat{T}_3$ contributions to the CCSDT $\hat{T}_2$ equations. D_{10b} and D_{10c} are relevant to the CCDT-1 approximation defined in the last section

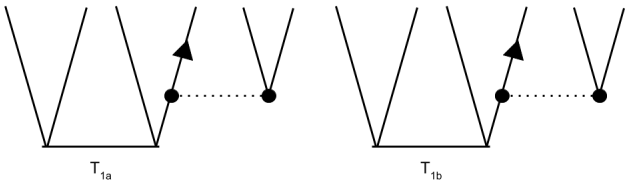


Figure 3.4 Antisymmetrized Goldstone diagrams representing the CCSDT $\hat{T}_3$ equations relevant to calculating the CCDT-1 approximation. For a full list of all antisymmetrized Goldstone diagrams for the CCSDT $\hat{T}_3$ equations, the reader is referred to Reference [80]

To show a complete coupled cluster calculation written out in diagrammatic form, we will represent the CCD amplitude equation in diagrammatic form in Equation (3.2). Though long, this collection of diagrams is significantly shorter than the entirely written out CCD amplitude equation using summations and Dirac notation, which can be found in Reference [35].

Finally, look at some diagrams from the $\hat{T}_3$ operator, which are important to the CCD(T) and CCDT-1 approximations. In fact, the terms T_{1a} and T_{1b} from the $\hat{T}_3$ amplitude equations form Equation (3.48). The amplitudes that are gained by this equation are then used to calculate the diagrams D_{10b} and D_{10c} for the $\hat{T}_2$ amplitudes [97].

3.5 Many-Body Methods Conclusion

In this chapter, we have developed three many-body methods we will use throughout this thesis. The most fundamental of these methods is the Hartree-Fock theory, which will recover a reasonable amount of the system's energy. It is an independent particle model, so it cannot account for the energy from interacting particles. The first post-Hartree-Fock method we developed, so-called because it improves the Hartree-Fock result, was many-body perturbation theory (MBPT) which provides a convenient truncation scheme that allows for systematic improvements to the Hartree-Fock results. The final many-body method, developed in great detail due to its importance later in this thesis, was coupled cluster theory, which is more accurate than MBPT but much more time-consuming. Coupled cluster theory also provides a convenient truncation scheme and, at the triple level, allows us to recover most of the system's energy even with the method being truncated. Now that we have developed our many-body methods, the next chapter will be devoted to developing the system to which these methods will be applied: infinite matter.

CHAPTER 4

INFINITE MATTER SYSTEMS

4.1 Introduction to Infinite Matter Systems

An infinite matter system is simply a system which contains infinite particles and occupies an infinite volume. We will be looking at two infinite matter systems in this chapter. The first, the homogeneous electron gas, which has critical applications in several areas of chemistry and condensed matter physics [9, 37, 38, 40, 43, 54, 76, 98–101]. Since the homogeneous electron gas is a more straightforward infinite matter system, it is also a test bed for developing methods that can be applied to more complex infinite matter systems. The second infinite matter system we will investigate in this chapter is infinite nuclear matter [9, 21, 35, 102, 103]. Specifically, pure neutron matter is an infinite nuclear matter system where all the particles are neutrons ([14, 20, 30, 32, 33, 104–107]), and symmetric nuclear matter, where half of the particles are protons and half are neutrons ([25, 108]). Studies of infinite nuclear matter play an essential role in nuclear physics and astrophysics [11, 22, 23, 26–31, 57, 60, 67, 77].

Performing coupled cluster calculations on infinite matter is not a new concept. For coupled cluster calculations of the electron gas, see, for example, References [3, 38, 40, 78], among many other interesting studies. For infinite matter, there are References [3, 11, 34, 35, 78] among others. However, this work is unique in the methodology we will take to remove some of the errors that result from these calculations, described towards the end of this chapter. It is important to note that for all infinite matter systems investigated in this thesis, the one-particle one-hole excitation operator ($\hat{T}_1$) in the cluster operator is zero due to symmetry considerations [35]. It is almost important to note that these infinite matter systems are momentum conserving and that the total momentum is zero [35].

4.2 Single Particle Basis for Infinite Matter

We can begin to define a single particle basis for infinite matter calculations in three dimensions by defining the single particle wavefunction as plane waves with the form:

$$\psi_{\vec{k},\sigma} = \frac{1}{\sqrt{\Omega}} e^{i\vec{k}\cdot\vec{r}} \xi_\sigma, \tag{4.1}$$

where ξ_σ has two possible values corresponding to spin-up particles:

$$\xi_{\sigma=\frac{1}{2}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{4.2}$$

or spin-down particles:

$$\xi_{\sigma=-\frac{1}{2}} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{4.3}$$

Additionally, in Equation (4.1), $\Omega = L^3$ corresponds to the volume of the infinite matter system. The limit $L \to \infty$ is taken to accurately simulate an infinite matter system after the various expectation values have been calculated [1, 76, 77].

When using the plane waves as the single particle wavefunctions and assuming periodic boundary conditions, we can define the single particle momentums to be:

$$k_i = \frac{2\pi}{L} n_i, \tag{4.4}$$

where $i = x$, y, or z and $n_i = 0, \pm 1, \pm 2, \ldots$ [76, 77]. From the momentum numbers, we can define the kinetic energy of an infinite matter system in second quantization to be:

$$\hat{T} = \sum_{\vec{p}\sigma_p} \frac{\hbar^2 k_p^2}{2m} a^\dagger_{\vec{p}\sigma_p} a_{\vec{p}\sigma_p}. \tag{4.5}$$

Finally, we can define the energy of each single particle state in terms of its quantum numbers:

$$\epsilon_{n_x,n_y,n_z} = \frac{\hbar^2}{2m} \left(\frac{2\pi}{L}\right)^2 (n_x^2 + n_y^2 + n_z^2). \tag{4.6}$$

We can rewrite Equation (4.6) to be in terms of the momentum using Equation (4.4), resulting in:

$$\epsilon_{n_x,n_y,n_z} = \frac{\hbar^2}{2m} (k_x^2 + k_y^2 + k_z^2). \tag{4.7}$$

Some of the single-particle states in a system will contain particles, which we will call occupied single-particle states. On the other hand, some will not contain particles, referred to as unoccupied single-particle states. When the system is arranged in its ground state formation (i.e., the lowest energy configuration), the occupied single-particle states fill the states with the lowest single-particle energies. Thus, the occupied single particle states will lie inside a so-called Fermi sphere, described by the Fermi momentum, k_F, and the Fermi energy, E_F. The Fermi level divides the occupied and unoccupied single-particle states in this ground state configuration.

The following set of equations can relate to the Fermi momentum and the Fermi energy:

$$E_f = \frac{\hbar^2 k_f^2}{2m} \longrightarrow k_f = \sqrt{\frac{2mE_f}{\hbar^2}}. \tag{4.8}$$

From the Fermi momentum, we can define the average density of the system, ρ_0, to be:

$$\rho_0 = \frac{k_f^3}{3\pi^2}. \tag{4.9}$$

The average density, ρ_0 can also be defined as $\rho_0 = N/\Omega$, where N is the number of particles in the system and like L, the limit $N \to \infty$ is taken. However, the limits $N, L \to \infty$ are taken such that ρ_0 remains at a fixed, finite value. Finally, N and k_F can be related with:

$$N = \frac{k_f^3 \Omega}{3\pi^2} \longrightarrow k_F = (\frac{3\pi^2 N}{\Omega})^{1/3}. \tag{4.10}$$

While in this theoretical model, there are an infinite number of single-particle states in the basis, in practice, due to computational limitations, the total number of single-particle states included in the calculation, M, must be truncated to a finite number. In this work, we will assume that the number of occupied and unoccupied single particle states correspond to closed-shell structure, and the total number of shells will be finite. Here we will define the a shell as a collection of single particle states with the same energy level. We will place a spherical energy cut-off on the quantum numbers such that $n_x^2 + n_y^2 + n_z^2 \leq N_{shells} - 1$, where N_{shells} is the total number of energy shells included in the calculation. It is important to note that $N_{shells} \neq N_{max}$. Rather, N_{max} corresponds to the maximum value of n_x, n_y, or n_z that is included in the calculation. This closed shell structure is also used for coupled cluster calculations in Reference [77], [78], and [35] among others.

Enforcing a closed-shell structure for the occupied and unoccupied single-particle states and truncating the total number of shells allowed in the system restricts the number of allowed particles and total single-particle states to a finite set of values. We refer to these allowed numbers as "magic numbers," They correspond to the total number of single-particle states when s shells are included in a calculation. The first magic number is two, as there are two single-particle states in the first energy shell. The second magic number is 14, as there are 14 single-particle states when two energy shells are included (2 in the first shell and 12 in the second shell). Continuing as such gives the following magic numbers: 2, 14, 38, 54, 66, 114, ... Note that these are only the magic numbers for the homogeneous electron gas and pure neutron matter. For symmetric nuclear matter, all magic numbers are doubled (4, 28, 76, 108, 132, 228, ...). In the context of this thesis, the term shells will refer to the total number of energy shells used in the calculation, M will refer to the total number of single-particle states in the system (both occupied and unoccupied), and open shells will refer to the number of shells above the Fermi level only.

4.3 The Homogeneous Electron Gas

The homogeneous electron gas (HEG) is an infinite matter system containing only electrons with a uniform positive background charge such that the overall net charge of the system is zero [3]. Though the HEG as a theoretical model exists in various dimensions [3], this thesis will only focus on the three-dimensional electron gas. The HEG in three dimensions is essential in density functional theory, where it is the cornerstone of the local density approximation [3]. It is also a reasonable model for several systems of interest in quantum chemistry and condensed matter physics, including the electrons in semiconductors and alkali metals [3]. Additionally, studies of the HEG can be used to build a framework that can be transferred to studies of infinite nuclear matter, which is essential for studying the equation of the state of nuclear matter and many-body studies of neutron stars. The HEG is used as a framework for developing analysis tools for other many-body systems because the HEG is a relatively simple infinite matter system, leading to many published properties (both analytical and numerical) for results to be compared to. Other studies have used the HEG as a system that allows for comparisons between the results of different many-body methods

(see, for example, Reference [3]).

The Hamiltonian for the HEG consisting of N electrons can be defined as

$$\hat{H} = \hat{K} + \hat{V}_{ee} + \hat{V}_{be} + \hat{V}_{bb}, \tag{4.11}$$

where $\hat{K}$ is the kinetic energy operator, $\hat{V}_{ee}$ is the interaction between all sets of two electrons, $\hat{V}_{be}$ describes the interaction of all electrons with the positive background charge, and $\hat{V}_{bb}$ describes the contribution of the background charge interacting with itself. Since the HEG comprises a positively charged background and negatively charged electrons, the Coulomb force is the primary force at play in the interactions. The Coulomb force is a long-range force that acts over an infinite space. However, instead of using the Coulomb interaction, we will instead use Ewald's interaction, which splits the electron interaction into a short-range term and a long-range term while simultaneously dealing with $\hat{V}_{be}$ and $\hat{V}_{bb}$. We can rewrite the HEG Hamiltonian using the Ewald interaction as:

$$\hat{H} = -\frac{1}{2}\sum_{\alpha} \nabla_{\alpha}^2 + \frac{1}{2}\sum_{\alpha \neq \beta} \hat{v}_{\alpha\beta} + \frac{1}{2} N v_M, \tag{4.12}$$

where α and β are electron indices [76]. The first term is the kinetic energy operator, and the final two terms comprise the Ewald interaction [76]. The term v_M is called the Madelung term and $\hat{v}_{\alpha\beta}$, the two-electron operator, is defined as:

$$\hat{v}_{\alpha\beta} = \frac{1}{V}\sum_{\vec{q}} v_{\vec{q}}\, e^{i\vec{q}(\vec{r}_{\alpha} - \vec{r}_{\beta})}, \tag{4.13}$$

where $V = L^3$ is the finite volume of the HEG and $v_{\vec{q}} = \frac{4\pi}{\vec{q}^2}$ if $\vec{q} \neq 0$ and $v_{\vec{q}} = 0$ if $\vec{q} = 0$ [76]. This choice of Hamiltonian leads to the plane wave basis for the single-particle states described in Section 2 of this chapter. While the Ewald interaction makes the HEG easier to work with by splitting up the long-range and short-range components of the electron-electron interaction, it cannot correctly describe the exchange-correlation energy [3].

Note that the density of the homogeneous electron gas is typically described by a parameter called the Wigner-Seitz radius. The Wigner-Seitz radius can be defined as

$$r_s = \frac{r_0}{r_B}, \tag{4.14}$$

where r_B is the Bohr radius, and r_0 is related to the size of the box containing the electron gas by

$$\frac{4}{3}\pi r_0^3 = \frac{N}{L^3}, \tag{4.15}$$

where the density of the HEG is $d = N/L^3$, measured in fm^{-3} [78]. A larger value of r_s means the modeled system is in a larger box and therefore is less dense than a smaller value of r_s for the same number of particles. It is important to note that the HEG behaves differently at different densities. At low densities (or high r_s), the electrons in the HEG are for a lattice. This process is called Wigner crystallization and results from the long-range repulsive interactions between the electrons [3]. At higher densities (or lower values of r_s), the HEG can better be described as a liquid instead of as a gas [3].

The convergence of the correlation energy for the electron gas using plane waves as a basis has yet to be thoroughly investigated, but some work has been done in Reference [76]. One reason why the correlation energy of the electron gas is an interesting problem is that it has substantial contributions from the electron-electron cusp. However, this electron-electron cusp does lead to trouble when making computational truncations as a finite set of single-particle states cannot accurately describe the behavior in this region [76].

It is important to note that the MBPT2 correlation energies are well-defined for a finite electron gas. However, they diverge in the thermodynamic limit of an infinite electron gas in three dimensions [76]. This happens at high densities due to the dominance of the particle-hole ring diagrams [3].

The single particle energies of the HEG when using the Ewald interaction are as follows for occupied single particle states:

$$\epsilon_i = \frac{\vec{k}_i^2}{2} - \sum_{j \neq i} \langle ij|\hat{v}_2|ji\rangle - \frac{v_M}{2}, \tag{4.16}$$

and for unoccupied single particle states:

$$\epsilon_a = \frac{\vec{k}_a^2}{2} - \sum_{j} \langle aj|\hat{v}_2|ja\rangle. \tag{4.17}$$

Here we have used the common denotation of indices where a, b, c,... represent states which are unoccupied in the Fermi ground state, i, j, k,... represent states which are occupied in the Fermi ground state, and p, q, r,... could represent either [76].

We can also define the two-electron matrix elements in integral notation as:

$$\langle ij|\hat{v}_2|ab\rangle = \delta_{ia}\delta_{jb}\frac{1}{\Omega^2}\int\int d\vec{r}_1 d\vec{r}_2 \psi_i^*(\vec{r}_1)\psi_j^*(\vec{r}_2)\hat{v}_2(\vec{r}_1,\vec{r}_2)\psi_a(\vec{r}_1)\psi_b(\vec{r}_2), \tag{4.18}$$

where $\hat{v}_2$ is defined in Equation (4.13) and ψ are plane waves defined in Sec. 4.2.

By convention, the energies calculated using the HEG will be reported in Hartrees (1 Hartree = $4.35x10^{-18}$ Joules). However, this differs from the units that will be used for the next system, infinite nuclear matter.

4.4 Interlude: Nuclear Forces

The properties of infinite nuclear matter and finite nuclear systems are determined by the nuclear forces that govern the interactions between the nucleons. These nuclear forces are given by two fundamental forces: the strong nuclear force, which binds the nucleons together, and the electromagnetic force, which causes repulsion between the protons. The strong force is stronger over short distances (which causes nuclei to hold together), but the electromagnetic force is a long-range force acting over large distances. The nature of the strong force needs to be better understood, and this can make it challenging to model the nuclear forces in theoretical calculations [18, 19]. Because of this, the first interaction used to perform calculations in this thesis is the "toy" interaction called the Minnesota potential, which is defined in Reference [109] and used in, for example, Refs [77, 78, 110] to model the nuclear interaction. The Minnesota potential is a local, nucleon-nucleon-only, moderately soft potential reproducing the nucleon-nucleon effective range parameters. It provides a reasonable approximation for the binding energies of light nuclei, but it is a simple interaction that is computationally inexpensive compared to more realistic nuclear interactions. It should be noted that the Minnesota potential is a phenomenological interaction, meaning that it was constructed by fitting experimental data instead of being derived from first principles. The matrix elements for the Minnesota potential are given by:

$$V_{i,j} = \frac{1}{2}[V_R + \frac{1}{2}(1+P_{ij}^{\sigma})V_t + \frac{1}{2}(1-P_{ij}^{\sigma})V_s](1+P_{ij}^{r}).$$

In the above equation, P^{σ} is the spin exchange operator, P^{r} is the space exchange operator, and V_R, V_t, and V_s are given by the following equations, where r_{ij} is the distance between two nucleons

and the constants are found by fitting to experimental data.

$$V_R = V_{0R} e^{-\kappa_R r_{ij}^2}$$

$$V_t = -V_{0t} e^{-\kappa_t r_{ij}^2}$$

$$V_t = V_{0s} e^{-\kappa_s r_{ij}^2}.$$

Though the Minnesota potential is computationally inexpensive and captures important aspects of the nuclear interactions, modern advancements have improved our models of nuclear forces [11, 18, 19, 104, 111–123]. The second nuclear force used to model infinite nuclear matter in this thesis, which is a more model interaction, is derived from practical field theory (EFT) [77] [2, 12–15, 18, 19, 29, 108, 119, 124–126]. The nuclear forces derived from EFT have an advantage over other nuclear forces in that the two-body and the many-body interactions can be derived in a mutually consistent matter [35]. Much recent progress has been made in deriving the nuclear forces based on chiral EFT and the nuclear Hamiltonian for many nucleonic matter calculations, not using forces from chiral EFT, including both nucleon-nucleon (NN) and three-nucleon (3N) forces [104, 112, 117, 118, 123, 127, 128]. For the results presented herein, we will use the parametrization NNLO$_{opt}$ for the NN and local 3N interactions.

It should be noted that implementing the 3N forces in the single particle basis used for infinite nuclear matter calculations is much simpler than implementing them in the harmonic oscillator basis commonly used for nuclei calculations [34]. However, the large number of matrix elements needed to compute 3N forces is still the computationally limiting factor in these calculations [34]. The specific chiral potentials used in this work are detailed in References [18] and [19]. They have optimized Δ-full interactions and are calibrated with nuclear matter properties [18]. The Δ-full interaction is detailed in References [120–122, 129–131]. The interaction employs a standard non-local regulator function of the form:

$$f(p) = e^{(p/\Lambda)^{2n}}, \tag{4.19}$$

where p is the relative momentum, n is 4, and Λ is 394 MeV. We are using these interactions at the next-to-next-to-leading order (NNLO). In this order, 17 low-energy coefficients (LECs)

parameterize the interaction and whose values are given in Reference [18]. This leads to a form of the Hamiltonian, which can be written as:

$$H(\alpha) = h_0 + \sum_{i=1}^{N_{LECs}=17} \alpha_i h_i, \tag{4.20}$$

where $h_0 = t_{kin} + v_0$, t_{kin} is the kinetic energy, v_0 is a constant potential that does not depend on the LECs, $\vec{\alpha}$ is a vector that denotes all of the LEC. Of the two interactions used in this thesis to model nuclear forces, the optimized Δ-full interaction is more accurate than the Minnesota potential, but has more complex matrix elements which leads to longer run times.

4.5 Infinite Nuclear Matter

Infinite nuclear matter is defined as a system containing infinite nucleons (protons and neutrons) that only interact via the nuclear forces [35]. Studies of infinite nuclear matter are essential for understanding the matter within dense astronomical objects such as neutron stars [77]. Neutron stars are exciting because they offer insights into nuclear processes and astrophysical observables. However, neutrons stars also contain matter that spans several orders of magnitude and contain many different compositions of matter [26–31, 77]. Neutron star matter occurs at densities of 0.1 fm^{-3} or greater and consists of various fractions of neutrons, protons, electrons, and muons. These particles exist in beta equilibrium (β-equilibrium) governed by the weak force.

Studies of infinite nuclear matter are also focused on determining the equation of state (EoS) . When considering applications to neutron stars, the EoS can help determine the mass range, the relationship between the star's mass and its radius, the thickness of the star's crust, and the rate at which the star will cool down [77]. Determination of the EoS also links neutron stars to the neutron skin in atomic nuclei and the symmetry energy of nuclear matter. The symmetry energy is crucial because it relates to the difference between proton and neutron radii in nuclei [35].

Solving the EoS depends on our ability to solve the many-body problem for infinite nuclear matter [35, 77]. The nuclear matter has been of interest in many-body studies since the early days of the field (see Reference [132] for a review of these early studies). These early calculations were performed with Brueckner-Bethe-Goldstone theory ([133, 134]). However, modern many-body

studies of nuclear matter are performed with a varied of methods, including coupled-cluster theory (CC) ([2, 3, 34, 35, 77, 78, 119, 125]), Monte Carlo methods ([16, 23, 32, 33, 57, 105, 106, 135, 136]), Green's function methods ([107, 108, 137, 138]) and methods from the renormalization group theory family ([77, 126, 139]). The coupled-cluster theory will be the many-body method of interest for the remainder of this thesis. Coupled-cluster calculations of nuclear matter date back to the 1970s and 1980s [35].

One property of infinite nuclear matter we are interested in is the proton fraction, which is defined as

$$x_p = \frac{\rho_p}{\rho}, \tag{4.21}$$

where ρ_p is the density of protons in the matter, ρ_n is the denisty of neutrons in the matter, and $\rho = \rho_p + \rho_n$ is the total density of the infinite nuclear matter [77]. Defining different proton fractions defines different types of infinite nuclear matter systems. For example, if $x_p = 0$, the infinite nuclear matter system contains only neutrons. We will refer to this system as pure neutron matter (PNM). If $x_p = 1/2$, then the system contains an equivalent number of protons and neutrons, and this system is called symmetric nuclear matter (SNM).

From the proton fraction, we can define the symmetry energy as the difference between the energy for pure neutron matter and symmetric nuclear matter at a set density [77]:

$$E_{sym}(\rho) = E(\rho, x_p = 0) - E(\rho, x_p = 1/2). \tag{4.22}$$

All energies for infinite nuclear matter calculations will be, by convention, reported in units of MeV.

4.6 Truncation Errors in Infinite Matter Calculations

As mentioned throughout this chapter, several truncations and approximations must be made to study these infinite matter systems in a computational framework. Though computational limitations require these truncations and approximations, they introduce errors into the calculations, making them undesirable [38, 76]. Though it is impossible to remove every error that comes from the various truncations and approximations, this thesis will focus on mitigating the effects of three

truncation errors: the error that results from truncating the number of single-particle states in the basis, the error that results from truncating the number of particles and volume of the infinite matter system, and the error that results from truncating the coupled cluster correlation operator, $\hat{T}$.

As discussed in Section 2 of this chapter, the number of single-particle states (M) in the calculation is truncated due to truncating the total number of energy shells allowed in the system. This introduces an error in the calculation called the basis incompleteness error. The basis incompleteness error can be mitigated by increasing the number of single-particle states (or energy shells) in the calculation. Unfortunately, this also drastically increases the computational time and resources needed. Coupled cluster theory has polynomial time scaling with respect to M, which, while being better than other many-body methods, still means that computational times can be prohibitive at large values of M. Additionally since increasing the number of single-particle states in the system increases the number of matrix elements in the calculations, computational resources are also increased.

While infinite matter systems theoretically contain infinite particles over an infinite volume, these values must be truncated to finite values computationally [35]. This introduces an error in the calculations called finite size error. Since the number of particles in the system and the volume of the system are related by a fixed density ($\rho_0 = N/\Omega$), it is sufficient to increase N to the infinite limit while keeping the density constant. However, the same problem occurs when attempting to increase M since coupled cluster theory also has polynomial time scaling with respect to the number of particles in the system; however, the power tends to be higher with particles than with single-particle states. Additionally, matrix elements involving occupied states (or particles) are computationally more complex than those involving only unoccupied states. Therefore, adding more particles to the system increases the computational resources needed than additional single-particle states.

As discussed in the previous chapter, the coupled cluster correlation operator is:

$$\hat{T} = \sum_{m=1}^{N} \hat{T}_m, \tag{4.23}$$

where N is the total number of particles in the system and $\hat{T}_m$ represents the m-particle m-hole excitation operator. We can make this operator specific to infinite matter systems by making two

changes. First, infinite matter systems contain particles, so $N = \infty$. Second, the 1-particle 1-hole excitation operator, $\hat{T}_1$, will always be zero for infinite matter systems due to symmetry in the momentum. Therefore, we can write the cluster operator for infinite matter systems as:

$$\hat{T} = \sum_{m=2}^{\infty} \hat{T}_m. \tag{4.24}$$

Also, as discussed in the previous chapter, the correlation operator is truncated in practice by setting excitation operators over a certain level to zero. In this thesis, the coupled cluster results will either be calculated using coupled cluster doubles (CCD) where $\hat{T} \approx T_2$ or approximations to coupled cluster doubles triples (CCDT) where $\hat{T} \approx T_2 + T_3$. Since the full cluster operator is not used in the calculations, this does reduce the accuracy of the calculations. However, in this thesis, we will not be extrapolating to the complete coupled cluster calculation since this would require performing coupled cluster calculations at more truncation levels. However, we will attempt to improve the accuracy of infinite matter calculations by predicting the CCDT result from the CCD result and the CCD result from the MBPT2 result, thus improving the accuracy while keeping computational time and resources small.

However, in the infinite matter calculations, there are sources of error from approximations and truncations that are not addressed in this thesis but could be the subject of future work. Some of them will be discussed briefly here. First, the Coulomb interaction, the predominant interaction in the homogeneous electron gas, is a long-range interaction that acts over an infinite distance. Truncating the volume of the electron gas to a finite size also truncates the Coulomb interaction. It is important to note that this is a finite size error occurring in the HEG but not in infinite nuclear matter [3]. This means that studies of the HEG at finite sizes will face additional complications compared to studies of infinite nuclear matter [3]. In addition to truncating the coupled cluster correlation operator, not all n-body interactions are included in the calculation because the Hamiltonian is also truncated. In this work, calculations are limited to including, at most, the two-body or three-body matrix elements. However, the highest level of matrix elements contributing to the system is k-body, where N is the total number of particles. Finally, while the Coulomb interaction, the primary interaction of the electron gas has a simple closed-form representation that can be calculated

precisely, the primary interaction in the infinite nuclear matter is the nuclear interaction. However, the nuclear interaction has no simple closed-form solution and cannot be represented precisely. Therefore all nuclear interactions are approximations with various levels of accuracy compared to real nuclear interaction.

4.7 Conclusion

In this chapter, we have explored the two systems of interest in this thesis: the homogeneous electron gas and infinite nuclear matter. We have also developed a single-particle basis for these calculations which provides a convenient method for truncating the number of particles in the system and the number of single-particle states, which is required due to computational limitations. We have also investigated two nuclear interaction models and seen some errors that can occur when trying to model an infinite system in a finite computational space.

Up until this point, this thesis has focused on the physical framework necessary for understanding this work. We first developed a many-body framework to define our many-body systems. We then developed three many-body methods which can be used to find the energies of our systems, and finally, we explored the two systems of interest. The following two chapters will develop the needed computational framework: machine learning and the basics of the SRE method.

CHAPTER 5

BAYESIAN MACHINE LEARNING

5.1 Machine Learning

Machine learning is the field that occurs at the intersection of data science and artificial intelligence; it is the science of programming a computer so that it can learn from a given data set [79]. Machine learning algorithms can solve these problems without being programmed with task-specific instructions; machine learning encompasses a set of generic algorithms that can be applied to various problems.

Machine learning can be applied to a wide variety of problems, but it is most useful when traditional styles of programming cannot solve the problem or would take a very long time to solve a problem. This includes problems where the traditional solution would have a long list of rules; machine learning can find patterns in the data set without problem-specific programming. Machine learning also excels in problems where a large amount of complex data is difficult to sort and work with by hand or with traditional programming [79].

Machine learning algorithms can be classified into one of three categories: supervised, unsupervised, and reinforcement learning. In supervised learning, the training data given to the algorithm is labeled, meaning it has both an X and a y component. Therefore, supervised learning aims to map every X to its corresponding y correctly. There are two types of supervised learning: classification (when y contains a finite number of possible values) and regression (when y contains an infinite number of possible values). All machine learning algorithms developed in this chapter will be supervised learning. Unsupervised learning algorithms work with unlabelled data, meaning they only receive the X component of the data set. Common unsupervised learning tasks are clustering and dimensionality reduction. Finally, reinforcement learning describes a set of algorithms where an agent learns to solve a problem by maximizing its reward. Supervised and unsupervised learning are common in machine learning applications in physics, while reinforcement learning in physics applications is less common.

Machine learning has recently become a popular tool in physics and is being applied to a wide range

of problems. Machine learning has found applications across all areas of physics. Specifically looking at neural networks, they have been used in nuclear physics to perform extrapolations (for example References [140], [66], [141],[142],[67],[68],[69],[52]) and to directly solve the many-body problem (for example Ref [53]). Neural networks can be problematic for *ab initio* data sets due to very small data sets but will be explored in this thesis as a possible machine learning method. Machine learning can also be used in many-body physics to simulate the wavefunction of a system to find the variationally lowest energy of the system [57]. More generally, machine learning algorithms have been used to predict the coupled cluster energies of molecules using a variety of machine learning algorithms. The work presented in this thesis, though unique, has been inspired by these previous applications of machine learning in the field of many-body physics.

The remainder of this chapter will be structured as follows. The following sections will develop four standard supervised learning algorithms: linear regression, its related algorithms ridge and kernel ridge regression, and finally, a brief discussion on neural networks and recurrent neural networks. Next, there will be a discussion of Bayesian statistics to prepare the reader for the development of the final two machine learning algorithms: Bayesian ridge regression and Gaussian processes.

5.2 Linear Regression

Linear regression is one of the most basic machine learning algorithms. However, it is often the first algorithm studied in a machine learning class or textbook because it still contains all of the essential elements of a supervised machine learning algorithm (a loss function, parameters, and optimization). One of the main reasons linear regression is still studied as a machine learning algorithm is that it has analytical expressions for its parameters. This contrasts with other supervised machine learning algorithms, such as neural networks, which use numerical algorithms to optimize the parameters. Furthermore, due to the analytical expressions for the parameters, linear regression will always return the same output if given the same input, again in contrast to other machine learning algorithms such as neural networks. This reproducibility makes linear regression and its related algorithms attractive for physical applications.

The output of the linear regression algorithm can be written as follows:

$$\hat{y}_{Linear} = X\theta, \tag{5.1}$$

where X is the input of the algorithm and θ is a vector of constant parameters, often called the weights. The "machine learning" part of the algorithm is tuning θ so that:

$$y_{True} = \hat{y}_{Linear}. \tag{5.2}$$

In machine learning, a loss function measures the error in the algorithm's prediction from the real data. The loss function for the linear regression algorithm is the mean-squared error function:

$$J_{Linear}(\theta) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \hat{y}_{Linear,i})^2, \tag{5.3}$$

where y = y_{True}.

The optimal values of θ are the ones that make the output of the linear regression algorithm the same as the actual data. Another way of phrasing this is that the optimal values of θ are the values that minimize the loss function. Thus linear regression becomes a simple minimization problem! Minimizing the loss function with respect to the parameters θ yields the optimal parameters, the values of θ that make the output of the linear regression algorithm as close to the actual data as possible.

$$\theta_{Linear} = (X^T X)^{-1} X^T y. \tag{5.4}$$

These optimized values of θ can then be multiplied by new data not in the training set to make predictions for new data.

5.3 Ridge Regression

Several regularized linear regression forms exist, including ridge, LASSO, and elastic net. This section focuses on ridge regression, where the base linear regression loss function is regularized by the L2 norm of the weights, θ [79, 143, 144].

The output of a ridge regression algorithm is the same as for linear regression:

$$\hat{y}_{Ridge} = X\theta, \tag{5.5}$$

The main difference from linear regression comes with the loss function. The ridge regression loss function, shown in Equation (5.6), consists of the standard mean-squared error as the first term and a second term known as a regularization term. The regularization term is simply the L2 norm of the weights, θ, scaled by a parameter λ, known as the strength of the regularization. Note that in some sources, this scaling factor is shown as $\lambda/2$, or sometimes as α.

$$J_{Ridge}(\theta) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \hat{y}_{Ridge,i})^2 + \lambda\sum_{i=0}^{n-1}\theta_i^2 \tag{5.6}$$

The optimal values of θ will again be found via a simple minimization which results in optimal values of θ which are close to θ_{Linear} but which contain an additional term from the regularization:

$$\theta_{Ridge} = (X^T X - \lambda \mathbf{I})^{-1} X^T y, \tag{5.7}$$

Ridge regression tends to be more accurate than linear regression due to the bias-variance trade-off. The addition of the regularization term to the loss function adds some amount of bias to the algorithm. This, in turn, causes a drop in the variance, leading to ridge regression being better able to generalize to data outside of its training set than linear regression. The regularization term also keeps the model's weights small, which can have computational benefits.

The value chosen for the hyperparameter λ can significantly affect the values of the optimized θ_{ridge} and the results of the future predictions. For example, if $\lambda = 0$, then $\theta_{ridge} = \theta_{Linear}$ (the ridge regression algorithm becomes linear regression because the regularization term is removed). On the other hand, if λ is very large, then the weights will all be forced to be very small, and the resulting predictions will be a straight line through the mean of the data set.

Since the results of a ridge regression model depend on the value of λ chosen, hyperparameter tuning is used to choose an optimal value. In hyperparameter tuning, a ridge regression model is trained using the training data set and various values of λ. The algorithm's performance with each new value of λ is checked against a validation data set, which can be a subset of the training set, the test set, or a separate data set. For a thorough hyperparameter tuning of λ, hundreds of different values are typically tested, spanning several orders of magnitude. The value of λ that is kept is the one that produces the lowest error when recreating the validation data set.

While finding a value of λ that produces an acceptable error in this manner is possible, it poses a few problems. First, there is a theoretical value of λ that will minimize the loss function when applied to the validation data set. However, it is unlikely that the exact value will be found in a traditional hyperparameter tuning process where the list of supplied values for λ are tested by brute force. This leaves the open question of if the value of λ that performed best in the tuning process is the best value overall. Secondly, while the hyperparameter tuning process can be fast (depending on the number of values tested), it is an additional step that needs to occur with each new data set and thus adds time to the ML analysis. Lastly, hyperparameter tuning requires a validation data set, either taken from the training or test data set, making them smaller, or generated separately. While this may be fine for some machine learning applications, especially those with large data sets or where data is easy to generate, this is a significant drawback for many-body applications since new data points can represent large computational requirements in time and resources.

5.4 Kernel Ridge Regression

A regression algorithm related to ridge regression is called kernel ridge regression (KRR) [62, 79, 143]. While it uses the same loss function as ridge regression, the inputs are altered using a kernel function. This is known as the kernel trick and adds non-linearity into the model, making it able to generalize to a broader range of data sets. KRR assumes that the output data, y, can be approximated by multiplying a set of weights, θ, by values from a kernel function:

$$\hat{y}_{KRR} = \sum_{i=0}^{n-1} \theta_i k(x, x_i), \tag{5.8}$$

where the points x_i correspond to the training data, and x is the point to produce a prediction. The function k is the kernel function and can take different forms depending on the application. Some common kernel functions are the polynomial kernel,

$$k(x, y) = (\gamma x^T y + c_0)^d, \tag{5.9}$$

and the sigmoid kernel,

$$k(x, y) = tanh(\gamma x^T y + c_0). \tag{5.10}$$

Each of these kernel functions comes with some hyperparameters: γ, c_0, and d for the polynomial kernel and γ and c_0 for the sigmoid kernel, for example. The user must set these hyperparameters before the algorithms are trained. Even the kernel function can be considered a hyperparameter since the user chooses it, and it affects the results of the trained algorithm.

The loss function for KRR is the same as ridge regression's, a regularized form of the mean-squared error function where the regularization term is the L2 norm of the weights, θ. Therefore, KRR still has the λ hyperparameter. Just like the previous algorithms, the optimized values of the weights are found by minimizing the loss function with respect to the weights yielding optimized values of the weights to be:

$$\theta_{KRR} = (K - \lambda \mathbf{I})^{-1} y, \tag{5.11}$$

where K is known as the kernel matrix and is defined as:

$$K_{i,j} = k(x_i, x_j). \tag{5.12}$$

While KRR is better at generalizing than linear regression and ridge regression due to the inclusion of the kernel function, this does introduce significantly more hyperparameters into the algorithm. Furthermore, this drastically increases the complexity of the hyperparameter tuning process since the kernel function and its hyperparameters must also be tuned in addition to λ. Therefore, while KRR can produce the most accurate predictions once trained of the three regression algorithms investigated here, the requirements for its hyperparameter tuning are a significant drawback when total run time and data set size are restrictions, as in the case with the research presented in this thesis.

5.5 Neural Networks

Neural networks are computational systems that can learn to perform tasks by considering examples, generally without being programmed with any task-specific rules. Another way to phrase this is that a neural network is a computational system that learns to match a given input to the correct output. They are a broad category of machine learning algorithms, including popular algorithms such as convolutional neural networks, recurrent neural networks, and deep learning.

Fully Connected Feedforward Neural Network

Though there are many types of neural networks, using just the phrase "neural network" typically refers to a type of neural network known as a fully connected feedforward neural network (FFNN). This type of network can also be known as a multilayer perceptron (MLP) if it has at least one hidden layer. These FFNNs contain many interconnected layers which transform the given input into an output [51]. The base unit of an FFNN is called a neuron, and these neurons are arranged into columns which are called layers [51]. Information in an FFNN moves only forward. Additionally, each neuron is connected to every neuron in the next layer, and there are no connections between neurons in the same layer. This means that the input to a layer in the neural network is simply the output from the previous layer. As we will see in a moment, each neuron receives a weighted sum of the outputs of all neurons in the previous layer.

5.5.0.1 Mathematics of Neurons and Layers

Each neuron is a mathematical function involving a column from a weight matrix, a scalar from a bias vector, and an activation function. We can represent the mathematical form of the ith neuron in the first hidden layer as:

$$\hat{y}_i = f(\sum_{j=1}^{M} w_{ij}x_j + b_i), \tag{5.13}$$

where x is the input to the neural network, w is the weight matrix that scales the input of the neuron, **b** is the bias vector that ensures the neuron output is non-zero. Finally, the function f is known as the activation function, which adds nonlinearity to the neuron [51]. The weights and biases are free parameters (meaning they are set for the neural network during training), but the activation function is a hyperparameter [51].

For each neuron, i, in the first hidden layer of a neural network, we can represent its mathematical form as:

$$\hat{y}_i^1 = f^1(\sum_{j=1}^{M} w_{ij}^1 x_j + b_i^1). \tag{5.14}$$

We can also write out the mathematical form for the entire first hidden layer as:

$$\hat{y}_1 = f^1(W_1\mathbf{x} + \mathbf{b}_1). \tag{5.15}$$

Similarly, for the second hidden layer, we can represent the mathematical form of each neuron as:

$$y_i^2 = f^2(\sum_{j=1}^{N} w_{ij}^2 y_j^1 + b_i^2). \tag{5.16}$$

Note here that the weights matrix is no longer multiplied by the inputs to the neural network (x) but rather by the output of the first hidden layer. This is because the input to the first hidden layer is the input to the neural network, but the input to the second hidden layer is the output of the first hidden layer. Therefore, we can expand the above equation to be more precise:

$$y_i^2 = f^2(\sum_{j=1}^{N} w_{ij}^2 f^1(\sum_{k=1}^{M} w_{kj}^1 x_k + b_j^1) + b_i^2). \tag{5.17}$$

We can also write a mathematical form for the entire second hidden layer in matrix-vector form as:

$$\hat{y}_2 = f^2(W_2 \hat{y}_1 + \mathbf{b}_2), \tag{5.18}$$

or more explicitly as

$$\hat{y}_2 = f^2(W_2 f^1(W_1(x) + \mathbf{b}_1) + \mathbf{b}_2). \tag{5.19}$$

Finally, we can use the pattern we have developed to write down the equation for the mathematical output for a neuron on the l-th hidden layer of the neural network. For the i-th neuron on the l-th layer, we can describe it mathematically as:

$$y_i^l = f^l(\sum_{j=1}^{N_l} w_{ij}^l y_j^{l-1} + b_i^l), \tag{5.20}$$

and more explicitly as

$$y_i^l = f^l(\sum_{j=1}^{N_l} w_{ij}^l f^{l-1}(\sum_{k=1}^{N_{l-1}} w_{kj}^{l-2} y_k^{l-1} + b_j^{l-1}) + b_i^l), \tag{5.21}$$

and finally, all the way expanded as

$$y_i^l = f^l(\sum_{j=1}^{N_l} w_{ij}^l f^{l-1}(\sum_{k=1}^{N_{l-1}} w_{jk}^{l-1}(\cdots f^1(\sum_{n=1}^{M} w_{mn}^1 x_n + b_m^1)\cdots) + b_k^{l-1}) + b_j^l). \tag{5.22}$$

We can also write a mathematical expression for the output of the entire l-th layer using matrix-vector format as:

$$\hat{y}_l = f^l(W_l \hat{y}_{l-1} + \mathbf{b}_l), \tag{5.23}$$

which can be expanded to

$$\hat{y}_l = f^l(W_l f^{l-1}(W_{l-1}\hat{y}_{l-2} + \mathbf{b}_{l-1}) + \mathbf{b}_l), \tag{5.24}$$

and finally to

$$\hat{y}_l = f^l(W_l f^{l-1}(W_{l-1}(\cdots f^1(W_1\mathbf{x} + \mathbf{b}_1)\cdots) + \mathbf{b}_{l-1}) + \mathbf{b}_l). \tag{5.25}$$

It is a complicated expression and can grow to be very large, but it is a set equation that describes the output of a neural network with l-1 hidden layers and an output layer. So it is also possible to rephrase the definition of a neural network to be an analytical function that maps a set of inputs to a set of outputs using a set of optimized parameters [51].

Activation Functions

The activation function is nonlinear. Its purpose is to allow the neural network to capture nonlinear patterns in the data set [51]. Without an activation function, a neural network could only produce an output that was a linear combination of the inputs. There are many choices for the activation function for the neurons, but standard activation functions are the sigmoid, the hyperbolic tangent function (tanh), and the rectified linear unit (ReLu) [51]. Some neural network architectures will have some layers with no activation function. For example, it is common to have no activation function on the output layer so as not to constrain the range of values the output layer can produce.

Loss Functions and Training Neural Networks

A loss function determines how much the output from a neural network differs from the real/expected result. There is no set loss function used with neural networks, but two common loss functions are the mean-squared error loss function and the mean absolute error function. The mean-squared error loss function (MSE) can be defined as:

$$J_{MSE}(W) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2,$$

where y is the true data set, $\hat{y}$ is the neural network prediction, N is the number of data points in the set, and W is the neural network's weights. The loss function depends on the weights of the neural network because changing the weights of the neural network changes its output.

The mean-absolute error loss function (MAE) has a similar form:

$$J_{MAE}(W) = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|,$$

A significant part of working with neural networks is a process known as training, where the weights of the neural network are optimized such that the cost function is minimized. This training process has two phases: the forward pass and the backpropagation.

The forward pass occurs when data is sent through the neural network (left to right on the above graph) to produce a predicted output. This predicted output is fed into the loss function with the actual data set to generate the loss value.

After the forward pass comes backpropagation, where the error from the loss function is backpropagated through the layers of the neural networks, and its weights are adjusted layer by layer so that the next forward pass will result in a reduced loss value. A simple way to optimize the weights of a neural network during backpropagation is through an optimization technique known as gradient descent. The weights of the neural network are adjusted by the derivative of the loss function with respect to the weights, scaled by a hyperparameter known as the learning rate:

$$W = W - r_l\frac{\partial J(W)}{\partial W},$$

The learning rate (r_l) is a number typically much less than one, and it is also a hyperparameter, so its value must be set before the neural network is run.

The process of training a neural network involves many different iterations of forward pass followed by backpropagation. Typically a training process will continue until a certain number of training iterations has been reached, or the difference in the current loss value compared to the value from the previous iteration is below a certain threshold. However, neural networks should not be trained for an overly long time because this will lead to something called overfitting, where the neural network learns to match the data set it is trained with very well (so it will show a small loss value) but loses all generality when given new data (so it will perform poorly when given the new data set).

Recurrent Neural Networks

There are many different types of neural networks besides the basic FFNN, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) [51]. However, RNNs will be the focus of this section as they are designed to handed ordered data, which may make them better at extrapolations than regular neural networks. Furthermore, recurrent neural networks can be used to analyze time series data; thus, they have typical applications in finance.

In feedforward neural networks, data only flows in one direction, from an input layer to the output layer. The architecture of a recurrent neural network will look very similar to a feedforward neural network, except each neuron feeds its output back into itself and sends its output to the next layer. For every data point the RNN receives, each neuron receives both the input from the previous layer (using the same equations as a feedforward neural network) and its output from the previous data point. This combination of inputs gives an RNN a memory, making it capable of predicting the future when analyzing time series data or being an extrapolator for other data sets.

The main problem RNNS faces is a vanishing or exploding gradient that can occur during the training process. It is possible to fix this problem using different types of RNN layers. Common alternative RNN layers are long-short-term memory layers (LSTM) and gated rectified unit layers (GRU).

A Discussion on Neural Network Hyperparameters

One of the drawbacks of neural networks is that they have many hyperparameters that need to be set by the user before the neural network can be trained. Examples of hyperparameters include the activation function, the learning rate, the use of dropout during the training process, etc. When using RNNs, the type of RNN layer (regular, LSTM, or GRU) adds another set of hyperparameters. Additionally, the number of layers in the neural network defines its depth, and the number of neurons in the hidden layers defines the width of the neural network [51]. Both of these are also hyperparameters. A network's ability to find patterns in complex data is governed by the number of hidden layers and neurons the network contains, but too many neurons and layers can make for a slow training process [51]. Thus, with the vast number of possible hyperparameter combinations present

in neural networks, hyperparameter tuning can be difficult and time-consuming. Additionally, since neural network weights are initialized randomly, and the optimization of the weights is typically not performed with a closed-form optimization algorithm, a neural network will return a different result every time it is trained. This means that to ensure that a neural network has a good set of hyperparameters to give accurate results reliably, it must be retrained many times, and its results averaged to determine its actual performance. It is not uncommon in physical science applications where producing reliable results is essential to use a "forest" of 100 identical neural networks and use the average results, the true answer, and the standard deviation across all the predictions as the uncertainty of the result [51].

5.6 Bayesian Ridge Regression

The following two machine learning algorithms we will investigate belong to a family of algorithms called Bayesian machine learning, so called because they rely on Bayes' theorem and Bayesian statistics to both make predictions and determine the uncertainty of those predictions. The main draw of using Bayesian machine learning algorithms over standard machine learning algorithms is that Bayesian algorithms determine their hyperparameters using Bayesian statistics to determine the most likely values. They also produce uncertainty in their results, which is essential for machine learning applications in the physical sciences [79, 143, 145].

Therefore an alternative to performing traditional ridge regression with hyperparameter tuning is to use Bayesian ridge regression, the Bayesian form of ridge regression where the training process sets the value of λ [79, 143, 146, 147]. In Bayesian ridge regression, the output data, y, is assumed to be in a Gaussian distribution around Xθ:

$$p(y|X, \theta, \lambda) = N(y|X\theta, \lambda), \tag{5.26}$$

Priors for the weights, θ, are give by a spherical Gaussian, where β^{-1} is taken to be the precision:

$$p(\theta|\beta) = N(\theta|0, \beta^{-1}\mathbf{I}), \tag{5.27}$$

where $\mathbf{I}$ is the identity matrix.

The priors over λ and β are assumed to be gamma distributions, and θ, λ, and β are estimated jointly when the model is fit. The Bayesian ridge regression analysis will result in different weights than a regular ridge regression analysis, but a Bayesian ridge regression algorithm is more robust. Additionally, since the value of λ is set when the algorithm is trained, using Bayesian ridge regression eliminates the need for hyperparameter tuning and a validation data set, making it a better choice than ridge regression when time savings is essential, and the size of the data sets is minimal.

5.7 Gaussian Processes

Gaussian processes (GP) are non-parametric models that use Bayesian statistics to provide a machine learning algorithm that can model complex relationships between the data, even if the predictions are based on uncertain information. GPs can be used for classification, but we will only consider regression applications in this thesis, meaning the GPs will learn to model a continuous function. GPs assume that the function to be modeled can be represented as a Gaussian distribution over the function values at every point [79, 143, 145].

Given a set of inputs **x** and a set of outputs **y**, the GP defines a joint distribution over all outputs such that the subsequent multivariate Gaussian distribution, N, is created:

$$p(\mathbf{y}|\mathbf{x}) = N(\mu, \Sigma),$$

The vector μ contains the mean of each element of y ($\mu_i = E[y_i]$), and Σ is the covariance matrix with elements $\Sigma_{i,j} = k(y_i, y_j)$, where k is known as the covariance or the kernel function.

The kernel function used in the GP analysis presented in this work is the rational quadratic (RQ) kernel, which is defined as:

$$k(x_i, x_j) = (1 + \frac{d(x_i, x_j)^2}{2\alpha l^2})^{-\alpha},$$

where α is a hyperparameter known as the scale mixture parameter, l is a hyperparameter that sets the length scale of the kernel, and the function d calculates the Euclidean distance between x_i and x_j. The values of the hyperparameters are set during the training process. The rational quadratic kernel can model functions with a mixture of local and global smoothness. It was chosen as the

kernel function for this analysis because it produced the lowest RMSE error on the extrapolations compared to other kernel functions.

GPs were chosen as a primary machine learning method because they can be thought analogous to the Bayesian implementation kernel ridge regression (KRR). KRR relies on user-chosen hyperparameters, complicating the analysis process. The values of these hyperparameters (one from the loss function, 2-5 from the kernel) are chosen through hyperparameter tuning, where many combinations of hyperparameters are tested to find the best combination. The drawback to this method is that it requires a validation data set to be used for the hyperparameter tuning, in addition to the training data set and the test data set. Additionally, it was found that the data sets used in this work are sensitive to the values of the hyperparameters (a slight change in hyperparameter value could lead to a significant change in extrapolated value), so a quiet through hyperparameter tuning would need to be performed to find the optimal values.

In the context of this research, we are interested in time savings in addition to prediction accuracy. Therefore the goal is to accurately generate the converged correlation energies with as few training points as possible and as fast as possible. Furthermore, eliminating the need to perform hyperparameter tuning saves a significant amount of time in the machine learning analysis and eliminates the need to generate a validation data set.

5.8 Conclusion

In this chapter, we have introduced machine learning and investigated different supervised learning methods. However, as they are commonly formatted, these algorithms will only make a good extrapolator for some-body data sets. RNNs, built for extrapolations, require too much training data to be helpful in a many-body application (where generating data points is computationally expensive), and the hyperparameter tuning process associated with RNNs is a significant drawback. However, the other machine learning algorithms we investigated here need to be built for extrapolation and thus perform poorly. Thus in the next chapter, we will create a method of formatting a data set, called sequential regression extrapolation (SRE), that will take a simple regression algorithm, such as ridge regression, and turn it into a powerful extrapolator.

CHAPTER 6

SEQUENTIAL REGRESSION EXTRAPOLATION (SRE)

6.1 Introduction to Methodology

Removing the basis incompleteness and finite size errors by formatting it as an extrapolation problem is a promising application of machine learning because it is well formatted. Furthermore, the extrapolations are known to have asymptotic values for the correlation energies (i.e., the converged values), there is a simple pattern in the data for the machine learning to pick up, and the training data is already near the converged values (but still far enough away that performing the extrapolation increases the accuracy of the results) [51].

Attempting to remove basis incompleteness and finite size errors from many-body calculations with machine learning is not a novel idea. For example, in Ref [51], the authors attempt to use a neural network to perform an extrapolation to remove the basis incompleteness error for a harmonic oscillator basis used in coupled cluster calculations of nuclei. However, the authors encountered the problems that would be expected when using neural networks for this application, including difficulty working with small data sets (leading to the need for interpolation before extrapolation) and neural network's inability to create exactly reproducible results due to its training process [51]. Another example of using machine learning to remove truncation errors from many-body calculations can be found in [54].

We should point out that other methods exist to eliminate the finite size errors other than performing calculations at higher N or extrapolating. One of these options is to use twisted boundary conditions instead of periodic boundary conditions, which provide much more accurate estimates of the correlation energies [3].

6.2 Formulation

As discussed in the previous chapter, supervised machine learning algorithms require label training data, meaning that both x and y components exist. Therefore, a supervised machine learning algorithm learns to approximate a function, f, during its training process such that $f(x) = y$ for

every value of x in the data set.

However, supervised machine learning algorithms tend to fail when asked to make predictions outside their training range (for example, see Reference [51]). Put another way, supervised machine learning algorithms tend to make more extrapolators and are thus not used for extrapolation applications. However, one type of supervised machine learning performed well with extrapolations; in fact, it was designed to make extrapolations on time series data. This algorithm is the recurrent neural network discussed in the last section. RNNs perform very well on extrapolations as their design inherently gives them some "memory" of previous data that they can use to predict new data. While RNNs may be a good choice for performing time series analysis and extrapolations, they have some significant drawbacks when applying them to *ab initio* data sets. First, RNNs, like all neural networks, require much training data to make accurate predictions and avoid overfitting. While this is not a problem for many applications of neural networks, this is a significant drawback when applying any neural network to *ab initio* data sets. Since each new point in an *ab initio* data set can represent significant computational time and resource investment, these data sets are usually relatively small, especially by neural network standards [51]. One way around this is to use an interpolation algorithm to increase the size of the data set artificially, but this represents an additional step in the workflow and possibly an additional source of error in the analysis [51]. Thus in this work, we want to avoid using interpolation as much as possible.

Secondly, due to how their weights are initialized, neural networks are inherently random [51]. This means a neural network, including RNNs, will produce slightly different results each time it is trained, even when the same training data is used. The irreproducibility is a significant drawback when using neural networks to predict physical values. Additionally, the uncertainty of a neural network's prediction is an open research question.

Finally, as mentioned in the previous chapter, neural networks generally have a large number of hyperparameters, including the number of hidden layers, the number of neurons per hidden layer, and the activation function, among many others. This leads to complicated and long tuning processes, dramatically increasing the runtime needed to perform an ML analysis.

While RNNs are not a good choice of a machine learning algorithm for this application, we can take inspiration from them and create a machine learning algorithm that can. A standard machine learning algorithm is trained to take a point from the training set, x_i, and match it to its corresponding y values, y_i. Thus training a standard machine learning algorithm has the following relations to be learned:

$$f_{ML}(x_1) = y_1, f_{ML}(x_2) = y_2, ... \tag{6.1}$$

While this training pattern makes many supervised machine learning algorithms excellent at matching new inputs to the correct output, they typically only perform well in the range of data encompassed by the training data.

However, there is a way to train RNNs around this problem. This training pattern is typically used when RNNs perform time-series analysis, which is common in the financial industry. Instead of learning the relationship between the x and y components of the data set, a time-series data formatting teaches the RNN to learn the pattern between a sequence of y values and the next y value in the training data. The training points in this form will look as follows:

$$f_{RNN}(y_{k-3}, y_{k-2}, y_{k-1}) = y_k, \tag{6.2}$$

making the RNN much better at extrapolating because it is trained to predict the next value in a sequence. Note that the sequence length in the inputs could be of any length, thus adding another hyperparameter.

The drawback of this form of training here is that the data must be sequential (have an order to it), and the data must be evenly spaced regarding the dependent variables. Additionally, since the RNN only sees the y component of the data, it may lose the information encoded in the x component. However, an RNN trained in this manner is an extremely powerful extrapolator.

Nevertheless, as explained earlier in this section, there are several drawbacks to using RNNs in this application. However, we can combine the training style of an RNN with a simpler machine-learning algorithm to make a more straightforward but still powerful extrapolator. Furthermore, since none of the data in this thesis is time-dependent, instead of calling this style of formatting the

data a time series, we will call it sequential formatting, as the data must be arranged sequentially. This gives rise to the name of the machine learning algorithm we are building to extrapolate many-body data sets, sequential regression extrapolation (SRE); so named because it combines sequential data formatting with a regression algorithm to create an extrapolator that can make predictions from small data sets. Thus we will train a regression algorithm using the following format:

$$f_R(y_{k-3}, y_{k-2}, y_{k-1}) = y_k. \tag{6.3}$$

The length of the input, now known as the SRE sequence length, can be any length. However, the larger the sequence length is, the fewer total data points present once the data has been formatted, so there is a balancing act between choosing a sequence length long enough to encode the sequential patterns in the data but not so long that there are very few points remaining after the formatting.

Now that we have developed the algorithm generally, we can apply it to remove the basis incompleteness and finite size errors arising from many-body calculations of infinite matter systems. The remainder of this section will develop the specific SRE formalism to remove these errors.

6.3 CCD Extrapolations With Respect to Number of Single Particle States

This section describes the SRE formulation to remove the basis incompleteness errors from CCD calculations of infinite matter.

Both the ΔE_{CC} and ΔE_{MBPT} converge as M increases. In that case, there must be a large value of M where their ratio becomes a constant:

$$\frac{\Delta E_{CC,Large\ M}}{\Delta E_{MBPT,Large\ M}} \longrightarrow constant. \tag{6.4}$$

One of the machine learning algorithms described in the previous chapter will be used to find this constant value using only data collected at low values of M, where the ratio still needs to be converged. The energies ΔE_{CC} and ΔE_{MBPT} are calculated for the training data set generated at small values of M. The exact values of M will depend on the value of N and the system in the calculations. However, the largest value of M used across all training data in this thesis is 2,090 single-particle states. For each data set (constant N and ρ_0/r_s), the training data for the regression

algorithm was created by dividing ΔE_{CC} by ΔE_{MBPT} at the exact value of M.

$$y = \frac{\Delta E_{CC,M}}{\Delta E_{MBPT,M}} = \frac{\Delta E_{CC,M_1}}{\Delta E_{MBPT,M_1}}, \frac{\Delta E_{CC,M_2}}{\Delta E_{MBPT,M_2}}, \frac{\Delta E_{CC,M_3}}{\Delta E_{MBPT,M_3}}, ... \tag{6.5}$$

Next, the machine learning algorithm, f_R, is trained on this data set using the SRE formulation developed earlier in this chapter. The sequence length shown here is three data points, but depending on the analysis being done, it could range from one to three. We are adding this hyperparameter into the analysis, whose value will have to be set, but in general, we want a smaller sequence length if there is less training data:

$$f_R(\frac{\Delta E_{CC,k-3}}{\Delta E_{MBPT,k-3}}, \frac{\Delta E_{CC,k-2}}{\Delta E_{MBPT,k-2}}, \frac{\Delta E_{CC,k-1}}{\Delta E_{MBPT,k-1}}) = \frac{\Delta E_{CC,k}}{\Delta E_{MBPT,k}}. \tag{6.6}$$

The SRE algorithm is then used to extrapolate this data set to many points until the ratio of correlation energies has converged. This value can then be taken as the slope of the graph created with ΔE_{CC} plotted as a function of ΔE_{MBPT}:

$$\lim_{k\to\infty} \frac{\Delta E_{CC,k}}{\Delta E_{MBPT,k}} = slope = m. \tag{6.7}$$

Finally, $\Delta E_{MBPT,Large\ M}$ is generated, a process that takes less than one second (not including the time to generate the matrix elements). This energy, $\Delta E_{MBPT,Large\ M}$, is multiplied by the slope, m, to approximate $\Delta E_{CC,Large\ M}$.

$$m\Delta E_{MBPT,Large\ M} \approx \Delta E_{CC,Large\ M} \tag{6.8}$$

Due to convergence, the number of single-particle states used to calculate the large M data sets will vary depending on the system. It is important to note that the data sets used to train a machine learning algorithm in this work consist of 3-16 points each, making these data sets some of the smallest used in physics applications of machine learning. Many machine learning algorithms need more points to be accurately trained, even up to 1-2 orders of magnitude more, as with some neural networks. For example, the ML-based process shown in Reference [62] can generate accurate molecular CCD correlation energies using only 12 data points with kernel ridge regression and a training process based on MP2 amplitudes. However, the MP2 amplitudes used in the training

data have a very high dimensionality which can increase the time needed for training. Some other studies designed machine learning algorithms to use small data sets but still needed around 100 data points, if not more (see Reference [63] for example) and are usually artificially extended with interpolation (for example, see [51].

It is also of note that machine learning is typically only used to make extrapolations in exceptional cases such as recurrent neural networks. When asked to make predictions outside their training range, many machine learning algorithms could improve. However, it will be shown that SRE can make accurate extrapolations using only a small training set, making it a unique machine-learning algorithm. There is, however, a drawback to the SRE method. Since it relies only on the y component of the training data set, it assumes that the data is evenly spaced with respect to the x variable and can only make predictions at the same spacing. This works well for this application since "evenly spaced" means a calculation at every closed shell of unoccupied states, and extrapolations are made until the result converges (the exact x value where this occurs is not essential for this study). However, this limitation does mean that the SRE method is only suitable for some applications.

Error Analysis on Prediction

The prediction has a measurable uncertainty since Bayesian methods calculate the slope, m. This can be transferred to an uncertainty on the predicted converged CCD correlation energy using the following scheme. Given that the converged CCD correlation energy is:

$$\Delta E_{CC} = m\Delta E_{MBPT}, \tag{6.9}$$

then we can relate the uncertainties on all three quantities using equation Equation 6.10. In Equation 6.10, δx refers to the uncertainty associated with quantity x.

$$\frac{\delta\Delta E_{CC}}{\Delta E_{CC}} = \frac{\delta\Delta E_{MBPT}}{\Delta E_{MBPT}} + \frac{\delta m}{m} \tag{6.10}$$

We will also assume that the uncertainty in the calculation of the MBPT2 correlation energy is zero, so 6.10 simplifies to:

$$\frac{\delta\Delta E_{CC}}{\Delta E_{CC}} = \frac{\delta m}{m}. \tag{6.11}$$

Now, Equation 6.11 is solved for $\delta\Delta E_{CC}$, or the uncertainty on the predicted converged CCD correlation energy, yielding:

$$\delta\Delta E_{CC} = \frac{\Delta E_{CC}}{m}\delta m. \tag{6.12}$$

However, this can be simplified using the following:

$$\Delta E_{CC} = m\Delta E_{MBPT} \longrightarrow \Delta E_{MBPT} = \frac{\Delta E_{CC}}{m}, \tag{6.13}$$

yielding as a final equation for the CCD correlation energy uncertainty:

$$\delta\Delta E_{CC} = \Delta E_{MBPT}\delta m. \tag{6.14}$$

Finally, to present the results as the correlation energy per particle, as is typically done with infinite nuclear matter calculations, the uncertainty in the predicted converged CCD correlation energy can be found by dividing both sides of Equation 6.14 by N.

$$\delta(\frac{\Delta E_{CC}}{N} = \frac{\Delta E_{MBPT}}{N}\delta m \tag{6.15}$$

6.4 CCD Extrapolations With Respect to Number of Particles

Removing the finite size error from truncating the number of particles in the system will be similar to removing the basis incompleteness error. We will start with a data set that is created by dividing the converged CCD correlation energies by the number of particles in the system for small values of N, resulting in:

$$y = \frac{\Delta E_{CCD}^{N_k}}{N_k} = \frac{\Delta E_{CCD}^{N_1}}{N_1}, \frac{\Delta E_{CCD}^{N_2}}{N_2}, ... \tag{6.16}$$

Then we will train a machine learning algorithm using the sequential formatting developed in this last section on the data set we have created:

$$f_R(\frac{\Delta E_{CCD}^{N_{k-3}}}{N_{k-3}}, \frac{\Delta E_{CCD}^{N_{k-2}}}{N_{k-2}}, \frac{\Delta E_{CCD}^{N_{k-1}}}{N_{k-1}}) = \frac{\Delta E_{CCD}^{N_k}}{N_k}. \tag{6.17}$$

The final step is to use the trained machine learning algorithm to extrapolate this ratio until convergence, resulting in the CCD correlation energy in the thermodynamic limit. Thus, we have:

$$\lim_{k\to\infty} = \lim_{k\to\infty} = \Delta E_{CCD}^{\infty}, \tag{6.18}$$

where ΔE_{CCD}^{∞} is the CCD correlation energy per particle in the thermodynamic limit. Additionally, since the prediction the machine learning algorithm makes is the result we are looking for, the uncertainty of that prediction does not need to be modified.

6.5 CCDT Extrapolations With Respect to Number of Single Particle States

For removing the basis incompleteness errors from calculations using the approximative triples methods, we will use the same method developed for removing the basis incompleteness errors from CCD calculations. Even though the MPBT2 correlation energies are still at a comparatively lower level of approximation than the CCDT approximations, the SRE methods will still extrapolate well because both MBPT2 and CCDT converge as the value of M increases, and both MBPT2 and CCDT converge at roughly the same rate. So, fortunately, CCD and CCDT correlation energies can be predicted with the same methods.

However, we will not perform extrapolations to the thermodynamic limit for the approximative triples calculations because we will use system sizes that already reduce the finite size effects. This will be explained in greater detail in Chapter 8.

6.6 Conclusion

In this chapter and the previous one, we have developed the computational framework needed for this thesis. First, we explored several supervised machine learning algorithms, and then we used these algorithms to develop the sequential regression extrapolation (SRE) algorithm. Though we have developed the SRE algorithm to work with any supervised machine learning algorithm, in practice, we will restrict ourselves to only using three in the following two results chapters: ridge regression, Bayesian ridge regression, and Gaussian processes. Several reasons for not using neural networks or recurrent neural networks for this application have been given in this chapter and the previous one. Though kernel ridge regression does perform well as the machine learning algorithm in an SRE analysis, the large amount of hyperparameters KRR has makes it unattractive. Many-body calculations are very time-consuming; thus, avoiding the need to generate a validation data set is optimal. Furthermore, suppose we wish to develop a machine learning extrapolation algorithm that is both accurate and saves time. In that case, the time costs of generating a validation data

set and performing an extensive tuning process over many hyperparameters are unattractive. Thus neural networks, recurrent neural networks, and kernel ridge regression are eliminated as possible algorithms.

The following two chapters will apply the methods we have developed in this chapter to coupled cluster calculations of infinite nuclear matter calculations. Chapter 7 will focus on calculations of the homogeneous electron gas, and Chapter 8 will focus on calculations of infinite nuclear matter; pure neutron matter and symmetric nuclear matter will be analyzed.

CHAPTER 7

THE HOMOGENEOUS ELECTRON GAS RESULTS

The first infinite matter system we will analyze with the SRE method is the homogeneous electron gas (HEG), described in Chapter 4. The HEG, as one of the simpler infinite matter systems, is an excellent sandbox to fully develop the SRE method described in the last chapter.

Figure 7.1 shows the results of computing the CCD correlation energy per electron for a HEG with r_s = 0.5 at various numbers of electrons and with four different numbers of single-particle states. We can see that the calculation of ΔE_{CC} depends heavily on the number of single-particle states in the system, and we can begin to see that the correlation energy converges as the number of single-particle states in the system increases. The calculations performed with only 514 single-particle states (or 15 shells) are the least accurate of the four plots in Figure 7.1 since they are performed with the fewest single-particle states. On the other hand, the calculations performed with 6,142 single-particle states (or 70 shells in total) are the most accurate of the four plots, and in fact, the convergence of ΔE_{CC} has been confirmed at this point. However, it is not always feasible to perform the calculations at this high number of single-particle states due to the associated high computational time and resource requirements.

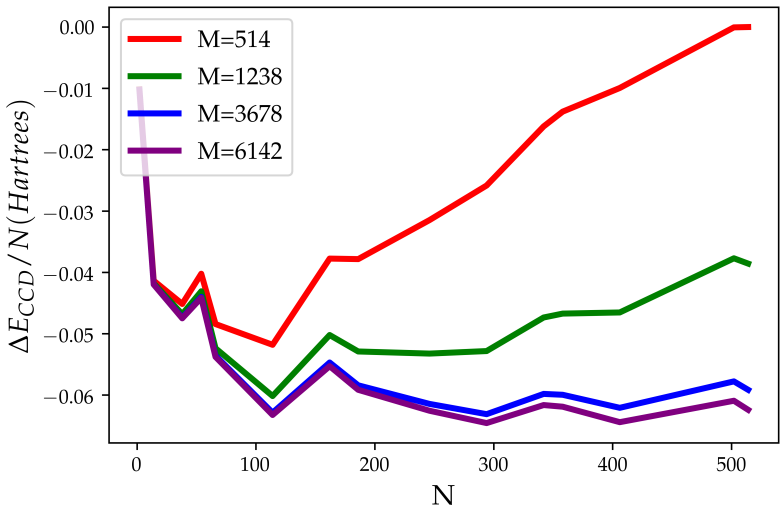


Figure 7.1 ΔE_{CC} per electron plotted against N *with calculations performed at four different values of M. As* M *increases, the error in the calculation decreases, but the run time drastically increases*

Figure 7.2 shows the total run time in node seconds for CCD calculations of the HEG with $r_s = 0.5$, $N = 66$, 294, or 514, and at various numbers of single-particle states. Node seconds/minutes/hours/-days will be used to report computational run times in this thesis so that run times generated with different numbers of MPI nodes can be compared. A *node second* (or minute, hour, or day) is defined as the run time of the coupled cluster program (in seconds/minutes/hours/days) times the number of MPI nodes used in the calculation. For the HEG, all calculations were performed using nodes from Michigan State University's high-performance computing center. Each node is an Intel Xeon processor with a reported clock speed of 2.4 GHz. Each coupled cluster program was run with four MPI nodes and 28 OpenMP threads. The specifications of the coupled cluster code used to perform the HEG and neutron matter calculations with the Minnesota potential shown in the section are described in Ref. [78].

In Figure 7.2, there is a consistent pattern in the data where more electrons and more single-particle states in a calculation require a longer run time. This is an obvious and expected result, and in fact, the CCD run times should scale polynomially with the number of particles and single particle states allowed in the calculation. For example, a CCD calculation of the HEG should scale as $O(N^4M^6)$, where typically M is much greater than N [38]. If we look at the results for $N = 66$ and assume that the converged CCD correlation energy occurs at M = 6,142 single-particle states (see Figure 7.1), the total run time for the CCD program is 653.88 node seconds (10.89 node minutes). However, generating the converged CCD correlation energy for N = 514 and M = 6,142 takes 42,791 node seconds (11.88 **node hours**). Thus while generating the converged CCD correlation energies at a smaller number of particles results in reasonable computational run times, performing the same calculation at a large number of electrons is not feasible in most studies.

Another factor we can look at is the amount of RAM needed to perform the CCD calculations at various numbers of electrons and single-particle states. The amount of RAM needed to perform the calculations should increase as both N and M increase as, naively, the Hamilton for the system should be $\binom{M}{N}$ on each side. However, modern advancements limit the number of matrix elements that must be computed and stored. Figure 7.3 shows the total amount of RAM (in gigabytes)

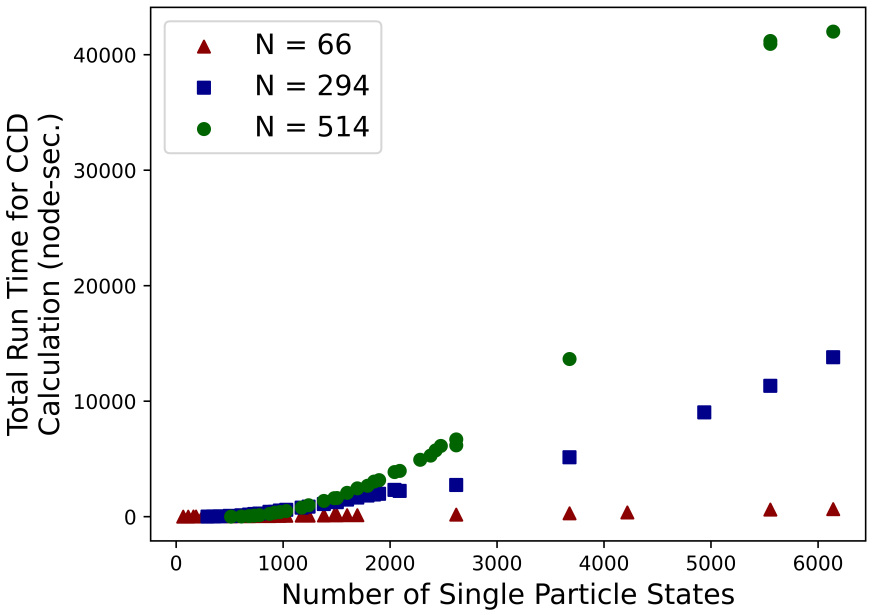


Figure 7.2 The total run time needed, in node seconds, to perform a CCD calculation on the HEG with $r_s = 0.5$ and at N = *66, 294, and 514. The CCD calculations are performed at many values of M, as shown on the x-axis. A node second is defined as the total run time of the program times the number of nodes used to run the program. Four nodes from Michigan State University's high-performance supercomputer were utilized in this case. These nodes have Intel Xeon processes with a clock speed of 2.4 GHz*

needed to perform CCD calculations for the HEG at $r_s = 0.5$. There are three different numbers of electrons represented in the figure: $N = 66$ (red triangles), $N = 294$ (blue squares), and $N = 514$ (green circles). These numbers of electrons span the range of interest for this work. The number of single-particle states varied from less than 1,000 to over 6,000. We see a consistent pattern in the data where the amount of RA*M* needed for the calculation increases as the number of electrons and single-particle states changes. The converged CCD correlation energy for 514 electrons (the highest amount of electrons in this thesis) and at 6,142 single particle states takes a startling 283.79 GB of RA*M* to run. The converged CCD correlation energy at 66 electrons and 6,142 single particle states still takes 106.49 GB of RAM, a very moderate number of particles to have in a calculation. Therefore, not only is computing the converged CCD correlation energies expensive in the amount of computational time required, but the converged correlation energies also take a considerable amount of computational resources, requiring a supercomputer for these calculations to be feasible. Thus, though increasing the number of single-particle states in a CCD calculation of the HEG

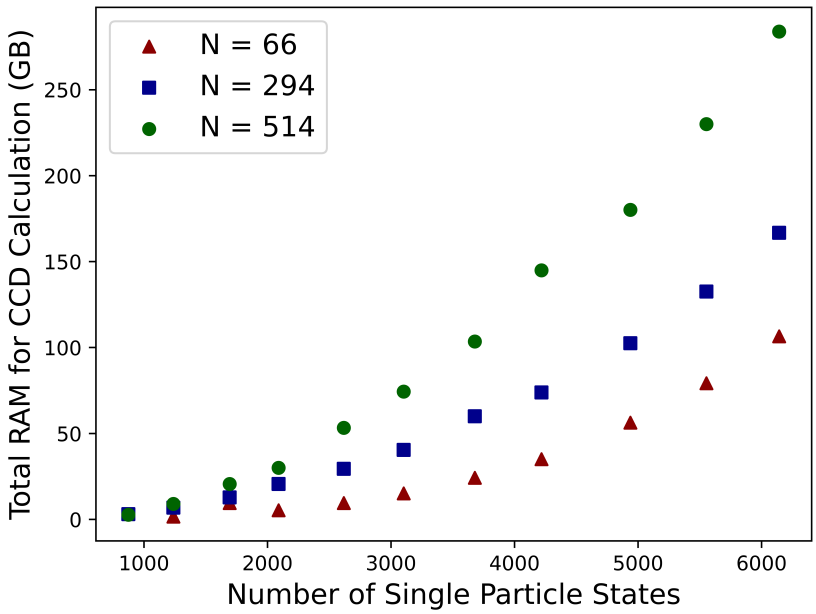


Figure 7.3 The total amount of RAM in gigabytes needed to perform CCD calculations on the HEG with $r_s = 0.5$ and with 66, 294, or 514 electrons. Calculations were performed at various numbers of single-particle states (x-axis)

increases the accuracy of the CCD correlation energy, it comes with high computational time and resource considerations that can make these calculations infeasible. Therefore, we are motivated to develop a method by which the CCD correlation energies at a high number of single-particle states can be recreated using only calculations performed at a lower number of single-particle states, where the computational resource and time requirements are significantly more reasonable.

In Figure 7.4, ΔE_{CCD} and ΔE_{MBPT2} is plotted as a function of the number of single-particle states in the calculation for a HEG system with r_s = 0.5 and 342 electrons. The CCD results are plotted in red, and the MBPT2 results are in blue. The calculations up to 2,090 single-particle states are plotted with the solid lines, and then the converged correlation energies are plotted with the dashed lines. It is much easier in this figure to see that the correlation energies converge as the number of single-particle states increases and that even at around 2,000 single-particle states, the correlation energies are not close to their converged values. However, the ranges of single-particle states shown in this figure with the solid lines are the calculations we wish to use to predict the converged CCD correlation energy, as at these ranges, the computational time and resource requirements are reasonable.

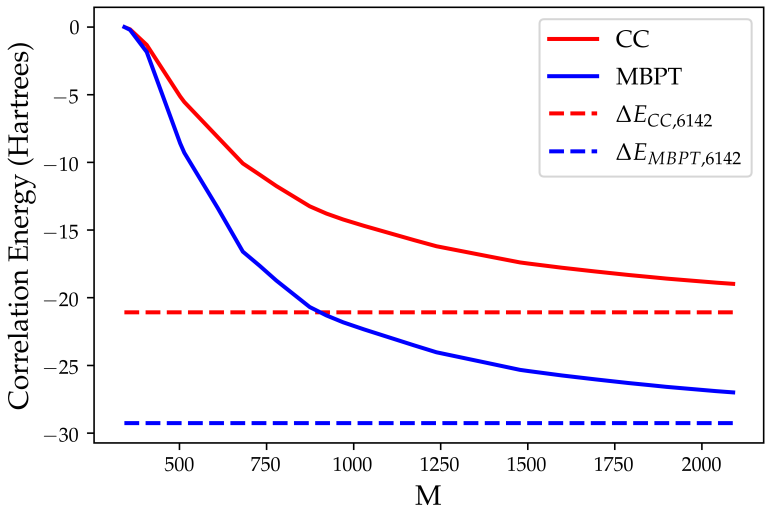


Figure 7.4 A plot of ΔE_{CC} and ΔE_{MBPT} versus M *for a HEG system with* N=*342 and r_s=0.5. The converged correlation energies calculated at M=6,142 are shown with dashed lines*

There are a few important things to note in Figure 7.4. First, there is a significant difference between the converged correlation energies, meaning that even though it is easier to generate ΔE_{MBPT} compared to ΔE_{CC}, ΔE_{MBPT} is not a good approximation to the true correlation energy. Second, though it takes 21,735.10 node seconds (6.03 node hours) to calculate $\Delta E^{342}_{CC,6142}$ it only takes 5.12 node seconds to generate $\Delta E^{342}_{MBPT,6142}$. Thus generating the converged MBPT2 correlation energies is significantly faster than generating the converged CCD correlation energies; the MBPT2 calculations are faster by four orders of magnitude. Finally, note that though the MBPT2 and CCD calculations converge to different values, they converge at roughly the same rate and display similar patterns in their convergence. This similarity in convergence was the inspiration for the SRE method.

Next, we will take Figure 7.4 and reformat it such that ΔE_{CC} is plotted as a function of ΔE_{MBPT} (Figure 7.5). Each point is at a constant value of M in this format, and M increases from the right to the left. In Figure 7.5, note that the graph is relatively linear, and in fact, it becomes more linear from right to left. Put another way, the graph of ΔE_{CC} versus ΔE_{MBPT} becomes more linear as M increases because as M increases, both correlation energies converge. We can check this using Figure 7.6

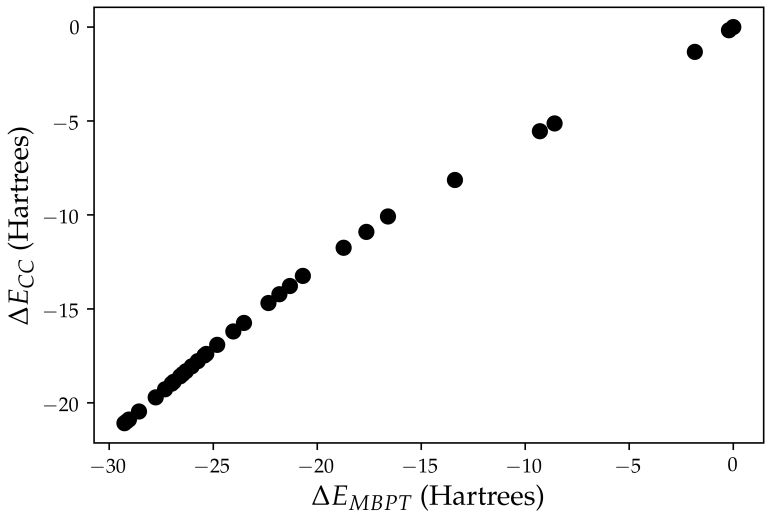


Figure 7.5 Figure 7.4 is transformed such that ΔE_{CC} are plotted as a function of $t\Delta E_{MBPT}$ (such that each point has the same M). The resulting plot has a linear relationship that becomes stronger to the left (as M *increases)*

Figure 7.6 shows the ratio of $\Delta E_{CC}/\Delta E_{MBPT}$ as a function of the number of single-particle states in the calculations for three different numbers of electrons for calculations of the HEG at $r_s = 0.5$. The ratio $\Delta E_{CC}/\Delta E_{MBPT}$ is the slope of the line in Figure 7.5. While the slopes are rapidly changing at lower values of M, by the end of the graph, all of the slopes are almost constant, though the slopes at higher numbers of electrons converge slower than the slopes at lower numbers of electrons. However, this graph does verify that the slopes will become constant, and the SRE method should work if we can accurately predict this slope.

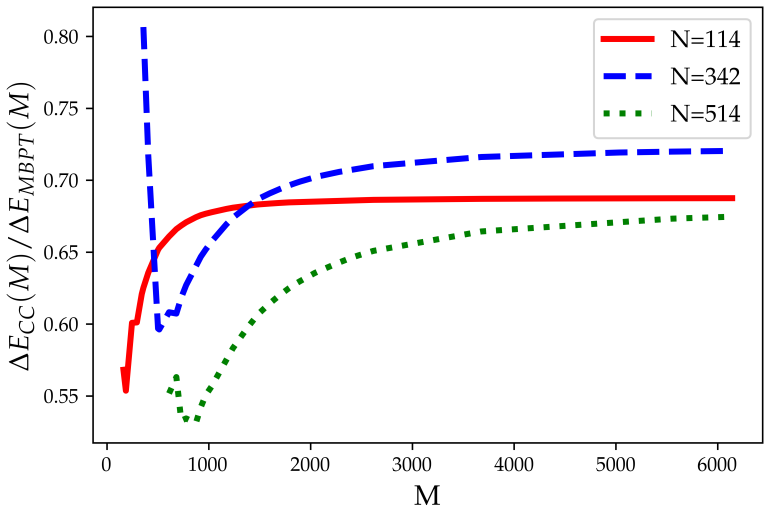


Figure 7.6 A plot of the ratio of ΔE_{CC} to ΔE_{MBPT} as a function of M *for N=114, 342, and 514 and r_s=0.5. As* M *increases, the ratio (or the slope of the line in Figure 7.5) becomes increasingly constant. However, it is important to note that the run time increases drastically as* M *increases*

Traditional Extrapolation Methods

Before we begin to develop in full the SRE method for determining the converged CCD correlation energies with respect to the number of single-particle states in the calculation, let us look at two traditional methods for removing the basis incompleteness error and two methods that may work based on the analysis from the last section.

The most common way to deal with a basis incompleteness error is to truncate the number of single-particle states at some high number. Figure 7.7 displays the results of truncating the calculations at 20 open shells, corresponding to between 922 and 2,090 single particle states depending on the number of electrons in the calculation for a HEG with r_s = 0.5. This graph, plotted in orange, is compared to the correlation energy calculation performed at 70 total energy shells (M = 6,142) in black, which are the fully converged results where the basis incompleteness error is eliminated. Of course, one way to negate the basis incompleteness error is to perform calculations at an higher numbers of single-particle states. However, as we saw in the last section, this is not necessarily feasible due to the high computational time and resource requirements, and, therefore, simply increasing the number of single-particle states in the system is not an attractive option for removing the basis incompleteness error.

Another common method is to find the correlation energy in the complete basis limit by using a power law of the form:

$$\Delta E_{CCD}(M) = E_{CCD,\infty} + AM^{-\alpha}, \tag{7.1}$$

where $E_{CCD,\infty}$ (the correlation energy in the complete basis limit) and A are fit to data from correlation energy calculations at several different values of M. The value of α is typically fixed to 1, so this method is generally referred to as a $1/M$ power law [38]. The results of performing this power law fit on a HEG system with $r_s = 0.5$ and using 16 data points per number of electrons (from 5 open shells to 20 open shells) are shown in Figure 7.7 in blue. We can see that this power law fit overestimates the HEG correlation energies by about the same margin that the truncation at 20 open shells underestimated them. This indicates that for the data available, neither of these traditional methods of removing the basis incompleteness error are attractive option. The power law does not provide a good fit and overestimates the correlation energies; while truncating the basis at a large number of single-particle states can provide an accurate result, the computational time and resource requirements make this option not feasible for many systems.

Next, we can look at two methods inspired by the analysis performed in the previous section. In Figure 7.5, the graph is reasonably linear if we plot the CCD correlation energies as a function of the MBPT2 correlation energies. The first approach to calculating the converged CCD correlation energies from this plot may be to take the CCD and MBPT2 calculation at a relatively high number of single-particle states, create the ratio $m = \Delta E_{CCD}/\Delta E_{MBPT2}$, and then take m and multiply it by the MBPT2 correlation energy at a very high number of single-particle states to approximate the CCD correlation energy at that same high number of single-particle states (here M = 6,142): $\Delta E_{CCD,6142} \approx m\Delta E_{MBPT2,6142}$. For this investigation, we derived the slope from calculations performed at 20 open energy shells (the same truncation level we used earlier), then calculated the slope and generated a prediction for the CCD correlation energy at 6,142 single-particle states. The results of this analysis are shown in green in Figure 8.11. We can see that this is a much better approximation of the converged CCD correlation energies than the previous two attempts. However, there are still significant deviations at higher numbers of electrons. Therefore, while this

is an improvement, we will seek a better method to accurately predict the converged correlation energies at all numbers of electrons, including the high values.

As a final method, we will use linear regression to fit a subset of the data shown in Figure 7.5. It should be noted that even though linear regression is quite a simple machine learning algorithm, this is our first machine learning approach to remove the basis incompleteness error from these calculations.

For training the linear regression algorithm, we will use 16 sets of MBPT2 and CCD correlation energies drawn from calculations between 5 and 20 open shells (the exact number of single-particle states depends on the number of electrons). First, the linear regression algorithm will be trained and then asked to predict $\Delta E_{CCD,6142}$ by giving $\Delta E_{MBPT2,6142}$. It is important to remember that generating MBPT2 correlation energies is a fast computation.

The results of performing this analysis are shown in red in Figure 7.7. These results are similar to the previous slope analysis, where the correlation energies at lower numbers of electrons are well matched, but there are deviations at higher numbers. We could improve this result by using training data drawn from higher M values. However, like the other methods shown here, that is an unattractive solution due to the increased computational time and resource requirements.

Thus the two traditional extrapolation methods we have looked at and the two new methods fail to satisfactorily remove the basis incompleteness errors without requiring data drawn from calculations with a much higher number of single-particle states. Thus, we need to be in a position to begin fully developing the SRE method.

Ridge Regression

Now that we have looked at performing the basis completeness extrapolations with traditional methods, we can develop a machine learning-based approach. A good machine learning algorithm will have a small number of hyperparameters to tune and give reproducible and accurate results for the extrapolation. Ridge regression only has one hyperparameter and a closed-form solution for its optimized weights. Therefore, it will always train the same way when given the same training data, making it a much more attractive alternative than neural networks, with their many hyperparameters

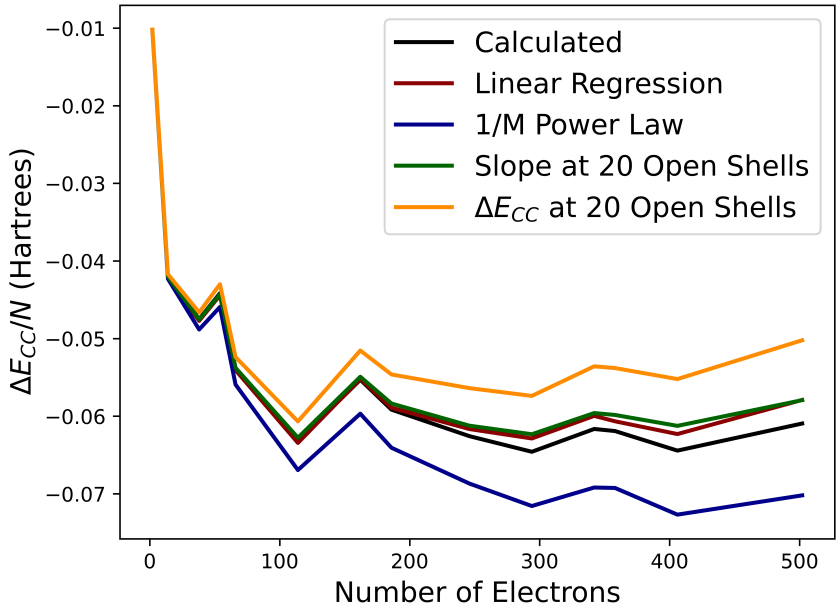


Figure 7.7 The converged CCD correlation energies (black) calculated at M = *6,142 for an HEG with* $r_s = 0.5$. *Also displayed are the results of predicting the converged correlation energies with a truncation at 20 open shells (yellow) which has a percent error of 6.87%, using a 1/*M *power law (blue) which has an average percent error of 7.00%, using the CCD to MBPT2 slope calculated at 20 open shells (green) which have an average percent error of 1.16%, and using linear regression to model the graph from Figure 7.5 (red) which has an average percent error of 0.95%*

and unrepeatable results.

The ridge regression algorithm contains one hyperparameter, α, which is the strength of the regularization. Since this is a hyperparameter, the user must set its value before training the ridge regression algorithm. However, its value can significantly impact the results of the ridge regression. For example, figure 7.8 shows the results of using the SRE algorithm with ridge regression and various α values to predict the converged ΔE_{CC} energies of the HEG with $r_s = 0.5$.

For Figure 7.8, the training data was chosen to be between 5 and 20 open shells (inclusive) or $114 \leq M \leq 2{,}090$ (exact values of M depend on N). The sequence length for the SRE algorithm was set to 3, and the data set was extrapolated until there were 50 total points and the convergence of the slope was confirmed. The value of ΔE_{CC} we get from this process is considered converged. Therefore we will compare it to converged calculations of the HEG at $M = 6{,}142$, where the correlation energy convergence has converged. In Figure 7.8 $\Delta E_{CC,6142}$ are plotted with scattered square points. These are considered to be the "true" or most accurate results. The solid lines show the results

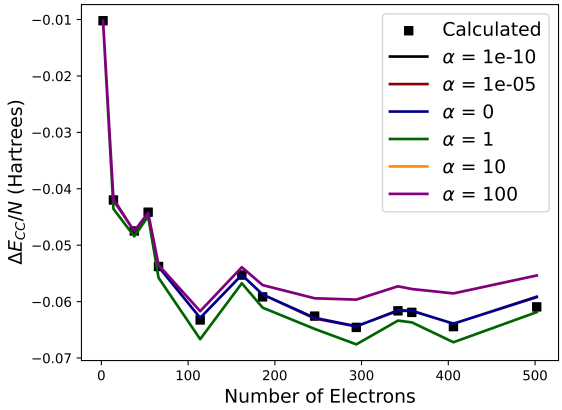


Figure 7.8 This figure shows the extrapolations' dependence on the hyperparameter α. The results of performing an SRE analysis on the HEG with $r_s = 0.5$ and at various numbers of electrons. The machine learning algorithm used for the SRE algorithm is ridge regression

of performing the SRE algorithm to converge the correlation energy with respect to the number of single-particle states using the SRE algorithm with a variety of values for the hyperparameter α. Note that several of the values of α produce essentially equivalent results, so there are not six distinct lines. For $\alpha = 10^{-10}$ and 10^{-5} (which provide nearly identical results), the results are very close to the fully calculated converged correlation energies, with the average percent error overall numbers of electrons of 0.49%. However, on the other hand, $\alpha = 1$ and $\alpha = 100$ provide results that visually appear to be suitable matches for the expected pattern in the data but do not match the converged calculations. However, if the fully converged calculations were not available for comparison, these usually could be considered reasonable predictions. For $\alpha = 1$, the average percent error overall numbers of electrons is 3.09%, and for $\alpha = 100$, the average percent error is 3.87%. Not shown in Figure 7.8 are the predictions for $\alpha = 0.01$ because the average percent error for the SRE algorithm's predictions and the actual data for this α value is 3.45×10^6%. While this produces an undeniably incorrect result even if the actual data was not available for comparison, this shows that wide variations in predictions can happen as α varies, and α values both larger and smaller than 0.01 provide reasonable predictions, but $\alpha = 0.01$ provides predictions that do not converge.

A hyperparameter is typically set through a process known as hyperparameter tuning. In hyperparameter tuning, many different values are tested for the hyperparameters to find the best ones. First,

the algorithm is defined using a given set of hyperparameters and trained using the training data; then, its accuracy is tested using a validation data set. The validation data set can be a subset of the training data, the test data, or a new data set altogether. This define-train-test cycle is repeated with many combinations of hyperparameters, and the hyperparameters that produce the lowest error on the validation data set are kept for future use.

For the HEG, we have data at five different values of r_s, so it was decided to choose one value of r_s for hyperparameter tuning. Therefore, for the chosen value of r_s, not only does the SRE training data need to be generated, but the converged correlations energies at M = 6,142 also need to be generated, which will be a time-consuming process. When performing the hyperparameter tuning process on α, 1,000 different values were tested, evenly spaced on a log scale between 10^{-10} and 100 (inclusive). For each value, a ridge regression algorithm was defined with that value for α. The ridge regression algorithm was used for an SRE analysis using the same training process described above. The predicted values for ΔE_{CC} were then compared to the fully converged calculations. The value for α that produced the lowest average percent error overall numbers of electrons was kept. This process was performed independently with all five different r_s data sets to see how the results compare.

When the hyperparameter tuning is performed using the data sets $0.1 \leq r_s \leq 0.75$, the optimal alpha value found was 10^{-10}. However, the hyperparameter tuning is performed on the data set at r_s = 0.05, and the optimal alpha value was found to be $4.27\text{x}10^{-4}$. When $\alpha = 10^{-10}$ the average percent error for all r_s values was 0.45%, which is quite good and is a very acceptable error on the actual data for a prediction made with unconverted data. However, when $\alpha = 4.27\text{x}10^{-3}$ the average percent error for all r_s values is $7.5\text{x}10^{11}$%, because some of the predictions do not converge. When analyzing the results, this visually is a wrong prediction, but as seen in Figure 7.8, values of α could be chosen that provide seemingly good but incorrect answers. Moving forward with the ridge regression analysis, we will use $\alpha = 10^{-10}$.

Figure 7.9 shows the results of performing an SRE analysis on the HEG data set using a ridge regression algorithm with $\alpha = 10^{-10}$. This value was chosen because it produces the smallest

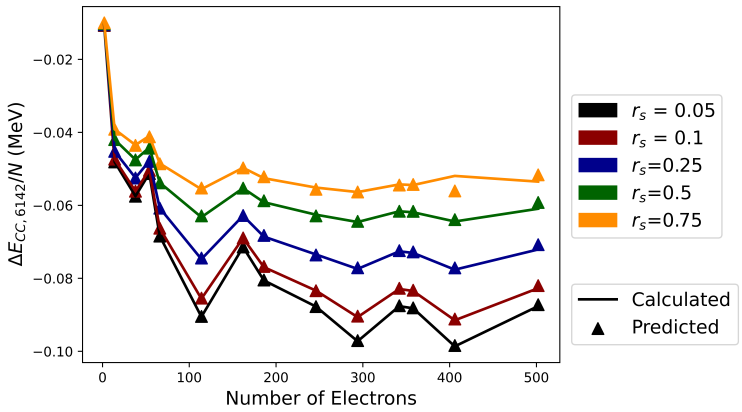


Figure 7.9 The results of performing an SRE analysis on the HEG to predict the converged CCD correlation energies for 14 different numbers of electrons and five different values of r_s. The SRE predictions, made using a ridge regression algorithm with $\alpha = 10^{-10}$, are shown with the triangular markers and the solid lines representing the full correlation energy calculations at M = *6,142. The average percent error between the SRE predictions and the full calculations is 0.45%*

validation error when performing hyperparameter tuning with the r_s = 0.5 data set. The SRE training procedure was the same as described earlier in this section. The average percent error across all data points in Figure 7.9 is 0.45%, which is tiny. This error is an acceptable level of accuracy of the machine learning predictions, especially considering the potential computational time and resource savings.

The time needed to generate the training data for the hyperparameter tuning process (i.e., the correlation energies from 5 to 20 open shells for r_s = 0.5) was 27.52 node hours. However, the time needed to generate the validation data set (i.e., the correlation energies at M = 6,142 for r_s = 0.5) was 42.6 node hours, meaning that the total time needed to generate the data needed for the hyperparameter tuning was 70.12 node hours so there is considerable time to find the optimal value of α. If we consider performing the SRE analysis on all five HEG data sets, the time needed to generate all training data is 153.28 node hours. Compared to the time needed to generate all $\Delta E_{CC,6142}$ at 219.44 nide hours, this leads to a time savings for the SRE method of 66.16 node hours, or over half a day of computational time saved using the SRE method over doing the complete calculations with no loss of relevant accuracy as the average percent error is less than 0.5%. However, when we add the time needed to generate the validation data set to tune α, the time savings drops to only 23.56 node hours, which is much less attractive savings.

Though the SRE algorithm with ridge regression can accurately recreate the converged ΔE_{CC} for the electron gas at various high densities, the results also display a significant dependence on the hyperparameter α value. Because of this dependence, if the fully calculated converged ΔE_{CC} are not also available, as they are in this work; for comparison, it could be easy to choose a value of α which created a set of converged ΔE_{CC} which visually look feasible but are not accurate (see Figure 7.8 for examples). Furthermore, the high computational time and resource requirements needed to generate the validation data set for the hyperparameter tuning process are unattractive. Therefore, we wish to develop a method that can provide the accuracy displayed in Figure 7.9 without dependence and reliance on hyperparameters. Thus, we will start to investigate the Bayesian machine learning implementation of ridge regression: Bayesian ridge regression.

Bayesian Ridge Regression

Bayesian ridge regression, the Bayesian machine learning implementation of ridge regression, should be a better choice for the machine learning algorithm for this SRE analysis. Since it is a Bayesian algorithm, it uses Bayesian statistics to find a good value for the hyperparameter, α. This means that we do not need to perform hyperparameter tuning or generate a validation data set. Therefore, using Bayesian ridge regression, the only CCD calculations that need to be performed are those for the training data, meaning this method should result in significant time savings between the SRE results and the computing fully converged energies. Additionally, since this is a Bayesian algorithm, it will give us an error in its predictions. Hence, we can get an uncertainty on our converged CCD correlation energy prediction.

This section attempts to predict the CCD correlation energy at M = 6,142 single-particle states (the converged correlation energy) using only the CCD and MBPT2 correlation energies from 5-20 opens shells and the MBPT2 correlation energy at M = 6,142. The exact values of M used in calculating the training data depend on the number of particles in the system but will range from 66 to 2,090 single-particle states. At each of these calculations, the number of single-particle states is not enough to converge the CCD correlation energy, thus, all of the training data suffers from basis incompleteness errors. Here we will be looking at calculations performed at five different values

of r_s, so these systems are of a high-density electron gas.

Then we used the implementation provided by the Python library Scikit-Learn for Bayesian ridge regression. The SRE sequence length was two, meaning the Bayesian ridge regression algorithm took two inputs (the previous two ratios of CCD correlation energy to MBPT2 correlation energy). Once the algorithm was trained using the 16 training points, the SRE algorithm was used to extrapolate the ratio until it was converged (resulting in a data set of 50 total points). The final ratio was extracted and multiplied by $\Delta E_{MBPT2,6142}$ to approximate $\Delta E_{CCD,6142}$. The analysis results on 14 numbers of electrons and five different r_s values are shown in Figure 7.10.

Figure 7.10 plots the converged CCD correlation energies calculated at M = 6,142 with solid lines. Calculations were performed at 14 different numbers of electrons and five values of r_s. Each color in the figure represents a different value of r_s. The SRE predictions for each correlation energy are shown with the triangular markers, and the shaded regions represent the uncertainty in the predictions, which comes from the Bayesian ridge regression algorithm. The average percent error for the entire graph is 0.47%. It should be noted that this is essentially the same result we achieve from the ridge regression analysis with the best value of α.

Furthermore, the time needed to generate the SRE training data for all predictions was 200.44 node hours, while the total time needed to generate all energies at M = 6,142 is 289.43 node hours. This leads to a time savings of 88.99 node hours or over 3.5 days of computational time saved with no loss of relevant accuracy. So not only can the SRE method with Bayesian ridge regression accurately predict the converged CCD correlation energies for the HEG across a variety of values of N and r_s, but it can also produce uncertainties on its predictions and results in considerable time savings over performing the complete calculations.

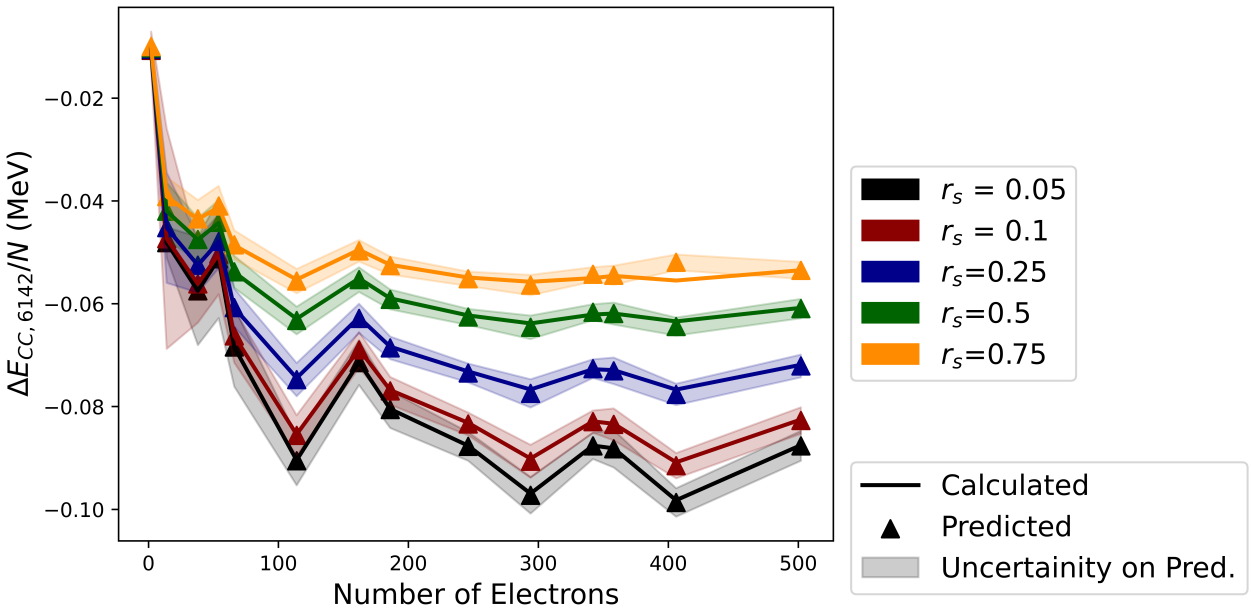


Figure 7.10 The results from performing an SRE extrapolation in terms of M *for various numbers of electrons and values of* r_s*. Results are plotted against* $\Delta E_{CC,6142}$

Finally, we can compare the performance of the SRE algorithm to the traditional extrapolation methods we analyzed in Figure 7.7. For the HEG at $r_s = 0.5$, if we add the SRE predictions to the others, we get Figure 7.11. We can see that the SRE predictions are much closer than any other methods and provide a reasonable approximation to the converged correlation energies, even at high numbers of electrons.

In this section, we have developed the first Bayesian implementation of the SRE method. It has the accuracy of the ridge regression implementation developed in the previous section but without the ambiguity of choosing the algorithm's hyperparameters. Additionally, Bayesian ridge regression provides uncertainties in its predictions, which are helpful in a scientific context. Overall, the SRE method with Bayesian ridge regression is the most promising of the methods attempted so far to remove the basis incompleteness errors.

Gaussian Process with Rational Quadratic Kernel

As a final study for the HEG, let us use the Gaussian process (GP) algorithm, another Bayesian machine learning algorithm, as the machine learning component in an SRE analysis. Here we will be using Gaussian processes as the machine learning component of the SRE algorithm. The kernel

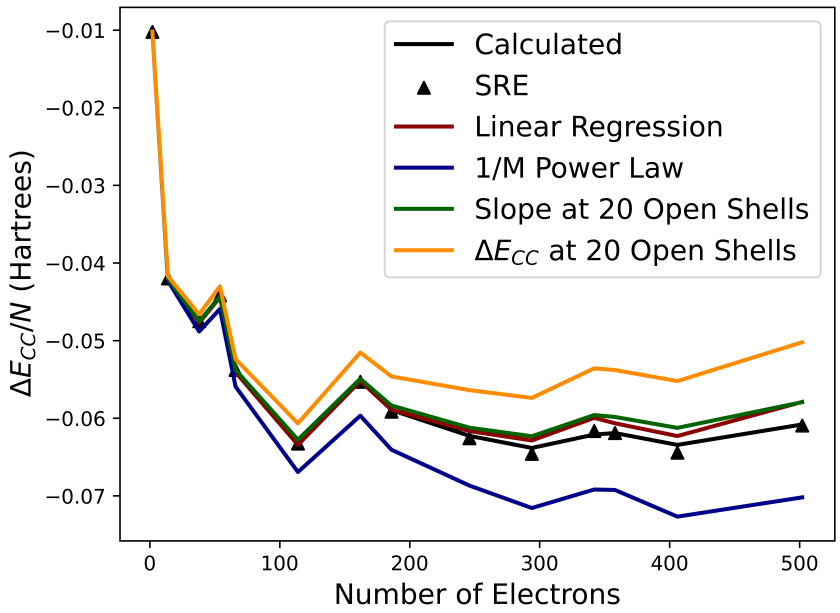


Figure 7.11 Here, with have recreated Figure 7.7, but have added the SRE results, which have an average percent error of 0.31%

function will be a modified rational quadratic kernel consisting of a constant kernel multiplied by a rational quadratic kernel and then added to a white kernel. All kernels and the GP algorithm are implemented using the Python library Scikit-Learn.

As a first study, we will use the same training data set as before, 16 training points taken from between five and 20 open shells. We will use an SRE sequence length of three for this training. The only other parameter we set is the α value of the GP algorithm, which we set to δy_{train}^4, where δy_{train} is the standard deviation y component of the training data set. Using the standard deviation when setting α seems to keep the resulting uncertainties small but has little effect on the prediction results.

Figure 7.12 shows the analysis results on the same data set used in the Bayesian ridge regression section. The average percent error between the SRE predictions for the converged correlation energies and the fully calculated correlation energies at M = 6,142 is 0.7three%. While this is larger than the error resulting from the Bayesian ridge regression analysis, it is still quite small and acceptable. Furthermore, it is essential to note that the uncertainties on this, represented by the shaded regions, are significantly smaller than those on the Bayesian ridge regression plot. They are

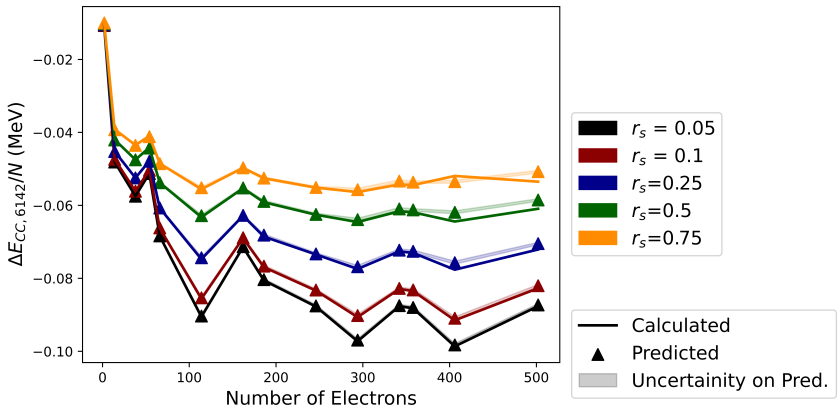


Figure 7.12 The results of performing an SRE analysis with Gaussian processes to predict the converged correlation energies for the HEG at different values of N *and* r_s. *The correlation energies calculated at* M = *6,142 are plotted with a solid line and are taken to be the true results, the SRE predictions are plotted with triangular markers, and the shaded region represents the uncertainty on the SRE predictions*

so small in places hidden entirely by the lines!

As a final analysis with the SRE method on the HEG, we will again use GP as the machine learning algorithm but will reduce the number of training points from 16 to only ten. Here we will use data collected from five to 14 open shells, thus removing the six training points with the highest number of single-particle states (and thus the highest computational costs). Here we have also reduced the SRE sequence length to 1 as there are a few training points.

The results of performing this analysis are shown in Figure 7.13, using the same plotting scheme as 7.12. The lower number of training points does lead to a higher average percent error of 1.16%, but this is still a relatively low error, and it does appear, based on the results graph, that most of that error is concentrated in the last two points of the r_s = 0.75 data set.

Next, we must point out the time savings we have gained by reducing the number of training points. When using 16 training points, the time savings is 88.99 node hours, which is not an insignificant amount of time saved. However, with only ten training points, and since the six training points we have removed have the most significant computational time, with this analysis, the amount of computational time saved is 224.43 node hours. That is equivalent to over a week in computational time saved for an error of only 1.16%!

Finally, we justify using the Gaussian processes as the machine learning algorithm over Bayesian

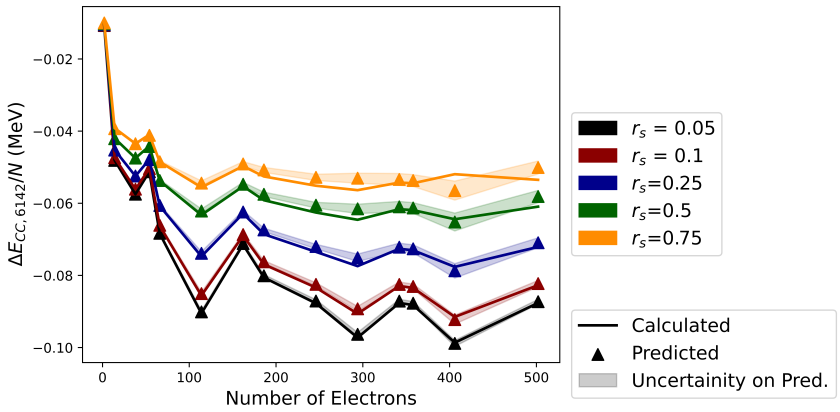


Figure 7.13 The results of performing an SRE analysis with Gaussian processes to predict the converged correlation energies for the HEG at different values of N *and* r_s *and only ten training points. The correlation energies calculated at* M = *6,142 are plotted with a solid line and are taken to be the true results, the SRE predictions are plotted with triangular markers, and the shaded region represents the uncertainty on the SRE predictions*

ridge regression when there is a small amount of training data. For example, suppose we were to perform this same analysis with Bayesian ridge regression, using ten training points and a sequence length of 1. In that case, the average percent error across all the correlation energies is 6.05%, much higher than the 1.16% we see with GP. Thus, the GP algorithm works better with smaller data sets and will be the primary algorithm we will use in the next chapter to analyze the infinite nuclear matter, as no training set in that chapter will contain more than ten points.

7.1 The Thermodynamic Limit

As a final look at the HEG, we can attempt to calculate the correlation energy per electron in the thermodynamic limit using the formalism developed in the previous chapter. We have used 14 training points, from $N = 2$ to $N = 502$, and an SRE sequence length of three. Again, the machine learning algorithm used to make these predictions is the Bayesian Ridge regression algorithm. We attempted this extrapolation using the correlation energies, which have been fully calculated at M = 6,142, and the correlation energies, which have been predicted by the SRE method and are shown in Figure 7.10. However, these two extrapolations result in identical correlation energies, with a percent error between the two extrapolated data sets being 0.0%. Thus, there is no reason to use the fully calculated data to predict the thermodynamic limit (TDL) correlation energies when the

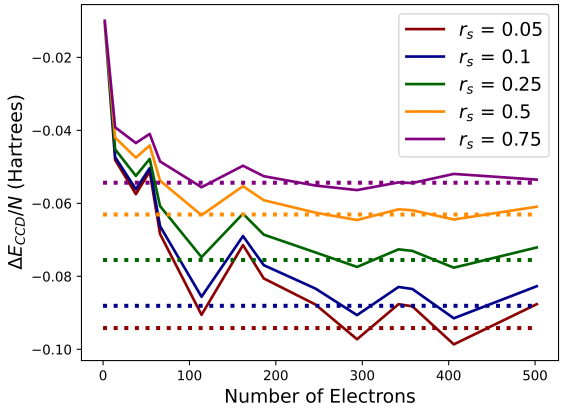


Figure 7.14 The converged CCD correlation energies for the HEG calculated at M = 6,142 (solid) and the TDL predictions for the correlation energy per particle calculated through the SRE method (dashed)

SRE predictions result in the same answer and yield significant time savings.

Figure 7.14 shows the correlation energies per particle for the HEG at a variety of values for N and r_s with solid lines. The horizontal dashed lines show the correlation energy at the TDL as predicted by the SRE algorithm. Even though the correlation energies show significant oscillations with respect to the number of electrons in the system (likely due to the periodic boundary conditions), the SRE predictions seem to provide a suitable match for where the oscillations will converge.

While there do exist literature values for the HEG in the TDL (for example, see the work by Bishop et al. in [39] and [40]), the results are highly dependent on the many-body method used, the values of r_s studies, and the specifications of the HEG system. Thus we could not find any currently published literature results to compare our predictions for the exact specifications we have reported. However, we can use a power law, a standard method to find the TDL energies (see Ref. [148]). Therefore, we will use a power law of the form:

$$\Delta E_{CCD}(N) = \Delta E^{\infty}_{CCD} + AN^{-\alpha}, \tag{7.2}$$

where ΔE^{∞}_{CCD}, A, and α are parameters that are fit using data generated at finite numbers of electrons and ΔE^{∞}_{CCD} the is correlation energy at the thermodynamic limit. Usually, α is fixed to one, which is used when calculating the correlation energy for the system at the TDL. We, however, are interested in calculating the correlation energy in the TDL per electron and thus will leave α

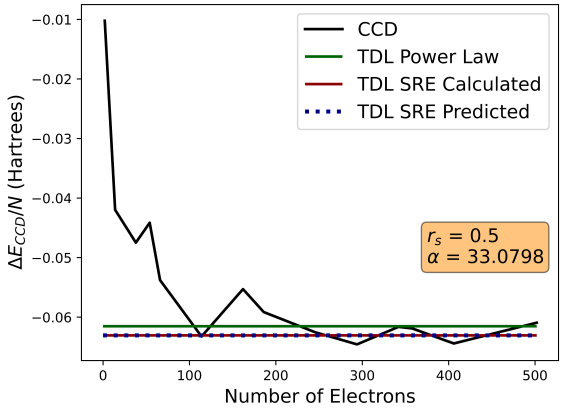


Figure 7.15 The CCD correlation energy per electron for an HEG with $r_s = 0.5$ and calculated at $M = 6{,}142$ (black). The power law prediction for the TDL correlation energy per electron is shown in green, and the SRE predictions for the TDL correlation energy per electron extrapolated from calculated and predicted energies are shown in red and blue, respectively

unfixed. The results for performing this power law fit are shown in Figure 7.15 for a HEG with r_s = 0.5 and are also plotted with the SRE predictions for the correlation energy at the TDL from complete calculations and the SRE predictions. Also shown are the complete calculations for the finite numbers of electrons shown at M = 6,142. While it is impossible to say which of these results is the most accurate, it does appear that the SRE predictions visually provide a better matching for the convergence of the correlation energies in the TDL. However, the best method of predicting the correlation energies in the TDL and the error in the predictions is still a topic of ongoing research and will require long running calculations at the TDL to ensure accuracy.

7.1.1 Homogeneous Electron Gas Conclusion

In this chapter, we have looked at several methods for removing the basis incompleteness error from CCD calculations of the HEG without performing calculations at a high number of single-particle states. Using the traditional methods of truncating the basis at some number of shells or using a $1/M$ power law fails to give reasonable predictions for the correlations energies in the complete basis limit, which increases the basis size of the complete calculations that need to be performed. Additionally, the two methods we looked at, which are based on the relationship of the convergence of the MBPT2 and CCD correlation energies, gave better predictions for the correlation energies in the complete basis limit but still failed to accurately reproduce the correlation energies in the

complete basis limit for a high number of electrons.

As we began to develop the SRE method, we had a method that could accurately recreate the correlation energies in the complete basis limit for various numbers of electrons and values of r_s with only minimal training data. However, when we developed the SRE method with ridge regression, we got an average percent error of 0.45% across 70 complete basis limit predictions if the best value of α was chosen. On the other hand, if the best value of α was not used, then the average percent error was infinity. This highlights the drawback of traditional machine learning methods: the abundance of hyperparameters and the need to perform hyperparameter tuning. Even though ridge regression only has one hyperparameter, α, finding its best value still requires hyperparameter tuning, which means we must create a validation data set already in the complete basis limit, which takes a long time to generate and reduces the time savings of using the SRE method to predict the correlation energies in the complete basis limit.

Thus we were motivated to use Bayesian machine learning, as these algorithms find the optimized values of their hyperparameters, eliminating the need for hyperparameter tuning and a validation data set. The first Bayesian algorithm we tested was Bayesian ridge regression, which produced all 70 correlation energies in the complete basis limit with an average error of only 0.47%, almost identical to the average ridge regression error. Additionally, since Bayesian ridge regression is a Bayesian algorithm, it produces uncertainties in its predictions, so we have error bars on our results. Since we only needed 16 training points to train the algorithm fully and no validation set, the total time saved using SRE to predict the total basis limit correlation energies instead of fully calculating them was over 3.5 node days, with an error of only 0.47%. Next, we attempted to reduce the amount of training data needed, and in order to accomplish this, we had to use a new Bayesian machine learning algorithm, Gaussian processes (GP). We could predict the correlation energies in the complete basis limit using only ten training points with the GP algorithm. This resulted in an average percent error of 1.16% across all 70 predictions, leading to over one week of computational time savings.

Finally, we looked at an initial analysis using the SRE method to predict the CCD correlation energy

in the thermodynamic limit. However, this work is still in progress because, when this thesis was written, it was impossible to know the TDL energies for the specific system we are modeling. Thus we cannot calculate the error in our predictions.

To conclude, the SRE method with Bayesian machine learning algorithms can accurately produce the correlations energies of the HEG in the complete basis limit while saving many hours of computational time and producing uncertainties in its results. As the electron gas is a test bed for developing methods that will eventually be applied to more complicated infinite matter systems, the analysis performed in this chapter motivates using the SRE method to predict correlation energies in the complete basis limit for other infinite matter systems.

CHAPTER 8

INFINITE NUCLEAR MATTER RESULTS

We have explored and developed the SRE method by applying it in various ways to the homogeneous electron gas. The electron gas, with its shorter computational times and more straightforward interactions, makes it an excellent test bed to develop methods that can be applied to other infinite matter systems. This chapter studies one of these more complicated systems: infinite nuclear matter. Therefore, we will be using SRE methods perfected on the electron gas. For this chapter, all SRE algorithms use Gaussian processes with a modified rational quadratic kernel that was developed in the last section of the previous chapter. We will start our studies in this chapter with pure neutron matter with both toy and realistic nuclear interactions and then move on to symmetric nuclear matter with realistic interactions before concluding with a brief study of the symmetry energy of infinite nuclear matter.

8.1 Pure Neutron Matter

8.1.1 Coupled Cluster Doubles and the Minnesota Potential

As a toy nuclear interaction, the Minnesota potential will provide us with approximations to the energies for a pure neutron matter system, but without a high degree of accuracy. However, the Minnesota potential is less computationally taxing than the more realistic chiral potentials and only includes two-body forces; the Minnesota potential, like the HEG, is a valuable test bed for further developing our SRE method before we apply it to the more realistic nuclear matter calculations with the chiral potentials.

Gaussian Processes

The training data for each combination of density and the number of neutrons was taken to be the CCD and MBPT2 correlation energies between five open energy shells and 14 open energy shells (inclusive). This data was formatted using the process described in Chapter 6. The machine learning algorithm used to analyze these data sets was Gaussian Process (GP). For each data set, the α value of the GP algorithm was set to be the standard deviation of the training data set. The kernel

for the GP algorithm was chosen to be a constant kernel multiplied by a rational quadratic kernel, added to a white kernel (all kernels are Scikit-Learn's implementation). The hyperparameters of the kernels were set by the GP algorithm during the training process. Once each GP was trained, it was asked to predict 100 data points, leading to a converged ratio of correlation energies. The final point in this data set was taken to be m and was multiplied by $\Delta E_{MBPT,6142}$ to give an approximation for $\Delta E_{CC,6142}$. Additionally, the GP algorithms produced their uncertainties on all of the plotted predictions and the results in Figure 8.1.

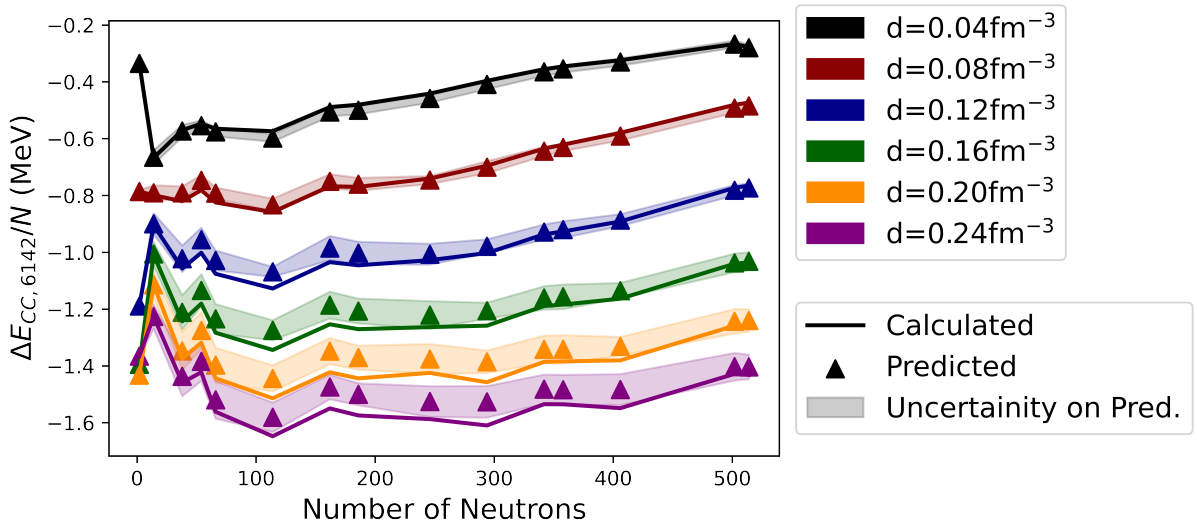


Figure 8.1 The CCD correlation energy per neutron as a function of the number of neutrons in the system for pure neutron matter with the Minnesota potential. All calculations are performed at 6,142 single particle states and with periodic boundary conditions. The full calculations, shown with the solid lines, were performed on MSU's ICER supercomputer. The SRE predictions, shown with the triangular data points, were predicted using only ten training data points from 5-14 open shells. The shaded region shows the Gaussian process uncertainty in predicting the converged correlation energy per neutron. The overall RMSE error between the full calculated and the predicted CCD correlation energies per neutron is 0.037 MeV

When looking at these results, it is also essential to consider the errors associated with other approximation schemes that do not involve machine learning—truncating the number of single-particle states in the system to some set number of energy shells. For example, if we limit the calculation to 14 open energy shells (15 to 30 shells), the error from the CCD correlation energies at 70 shells (M= 6,142) is 0.56 MeV, much larger than the GP result of 0.037 MeV. Furthermore, we can include more energy shells before truncating. For example, if 40 total shells are included

in the calculation, then the RMSE error from the results at 70 shells is 0.26 MeV; for truncating at 50 total shells, it is 0.15 MeV, and for truncating at 65 total shells, it is 0.045 MeV. Therefore, even when exerting significant computational time and resources to raise the truncation level of the calculations, the GP prediction made with very little training data collected at a low number of energy shells is still the most accurate.

Accuracy is not the only metric we must consider; we must also consider the computational time and resources needed to generate the training data and perform the ML analysis versus just performing the complete calculation. The timing data presented in this paragraph was collected on Michigan State University's HPCC supercomputer on Intel Xeon processors with 2.40 GHz and 240 GB of RAM. The coupled cluster code was memory-optimized using the diagonal block structure developed in Ref. [78] and was parallelized using four compute nodes and 28 threads per node. Figure 8.1 contains 90 points, each of which has to be individually calculated or predicted with the machine learning algorithm. Since each machine learning process has 10 points, 900 training points are needed to produce Figure 8.1. However, their run times are comparatively short since they are taken from 5-14 open energy shells. Generating all 900 training data points takes 136 node hours. It takes about 20 seconds to perform all 90 ML analyses, thus adding a negligible amount of time to the total ML analysis run time, which will be dominated by the time to generate the training data. To generate all 90 data points in Figure 8.1 at 70 energy shells each, the total run time is 484 node hours, meaning that by using the ML extrapolation method instead, 348 node hours of computational run time is saved. That is equivalent to over 14 node days of computational time saved and a resulting error of only 0.037 MeV!

It is possible to achieve better accuracy with this prediction method by changing the alpha hyperparameter of the Gaussian process algorithm. Instead of using the standard deviation of the training data as the α parameter (as used to generate Figure 8.1), instead we will use the fourth root of the standard deviation of the training data ($\alpha = \delta y_{train}^{1/4}$). However, while doing this increases the accuracy (the RMSE drops from 0.037 MeV to 0.025 MeV), the uncertainty drastically increases. The results for performing this extrapolation are shown in Figure 8.2, where the uncertainties are

shown in (a) but are removed in (b) for clarity.

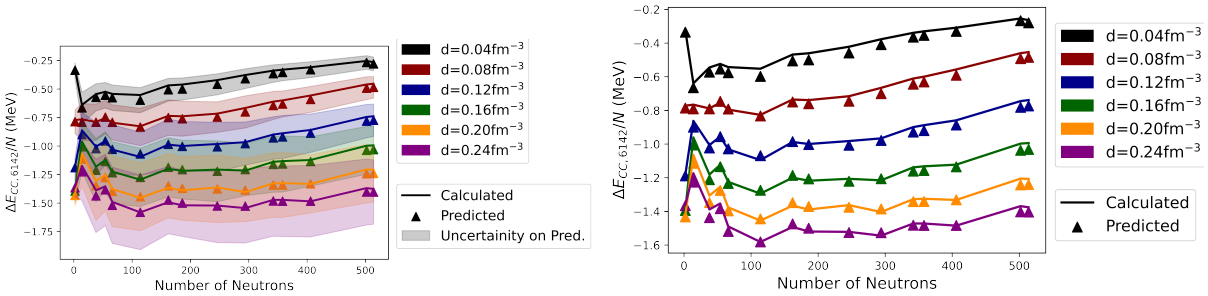


Figure 8.2 An SRE analysis of pure neutron matter with the Minnesota potential using a Gaussian process machine learning algorithm with $\alpha = \delta y_{train}^{1/4}$. This increases the accuracy of the preductions but also increases the uncertainity on the predictions. The results are shown with uncertainities (left) and without (right)

8.1.2 Coupled Cluster Doubles and Realistic Potentials

In this section, we will perform CCD correlation energy calculations on a pure neutron matter system, using chiral potentials derived from effective field theory (EFT) to model the nuclear interaction. These potential more realistically models the nuclear interaction when compared to the Minnesota potential, so these results are expected to be more accurate than the Minnesota potential results from the previous section. Specifically, we use next-to-next-to-leading order (NNLO) chiral potentials in this section, which were described in detail in Chapter 4.

We will limit our calculations to a 66 neutron system for the chiral potential. This is done for two reasons. Ref. [35] and [34] show that, especially for pure neutron matter, calculations for a system of 66 neutrons are nearly indistinguishable from calculations performed at the infinite matter limit. So for 66 neutrons, the finite size error in a pure neutron matter calculation is minimized. Therefore, we do not need to perform calculations at higher values of N to perform an $N \to \infty$ extrapolation. Secondly, the CCD correlation energy calculations for pure neutron matter with chiral potential are very time intensive, much more so than pure neutron matter calculations with the Minnesota potential. For the Minnesota potential, the average time to generate converged CCD correlation energies across all densities in Figure 8.1 for 66 neutrons is 23.48 node minutes. These calculations were all performed at 70 total energy shells or 6,142 single-particle states. However, when we perform the same calculations with the realistic chiral potentials from EFT, the average run time

across all densities shown in Figure 8.4 is 27.84 node hours per calculation. Moreover, these calculations converge at a much lower number of single-particle states, so these calculations were performed at 1,478 single-particle states or 27 total energy shells. Note that these calculations were performed with different codes, written in different languages, and performed with different high-performance computing set-ups. Hence, the run times are not directly comparable. However, it is sufficient to say that performing these calculations with realistic chiral potentials is much more computationally challenging than using the Minnesota potential.

As a further motivation for applying the SRE method to this system, let us examine the computational time and resources needed to perform calculations on a pure neutron matter system with 66 neutrons, a density of 0.16 fm^{-3}, and chiral NNLO potentials. Figure 8.3 shows the computational time in node seconds needed to perform the CCD correlation energy calculations as the number of single-particle states increases.

Figure 8.4 shows CCD calculations and ML predictions for pure neutron matter at 66 neutrons at two different density ranges. Figure 8.4a shows a density range around nuclear density (which nuclear density being around 0.16 fm^{-3}), and Figure 8.4b shows a low-density range. Densities around nuclear density are essential for studies of nuclei. The outer layers of a neutron star are thought to have densities around nuclear density ([77] [34] [2]), while low-density nuclear matter is also of interest [57]. The ML specifications for the plots in Figure 8.4 are as follows. For 8.4a, each ML prediction was made using only three training data points, MBPT2 and CCD correlation energy calculations between 114 and 186 single-particle states (or between one and three open shells). We were able to use such a small number of training points here because the correlation energies converge rather quickly. The SRE sequence length was chosen to be one here since there are only three training points. The converged CCD calculations in the figure were calculated at M=1,478 single particle states or 27 total energy shells (22 open energy shells). This number of single-particle states was chosen as the converged values because it is similar to the number of single-particle states used to calculate converged values in similar studies (Refs. [35], [34], and [11]). The same Gaussian Process algorithm and kernel as the Minnesota potential analysis was

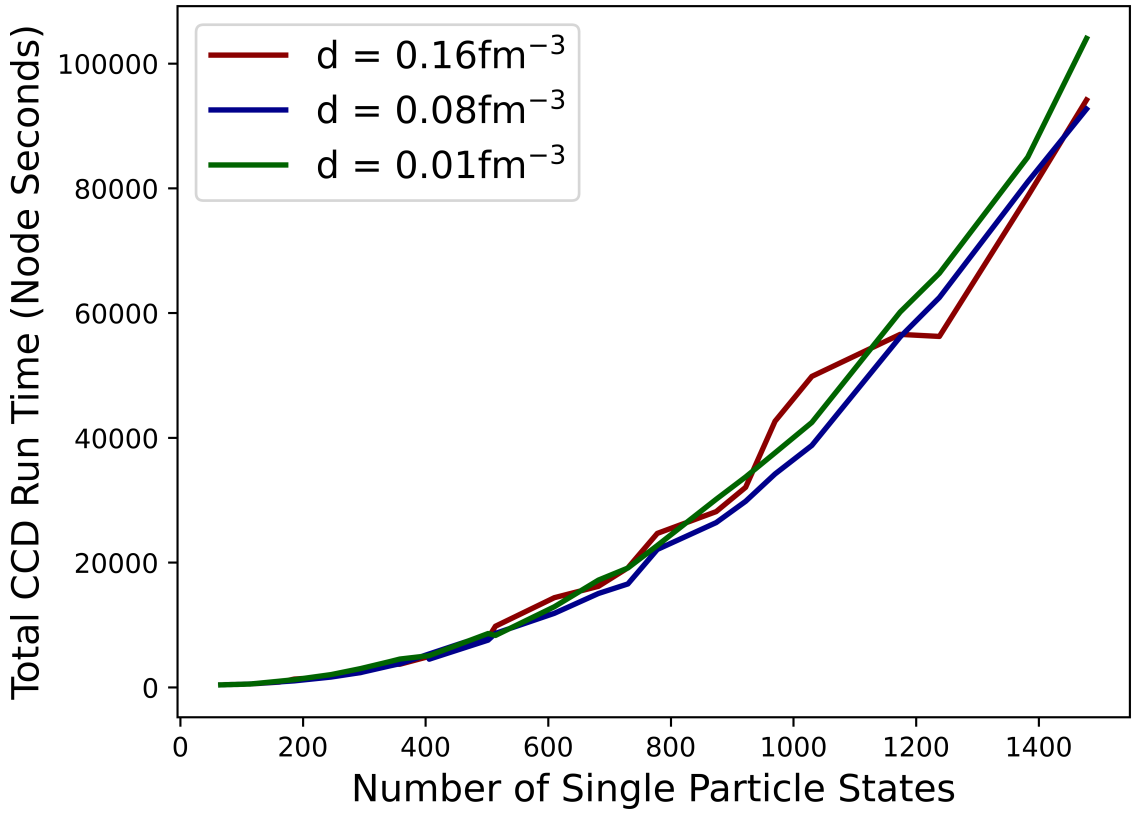
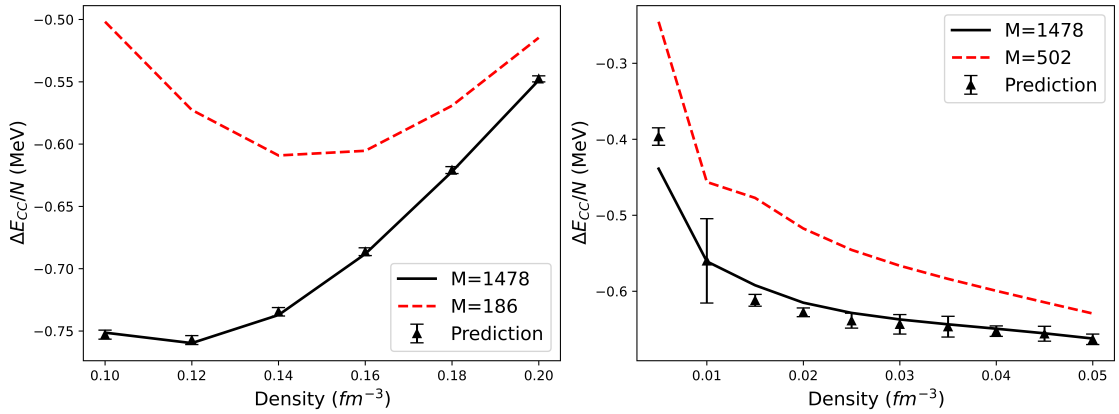


Figure 8.3 The run times, in node seconds, for coupled cluster doubles calculations of pure neutron matter as a function of the number of single-particle states in the calculation. Calculations performed at three different densities are shown

also used here, but the variance of the algorithm was chosen to be the standard deviation of the training data to the fourth power ($\alpha = \delta y_{train}^4$). For Figure 8.4b, the GP specifications are the same, but for the training data, seven training points were used from 186 to 502 single-particle states (three to nine open energy shells). This change was made because the correlation energy converges slower at lower densities. The SRE sequence length was also increased to three since more training data is available. The CCD correlation energy at M= 1,478 was still used at the fully converged value.

For Figure 8.4a, the average percent error between the fully converged CCD correlation energies and the ML extrapolated CCD correlation energies is 0.25%, and for Figure 8.4b it is 1.90%. The slightly higher average percent error for the second figure is easily explained. The convergence of the correlation energy is slower at lower densities. Especially at d = 0.005 fm^{-3} and d = 0.01



(a) CCD calculations of PNM around nuclear density

(b) CCD calculations of PNM at low densities

*Figure 8.4 The results from predicting the converged $\Delta E_{CC}/N$ for pure neutron matter using the coupled cluster doubles approximation. Figure (a) shows pure neutron matter for a range of densities around nuclear density ($0.10 \leq d \leq 0.20$), and figure (b) shows the results for a low-density pure neutron matter ($d \leq 0.05$). For both graphs, the solid black line plots the fully converged $\Delta E_{CCD}/N$, calculated at 1,478 single-particle states; the triangular markers represent the SRE predictions at each density of interest. The dashed red lines represent the point used to train the GP algorithms with the highest number of single-particle states (*M= *186 for Figure (a) and* M= *502 for Figure (b)) to show that the training set has not reached convergence*

fm^{-3}, the correlation energies had not sufficiently converged at 1,478 single-particle states. This explains why there is a discrepancy between the ML prediction and the complete calculation at d = 0.005fm^{-3}. Additionally, the ML prediction at d = 0.01 fm^{-3} closely matches the complete calculations, but there are pretty large error bars on the prediction when compared to the other predictions. This could also be explained by the slow convergence of the correlation energy at this point.

The timing data for Figure 8.4 was collected using Michigan State University's high-performance computing center using Intel Xeon compute nodes running at 2.4 GHz and with 240GB of RAM each. For these calculations, 12 MPI nodes were used per calculation. For example, for Figure 8.4a, generating the six fully converged CCD correlation energies at M = 1,478 single particle states took 156 node hours. However, generating the training data for the six GP algorithms took only 3.84 node hours, leading to a time savings of using the ML extrapolations of 152 node hours. Additionally, for Figure 8.4b, it takes 289 node hours to generate all ten fully converged CCD correlation energies and M= 1,478 single-particle states. However, it only takes 69.6 node hours

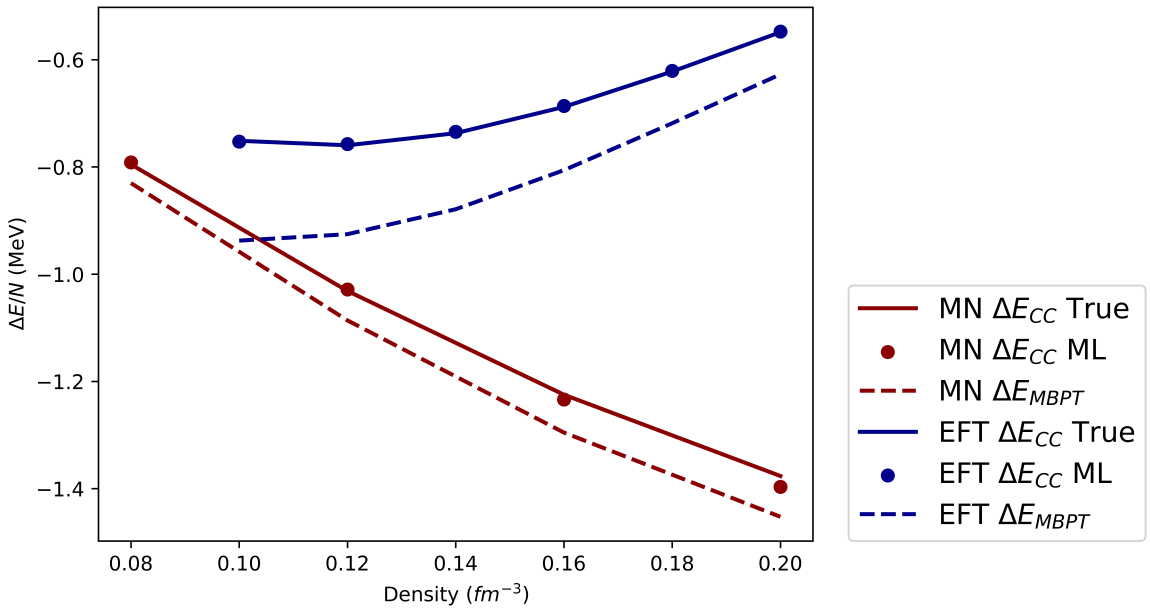


Figure 8.5 A comparison of correlation energy calculations for pure neutron matter near nuclear density for 66 neutrons. The red data sets were calculated with the Minnesota potential, and the blue data sets with the chiral potentials. The dashed lines represent the converged correlation energy per particle from MBPT2, the solid lines represent the converged CCD correlation energies per particle, and the circular makers represent the ML predictions for the converged CCD correlation energies per particle

to generate the ten GP training data sets. Therefore using the ML extrapolation over doing the complete calculations saves 219 node hours. That is a total computational time savings of over 15 node days for Figure 8.4

When considering both the accuracy and the time savings, the SRE method applied to CCD calculations of pure neutron matter with chiral potentials appears successful. The CCD correlation energies per particle can be accurately recreated for all but the smallest densities with very little training data and significant time savings.

Figure 8.5 compares pure neutron matter converged correlation energies per particle calculated with MBPT2 and CCD and for the Minnesota and chiral potentials. The Minnesota potential results are from calculations performed at 6,142 single-particle states of 70 total energy shells, and the chiral potential results are from 1,478 single-particle states or 27 energy shells. However, both the MBPT2 and CCD correlations energies have sufficiently converged. The ML predictions for the converged CCD correlation energy per particle are also shown. The results shown are for

calculations involving 66 neutrons and near nuclear density. There are a few critical observations we can make from this graph. First, we know that the Minnesota and chiral potentials will produce different results as they model the nuclear interaction differently and to different levels of accuracy. The results differ significantly, with the chiral potentials yielding lower energy for both MBPT2 and CCD calculations. Additionally, the difference in the results increases as density increases, indicating that there will be quite a significant difference between the two potentials at densities greater than nuclear density. Secondly, we can look at the difference between the MBPT2 and CCD results for the two potentials. For both potentials, the MBPT2 correlation energy per particle is lower than the CCD correlation energy per particle. However, for the Minnesota potential, the MBPT2 correlation energies are only slightly below the CCD correlation energies, and the gap between the MBPT2 correlation energies and the CCD correlation energies is roughly consistent over the entire density range shown. There is a more significant discrepancy between the MBPT2 and CCD correlation energies for the chiral potential, and this gap grows as the density decreases. Finally, it is essential to note that for each potential, the ML predictions to the converged CCD correlations energies are in good agreement with the total calculations, so the SRE method is potential independent.

8.1.3 Coupled Cluster Triples Approximations and Realistic Potentials

Beyond the CCD approximation, we can also look at the CCDT-1 coupled-cluster approximation, adding some aspects of the $\hat{T}_3$ excitation operator into the calculations without the full-time needed for a complete CCDT calculation. In the next section, we will also consider the CCDT-1 and CCD(T) approximation, another coupled-cluster approximation that adds some aspects of the $\hat{T}_3$ operator into the calculations using a different algorithm.

First, we should verify that the CCDT-1 approximation with the same NNLO chiral potentials gives a different (and more accurate) approximation than the CCD approximation for pure neutron matter. In Figure 8.6, the CCD correlation energies per particle are plotted as a function of the density of the pure neutron matter system for 66 neutrons and at densities around nuclear density with the solid red line. The CCD calculations shown were performed at 1,478 single-particle states, where

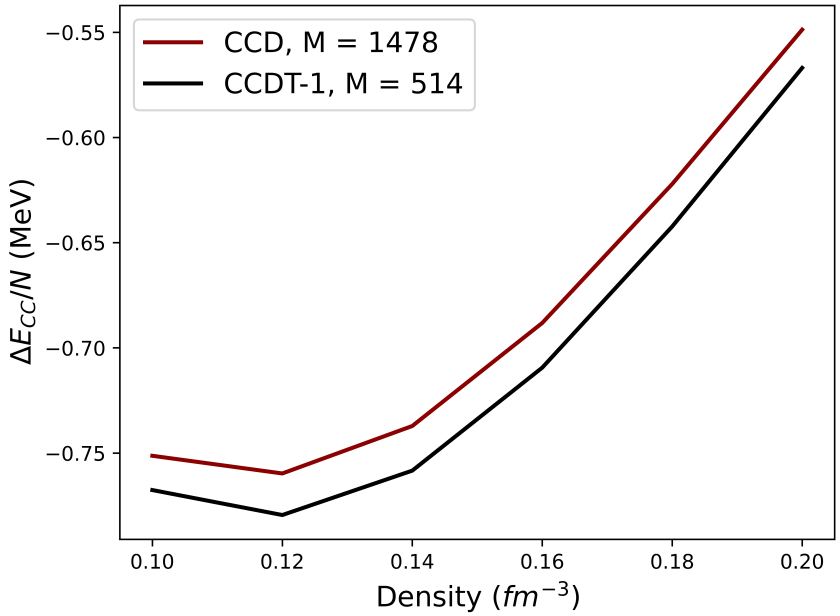


Figure 8.6 A comparison between the CCD and CCDT-1 correlation energies for a pure neutron matter system (N = 66 neutrons) with NNLO chiral potentials and for densities around nuclear density. The CCDT-1 approximation consistently gives a lower correlation energy than CCD correlation energy at all densities in the graph

the calculations converged. For the CCDT-1 calculations, shown with the solid black line, the calculations were also performed with the NNLO chiral potentials for pure neutron matter and 66 neutrons and at densities around nuclear density. As a result, the correlation energy calculations using CCDT-1 converge much faster than for CCD and only need 514 single-particle states to reach convergence. Additionally, since CCDT-1 calculations scale as $O(M^7)$ compared to a CCD calculation at $O(M^6)$, performing CCDT-1 calculation at higher values of $M$is computationally prohibitive. As shown in Fig 8.6, the CCD and CCDT-1 approximations give significantly different predictions for the correlation energies at each density value shown.

We can also compare the run times needed to complete a CCDT-1 calculation over a CCD calculation. Figure 8.7 shows the total run time needed to complete a CCDT-1 calculation on Michigan State University's High-Performance Computing Center using Intel Xeon nodes with a clock speed of 2.4 GHz and 24 MPI nodes per run. The results in this figure were collected by performing CCDT-1 calculations on a pure neutron matter system with chiral NNLO potentials and with 66 neutrons and a density of 0.16 fm^{-3}. The results are reported in node hours. It is obvious that the

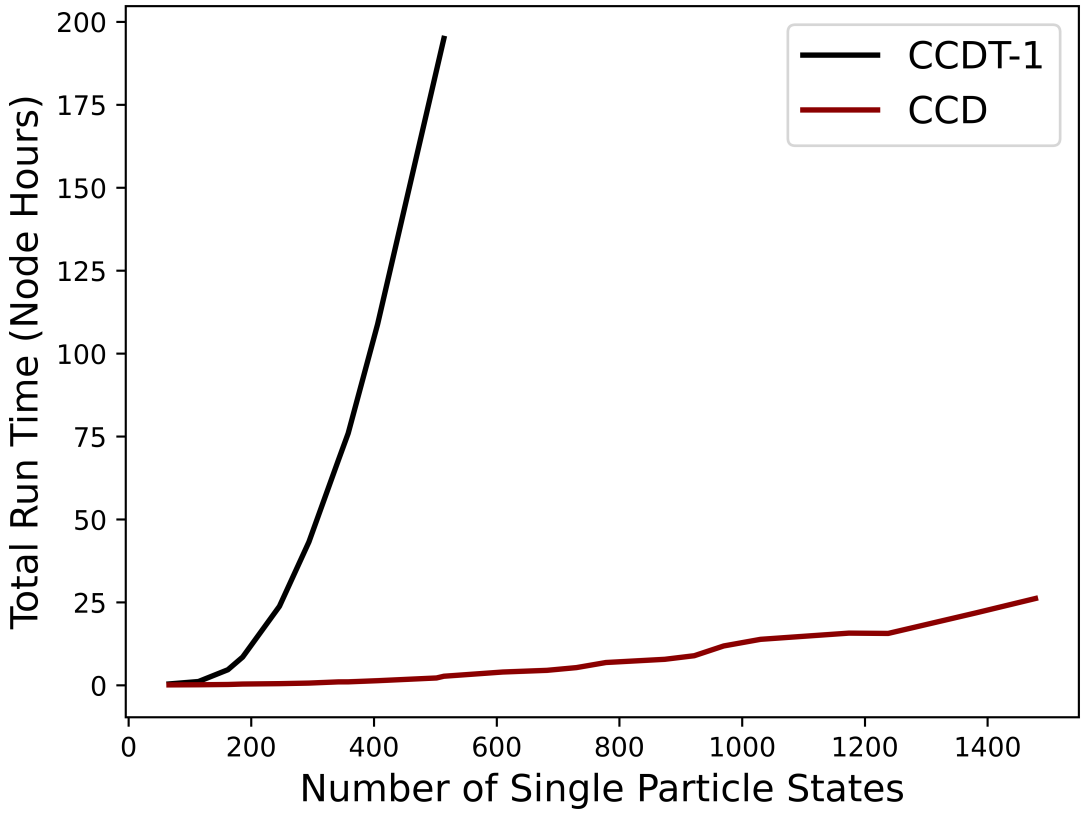


Figure 8.7 The run times needed to complete CCD and CCDT-1 calculations for a PNM system reported in node hours

run time increases very quickly with the number of single-particle states in the calculation. This rapid increase in run time is even more apparent if we plot the CCD and CCDT-1 run times on the same plot, as in Figure 8.7. Though the CCDT-1 calculation converges at a smaller number of single-particle states, the total run time needed is drastically higher than that needed for the CCD calculations. At 514 single-particle states, the CCDT-1 calculation takes over 15 times longer than the CCD calculation at the same number of single-particle states. So even though we were motivated to use the SRE method to save time on the CCD calculations with the chiral NNLO potentials due to their run times, there is even more of a need for an accurate extrapolation method for the CCDT-1 calculations as their run times are an order of magnitude larger.

Next, we will look at the feasibility of predicting the CCDT-1 correlation energy from the MBPT2 correlation energy using the same SRE and machine learning setup as the above doubles predictions. We will use the SRE method described in Chapter 6 to predict the CCDT-1 correlation energies

from the MBPT2 correlation energies using three training points (from one to three open shells or 114 to 186 single particle states) with a sequence length of one and a Gaussian process algorithm with its alpha value set to the standard deviation of the training data to the fourth power.

Figure 8.8 shows the results of predicting the converged CCDT-1 correlation energies using the method described above for a pure neutron matter system with 66 neutrons and d = 0.16 fm^{-3}. For the plot, the complete calculations are shown with solid lines, the training data with points, and the SRE predicted converged correlation energy with the horizontal dashed line. The percent error between the converged ΔE_{CCDT1} that has been fully calculated at 514 single particle states and the SRE prediction is 0.51%. As a comparison, the percent error between the converged ΔE_{CCDT1} that has been fully calculated at 514 single particle states and the largest point in the training data (at 186 single particle states) is 12.19%. This means that in the time needed to generate the training data, which is 14.29 node hours for the three points, the accuracy of the correlation energy can be improved from 12.19% to just 0.51%. Furthermore, the time needed to calculate the fully converged correlation energy at 514 single-particle states is 194.97 node hours, meaning that just for this one density, the SRE method saves 180.68 node hours while only sacrificing 0.51% in accuracy.

Now that we have shown that the SRE method can be feasibly applied to predict converged CCDT-1 correlation energies using only the MBPT2 energies, we can use this method to predict the correlation energies at various densities. Figure 8.9 compares the correlation energies from the last point in the training data at M= 186 (red dashed line), the fully calculated converged ΔE_{CCDT1} at 514 single-particle states (black line), and the SRE prediction for the converged correlation energies at densities around nuclear density. We can see that the training data is far from the converged values, and in fact, the difference between the M= 186 plot and the M= 514 plot is highest at the lower densities shown here. This is to show that the correlation energies used to train the SRE algorithm are not converged, and thus, a noticeable improvement is made over just using the training data. For Figure 8.9, the average percent error between the SRE predictions and the complete calculations is 0.45%. Furthermore, the total computational time saved when generating the six data points with the SRE method over fully generating them at 514 single-particle states is 917.30 node hours.

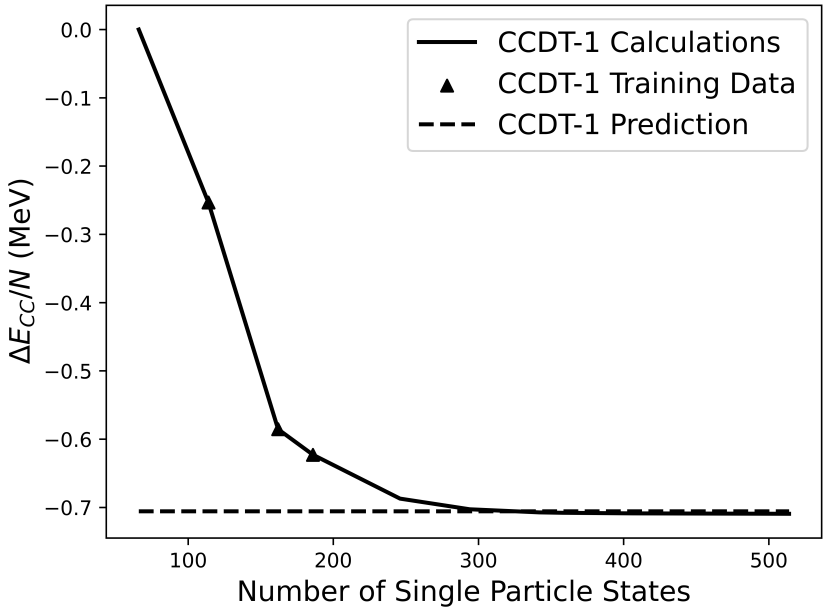


Figure 8.8 The results from performing coupled-cluster calculations with the CCDT-1 approximation on a pure neutron matter system with chiral NNLO potentials, 66 neutrons, and $d = 0.16\ fm^{-3}$. The full calculations are shown with a solid line, the training data used for the SRE algorithm is shown with the triangular markers, and the SRE prediction for the converged correlation energy is shown with the dashed line. The percent error between the SRE prediction and the fully calculated converged correlation energy is 0.51%, and the time savings for performing the SRE method is 180.68 node hours

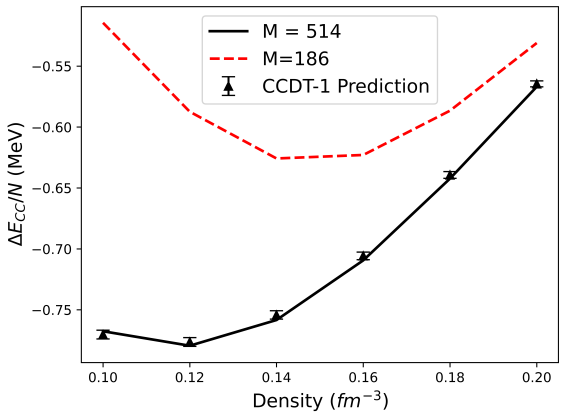


Figure 8.9 The results from predicting the converged $\Delta E_{CCDT1}/N$ for pure neutron matter at densities around nuclear density. The average percent error across all densities is 0.45%, and the time savings from generating only the training data versus the fully converged $\Delta E_{CCDT1}/N$ is 917.30 node hours

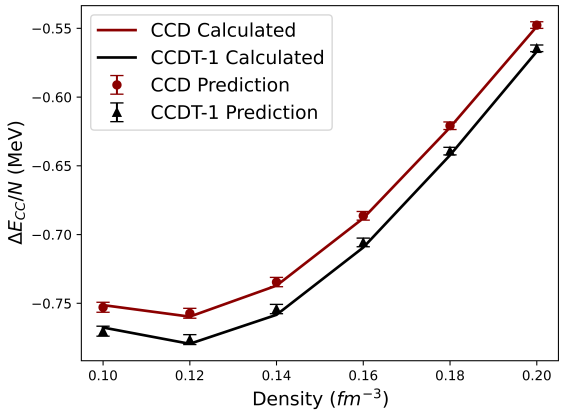


Figure 8.10 CCD (red) versus CCDT-1 (black) correlation energies for a pure neutron matter system with 66 neutrons and chiral potentials. The converged correlation energies from full calculations are shown with solid lines, and the points represent the machine-learning predictions. The machine learning prediction is shown with the uncertainties from the Gaussian process algorithm

Finally, let us compare the CCDT-1 results to the CCD results for densities around nuclear density with the SRE predictions (Figure 8.10). Not only does the SRE method have good predictions for the CCD calculations in the density range shown, it also performs equally well on the more complex CCDT-1 calculations.

As a final set of calculations with pure neutron matter, we can look at performing CC calculations using the CCD(T) approximation.

First, let us compare the correlation energies of a PNM system for densities around nuclear density calculated using CCD, CCDT-1, and CCD(T). It has already been shown that CCD and CCDT-1 give significantly different values for the correlation energies. Still, we want to ensure a significant difference between these two data sets and the CCD(T) energies. Figure 8.11 shows the fully converged correlation energies for a PNM system calculated with the three CC approximations. The CCD and CCD(T) were performed at M= 1,478, and the CCDT-1 calculations were performed at M= 514. It does appear that the three methods give significantly different answers, so it will be an exciting test for the SRE method to compare its performance on all three methods.

Now, we can compare the run times of each method when applied to PNM. The expected run time of a CCD calculation is an iterative $O(M^6)$, and the expected run time of a CCDT-1 calculation is an iterative $O(M^7)$. In comparison, the expected run time of a CCD(T) calculation falls in between

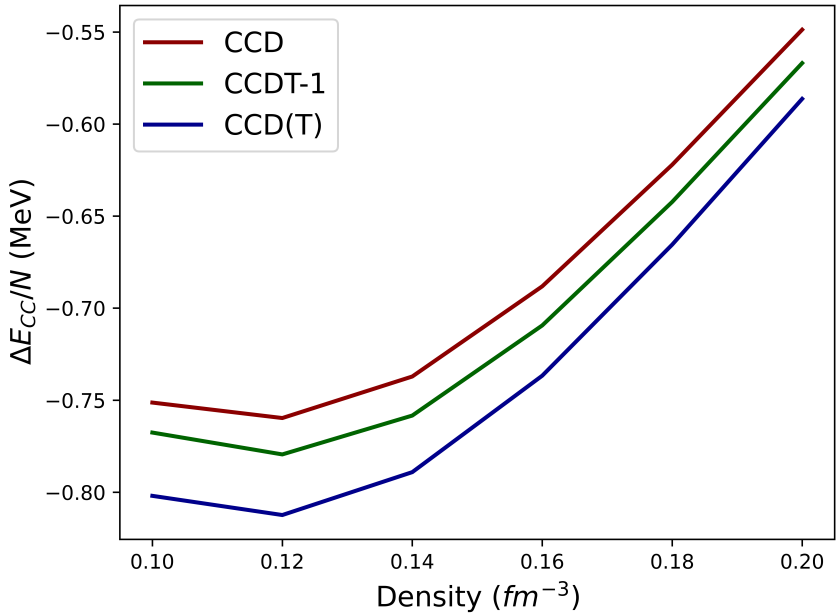


Figure 8.11 Correlation energies of PNM calculated with CCD (red), CCDT-1 (green), and CCD(T) (blue) for densities around nuclear density

these two with an iterative $O(M^6)$ followed by a non-iterative step of $O(M^7)$. And in fact, that is what we see in Figure 8.12 where CCD(T) is faster than CCDT-1 but slower than CCD. Note that the times are reported here in node hours per GHz. The CCD(T) calculations for PNM were performed on Oak Ridge National Laboratory's Andes supercomputer on AMD processors, which have a clock speed of 3.0 GHz and use 64 MPI nodes. Just as we defined a node hour as the total run time of a calculation multiplied by the number of MPI nodes in the calculations to compare the run times computed with different numbers of MPI nodes, here we define node hours per GHz (node hours divided by the clock speed of the processors) to compare results which have been performed on different supercomputers.

Now we can apply the SRE method to attempt to predict the converged CCD(T) correlation energies for PNM. We have used three training points, like with the other PNM and chiral potential extrapolations. Still, here we have moved the training data to slightly higher numbers of single-particle states (M= 186, 246, and 294) due to the slower convergence of the CCD(T) correlation energies with respect to the number of single-particle states. We have kept the SRE sequence length as 1. Figure 8.13 shows the results of performing this SRE analysis with the full CCD(T)

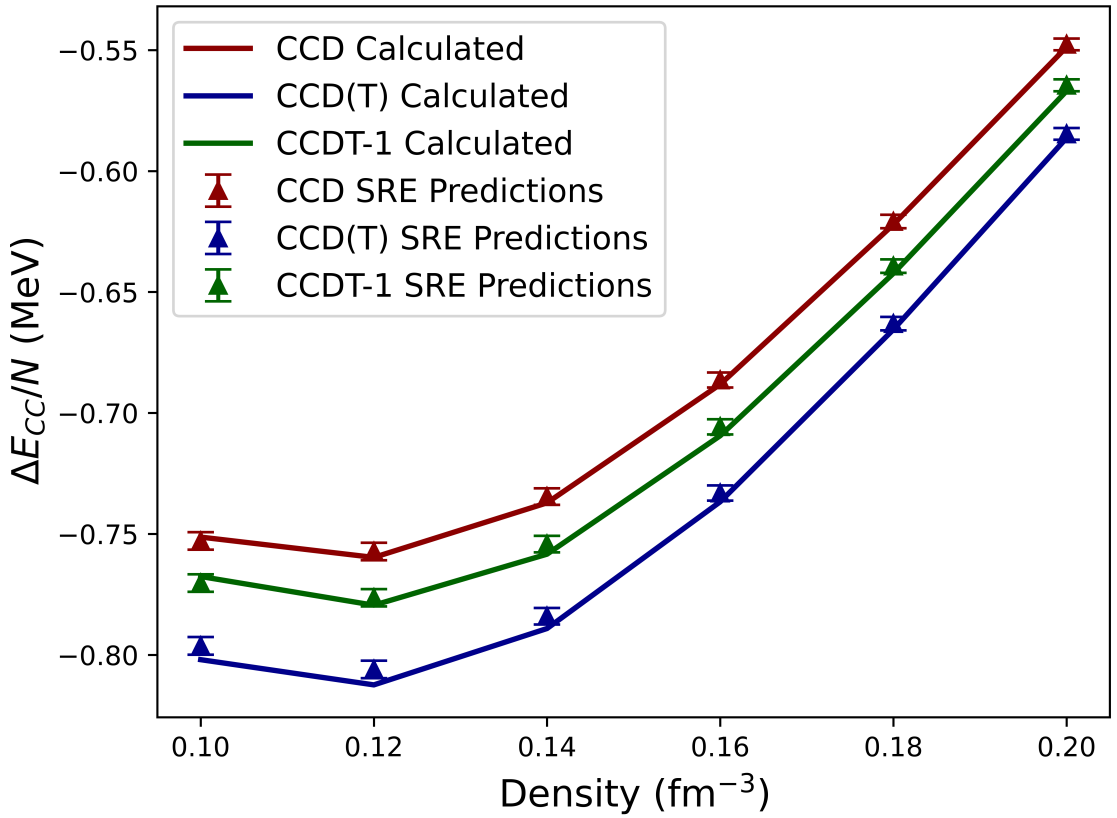


Figure 8.12 The run time (in node hours per GHz) to calculate the correlation energies for a PNM system at $d = 0.16 fm^{-3}$ calculated using CCD (red), CCDT-1 (green), and CCD(T) (blue) as a function of the number of single-particle states in the calculation

correlation energy calculations performed at M= 1,478 shown with the solid line and the SRE predictions with the triangular markers. Here the average percent error between the two data sets is 0.54%. Furthermore, the time needed to perform the six CCD(T) calculations at M= 1,478 shown in Figure 8.13 took 594.15 node hours, but the time needed to generate the SRE training data was only 32.16 node hours. This leads to a time savings of 561.99 node hours or 23.41 *node days*.

Finally, we are in a position to compare the SRE predictions for CC calculations of PNM using CCD (red), CCDT-1 (green), and CCD(T) (blue), as seen in Fig 8.14. For all three CC methods, the complete calculations are shown with solid lines. The SRE predictions are shown with triangular markers with error bars, with the error bars representing the uncertainties from the Gaussian processes algorithm. For all three data sets, the SRE method appears to provide a good match for all the correlation energies across all the densities shown. The average percent error of this

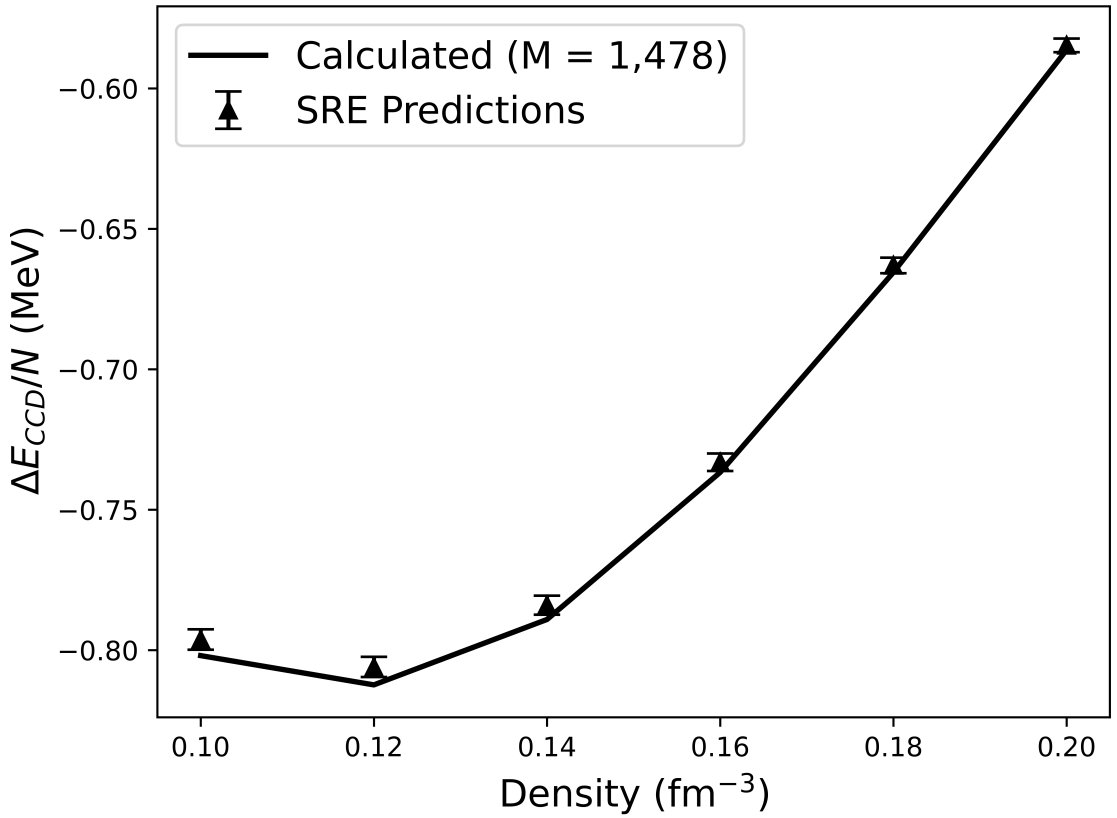


Figure 8.13 The CCD(T) correlation energies for PNM for densities around nuclear density were calculated at M= *1,478 (solid line) and predicted with the SRE method (triangular markers). The uncertainties from the Gaussian processes algorithm are shown with error bars on the markers*

figure is 0.42%. Furthermore, the entire time saved using the SRE method to predict the converged correlation energies rather than fully calculating them is 796.89 node hours or 33.20 *node days*. Therefore, in calculating just 18 points for the PNM system, we save over a month of computational time using the SRE method.

In this section, we have performed coupled cluster calculations of pure neutron matter using two different models of nuclear interaction and three different levels of coupled cluster approximations. However, it is essential to note that the main point of this section is not to compare the effects of different coupled cluster calculations and interactions, though we have done this. Instead, the main point of this section is to determine the performance of the SRE methods on different types of coupled cluster calculations. As we have seen throughout this section, the SRE method can accurately extrapolate to the converged correlation energy of the pure neutron matter system,

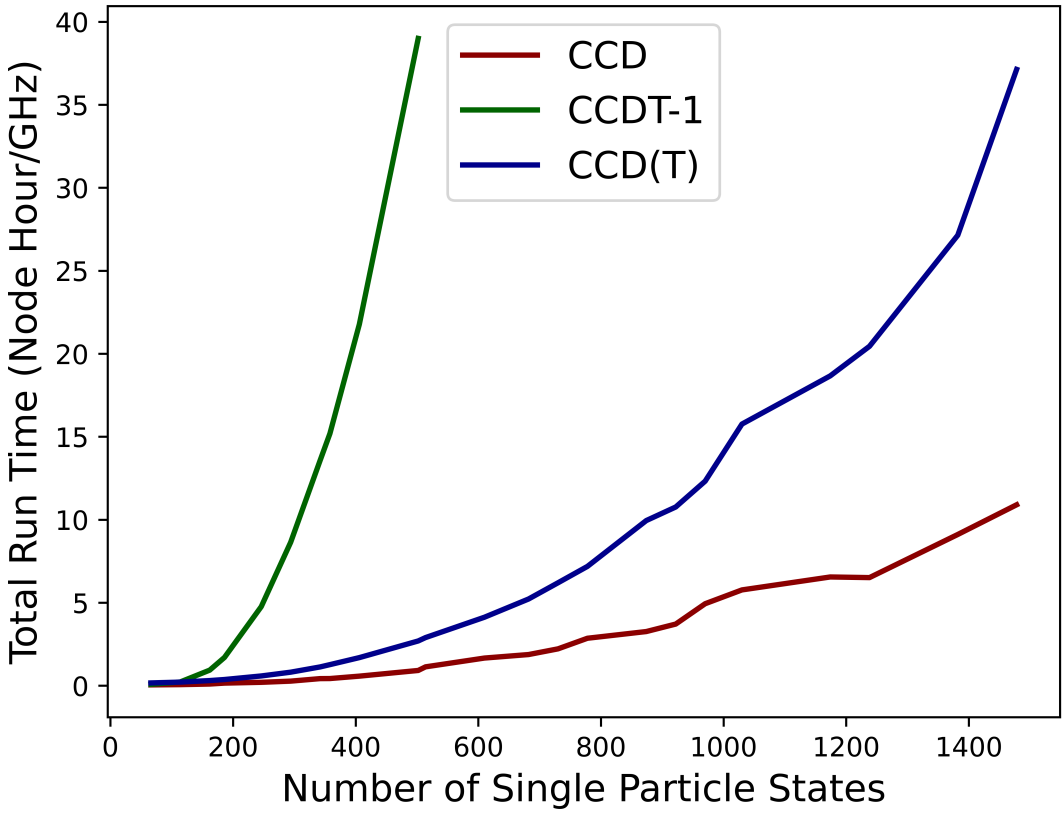


Figure 8.14 The correlation energies for PNM were calculated using CCD (red), CCDT-1 (green), and CCD(T) (blue). The fully converged calculations are shown with the solid lines, and the SRE predictions are shown with the triangular markers

regardless of the combination of nuclear interaction and coupled cluster approximation used. Furthermore, over a month of computational time is saved in this section by using the SRE method with no loss of relevant accuracy. Finally, it is essential to note that even though we are using machine learning, the training data sets we have used range from 3 - 10 data points. These are some of the smallest training data sets found in the literature for machine learning applications in the physical sciences and emphasize an essential feature of the SRE method. That being that the SRE method can make very accurate extrapolations using very little training data.

8.2 Symmetric Nuclear Matter

8.2.1 Coupled Cluster Doubles with Realistic Nuclear Interactions

Now, we can move on from analyzing pure neutron matter and instead turn our attention to symmetric nuclear matter (SNM), which comprises an equivalent number of protons and neutrons. We will

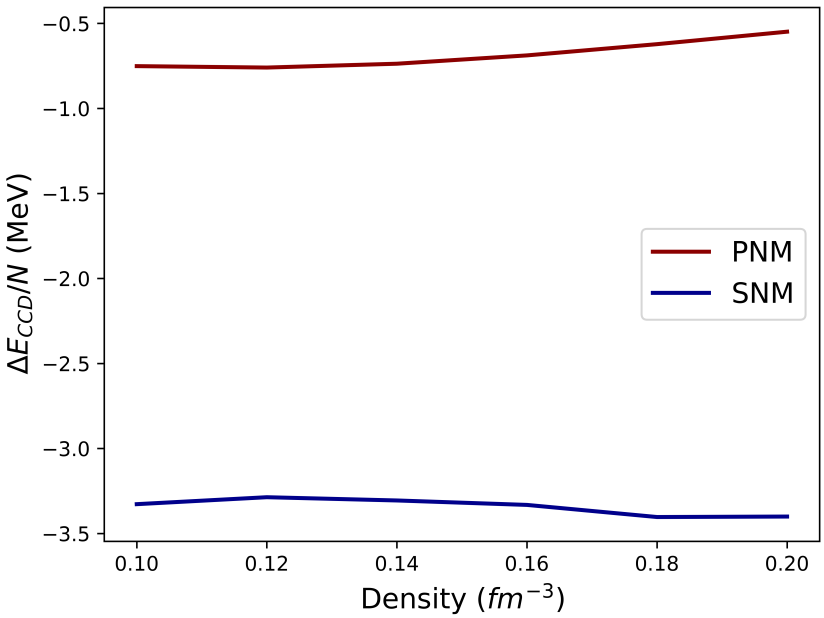


Figure 8.15 The CCD correlation energies for PNM (red) and SNM (blue) for densities around nuclear matter and with a chiral NNLO potential

continue to use the chiral NNLO potentials. Here all SNM systems will be calculated using 132 nucleons (66 protons and 66 neutrons) since these results will be similar to the energies at the TDL [34]. Furthermore, the immense computational costs of SNM calculations also prohibit studies and higher particle numbers. Compared to a PNM calculation, an SNM calculation will need to double the number of single-particle states to accommodate double the number of particles. It will need approximately three times the computational time and resources. However, it is essential to perform calculations of SNM in addition to PNM, even at the higher computational costs, because the addition of protons into the system causes different correlations in the system, leading to a correlation energy that converges slower with respect to the number of single-particle states. Thus SNM should be a more challenging system for the SRE method and will genuinely test its predictive abilities. Additionally, as we think about moving towards calculations of finite nuclei in future works, accurate calculations of both PNM and SNM are essential for these studies. Figure 8.15 compares the CCD correlation energies per nucleon for PNM (red) and SNM (blue). We can see that the SNM correlation energies are significantly larger than the PNM ones.

The CCD data set for SNM was generated using a supercomputer available through Michigan State

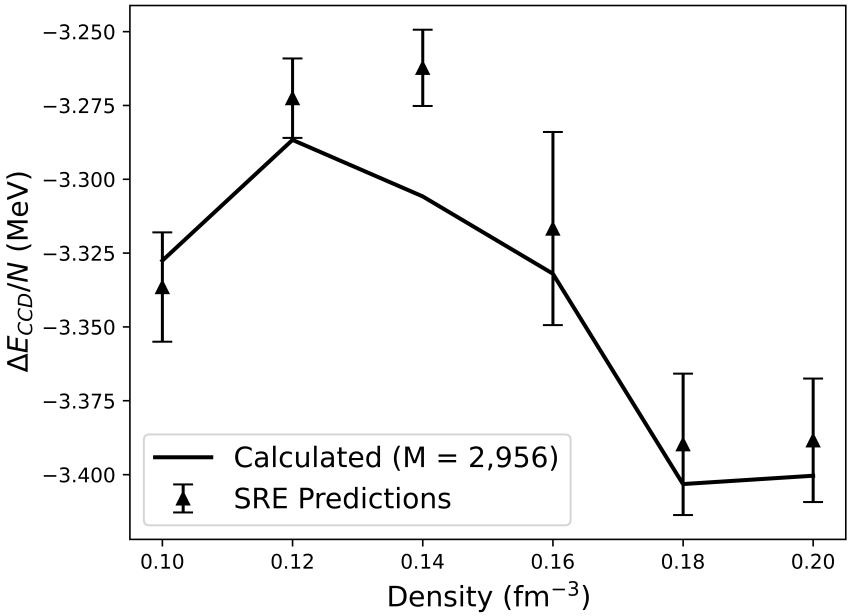


Figure 8.16 The CCD correlation energies for SNM calculated fully at M = *2,956 (solid line) and using the SRE method to predict them (triangular marker). The average percent error of this graph is 0.54% and the time savings of using SRE over performing the full calculations is 6.48 node days*

University's Institute for Cyber-Enabled Research (ICER) using Intel Xeon processors, which have a clock speed of 2.4 GHz. For the MPI parallelization, 24 nodes were used. It takes, on average, 26.07 node hours to generate a converged CCD energy for a PNM system (at M = 1,478). However, since SNM is a more challenging case, and the number of single-particle states doubles, it takes, on average, 84.12 node hours to perform a converged CCD correlation energy calculation for SNM at M = 2,956. This means the computational resources needed for an SNM calculation are more significant than those needed for a PNM calculation by about a factor of three.

The SRE predictions in Figure 8.16 were generated using seven training points from M = 324 to M = 1,004. The SRE sequence length is used as three since there are more training data points here compared to the PNM case. In addition, the SNM CCD energy has a slower convergence than the PNM energies, leading to the use of more training data.

When predicting the CCD correlation energies using the SRE method, the average percent error between the predictions and the complete calculations at M = 2,956 is 0.54%. This is compared to the average percent error between the calculations at M = 2,956 and M = 1,004 (the largest training data point) of 9.52%. Thus the SRE provides a much more accurate method than truncating the

calculations at the level of the training data. Furthermore, it takes 504.77 node hours to compute all six data points shown in Figure 8.16 at M = 2,956 but only 349.17 node hours to generate the training data needed by the SRE algorithm to predict all 6 points. This leads to a computational time savings of 155.61 node hours or 6.48 *node days*. Thus the SRE method can accurately recreate the CCD correlation energies of an SNM system with significant time savings.

8.2.2 Coupled Cluster Doubles with Perturbative Triples and Realistic Potentials

As a final data set with SRE, we will attempt to predict the converged correlation energies for SNM using the CCD(T) approximation. The CCD(T) approximation has significantly higher time requirements than the CCD approximation, as we saw with the PNM case. But this difference in time will be even higher more drastic for SNM since there will be a higher number of single-particle states. This extra factor of M can contribute significantly to the run time since it can be as large as M = 2,956 for the fully converged calculations. However, for this case, due to the high computational costs, we have truncated M to only 23 energy shells or M = 2,060. At this level, the correlation energies have converged to an order of 10^{-3} or less, so it should be sufficient for this work, but it is important to note that if the calculations were carried all the way out to M = 2,956 then the run times would be much higher. We can see in Figure 8.17 that the CCD(T) run times (blue) are, in fact, much higher than the CCD run times. The CCD(T) data was collected on Oak Ridge National Laboratory's Andes supercomputer using 3.0 GHz and 64 MPI nodes. Since the CCD and CCD(T) data was collected on different supercomputers and with a different number of MPI nodes, in Figure 8.17, we have standardized the run time by reporting it as node hour per GHz.

The average time to perform a CCD calculation of SNM at M = 2,956 is 84.12 node hours. However, the average time to complete a CCD(T) calculation of SNM at only M = 2,060 is 335 node hours. This significant increase in run time is necessary as the CCD(T) approximation adds some aspects of the $\hat{T}_3$ operator into the calculations neglected in the CCD approximation. This does make the calculations more accurate (more comparable to an actual SNM system), but we do have to pay the much higher computational cost. This further motivates the use of the SRE method because here we are taking almost 14 *node days* per density of interest, making studies of, for example, the

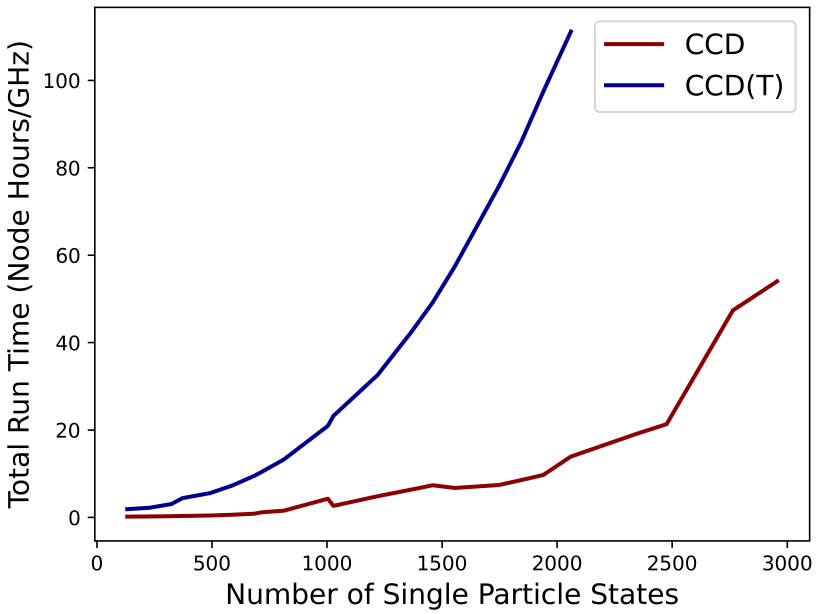


Figure 8.17 The run times (in node hours per GHz) for CCD calculations of SNM (red) and CCD(T) calculations of SNM (blue). The CCD calculations were performed on processors which ran at 2.4GHz and used 24 MPI nodes, while the CCD(T) calculations were performed on processors that ran at 3.0GHz and used 64 MPI nodes

equation of state of nuclear matter prohibitively expensive.

Figure 8.18 shows the results of performing an SRE analysis on CCD(T) calculations of SNM. The calculations at M = 2,060 are shown with the solid lines, and then SRE predictions are shown with triangular markers, which have error bars that come from the uncertainty of the Gaussian process algorithm; here, we have used six training points (from M = 162 to M = 406), and we have set the SRE sequence length to one. This results in an average percent error of 0.65%. Compared to the average percent error between the data at M = 406 (the largest training point) and the converged energies at M = 2,060 of 14.70%, the SRE method accurately predicts the converged CCD(T) energies from unconverged energies. Looking at the run times, it takes a total of 2,000 node hours to generate the six data points at M = 2,060, but it only takes 721 node hours to create the training data needed by the SRE algorithm, which is required to produce these converged correlation energies accurately. This results in a time savings of 1,278 node hours or 53.28 *node days*.

Finally, we can compare the CCD correlation energies for SNM (red) and the CCD(T) correlation energies for SNM (blue) in Figure 8.19. The complete calculations at M = 2,956 for CCD and M

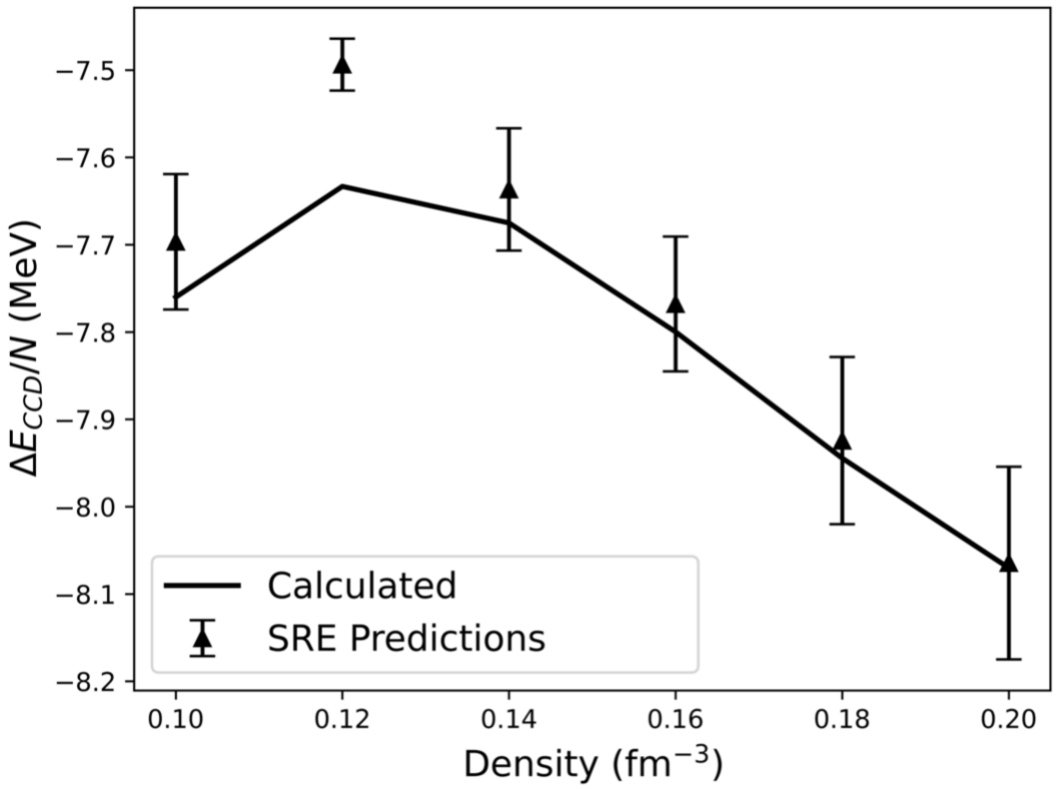


Figure 8.18 The CCD(T) correlation energies for SNM calculated at M = *2,060 (solid line) and predicted with SRE (triangular markers)*

= 2,060 for CCD(T) are shown with solid lines, and the SRE markers are shown with triangular markers. We draw two important conclusions from this figure. First, the CCD and CCD(T) correlation energies are significantly different, so this does justify the higher computational cost of the CCD(T) calculations as the CCD(T) results should more closely model a real system as they take into account some aspects of the $\hat{T}_3$ operator. Second, we can see that the SRE method performs well on both CCD and CCD(T) data sets, again showing that SRE works well on different coupled cluster data sets with no modifications of the algorithm besides changing the two hyperparameters: the number of training data points and the SRE sequence length.

8.3 Symmetry Energy

While we are limiting our infinite matter calculations only to pure neutron matter ($x_p = 0.0$) and symmetric nuclear matter ($x_p = 0.5$), we can still calculate a quantity that is important to studies

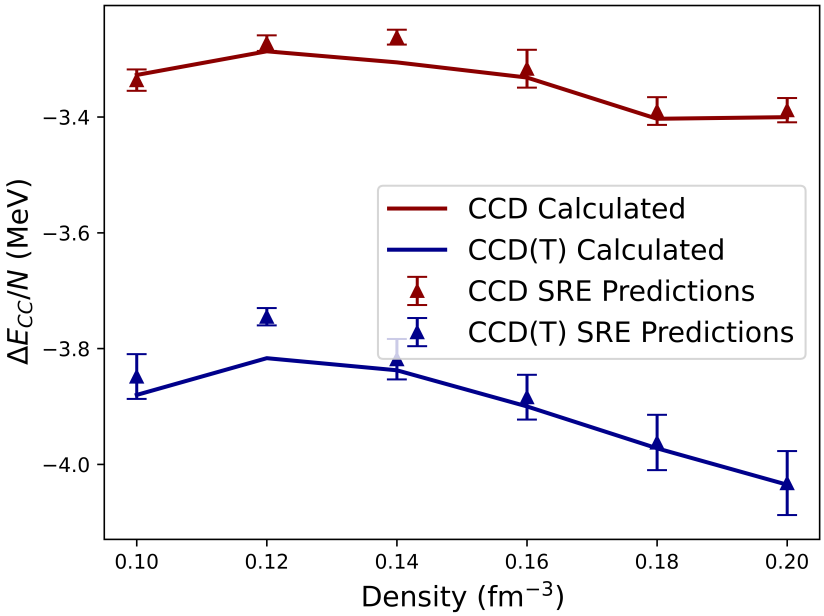


Figure 8.19 The correlation energies for SNM were calculated with CCD (red) and CCD(T) (blue). The full calculations at M = *2,956 (CCD) and* M = *2,060 (CCD(T)) are shown with solid lines, and the SRE predictions are shown with triangular markers*

of infinite nuclear matter and the nuclear equation of state. The quantity is the symmetry energy which is defined as:

$$E_{sym}(\rho) = E(\rho, x_p = 0) - E(\rho, x_p = 1/2). \tag{8.1}$$

Figure 8.20 shows the symmetry energies calculated for both the CCD approximation (red) and the CCD(T) approximation (blue) for densities around nuclear density. The symmetry energies calculated with full converged energies at M = 1,478 for PNM and M = 2,956 for SNM are shown with the solid lines, and the symmetry energies calculated with SRE predictions are shown with triangular markers. Additionally, the uncertainty on the SRE predictions is shown with error bars. This uncertainty comes from the Gaussian process algorithm. Note that this plot is energy per nucleon instead of correlation energy per nucleon as the other plots in this chapter.

For the CCD approximation, the average percent error between the two data sets is 0.056%, and for the CCD(T) approximation, it is 0.078%. Note that these percent errors are much smaller than previous results because we are considering the total energy instead of just the correlation energies so small differences have much less of an impact. We also need to consider time savings

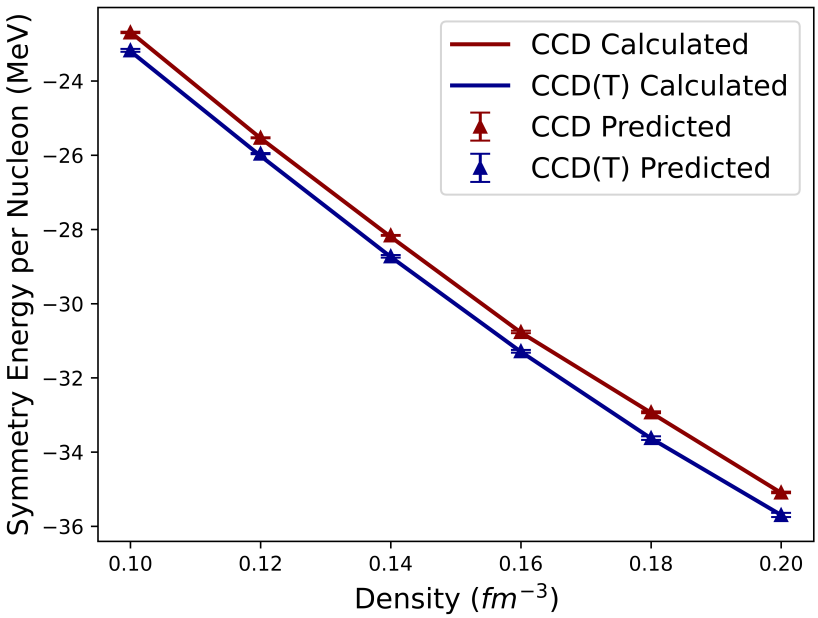


Figure 8.20 The symmetry energy for infinite nuclear matter is calculated using the CCD approximation (red) and the CCD(T) approximation (blue). The symmetry energy calculated from fully converged calculations is shown with the solid lines and SRE predictions with the triangular points. Error bars also represent the uncertainty on the symmetry energy from SRE predictions from the Gaussian process algorithm

as a justification for performing the SRE method over the complete calculations for the symmetry energy. The time needed to generate all 12 points (6 for PNM and 6 for SNM) needed to calculate the symmetry energy at M = 1,478, or 2,956, is 661 node hours for the CCD approximation and 2608 node hours for the CCD(T) approximation. The time needed to generate all training data for the CCD approximation is 353 node hours and 1472 node hours for the CCD(T) approximation. The training data requires just 55.8% of the computational time it takes to generate the fully converged energies and most of this time is due to the high cost of SNM calculations. This leads to a time savings of 20.11 *node days* for the CCD approximation with an average error of 0.056%. The time savings for the CCD(T) approximation is 47.33 *node days* with an average percent error of only 0.078%. Therefore, when calculating the symmetry energy, the SRE method provides an accurate prediction and can save many days of computational time.

8.4 Final Remarks

As pointed out in the pure neutron matter section of this chapter, the main goal of this chapter is not to compare coupled cluster calculations with different levels of approximation, different nuclear interactions, and different infinite nuclear matter systems. Though we have made some of these comparisons, the main point of this chapter is to analyze the accuracy and time savings of the SRE algorithm. As we have seen throughout this chapter, coupled cluster calculations of nuclear systems have very large (and possibly prohibitive) computational times, especially when using realistic nuclear interactions and higher-order coupled cluster approximations. The SRE method, with its ability to make accurate extrapolations with a small amount of training data, can make these studies more feasible by drastically reducing the run times needed to perform the calculations. This is especially important when we realize that studies of the nuclear equation of state (an essential future work for this thesis) will require accurate calculations at different densities and proton fractions. Thus, the development of the SRE method, by reducing the computational time needed to perform accurate calculations, makes large-scale studies of infinite nuclear matter much more feasible, paving the way for novel studies to occur in the future.

CHAPTER 9

CONCLUSION

The sequential regression extrapolation (SRE) method developed here based on Bayesian regression algorithms is an accurate and valuable extrapolator for removing basis incompleteness errors from coupled cluster calculations of infinite matter systems. Furthermore, when the infinite matter system needs to be taken to the thermodynamic limit, it is possible to use SRE to perform this task. Since the SRE algorithm uses training data taken from calculations at small numbers of single-particle states to predict the correlation energy at many single-particle states, the SRE algorithm can offer significant time savings over performing fully converged correlation energy calculations. As shown in this thesis, using the SRE algorithm to save over 100 node hours in the calculations of just one correlation energy is possible. Furthermore, this huge time savings does come with a loss in the accuracy of performing the whole savings. However, the average percent error between the SRE prediction and the fully calculated result was typically less than 1%, making it quite a slight difference compared to the large amount of computational time that has been saved.

Furthermore, besides developing the SRE method, we also compared different methods of performing coupled cluster calculations of systems of infinite nuclear matter. First, we compared the results from two different interactions: the Minnesota potential, a toy interaction, and chiral NNLO potentials, which are much more realistic. By comparing these two, we learned that they differ quite significantly around densities of infinite nuclear matter that are similar to nuclear densities and that calculations containing NNLO potentials do take much longer to compute compared to a Minnesota potential applied to the same system. Furthermore, we could also compare the difference between the coupled cluster approximations coupled cluster doubles (CCD), the iterative coupled cluster triples approximation (CCDT-1), and coupled cluster doubles with perturbative triples (CCD(T)) on calculations of infinite matter systems. We found that the triples approximations give significant results when compared to the CCD approximation, so they are worth performing even though they provide an increase in computational time.

Though it has been mentioned throughout this thesis, it is essential to emphasize the size of the

training data sets used in this work. This work's most extensive training data set used only 16 training points, and the smallest training set used only 3 points. Some areas of physics could be faster to adopt machine learning because of the vast amount of training data that some machine learning algorithms require. However, this work has shown that accurate machine learning predictions can be made with very few training points, thus encouraging using machine learning as a tool in fields with small data sets.

Possible Future Works

A few notable future works stem directly from the work presented here. First, while we showed results from both the CCD, CCDT-1, and CCD(T) approximations, we did not have the capabilities to produce coupled cluster correlation energies using a complete CCDT calculation. This is mainly due to the high computational costs of a complete triple calculation ($O(M^8)$), but recent computational advancements make this more achievable.

Additionally, while we only looked at pure neutron matter and symmetric nuclear matter here, there are other proton fractions of interest. If we want to model the equation of the state of nuclear matter thoroughly, then we need to be able to accurately predict the properties of neutron matter at proton fractions beyond just 0.0 and 0.5.

While truncating the number of particles in a calculation is limited to infinite matter and other large systems, truncations occur in every *ab initio* many-body calculation. Basis truncation is especially common and occurs in almost every calculation except some simple toy models. The last part of this thesis will be dedicated to exploring some possible future applications of the SRE methodology which has been developed.

An extension of the work presented in this thesis is to apply the SRE method to remove basis incompleteness errors from coupled cluster calculations of nuclei. Though nuclei are finite systems and the number of nucleons in the system generally does not need to be truncated, the number of single-particle states is still truncated, leading to a need to extrapolate to an infinite model space [51]. This is especially true for heavy nuclei and nuclei that are weakly bound [51]. There are methods to perform these extrapolations on nuclei calculations, but when using the harmonic

oscillator basis, which mixes the ultraviolet and infrared cutoffs, these extrapolation methods can fail [51]. Machine learning has been used to perform similar extrapolations (see Reference [51], [64], [65], for example), but these were performed with neural networks and thus incurred all of the problems that were experienced with neural networks in this thesis.

Additionally, the SRE method has no reason to be restricted to only predicting CC energies using MBPT2 energies. Extending the SRE method to other many-body methods should also be possible. One of the most accurate yet restricted *ab initio* many-body methods is full configuration interaction theory (FCI), which uses a variationally optimized linear combination of the full set of Slater determinants. Full configuration interaction is used in nuclear physics and electronic-structure theory, but its complexity limits it to only the smallest of systems. The ground state energy, which FCI finds, is the lowest (variationally) and most accurate that can be achieved. If an infinite single particle basis is used, FCI produces the solutions to the Schrödinger equation. However, due to computational limitations, FCI calculations must be performed with a finite basis, meaning they will fail to retrieve the total energy [76].

While most of this thesis, except for the sections on the electron gas, have been devoted to nuclear physics applications, *ab initio* many-body methods occur in many other fields besides nuclear physics. For example, similarity renormalization group theory (RG), coupled cluster theory, and other many-body methods are prevalent in other fields of physics and quantum chemistry. Therefore, the development of the SRE method should improve calculations outside of the realm in which it was developed.

BIBLIOGRAPHY

[1] W.H. Young N.H. March and S. Sampanthar. *The Many-Body Problem in Quantum Physics*. Dover Publications, 1995.

[2] G Hagen et al. "Coupled-cluster computations of atomic nuclei". In: *Reports on Progress in Physics* 77.9 (Sept. 2014), p. 096302. DOI: 10.1088/0034-4885/77/9/096302.

[3] Gustav Baardsen. "Coupled-cluster theory for infinite matter". Ph.D.. University of Oslo, 2014.

[4] D. J. Dean and M. Hjorth-Jensen. "Coupled-cluster approach to nuclear physics". In: *Physical Review C* 69 (5 May 2004), p. 054320. DOI: 10.1103/PhysRevC.69.054320.

[5] Jiří Čížek. "On the Correlation Problem in Atomic and Molecular Systems. Calculation of Wavefunction Components in Ursell-Type Expansion Using Quantum-Field Theoretical Methods". In: *The Journal of Chemical Physics* 45.11 (May 2004), pp. 4256–4266. DOI: 10.1063/1.1727484.

[6] J. Čižek and J. Paldus. "Correlation problems in atomic and molecular systems III. Rederivation of the coupled-pair many-electron theory using the traditional quantum chemical methodst". In: *International Journal of Quantum Chemistry* 5.4 (1971), pp. 359–379. DOI: https://doi.org/10.1002/qua.560050402.

[7] Rodney J. Bartlett and Monika Musiał. "Coupled-cluster theory in quantum chemistry". In: *Review of Modern Physics* 79 (1 Feb. 2007), pp. 291–352. DOI: 10.1103/RevModPhys.79.291.

[8] H. Kümmel, K.H. Lührmann, and J.G. Zabolitzky. "Many-fermion theory in expS- (or coupled cluster) form". In: *Physics Reports* 36.1 (1978), pp. 1–63. DOI: https://doi.org/10.1016/0370-1573(78)90081-9.

[9] R. F. Bishop. "An overview of coupled cluster theory and its applications in physics". English. In: *Theoretica Chimica Acta* 80 (1991), pp. 95–148. DOI: 10.1007/BF01119617.

[10] Jochen H. Heisenberg and Bogdan Mihaila. "Ground state correlations and mean field in ^{16}O". In: *Physical Review C* 59 (3 Mar. 1999), pp. 1440–1448. DOI: 10.1103/PhysRevC.59.1440.

[11] Baishan Hu et al. "Ab initio predictions link the neutron skin of 208Pb to nuclear forces". In: *Nature Physics* 18.10 (Aug. 2022). DOI: 10.1038/s41567-022-01715-8.

[12] A. Ekström et al. "Accurate nuclear radii and binding energies from a chiral interaction". In: *Physical Review C* 91 (5 May 2015), p. 051301. DOI: 10.1103/PhysRevC.91.051301.

[13] L. Coraggio et al. "Nuclear-matter equation of state with consistent two- and three-body perturbative chiral interactions". In: *Physical Review C* 89 (4 Apr. 2014), p. 044321. DOI:

10.1103/PhysRevC.89.044321.

[14] F. Sammarruca et al. “Toward order-by-order calculations of the nuclear and neutron matter equations of state in chiral effective field theory”. In: *Physical Review C* 91 (5 May 2015), p. 054311. DOI: 10.1103/PhysRevC.91.054311.

[15] Robert Roth et al. “Medium-Mass Nuclei with Normal-Ordered Chiral NN+3N Interactions”. In: *Physical Review Letters* 109 (5 July 2012), p. 052501. DOI: 10.1103/PhysRevLett.109.052501.

[16] Alessandro Lovato et al. “Comparative study of three-nucleon potentials in nuclear matter”. In: *Physical Review C* 85 (2 Feb. 2012), p. 024003. DOI: 10.1103/PhysRevC.85.024003.

[17] Francesca Sammarruca and Randy Millerson. “Exploring the relationship between nuclear matter and finite nuclei with chiral two- and three-nucleon forces”. In: *Physical Review C* 102 (3 Sept. 2020), p. 034313. DOI: 10.1103/PhysRevC.102.034313.

[18] A. Ekström et al. “Δ isobars and nuclear saturation”. In: *Physical Review C* 97 (2 Feb. 2018), p. 024332. DOI: 10.1103/PhysRevC.97.024332.

[19] W. G. Jiang et al. “Accurate bulk properties of nuclei from $A = 2$ to ∞ from potentials with Δ isobars”. In: *Physical Review C* 102 (5 Nov. 2020), p. 054301. DOI: 10.1103/PhysRevC.102.054301.

[20] S. Mishev and M. Savova. “Coupled-cluster method for nucleonic matter using GPU tensor cores”. In: *AIP Conference Proceedings* 2522.1 (Sept. 2022). 090007. DOI: 10.1063/5.0101117.

[21] L Satpathy. “Infinite nuclear matter based for mass of atomic nuclei”. In: *Journal of Physics G: Nuclear Physics* 13.6 (June 1987), p. 761. DOI: 10.1088/0305-4616/13/6/009.

[22] J.W. Negele and D. Vautherin. “Neutron star matter at sub-nuclear densities”. In: *Nuclear Physics A* 207.2 (1973), pp. 298–320. DOI: https://doi.org/10.1016/0375-9474(73)90349-7.

[23] S. Gandolfi, J. Carlson, and Sanjay Reddy. “Maximum mass and radius of neutron stars, and the nuclear symmetry energy”. In: *Physical Review C* 85 (3 Mar. 2012), p. 032801. DOI: 10.1103/PhysRevC.85.032801.

[24] W. G. Jiang et al. “Emulating \emphab initio computations of infinite nucleonic matter”. In: (Dec. 2022).

[25] H. Moeini and G.H. Bordbar. “Nuclear matter calculations with the phenomenological three-nucleon interaction”. In: *Nuclear Physics A* 1017 (2022), p. 122339. DOI: https://doi.org/10.1016/j.nuclphysa.2021.122339.

[26] "Neutron Star Models: Masses and Radii". In: *Black Holes, White Dwarfs, and Neutron Stars*. John Wiley & Sons, Ltd, 1983. Chap. 9, pp. 241–266. DOI: https://doi.org/10.1002/9783527617661.ch9.

[27] J. M. Lattimer and M. Prakash. "Neutron Star Structure and the Equation of State". In: *The Astrophysical Journal* 550.1 (Mar. 2001), p. 426. DOI: 10.1086/319702.

[28] James M. Lattimer and Madappa Prakash. "Neutron star observations: Prognosis for equation of state constraints". In: *Physics Reports* 442.1 (2007). The Hans Bethe Centennial Volume 1906-2006, pp. 109–165. DOI: https://doi.org/10.1016/j.physrep.2007.02.003.

[29] Andrew W. Steiner, James M. Lattimer, and Edward F. Brown. "THE EQUATION OF STATE FROM OBSERVED MASSES AND RADII OF NEUTRON STARS". in: *The Astrophysical Journal* 722.1 (Sept. 2010), p. 33. DOI: 10.1088/0004-637X/722/1/33.

[30] James M. Lattimer. "The Nuclear Equation of State and Neutron Star Masses". In: *Annual Review of Nuclear and Particle Science* 62.1 (2012), pp. 485–515. DOI: 10.1146/annurev-nucl-102711-095018.

[31] Henning Heiselberg and Morten Hjorth-Jensen. "Phases of dense matter in neutron stars". In: *Physics Reports* 328.5 (2000), pp. 237–327. DOI: https://doi.org/10.1016/S0370-1573(99)00110-6.

[32] S. Gandolfi et al. "Quantum Monte Carlo calculation of the equation of state of neutron matter". In: *Physical Review C* 79 (5 May 2009), p. 054005. DOI: 10.1103/PhysRevC.79.054005.

[33] S. Gandolfi et al. "Equation of state of low-density neutron matter, and the $^1S_0$ pairing gap". In: *Physical Review C* 80 (4 Oct. 2009), p. 045802. DOI: 10.1103/PhysRevC.80.045802.

[34] G. Hagen et al. "Coupled-cluster calculations of nucleonic matter". In: *Physical Review C* 89 (1 Jan. 2014), p. 014319. DOI: 10.1103/PhysRevC.89.014319.

[35] G. Baardsen et al. "Coupled-cluster studies of infinite nuclear matter". In: *Physical Review C* 88 (5 Nov. 2013), p. 054312. DOI: 10.1103/PhysRevC.88.054312.

[36] Sean Bruce Sangolt Miller. "Quantum Mechanical Studies of Infinite Matter by the use of Coupled-Cluster Calculations, with an Emphasis on Nuclear Matter". Master's. University of Oslo, 2017.

[37] Murray Gell-Mann and Keith A. Brueckner. "Correlation Energy of an Electron Gas at High Density". In: *Physical Review* 106 (2 Apr. 1957), pp. 364–368. DOI: 10.1103/PhysRev.106.364.

[38] James J. Shepherd. "Communication: Convergence of many-body wave-function expansions using a plane-wave basis in the thermodynamic limit". In: *The Journal of Chemical Physics*

145.3 (2016), p. 031104. DOI: 10.1063/1.4958461.

[39] R. F. Bishop and K. H. Lührmann. "Electron correlations. II. Ground-state results at low and metallic densities". In: *Physical Review B* 26 (10 Nov. 1982), pp. 5523–5557. DOI: 10.1103/PhysRevB.26.5523.

[40] R.F. Bishop and K.H. Lührmann. "Coupled clusters and the electron gas at metallic densities". In: *Physica B+C* 108.1 (1981), pp. 873–874. DOI: https://doi.org/10.1016/0378-4363(81)90741-5.

[41] Verena A. Neufeld, Hong-Zhou Ye, and Timothy C. Berkelbach. "Ground-State Properties of Metallic Solids from Ab Initio Coupled-Cluster Theory". In: *The Journal of Physical Chemistry Letterss* 13.32 (2022). PMID: 35939802, pp. 7497–7503. DOI: 10.1021/acs.jpclett.2c01828.

[42] Tina N. Mihm, Bingdi Yang, and James J. Shepherd. "Power Laws Used to Extrapolate the Coupled Cluster Correlation Energy to the Thermodynamic Limit". In: *Journal of Chemical Theory and Computation* 17.5 (2021). PMID: 33830754, pp. 2752–2758. DOI: 10.1021/acs.jctc.0c01171.

[43] Verena A. Neufeld and Alex J. W. Thom. "A study of the dense uniform electron gas with high orders of coupled cluster". In: *The Journal of Chemical Physics* 147.19 (Nov. 2017). 194105. DOI: 10.1063/1.5003794.

[44] Andreas Köhn and David P. Tew. "Towards the Hartree–Fock and coupled-cluster singles and doubles basis set limit: A study of various models that employ single excitations into a complementary auxiliary basis set". In: *The Journal of Chemical Physics* 132.2 (Jan. 2010). 024101. DOI: 10.1063/1.3291040.

[45] Thomas B. Adler and Hans-Joachim Werner. "Local explicitly correlated coupled-cluster methods: Efficient removal of the basis set incompleteness and domain errors". In: *The Journal of Chemical Physics* 130.24 (June 2009). 241101. DOI: 10.1063/1.3160675.

[46] Ke Liao and Andreas Grüneis. "Communication: Finite size correction in periodic coupled cluster theory calculations of solids". In: *The Journal of Chemical Physics* 145.14 (Oct. 2016). 141102. DOI: 10.1063/1.4964307.

[47] Tina N. Mihm, Laura Weiler, and James J. Shepherd. "How the Exchange Energy Can Affect the Power Laws Used to Extrapolate the Coupled Cluster Correlation Energy to the Thermodynamic Limit". In: *Journal of Chemical Theory and Computation* 19.6 (2023). PMID: 36918372, pp. 1686–1697. DOI: 10.1021/acs.jctc.2c00737.

[48] Xin Xing and Lin Lin. "Origin of inverse volume scaling in periodic coupled cluster calculations towards thermodynamic limit". In: (Apr. 2023).

[49] Tina N. Mihm, William Z. Van Benschoten, and James J. Shepherd. "Accelerating convergence to the thermodynamic limit with twist angle selection applied to methods beyond many-body perturbation theory". In: *The Journal of Chemical Physics* 154.2 (Jan. 2021). 024113. DOI: 10.1063/5.0033408.

[50] T.N. Mihm, T. Schäfer, and S.K. et al. Ramadugu. "A shortcut to the thermodynamic limit for quantum many-body calculations of metals". In: *Nat Comput Sci* 1 (2021), pp. 801–808. DOI: 10.1038/s43588-021-00165-1.

[51] W. G. Jiang, G. Hagen, and T. Papenbrock. "Extrapolation of nuclear structure observables with artificial neural networks". In: *Physical Review C* 100 (5 Nov. 2019), p. 054326. DOI: 10.1103/PhysRevC.100.054326.

[52] R. Utama and J. Piekarewicz. "Validating neural-network refinements of nuclear mass models". In: *Physical Review C* 97 (1 Jan. 2018), p. 014306. DOI: 10.1103/PhysRevC.97.014306.

[53] Giuseppe Carleo and Matthias Troyer. "Solving the quantum many-body problem with artificial neural networks". In: *Science* 355.6325 (2017), pp. 602–606. DOI: 10.1126/science.aag2302.

[54] Laura Weiler, Tina N. Mihm, and James J. Shepherd. "Machine learning for a finite size correction in periodic coupled cluster theory calculations". In: *The Journal of Chemical Physics* 156.20 (May 2022). 204109. DOI: 10.1063/5.0086580.

[55] Justin Smith et al. "Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning". In: *Nature Communications* 10 (July 2019). DOI: 10.1038/s41467-019-10827-4.

[56] M. R. Mumpower et al. "Physically interpretable machine learning for nuclear masses". In: *Physical Review C* 106 (2 Aug. 2022), p. L021301. DOI: 10.1103/PhysRevC.106.L021301.

[57] Bryce Fore et al. "Dilute neutron star matter from neural-network quantum states". In: (Dec. 2022).

[58] Rodrigo A. Vargas-Hernández et al. "Extrapolating Quantum Observables with Machine Learning: Inferring Multiple Phase Transitions from Properties of a Single Phase". In: *Physical Review Letters* 121 (25 Dec. 2018), p. 255702. DOI: 10.1103/PhysRevLett.121.255702.

[59] Léo Neufcourt et al. "Bayesian approach to model-based extrapolation of nuclear observables". In: *Physical Review C* 98 (3 Sept. 2018), p. 034318. DOI: 10.1103/PhysRevC.98.034318.

[60] R. Utama, J. Piekarewicz, and H. B. Prosper. "Nuclear mass predictions for the crustal composition of neutron stars: A Bayesian neural network approach". In: *Physical Review C* 93 (1 Jan. 2016), p. 014311. DOI: 10.1103/PhysRevC.93.014311.

[61] Marco Knöll et al. “Machine learning for the prediction of converged energies from ab initio nuclear structure calculations”. In: *Physics Letterss B* 839 (2023), p. 137781. DOI: https://doi.org/10.1016/j.physletb.2023.137781.

[62] Johannes T. Margraf and Karsten Reuter. “Making the Coupled Cluster Correlation Energy Machine-Learnable”. In: *The Journal of Physical Chemistry A* 122.30 (2018). PMID: 29985611, pp. 6343–6348. DOI: 10.1021/acs.jpca.8b04455.

[63] Ying Zhang and Chen Ling. “A strategy to apply machine learning to small datasets in materials science”. In: *npj Computational Materials* 4 (May 2018). DOI: 10.1038/s41524-018-0081-z.

[64] Gianina Alina Negoita et al. “Deep Learning: A Tool for Computational Nuclear Physics”. In: (Mar. 2018).

[65] Gianina Alina Negoita et al. “Deep learning: Extrapolation tool for ab initio nuclear theory”. In: *Physical Review C* 99 (5 May 2019), p. 054308. DOI: 10.1103/PhysRevC.99.054308.

[66] S. Athanassopoulos et al. “Nuclear mass systematics using neural networks”. In: *Nuclear Physics A* 743.4 (2004), pp. 222–235. DOI: https://doi.org/10.1016/j.nuclphysa.2004.08.006.

[67] R. Utama, J. Piekarewicz, and H. B. Prosper. “Nuclear mass predictions for the crustal composition of neutron stars: A Bayesian neural network approach”. In: *Physical Review C* 93 (1 Jan. 2016), p. 014311. DOI: 10.1103/PhysRevC.93.014311.

[68] R Utama, Wei-Chia Chen, and J Piekarewicz. “Nuclear charge radii: density functional theory meets Bayesian neural networks”. In: *Journal of Physics G: Nuclear and Particle Physics* 43.11 (Oct. 2016), p. 114002. DOI: 10.1088/0954-3899/43/11/114002.

[69] R. Utama and J. Piekarewicz. “Refining mass formulas for astrophysical applications: A Bayesian neural network approach”. In: *Physical Review C* 96 (4 Oct. 2017), p. 044308. DOI: 10.1103/PhysRevC.96.044308.

[70] Amber Boehnlein et al. “Colloquium: Machine learning in nuclear physics”. In: *Review of Modern Physics* 94 (3 Sept. 2022), p. 031003. DOI: 10.1103/RevModPhys.94.031003.

[71] Nawar Ismail and Alexandros Gezerlis. “Machine-learning approach to finite-size effects in systems with strongly interacting fermions”. In: *Physical Review C* 104 (5 Nov. 2021), p. 055802. DOI: 10.1103/PhysRevC.104.055802.

[72] Sota Yoshida. “Nonparametric Bayesian approach to extrapolation problems in configuration interaction methods”. In: *Physical Review C* 102 (2 Aug. 2020), p. 024305. DOI: 10.1103/PhysRevC.102.024305.

[73] Bryce Meredig et al. “Can machine learning identify the next high-temperature superconductor? Examining extrapolation performance for materials discovery”. In: *Mol. Syst. Des. Eng.*

3 (5 2018), pp. 819–825. DOI: 10.1039/C8ME00012C.

[74] Ryosuke Jinnouchi, Hirohito Hirata, and Ryoji Asahi. "Extrapolating Energetics on Clusters and Single-Crystal Surfaces to Nanoparticles by Machine-Learning Scheme". In: *The Journal of Physical Chemistry C* 121.47 (2017), pp. 26397–26405. DOI: 10.1021/acs.jpcc.7b08686.

[75] Rodrigo Navarro Pérez and Nicolas Schunck. "Controlling extrapolations of nuclear properties with feature selection". In: *Physics Letterss B* 833 (2022), p. 137336. DOI: https://doi.org/10.1016/j.physletb.2022.137336.

[76] James J. Shepherd et al. "Convergence of many-body wave-function expansions using a plane-wave basis: From homogeneous electron gas to solid state systems". In: *Physical Review B* 86 (3 July 2012), p. 035111. DOI: 10.1103/PhysRevB.86.035111.

[77] Justin G. Lietz et al. "Computational Nuclear Physics and Post Hartree-Fock Methods". In: *An Advanced Course in Computational Nuclear Physics: Bridging the Scales from Quarks to Neutron Stars*. Ed. by Morten Hjorth-Jensen, Maria Paola Lombardo, and Ubirajara van Kolck. Cham: Springer International Publishing, 2017, pp. 293–399. DOI: 10.1007/978-3-319-53336-0_8.

[78] Justin Gage Leitz. "Computational developments for ab initio many-body theory". Ph.D.. Michigan State University, 2019.

[79] Aurélien Géron. *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. Sebastopol, CA: O'Reilly Media, 2017.

[80] Isaiah Shavitt and Rodney J. Bartlett. *Many-Body Methods in Chemistry and Physics: MBPT and Coupled-Cluster Theory*. Cambridge Molecular Science. Cambridge University Press, 2009. DOI: 10.1017/CBO9780511596834.

[81] J. Paldus and J. čížek. "Time-Independent Diagrammatic Approach to Perturbation Theory of Fermion Systems". In: ed. by Per-Olov Löwdin. Vol. 9. Advances in Quantum Chemistry. Academic Press, 1975, pp. 105–197. DOI: https://doi.org/10.1016/S0065-3276(08)60040-4.

[82] D. R. Hartree. "The Wave Mechanics of an Atom with a Non-Coulomb Central Field. Part I. Theory and Methods". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 24.1 (1928), pp. 89–110. DOI: 10.1017/S0305004100011919.

[83] V. Fock. "Näherungsmethode zur Lösung des quantenmechanischen Mehrkörperproblems". In: *Zeitschrift fur Physik* 61.1-2 (Jan. 1930), pp. 126–148. DOI: 10.1007/BF01340294.

[84] Chr. Møller and M. S. Plesset. "Note on an Approximation Treatment for Many-Electron Systems". In: *Physical Review* 46 (7 Oct. 1934), pp. 618–622. DOI: 10.1103/PhysRev.46.618.

[85] J. Hubbard. "The description of collective motions in terms of many-body perturbation

theory". In: *Proceedings of the Royal Society A* 240 (1223 July 1957), pp. 539–560. DOI: 10.1098/rspa.1957.0106.

[86] N.M. Hugenholtz. "Perturbation theory of large quantum systems". In: *Physica* 23.1 (1957), pp. 481–532. DOI: https://doi.org/10.1016/S0031-8914(57)92950-6.

[87] F. Coester. "Bound states of a many-particle system". In: *Nuclear Physics* 7 (1958), pp. 421–424. DOI: https://doi.org/10.1016/0029-5582(58)90280-3.

[88] F. Coester and H. Kümmel. "Short-range correlations in nuclear wave functions". In: *Nuclear Physics* 17 (1960), pp. 477–485. DOI: https://doi.org/10.1016/0029-5582(60)90140-1.

[89] Andrew G. Taube and Rodney J. Bartlett. "Improving upon CCSD(T): Λ CCSD(T). I. Potential energy surfaces". In: *The Journal of Chemical Physics* 128.4 (Jan. 2008). 044110. DOI: 10.1063/1.2830236.

[90] Stanisław A. Kucharski and Rodney J. Bartlett. "Noniterative energy corrections through fifth-order to the coupled cluster singles and doubles method". In: *The Journal of Chemical Physics* 108.13 (Apr. 1998), pp. 5243–5254. DOI: 10.1063/1.475961.

[91] Rodney J. Bartlett et al. "Non-iterative fifth-order triple and quadruple excitation energy corrections in correlated methods". In: *Chemical Physics Letters* 165.6 (1990), pp. 513–522. DOI: https://doi.org/10.1016/0009-2614(90)87031-L.

[92] Krishnan Raghavachari et al. "A fifth-order perturbation comparison of electron correlation theories". In: *Chemical Physics Letterss* 157.6 (1989), pp. 479–483. DOI: https://doi.org/10.1016/S0009-2614(89)87395-6.

[93] W. Kutzelnigg. "Error analysis and improvements of coupled-cluster theory." In: *Theoret. Chim. Acta* 80 (May 1991), pp. 349–386. DOI: 10.1007/BF01117418.

[94] Yuan He, Zhi He, and Dieter Cremer. "Comparison of CCSDT-n methods with coupled-cluster theory with single and double excitations and coupled-cluster theory with single, double, and triple excitations in terms of many-body perturbation theory - What is the most effective triple-excitation method?" In: *Theoretical Chemistry Accounts: Theory, Computation, and Modeling (Theoretica Chimica Acta)* 105 (Jan. 2001), pp. 182–196. DOI: 10.1007/s002140000196.

[95] Krishnan Raghavachari et al. "A fifth-order perturbation comparison of electron correlation theories (Reprinted from Chemical Physics Letterss)". In: *Chemical Physics Letterss* 589 (Dec. 2013), pp. 37–40. DOI: 10.1016/j.cplett.2013.08.064.

[96] Miroslav Urban et al. "Towards a full CCSDT model for electron correlation". In: *The Journal of Chemical Physics* 83 (Oct. 1985), pp. 4041–4046. DOI: 10.1063/1.449067.

[97] Audun Skau Hansen. "Coupled cluster studies of infinite systems". Master's. University of

Oslo, 2015.

[98] Katuro Sawada. "Correlation Energy of an Electron Gas at High Density". In: *Physical Review* 106 (2 Apr. 1957), pp. 372–383. DOI: 10.1103/PhysRev.106.372.

[99] Yu-Liang Liu. "The intrinsical character of the electronic correlation in an electron gas". In: *Physics Letterss A* 383.12 (2019), pp. 1336–1340. DOI: https://doi.org/10.1016/j.physleta.2019.01.029.

[100] SHANG-kENG MA and KEITH A. BRUECKNER. "Correlation Energy of an Electron Gas with a Slowly Varying High Density". In: *Physical Review* 165 (1 Jan. 1968), pp. 18–31. DOI: 10.1103/PhysRev.165.18.

[101] W. J. Carr and A. A. Maradudin. "Ground-State Energy of a High-Density Electron Gas". In: *Physical Review* 133 (2A Jan. 1964), A371–A374. DOI: 10.1103/PhysRev.133.A371.

[102] J.P. Blaizot. "Nuclear compressibilities". In: *Physics Reports* 64.4 (1980), pp. 171–248. DOI: https://doi.org/10.1016/0370-1573(80)90001-0.

[103] L Satpathy. "Infinite nuclear matter based for mass of atomic nuclei". In: *Journal of Physics G: Nuclear Physics* 13.6 (June 1987), p. 761. DOI: 10.1088/0305-4616/13/6/009.

[104] K. Hebeler and A. Schwenk. "Chiral three-nucleon forces and neutron matter". In: *Physical Review C* 82 (1 July 2010), p. 014314. DOI: 10.1103/PhysRevC.82.014314.

[105] J. Carlson et al. "Quantum Monte Carlo calculations of neutron matter". In: *Physical Review C* 68 (2 Aug. 2003), p. 025802. DOI: 10.1103/PhysRevC.68.025802.

[106] Alexandros Gezerlis and J. Carlson. "Low-density neutron matter". In: *Physical Review C* 81 (2 Feb. 2010), p. 025803. DOI: 10.1103/PhysRevC.81.025803.

[107] M Baldo and G F Burgio. "Properties of the nuclear medium". In: *Reports on Progress in Physics* 75.2 (Jan. 2012), p. 026301. DOI: 10.1088/0034-4885/75/2/026301.

[108] Arianna Carbone, Artur Polls, and Arnau Rios. "Symmetric nuclear matter with chiral three-nucleon forces in the self-consistent Green's functions approach". In: *Physical Review C* 88 (4 Oct. 2013), p. 044302. DOI: 10.1103/PhysRevC.88.044302.

[109] D.R. Thompson, M. Lemere, and Y.C. Tang. "Systematic investigation of scattering problems with the resonating-group method". In: *Nuclear Physics A* 286.1 (1977), pp. 53–66. DOI: https://doi.org/10.1016/0375-9474(77)90007-0.

[110] S. K. Bogner et al. "Testing the density matrix expansion against ab initio calculations of trapped neutron drops". In: *Physical Review C* 84 (4 Oct. 2011), p. 044306. DOI: 10.1103/PhysRevC.84.044306.

[111] B. Friman and A. Schwenk. “Three-Body Interactions in Fermi Systems”. In: *From Nuclei to Stars*, pp. 141–156. DOI: 10.1142/9789814329880_0006.

[112] J. W. Holt, N. Kaiser, and W. Weise. “Density-dependent effective nucleon-nucleon interaction from chiral three-nucleon forces”. In: *Physical Review C* 81 (2 Feb. 2010), p. 024002. DOI: 10.1103/PhysRevC.81.024002.

[113] K. Hebeler et al. “Improved nuclear matter calculations from chiral low-momentum interactions”. In: *Physical Review C* 83 (3 Mar. 2011), p. 031301. DOI: 10.1103/PhysRevC.83.031301.

[114] Hans-Werner Hammer, Andreas Nogga, and Achim Schwenk. “Colloquium: Three-body forces: From cold atoms to nuclei”. In: *Review of Modern Physics* 85 (1 Jan. 2013), pp. 197–217. DOI: 10.1103/RevModPhys.85.197.

[115] W.N. Polyzou and W. Glöckle. “Three-body interactions and on-shell equivalent two-body interactions”. In: *Few-Body System* 9 (Feb. 1990), pp. 97–21. DOI: 10.1007/BF01091701.

[116] Jun-ichi Fujita and Hironari Miyazawa. “Pion Theory of Three-Body Forces”. In: *Progress of Theoretical Physics* 17.3 (Mar. 1957), pp. 360–365. DOI: 10.1143/PTP.17.360.

[117] A. Ekström et al. “Optimized Chiral Nucleon-Nucleon Interaction at Next-to-Next-to-Leading Order”. In: *Physical Review Letters* 110 (19 May 2013), p. 192502. DOI: 10.1103/PhysRevLett.110.192502.

[118] G Hagen et al. “Emergent properties of nuclei from ab initio coupled-cluster calculations*”. In: *Physica Scripta* 91.6 (May 2016), p. 063006. DOI: 10.1088/0031-8949/91/6/063006.

[119] G Hagen et al. “Emergent properties of nuclei from ab initio coupled-cluster calculations*”. In: *Physica Scripta* 91.6 (May 2016), p. 063006. DOI: 10.1088/0031-8949/91/6/063006.

[120] Sven Binder et al. “Ab initio path to heavy nuclei”. In: *Physics Letterss B* 736 (2014), pp. 119–123. DOI: https://doi.org/10.1016/j.physletb.2014.07.010.

[121] R. Navarro Pérez, J. E. Amaro, and E. Ruiz Arriola. “Low-energy chiral two-pion exchange potential with statistical uncertainties”. In: *Physical Review C* 91 (5 May 2015), p. 054002. DOI: 10.1103/PhysRevC.91.054002.

[122] R. J. Furnstahl et al. “Quantifying truncation errors in effective field theory”. In: *Physical Review C* 92 (2 Aug. 2015), p. 024005. DOI: 10.1103/PhysRevC.92.024005.

[123] Michael I. Haftel and Frank Tabakin. “Nuclear saturation and the smoothness of nucleon-nucleon potentials”. In: *Nuclear Physics A* 158.1 (1970), pp. 1–42. DOI: https://doi.org/10.1016/0375-9474(70)90047-3.

[124] Petr Navrátil et al. “Unified ab initio approaches to nuclear structure and reactions”. In: *Physica Scripta* 91.5 (Apr. 2016), p. 053002. DOI: 10.1088/0031-8949/91/5/053002.

[125] Sven Binder et al. “Extension of coupled-cluster theory with a noniterative treatment of connected triply excited clusters to three-body Hamiltonians”. In: *Physical Review C* 88 (5 Nov. 2013), p. 054319. DOI: 10.1103/PhysRevC.88.054319.

[126] H. Hergert et al. “The In-Medium Similarity Renormalization Group: A novel ab initio method for nuclei”. In: *Physics Reports* 621 (2016). Memorial Volume in Honor of Gerald E. Brown, pp. 165–222. DOI: https://doi.org/10.1016/j.physrep.2015.12.007.

[127] D. R. Entem and R. Machleidt. “Accurate charge-dependent nucleon-nucleon potential at fourth order of chiral perturbation theory”. In: *Physical Review C* 68 (4 Oct. 2003), p. 041001. DOI: 10.1103/PhysRevC.68.041001.

[128] G. Hagen et al. “Medium-Mass Nuclei from Chiral Nucleon-Nucleon Interactions”. In: *Physical Review Letters* 101 (9 Aug. 2008), p. 092502. DOI: 10.1103/PhysRevLett.101.092502.

[129] P. Navrátil, G. P. Kamuntavi čius, and B. R. Barrett. “Few-nucleon systems in a translationally invariant harmonic oscillator basis”. In: *Physical Review C* 61 (4 Mar. 2000), p. 044001. DOI: 10.1103/PhysRevC.61.044001.

[130] I. Angeli and K.P. Marinova. “Table of experimental nuclear ground state charge radii: An update”. In: *Atomic Data and Nuclear Data Tables* 99.1 (2013), pp. 69–95. DOI: https://doi.org/10.1016/j.adt.2011.12.006.

[131] H. Kümmel, K.H. Lührmann, and J.G. Zabolitzky. “Many-fermion theory in expS- (or coupled cluster) form”. In: *Physics Reports* 36.1 (1978), pp. 1–63. DOI: https://doi.org/10.1016/0370-1573(78)90081-9.

[132] B. D. DAY. “Elements of the Brueckner-Goldstone Theory of Nuclear Matter”. In: *Review of Modern Physics* 39 (4 Oct. 1967), pp. 719–744. DOI: 10.1103/RevModPhys.39.719.

[133] K. A. Brueckner. “Many-Body Problem for Strongly Interacting Particles. II. Linked Cluster Expansion”. In: *Physical Review* 100 (1 Oct. 1955), pp. 36–45. DOI: 10.1103/PhysRev.100.36.

[134] K. A. Brueckner, C. A. Levinson, and H. M. Mahmoud. “Two-Body Forces and Nuclear Saturation. I. Central Forces”. In: *Physical Review* 95 (1 July 1954), pp. 217–228. DOI: 10.1103/PhysRev.95.217.

[135] A. Gezerlis et al. “Quantum Monte Carlo Calculations with Chiral Effective Field Theory Interactions”. In: *Physical Review Letters* 111 (3 July 2013), p. 032501. DOI: 10.1103/PhysRevLett.111.032501.

[136] J. Carlson et al. “Quantum Monte Carlo methods for nuclear physics”. In: *Review of Modern Physics* 87 (3 Sept. 2015), pp. 1067–1118. DOI: 10.1103/RevModPhys.87.1067.

[137] V. Somà et al. “Chiral two- and three-nucleon forces along medium-mass isotope chains”. In: *Physical Review C* 89 (6 June 2014), p. 061301. DOI: 10.1103/PhysRevC.89.061301.

[138] W.H. Dickhoff and C. Barbieri. “Self-consistent Green’s function method for nuclei and nuclear matter”. In: *Progress in Particle and Nuclear Physics* 52.2 (2004), pp. 377–496. DOI: https://doi.org/10.1016/j.ppnp.2004.02.038.

[139] T. D. Morris, N. M. Parzuchowski, and S. K. Bogner. “Magnus expansion and in-medium similarity renormalization group”. In: *Physical Review C* 92 (3 Sept. 2015), p. 034331. DOI: 10.1103/PhysRevC.92.034331.

[140] J. Clark et al. “Statistical Modeling of Nuclear Systematics”. In: (Oct. 2001). DOI: 10.1142/9789812811127_0008.

[141] N. J. Costiris et al. “Decoding β-decay systematics: A global statistical model for β^- half-lives”. In: *Physical Review C* 80 (4 Oct. 2009), p. 044332. DOI: 10.1103/PhysRevC.80.044332.

[142] S Akkoyun et al. “An artificial neural network application on nuclear charge radii”. In: *Journal of Physics G: Nuclear and Particle Physics* 40.5 (Mar. 2013), p. 055106. DOI: 10.1088/0954-3899/40/5/055106.

[143] Kevin P. Murphy. *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.]: MIT Press, 2013.

[144] Wessel N. van Wieringen. “Lecture notes on ridge regression”. In: *arXiv: Methodology* (2015).

[145] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006, pp. I–XVIII, 1–248.

[146] David J. C. MacKay. “Bayesian Interpolation”. In: *Neural Computation* 4.3 (May 1992), pp. 415–447. DOI: 10.1162/neco.1992.4.3.415.

[147] Michael E. Tipping. “Sparse Bayesian Learning and the Relevance Vector Machine”. In: *J. Mach. Learn. Res.* 1 (Sept. 2001), pp. 211–244. DOI: 10.1162/15324430152748236.

[148] A.J. Kallio, E. Pajanne, and R.F. Bishop. *Recent Progress in Many-body Theories*. Recent progress in many-body theories no. 1. Plenum, 1987.