Stochastic Self-Energy in a Self-Consistent Second-Order Green's Function Scheme

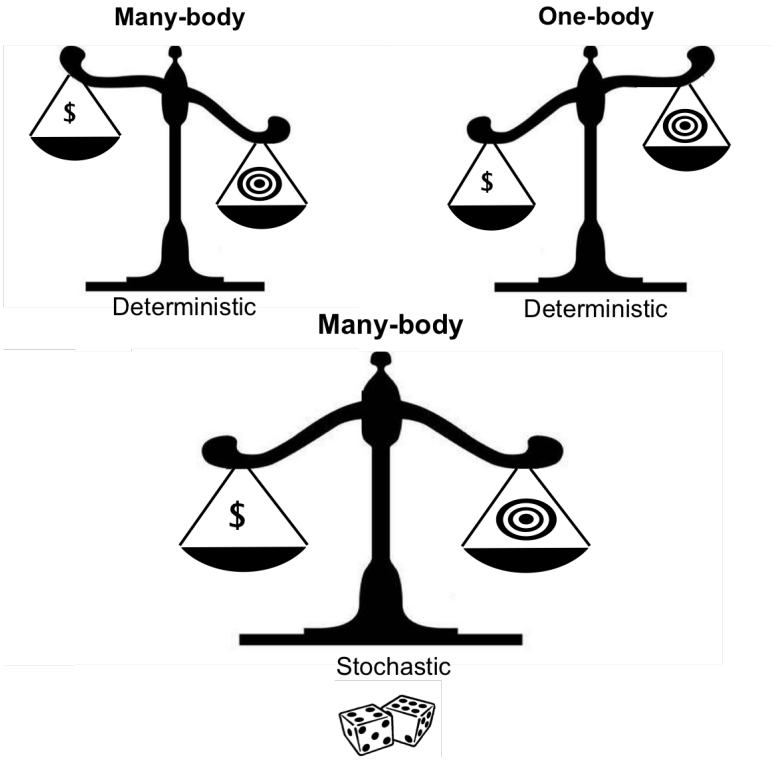
by

Blair Winograd

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Chemistry)
in the University of Michigan
2020

Doctoral Committee:

Associate Professor Dominika Zgid, Chair
Professor Eitan Geva
Associate Professor Emmanouil Kioupakis
Professor Adam Matzger


Many-body
One-body
$
$
Deterministic
Deterministic
Many-body
$
Stochastic

Blair Winograd
bawinogr@umich.edu
ORCID iD: 0000-0001-8616-9082

DEDICATION

I would like to dedicate this thesis to my son, Mason, and daughter, Brooklynn.

ACKNOWLEDGMENTS

In completing this thesis and developing my skills as a scientist, I am deeply indebted to many important people I have met along the way. Not only have my colleagues and faculty been important supporters of my scientific development, but also the Graduate and Chemistry program at the University of Michigan. This program has provided me with the essential tools required to complete my Graduate studies and find success, and I am beyond elated that I was accepted and chose to come to this program.

While many people have shaped my experience and progression as a researcher, I would like to acknowledge just a few of the people who helped me through this experience. First, I would like to thank my advisor Professor Dominika Zgid. I am unbelievable lucky to have had her as my PhD advisor with her incredible knowledge, wisdom, sense of humor, and kindness. While I did not come to Michigan with the plan to join her group (or even study theoretical chemistry), she made the journey worth it. Through working for her I developed a computational and mathematical skill-set I never imagined I could obtain. Before working for her, I did not know I was capable of learning to code or implement the numerical analysis required of theoretical physics/chemistry. Yet, even though I questioned myself she

never once questioned my ability and only empowered me to persist. She also taught me how to attack new research problems in an efficient manner. At the same time, I learned the power that comes with leading with compassion and patience.

Additionally, I must thank Professor Eitan Geva. Initially rotating in his group opened my eyes to theoretical chemistry. He allowed to me to pursue chemistry education research which helped me to migrate each of my interests in a way I didn't know was possible. Thank you to my group members, Alexei, Alex, and Avijit who were always willing to help. Thank you to the group's newer members, Diksha and Yanbing, who helped me gain confidence in my understanding of quantum mechanics. Without our weekly meetings, I am not sure where my project would have gone. And especially, thank you to Alicia for putting endless hours into helping me master the art of debugging when she, herself, was overloaded in her own work. She has also become a dear lifelong friend and was an incredible support system throughout my PhD.

Finally, I must thank my two children and my partner Eyal. Eyal always encouraged me to complete my studies, and Mason helped ground me after stressful days of failing research. Also, thank you to my daughter Brooklynn whose arrival weeks before my defense provided endless motivation. Lastly, but most importantly, I want to acknowledge my parents for always pushing me to further my education and showing pride in my accomplishments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The description of electron correlation has been a critical problem among theoretical and computational chemistry researchers. To describe the physics of many new materials, this interaction is crucial. Typically, chemists and physicists have constructed approximations to classic electronic structure methods - wavefunction methods and density functional theory - to attempt to solve this problem. Both have had many successes in the computational chemistry field, but are hindered by either their computational cost or ability to rigorously describe the electron correlation. More recently, researchers have come back to investigating how Green's functions can be of use to study this correlation. This thesis focuses on the second-order Green's function which has shown successes in its moderate computational cost and ability to describe electron correlation. However, in solid-state chemistry where the systems to be studied are quite large, the method still must be implemented by other means. Instead of exploring this method deterministically, this thesis moves towards investigating the method via a stochastic method. While stochastic methods such as diagrammatic Monte Carlo are quite common in the physics community, they are far lesser explored among theoretical chemists.

This work attempts to develop a computationally feasible way to implement diagrammatic Monte Carlo within the second-order Green's function scheme. This method is preferred to be implemented in a fully self-consistent matter. Thus, chapter 4 of this work will explore the algorithmics required to sample for the second-order self-energy component of the method. Further in chapter 5, the correct statistical exploration required to obtain important expectation values such as the one-body and two-body energies are

discussed. It is found that due to the non-linearity of the data, non-parametric statistical resampling methods are required. In this work, it is shown that via a jackknife algorithm built into the second-order Green's function scheme, the stochastic error from a Monte Carlo evaluation of the self-energy can be controlled. To show the power of this analysis, in chapter 6 self-consistency is shown to be possible via calculations of a few model systems as well as larger more realistic chemical systems.

Finally, chapter 7 of this thesis segues into a problem within the chemistry community. Coding skills are at the core of developing the methods described in this thesis; however, coding is not required in the chemistry curriculum in the United States. This chapter describes a curriculum that encouraged chemistry students to develop coding skills in a low stakes environment. While the overall hypothesis going into the study was that students who use coding to study a quantitative problem in chemistry will increase their understanding of that problem, the study left with further results. Via surveys, it was discovered that many chemistry students desire to learn coding and feel that developing the skill positively impacts their studies and future goals.

Overall, this thesis has made great progress in developing a diagrammatic Monte Carlo technique that could be of interest to the chemistry community. The method has proven to be computationally feasibly and quantitatively correct in comparison to the analogous deterministic method. This work has formulated a scheme that can provide an accessible approach to solving other Green's function methods of interest and hopefully bring us closer to finding a method that is computationally approachable to quantitatively describing electronic correlation.

Part I:

Introduction

CHAPTER 1

Introduction

Materials research is becoming increasingly relevant and diverse. Researchers are looking for materials that will pave the way for new technologies or new sources of energy. High-quality and uniquely featured materials, such as the batteries that power our phones, or the semiconductor screens on our tablets, are currently needed for our everyday use. Discovery and analysis of these materials have mostly come from tedious synthetic research. Since the 1960s, theoretical and computational chemistry have been on the rise for the study of molecular systems. This particular field of quantum chemistry research focuses its work on the motion of electrons. The behavior of the movement of electrons in relation to the other electrons in materials, also known as electron correlation, cannot be calculated exactly due to the immense computational size of the problem. Even so, at the molecular level, the methods that have been developed have been fine-tuned and well examined to give informative results for many physical properties. Much of this research has been directly used by synthetic chemists to streamline their research and development process. However, the lack of computational power has created a threshold for the size of the systems that could be examined. Many of the materials that we as a society are interested in are large periodic solids. The methods that were largely developed for the study of molecular systems are not suitable for this type of calculation. In the case of materials development, new physical problems arise even beyond a solid's cumbersome system size.

When studying a solid, two major aspects of physics interplay. One is the interactions such as exchange interactions between spins. The other aspect is band theory which describes that the

states of electrons in solid materials can only have values of energy within specific ranges. An electron is correlated to all the other particles around it; in a solid, this issue becomes increasingly important to analyze along with both the interactions and band states. For example, the Mott insulator is an important type of material that is notoriously difficult to simulate the properties of. Band theory would predict an insulating state when all bands are occupied or empty, yet a metallic state occurs under certain conditions. In many cases, these conditions for a metallic state can be dominated by a Coulomb interaction. Highly sought after materials such as transition metal oxides are often Mott insulators. Choosing or developing the right theory that can describe the physics of these systems accurately as well as simulate the material within a feasible computational cost is difficult. As of recent, high performance computing (HPC) power has grown at a spectacular pace. While the physics of this problem is still not exactly solvable, the growth in computational resources has made the study of electron correlation of large chemical systems more feasible and of great current interest. Via theoretical chemistry, mathematical simulation and HPC, there are many paths to follow in the pursuit of accurate calculations of the electron correlation in solids. One of the most common paths taken is Density Functional Theory (DFT) because of its computational tractability. However, DFT typically fails in the case where electron correlation effects are important. For example, DFT alone cannot reliably capture the physics of the Mott insulator.[3,4,5] In light of this, the following work will consider another path known to capture the relevant physics of such difficult problems - the second order Matsubara Green's function (GF2).

As introduced, this field of electronic structure and studying electron correlation is important academically as well as industrially in that computational discoveries have the ability to help guide experimentalists. Computational discoveries demand in depth computer skills, and in many cases, the ability to code. The chemistry curriculums at most United States universities do not include a component on learning to code. Yet, The majority of our research level chemists have been trained at these institutions. Across many sub-disciplines in chemistry, coding skills are an important tool to have in a chemist's arsenal. In many cases, coding skills may help an analytic chemist analyze their data. Coding skills could come in handy for a biochemist handling sequencing data. Further-

more, students who pursue theoretical chemistry will typically need to code either small scripts for efficiency or even new theoretical models. Additionally, in the process of learning chemistry, there is a huge and intricate area of quantitative modeling. Many students struggle when learning these concepts. It is interesting to consider whether a student who programs these models, as opposed to simply reading about them, may gain further insight and confidence in using chemistry models. This thesis further includes an introductory advance in research that combines coding into the chemistry curriculum at the university level. Initial outcomes are given, and further directions in which this research should be pursued in will be discussed.

The following chapters will discuss this new method in electronic structure as well as a chemistry education process crucial to making our future chemists more proficient. Chapter two gives a general introduction to the well-known and a few lesser-developed methods in electronic structure theory. Chapter three overviews the Monte Carlo integration method and how it can be useful in computationally intensive electronic structure calculations. Chapter four suggests a new stochastic algorithm to give insight into electronic structure problems. Chapter five considers how to control the error in the unique stochastic algorithm and the validation needed to execute the methods described in chapter four. Chapter six considers the quality and benefits of this type of method, as well as summarizes the preliminary results obtained from the new stochastic method. Chapter seven segues into giving an overview as to how coding can be introduced into the chemistry curriculum, and the obvious benefit it has on students. Chapter eight summarizes the outcomes, discusses future directions, and concludes this thesis.

CHAPTER 2

Electronic Structure Background

How does science describe a molecule?

The composition of a molecule is the collection of two or more interacting atoms of the same or different elements. These elements exist as a collection of two types of charged particles: their positive nuclei and their negative electrons. The interaction of two or more particles is described by the potential energy,

$$V_{ij} = V(r_{ij}) = \frac{q_i q_j}{r_{ij}}.$$

This interaction between particles, V_{ij}, the attraction or repulsion, is the only physical (aka experimentally measurable) quantity required to describe chemistry.

The potential energy only describes a snapshot in time. In reality, the particles that make up a molecule are in constant motion. Moreover, there are also interactions ”forgotten” by this term that must be accounted for when studying the properties of a molecule. When more than one electron is present, the full description of these interactions is already impossible to solve analytically.

Now consider a solid material. In general, solid materials can be thought of as a regular or irregular repeating units of a molecule (crystal). These systems, in most cases, can contain 10^{23} electrons per cm^3. If a molecule is already impossible to solve analytically, or without a computer, it should be apparent that a regularly repeating molecule proves to be an even more daunting task. This thesis discusses one route that may bring us closer to describing the electronic structure, including the physical and non-physical interactions of electrons, of solid materials.

2.1 Electron Correlation

In classical mechanics, how a system and its particles evolve in time is described by Newton's second law. In quantum mechanics, the behavior of particles are typically described in terms of a wave-function and can be solved via the time-dependent Schrödinger Equation.

$$\hat{H}\Psi(r,t) = ih\frac{\delta\Psi(r,t)}{\delta t}.$$

If $\hat{\mathcal{H}}$, the hamiltonian operator, is time-independent, the time-independent Schrödinger equation can be written as follows.

$$\hat{H}\Psi(r) = E\Psi(r).$$

The time-independent Schrödinger equation describes the particle-wave duality of electrons. Taking a system or molecule with N-particles, the Hamiltonian must contain a sum consisting of the kinetic ($\hat{T}$) and potential ($\hat{V}$) energy:

$$\hat{H} = \hat{T} + \hat{V}.$$

By describing the Hamiltonian this way, the dynamics of all electrons and nuclei in the system should be taken into account. The Hamiltonian is then further simplified via the Born-Oppenheimer approximation which states that because nuclei are much heavier than electrons, the velocity of the nuclei is negligible in comparison. The Schrödinger equation can then be split into two parts where one part describes the electronic wavefunction for the fixed nuclear geometry and the second part describes the nuclear wavefunction where the electronic energy is the potential energy.

Unfortunately, even for small molecules, the Schrödinger equation cannot be solved analytically in its exact form. Instead, the methods to solve the time-independent Schrödinger equation must make approximations. One of the most basic and widely known methods is the Hartree-

Fock (HF) approximation. The HF method poses that each electron's motion in a system can be described by an orbital (a single-particle function). This orbital does not explicitly depend on any instantaneous motions of the other electrons. It effectively replaces the Schrödinger equation by a set of these orbitals, in which each electron moves in an effective field (typically known as the mean-field). The mean field is approximated to be the same for all electrons. This replaces the many body Schrödinger equation into many one-electron problems. The computational effort formally scales as M^4 (where M is the number of basis functions).

In the study of real applications, where the systems are highly correlated or not near equilibrium geometry, it becomes clear that the HF approximation does not adequately describe electron dynamics. Where HF falls short is the correlation energy. The correlation energy is defined as,

$$E_{corr} = E_{exact} - E_{HF}$$

Again, as alluded to in the introduction, in the Hartree-Fock approximation electrons interactions are not treated explicitly, and electrons are then allowed to occupy orbitals in formations that are physically too close. Thus, because the Coulomb correlation is not taken into account, a part of the electron repulsion is completely forgotten. This interaction that is missing from the Hartree-Fock description of the energy is the correlation energy. One of the most important aspects of research in the field of electronic structure theory is finding a computationally feasible method to calculate the electron correlation. The next section will explore some of those available methods.

2.2 Common Methods in Modern Electronic Structure

As computational power increased, the opportunity to calculate the many body Schrödinger equation at a more exact level than HF became enticing. Many methods have been developed to more accurately describe the correlation energy. In most cases, HF is used as the starting point for these correlated methods.

In Hartree-Fock, it is clear that using a single slater determinant does not fully describe that

Cost	Method
N^4	HF
N^5	MP2
N^6	MP3, CISD, MP4SDQ CCSD, QCISD
N^7	MP4, CCSD(T) QCISD(T)
N^8	MP5, CISDT, CCSDT
N^9	MP6
N^{10}	MP7, CISDTQ, CCSDTQ

Table 2.1: This table gives a summary of many of the popularly explored wave function methods in theoretical chemistry. The formal computational cost of each method is given.

the electrons can actually occupy excited states, because the ground and excited states are only approximate. One method that can completely account for these excitations is the full configuration interaction (FCI). This method is built off of an N-electron wavefunction made up of a linear combination of slater determinants describing the ground state, all single, double, etc, excitations possible.

Some of the strengths of FCI are that it can apply in situations where many other methods fail. For example, FCI applies to excited states, to open-shell systems, and to systems far from their equilibrium geometries. However, it comes at a steep cost. For a chemical system with a finite number of orbitals (K) in its basis set, and N electrons, the number of possible configurations to be described in the wavefunction are:

$$\frac{(2K)!}{N!(2K-N)!}$$

Solving the full wavefunction becomes an intractable problem for anything more than the smallest molecules. However, if a complete basis set is used and we had the computational power, the resulting energy, E_{CI}, subtracted by the Hartree-Fock energy would give us the exact correlation energy. Thus, in reality, to calculate information about the correlation energy of chemical systems, one has to make a trade-off between accuracy and computation time/cost.

Many other wavefunction-based approximations beyond the level of Hartree-Fock can be made. Some of the most common wavefunction methods used are second-order Møller-Plesset Perturba-

tion Theory (MP2) and coupled cluster. However, MP2 fails for many systems in their stretched regime, and coupled cluster quickly becomes far too expensive. Beyond wavefunction methods, other routes have been explored. The most ubiquitous of those paths is Density Functional Theory (DFT). While the theories so far described work with the many body wavefunction to solve the electronic Schrödinger equation, these methods are dealt the unfortunate card of managing the $3N$ coordinates contained in the wavefunction. If available, it would be useful to instead work with an object that only depends on one main variable. In DFT, this is exactly what is done. The many body problem is essentially mapped to a single-body problem where the key variable is the electronic density:

$$n(\overrightarrow{r}) = N \int d^3\overrightarrow{r_2} ... \int d^3\overrightarrow{r_N} \Psi^*(\overrightarrow{r}, \overrightarrow{r_2}, ..., \overrightarrow{r_N}) \Psi(\overrightarrow{r}, \overrightarrow{r_2}, ..., \overrightarrow{r_N})$$

A material with N electrons can be studied as a combined set of N one-electron equations that look similar to Schrödinger equations: these are known as the Kohn-Sham equations.

The method further depends on a theorem that defines an energy functional for the system and proves that the correct ground state electron density minimizes this energy functional. A dominating problem with DFT is that the exact functionals that describe both the exchange and correlation for any material is not known. While many rigorous approximations have been made and left us with what feels like an endless list of functionals, choosing a functional to best describe the physics of the material being calculated is crucial. Yet, this work pursues development of new materials. If a material is not yet in existence, knowledge of the physical properties are unknown. In this case, a scientist must guess the "correct" functional which is clearly impractical. The following section gives insight into a lesser-known and far lesser-studied non-empirical method which has shown promise in giving insight to the important physical properties of a solid at a practical computational cost.

2.3 The Green's Function and Self-Energy

Some very well understood methods to solve the Schrödinger equation have been discussed. These methods fall into two categories: the wavefunction and the one-electron density. While these methods have several uses, they also suffer from several shortcomings. Yet another category of method, one perhaps less explored can also be used, this is the Green's function.

As discussed, the many-particle system is complex; the detailed form of its energy spectrum and wavefunctions is neither exactly calculable nor measurable. In most experimental measurements involving a many-particle system, a system in equilibrium is weakly perturbed by either a particle being added or removed, by applying a weak electromagnetic field, or by a beam of electrons or neutrons. It can be rather useful to try to understand how a system responds to such external perturbations. The Green's function method is an excellent mathematical tool to do this.

The Green's function formalism has many unique aspects that make it a useful method to study. For example, in certain formalisms the Green's function can be temperature dependent. Additionally, it is directly related to experimentally measured quantities. It should be noted that the Green's function method is widely applicable. It can be applied to systems at 0K, finite temperature, systems in equilibrium, systems out of equilibrium, time-dependent cases and time-independent cases. There are many excellent textbooks that outline this formalism in detail.[6,7] In the remaining sections of this chapter, I will briefly discuss the parts of the Green's function method that are relevant to this work. Specifically, I will focus on the imaginary time Green's function formalism.

2.4 Green's function and Inhomogeneous Differential Equations

Green's functions are named after the British mathematical physicist, George Green, who wrote "An Essay on the Application of Mathematical Analysis to the Theories of Electricity and Magnetism." Through the mathematical lens, the Green's function describes the impulse response of an inhomogenous differential equation. The Green's function is a method to solve inhomogenous

differential equations such as:

$$[z - L(r)]u(r) = f(r),$$

where the solution is given as,

$$u(r) = \int dr' G(r, r'; z) f(r').$$

In mathematics, the Green's function is the impulse response. In fact, in many fields, Green's functions are actually called response functions. A complete mathematical description of the Green's function formalism as well as its basic properties in both time-independent and time-dependent cases is given in "Green's Functions in Quantum Physics".[8]

2.5 Green's function and Quantum Mechanics

In quantum mechanics, the time independent Schrödinger equation again can be written as:

$$\hat{H}\psi = E\psi$$

Or equivalently,

$$[E - \hat{H}(r)]\psi(r) = 0$$

Here $\hat{H}$ is the Hamiltonian operator and E denotes the discrete eigenvalues of the Hamiltonian. The mathematical formalism introduced in section 2.4 can be extended to solving the Schrödinger equation as follows:

$$[E_H(r)]G(r, r'; E) = \delta(r - r')$$

Furthermore, this can be extended to the time dependent Schrödinger equation as:

$$\left(ih\frac{\partial}{\partial t} - \hat{H}\right)|\psi(t)\rangle = 0$$

Defining the time evolution operator as,

$$\hat{U}(t - t_0) = e^{-i(t-t_0)\hat{H}/h}$$

the Green's function for the operator can be defined as:

$$\hat{U}(t - t_0) = ihG(t - t_0)$$

The time-dependent Schrödinger equation in terms of the time evolution operator takes the following form,

$$|\psi(t)\rangle = \hat{U}(t - t_0)\,|t_0\rangle\,.$$

This equation can then be rewritten in the r, or position, representation to obtain the equation:

$$\psi(r, t) = \psi_0(r, t_0) + ih\int G(r, r', t - t_0)\psi(r', t_0)dr'$$

It can be seen that from the quantum-mechanical view, the Green's function is a type of propagator in that it shows that a particle is being propagated from a position r' and time t_0 to a later position r and time t. To completely solve the entire Green's function is almost as difficult as solving for the wavefunction. Thus, just like in wavefunction methods, approximations are made when trying to solve for Green's functions of practical chemical systems. In many cases, one begins from the non-interacting Green's function and applies perturbation theory to calculate the full, or some approximation of, Green's function for the interacting case. The following section, 2.6, will give background into how using perturbation theory within the Green's function formalism can bring

us closer to accessibly modeling electron correlation.

2.6 Green's Function Perturbation Theory and the Self-Energy

Going back and reconsidering the eigensolutions of the time-dependent Schrödinger equation,

$$\left(ih\frac{\partial}{\partial t} - \hat{H}\right)|\psi(t)\rangle = 0,$$

we can also rewrite the time-dependent Schrödinger equation including a potential, $\hat{V}$, that will be considered as the "perturbation".

$$\left(ih\frac{\partial}{\partial t} - \hat{H} - \hat{V}\right)|\psi(t)\rangle = 0$$

The corresponding Green's function will be:

$$\psi(r,t) = \psi_0(r,t_0) + ih\int G_0(r,r',t-t_0)\hat{V}(r')\psi(r',t_0)dr'$$

For convenience, dropping the integration variables the following expansion can be seen:

$$\psi(r,t) = \psi_0 + G_0\hat{V}(\psi_0 + G_0\hat{V}\psi)$$

$$\psi(r,t) = \psi_0 + G_0\hat{V}\psi_0 + G_0\hat{V}G_0\hat{V}(\psi_0 + G_0\hat{V}\psi)$$

$$\psi(r,t) = \psi_0 + G_0\hat{V}\psi_0 + G_0\hat{V}G_0\hat{V}G_0 + ...)\hat{V}\psi_0$$

This lays a foundation for perturbation themes in terms of successive orders of a scattering potential, $\hat{V}$. From here, one ends up with a formalism known as the Dyson equation:

$$G = G_0 + G_0\hat{V}G$$

or equivalently,

$$G^{-1} = G_0^{-1} - \hat{V}$$

The Dyson equation represents the amplitude of probability of going from (r, t_0) to (r', t) without collision (G_0), with one collision ($G_0\hat{V}G_0$), with two collisions ($G_0\hat{V}G_0\hat{V}G_0$) and so on. When describing an interacting system, the interacting Green's function can be treated as having a series of collisions with a potential called the self-energy. The self-energy represents the interactions between the particle and its system. The system, in this case, may be a material. This means that the self-energy represents the potential felt by the single electron interacting with the other electrons in the material, and this interaction in turn also interacting with the single electron. Here we will use the greek letter, Σ, to denote the self-energy. Thus, the Dyson equation becomes:

$$G^{-1} = G_0^{-1} - \Sigma \tag{2.1}$$

The guiding purpose of this work is to use the self-energy to obtain information about the energy of a system. Thus, I will rewrite the Dyson equation in the form that solves for the self-energy:

$$\Sigma = -G^{-1} + G_0^{-1} \tag{2.2}$$

Specifically, the self-energy we are interested in is at a level of perturbation that includes some form of correlation. The calculations to obtain this information will be further described in the next section. However, it is important to mention that an important aspect of perturbation theory is choosing what level of perturbation to use. In other words, what level of approximation will be made to make this problem soluble. For the purpose of this thesis, the self-energy and corresponding Green's function will always be considered approximated at the level of one collision,

also considered the second-order Green's function:

$$G^2 = G_0\Sigma^2 G_0$$

2.7 Luttinger-Ward Functional

There are several Green's function formalisms. The only one that is discussed in this work is the Matsubara formalism. The Matsubara Green's function is built on an imaginary frequency axis in equilibrium, rather than real time. This formalism is temperature dependent and mathematically attainable. For a system of interacting electrons, it is imperative to know the relationship between static quantities which describe the thermodynamics of a system and dynamic quantities that describe one-particle excitations. For the one-electron many body Green's function, those dynamic quantities are the Green's Function and the self-energy. The Luttinger-Ward functional relates thermodynamic quantities to the Green's function and self-energy via the grand canonical potential, Ω.[9]

$$\Omega = \Phi + \mathrm{Tr}\ln G + \mathrm{Tr}\,\Sigma G$$

This relationship is useful when calculating thermodynamically meaningful quantities.[10] For the case of this work, those quantities of interest are specifically focused on the internal energy of a system. Thus the energy can be calculated as:

$$E = \frac{1}{2}\mathrm{Tr}\Big[(\hat{H} + F)\gamma\Big] + \frac{2}{\beta}\sum Re(\mathrm{Tr}[G\Sigma])$$

where γ is the one-body density matrix, $\hat{H}$ is the one-body Hamiltonian, F is the Fock matrix. More than just a connection to thermodynamics, the Luttinger-Ward functional has many unique properties. Some of those properties are introduced here as they can be useful to algorithm design.

First, the Luttinger-Ward functional is defined as:

$$\Phi = \sum_{m=1}^{\infty} \frac{1}{2m} \mathrm{Tr}\left[\sum_{n} \Sigma^{(m)}(i\omega_n) G(i\omega_n)\right]$$

where m defines the order of the self-energy. The functional, $\Phi[G]$, is just a scalar that is dependent on the Green's function. Additionally, one can think of the Luttinger-Ward functional as a functional of the self-energy, $\Phi[\Sigma]$. The fact that it is a scalar quantity directly relevant to the multidimensional Green's function can be convenient in certain schemes. This will become important and will carefully be reiterated in chapter four of this thesis.

2.8 A Note on Particle Conservation

The assertion to use many body perturbation theory using partial summation of self-energy diagrams has been made. At the second order it has proven efficient as well as convergent.[11] Moreover, the desire to obtain ground-state properties like the total energy has been made. Evaluation in this method circumvents the expense and complicated calculations via wavefunction methods as well as avoids the limiting approximations used in DFT. The many body method of integration to evaluate the total energy has a consequence: the requirement of particle conservation via the grand canonical potential may not be obtained in a single iteration. Therefore, the second-order Green's function method, as well as diagrammatic many body perturbation theory in general, must be solved fully self-consistently to ensure that the particle number and total energy are conserved.[12] While it has been proven that the Green's function obtained self-consistently from the Dyson equation does yield the exact particle number,[1] many formalisms such as G_0W_0 forgo self-consistency due to its expense. In this work, however, I argue self-consistency is imperative to formulating both qualitative and quantitative convergent energy results from the GF2 method. More discussion of the self-consistent procedure will be given in chapter five.

CHAPTER 3

Monte Carlo Integration

3.1 Stochastic vs Deterministic Methods: Monte Carlo Methods

Thus far, I have discussed a series of electronic structure methods. First discussed was FCI, an exact method at the basis set limit. However, the dimensionality of this problem is impossible to solve for any reasonably sized system. Further, I went on to discuss other approximate wavefunction methods. These had their benefits, such as reduced cost, yet they still can be quite cumbersome to solve for systems with many orbitals. Furthermore, I discussed the profound impact DFT has had on the field of electronic structure. While its low computational cost makes calculations of large systems reasonable, its results are often inaccurate or require too many empirical observations. Lastly, I introduced the benefit of Green's function methods - specifically the second-order Green's function (GF2). These methods have a reduced scaling and can give insight into the electronic structure of materials that other wavefunction methods cannot. This exploration of possible methods can be summarized in one way: how do we compromise between lost accuracy by way of approximation and the ever increasing computational cost? When the dimensionality of a problem is too large to approach via analytic methods, and deterministic approaches to integration are too expensive, a method called Monte Carlo can be of use.

Initially, Monte Carlo methods were discovered as useful for creating an estimate of integrals that could not be analytically solved. Many advances have been made in probability theory and

statistics which are now included in the broader field of "Monte Carlo Methods"; however, this thesis only requires an understanding of the corner of this field called Monte Carlo integration.

In a deterministic approach to numerical integration, each attempt to solve the problem provides the same outcome. In a Monte Carlo, or stochastic, approach, each attempt to solve the problem provides a different outcome. In the Monte Carlo approach, the outcome is an approximation to the correct value likely to be within predetermined error bars.

In this chapter, the principal ideas behind quantum Monte Carlo methods are outlined. The theory behind Monte Carlo integration will be presented. As will be described, finding an efficient implementation of Monte Carlo integration for the object to be studied is important. Any inefficiencies increase computational overheads and will create limitations to the size of system and statistical accuracy that may be obtained. A few examples of how to promote efficient calculations are given in this chapter; however, full details of the specific approach used in this thesis and its implementation are presented in the following chapter.

3.2 Monte Carlo Integration

A very simple explanation of Monte Carlo is the application to evaluating a one-dimensional definite integral. For example, take the following one-dimensional integral:

$$E = \int_a^b f(x)dx$$

Different ways to approximate this integral can be made. One method is to apply the mean value theorem. Within the mean value theorem, the integral can be approximated as,

$$E_N = \frac{b-a}{N} \sum_{i=1}^{N} f(x_i)$$

As the number of instances of N increases, the value of E_N is known to evaluate closer and closer to the true value of E. A standard choice for the points x_i would be a completely uniform grid.

More accurate methods of quadrature instead use grids made of a weighted average of the points:

$$E_N = \frac{b-a}{N} \frac{\sum_{i=1}^{N} w_i f(x_i)}{\sum_{i=1}^{N} w_i}$$

The computational cost of evaluating an integral (to a certain determined accuracy) of dimensionality d increases as N^d. However, for many functions actually evaluating the sum of every point may not be necessary. For a system with a very large grid, a more efficient approach is to select points x_i randomly from a given probability distribution. This can be done by a Monte Carlo simulation. If the points are selected truly at random over the interval $[a, b]$, the Monte Carlo estimate of the integral becomes

$$E_N = (b-a)\bar{f} = \frac{b-a}{N} \sum_{i=1} N f(x_i) + O(1/\sqrt{N})$$

with the error behaving as,

$$\sigma_N = \sqrt{\frac{\frac{b-a}{N} \sum_{i=1}^{N} f^2(x_i) - E_N^2}{N-1}}.$$

Assuming a large number of independent samples, the normalized sum should tend towards a normal distribution. Thus, the probability that E is within $E_N \pm \sigma_N$ is ≈ 0.68 and the probability of being with $2\sigma_N$ is ≈ 0.95. This error decays as $1/\sqrt{N}$ independent of the dimensionality of the integral. A great benefit of Monte Carlo simulation is that the behavior of Monte Carlo error is well understood and can be predicted. Statistical knowledge of how the error behaves in Monte Carlo simulations is crucial when developing and validating new algorithms dependent on Monte Carlo sampling. This important aspect of Monte Carlo as it pertains to this thesis will be further discussed in Chapter five.

3.3 Importance Sampling

Simple schemes for Monte Carlo integration can be of huge benefit. However, in some cases, these simple schemes may suffer from low efficiency. For example, if sampling a higher dimensional function, the function of interest may only have significant weight in certain regions. These regions may be considered more "important" than other regions. For example, most of the contributions to an integral of a simple Gaussian are located near the central peak as seen in figure 3.1. If a simple

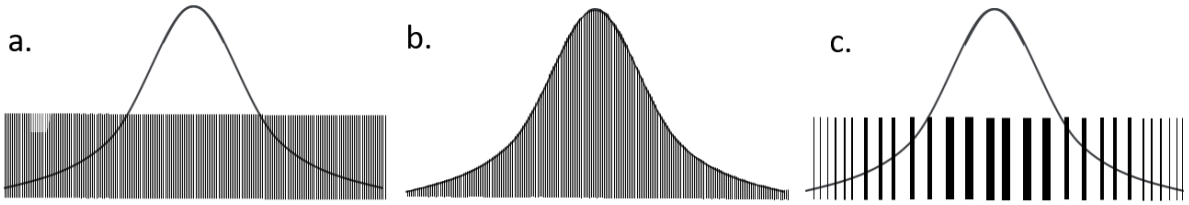


Figure 3.1: Importance sampling of a Gaussian integral would involve the following three steps shown in images a, b, and c. In a sample from a prior belief $q(x)$, for example the distribution is uniform. In b, compute importance weights of different portions of the function $w(x) = \frac{p(x)}{q(x)}$. In c, resample according to importance weights to get $p(x)$. Samples with greater weight should be sampled more times as portrayed by the greater density of the lines.

Monte Carlo integration simulation scheme was used to sample a Gaussian, the points would be sampled uniformly. This means that considerable effort would be wasted by sampling the tails of the Gaussian. The techniques used to overcome this problem attempt to increase the density of points in regions of interest and, therefore, improve the overall efficiency of the simulation. These techniques of choosing points based on weight are called importance sampling.

In an importance sampling scheme, instead of choosing points from a uniform distrubution, points are sampled from a non-uniform distribution $w(x)$. $w(x)$ is chosen to always be positive and chosen to approximate $f(x)$ over the region of interest. This allows the integral to be evaluated by selecting points from the probability distribution $p(x)$,

$$p(x) = \frac{w(x)}{\int_a^b w(x)dx}$$

From here, points x_i are generated from a probability distribution of $p(x)$ and the integral may be evaluated.

$$E = \int_a^b \frac{f(x)}{p(x)} p(x)dx \approx \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}$$

The integration procedure is more efficient than sampling without importance, and the error still decays as $1/\sqrt{N}$.

3.4 Broader Applications

The method of Monte Carlo integration is an extremely useful tool for numerical integration. Its most well-known advantages are its simplicity, its easy adaptation to multi-dimensional integrals, and the fact that it is unbiased. One additional point important for very large calculations is that Monte Carlo integration is consistently parallel in nature: each processor of a parallel computer can be assigned the task of simulating each random trial. The fact that it can be completely general to almost any function, assuming issues such as sign error can be avoided, makes it a useful method in many areas of mathematical modeling. In this work, I will show how Monte Carlo integration can be a means to study the imaginary time self-energy described in section 2.6 at a second-order approximation. The resulting stochastically evaluated self-energy can in turn be used within a slightly altered GF2 self-consistent scheme which will be called stochastic GF2 (sGF2). This study is a type of application of diagrammatic perturbation theory and, more specifically, diagrammatic Monte Carlo. This implementation described in the following chapters is completely unique. I will discuss how this method is employed, the error evaluation scheme, as well as a set of introductory results for small toy systems to moderately-sized systems. In continuation, I will discuss how it

may provide important insight into the second-order Green's function as well give mention to the insight it may provide to other Green's function methods.

Part II:

Methods and Results

CHAPTER 4

Monte Carlo Second-Order Self-Energy: Algorithmics

Studying the exact solution of the many-electron problem has an exponential computational cost. In most electronic structure methods designed to study molecular systems two major decisions are made, one: choose an approximation to reduce excessive computational cost but forgo accuracy, or two: treat the electrons more accurately and endure the great computational cost. Another branch of electronic structure methods, those that employ Monte Carlo (MC) simulations, can be a way to explore those more accurate electronic structure methods. These stochastic methods scale more eloquently with increasing system size than deterministic methods, but will retain an error associated with the calculation. Luckily, this error - the stochastic error - is well understood and can be easily monitored.

Some of the more common stochastic electronic structure methods fall under the umbrella of "Quantum Monte Carlo" such as variational or diffusion monte carlo. Other types of stochastic methods such as sGW, sMP2, and stochastic Coupled Cluster have shown promise. However, in many of these methods, such as a suggested implementation of sGF2,[13] the method is evaluated in "one shot" and forgoes self-consistency. Furthermore, this particular method suggests an empirical method to evaluate the stochastic error instead of a statistically and repeatably rigorous approach. Another option that hasn't been explored is a stochastic procedure in the evaluation of a self-consistent method. The deterministic second-order Green's Function (GF2) is an interesting method to study through a MC simulation. GF2 can distinguish between metallic, insulating, and

mott regimes. This is an important quality when studying larger realistic systems; however, with a computational cost of $On_\tau(N^5)$ it is quite prohibitive in studying large realistic systems. A stochastic evaluation of GF2 can be a beneficial method to improve the cost. Moreover, we can take advantage of GF2's ability to distinguish between electronic regimes by employing a metropolis algorithm.

4.1 Introduction to the Method

The self-consistent Green's function theory at a second order approximation (GF2) is solved by iterative solution of the Dyson equation: $G(i\omega_n) = \left[[G_0(i\omega_n)]^{-1} - \Sigma^{(2)}(i\omega_n)\right]^{-1}$ Here the term, $G_0(\omega)$ is the non-interacting Green's function and $\Sigma^{(2)}(\omega)$ is the self-energy. In GF2, the self-energy is truncated at the second order and can be expressed as a functional of the Green's function $\Sigma[G(\omega)]$. This work is built upon the deterministic self-consistent procedure used to obtain electronic structure elements from the second order Green's function (GF2). An important aspect in the self-consistent GF2 scheme is that it uses the Matsubara imaginary frequency Green's function. The Matsubara imaginary frequency Green's function is particularly useful because it can be transformed to the imaginary time Green's function, $G(\tau)$, which is a smooth function. This smoothness makes it convenient for stochastic sampling. In this chapter, section 4.2 will give a general overview of the deterministic GF2 algorithm, section 4.3 discusses a stochastic scheme to sample for the second-order self-energy, section 4.4 gives further insight into the configuration space chosen, section 4.5 and 4.6 discuss the type of importance sampling scheme chosen, and lastly section 4.7 gives mention to other variations in the metropolis schemes that could be chosen.

4.2 Description of the GF2 Algorithm

In this work, the second order self-energy is the object of interest. The self-energy and Green's function are mutually dependent and the Green's function can be thought of as a functional of the self-energy, $G[\Sigma]$. Deterministically, the GF2 method is solved self-consistently on an imaginary

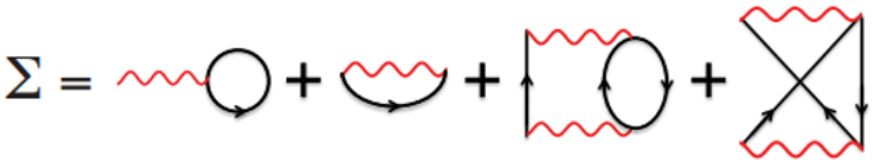


Figure 4.1: Diagrammatic expansion of the self-energy cut off at the second order. Within this subset of diagrams, the first two are frequency-independent and are included in in the Fock matrix. The second two diagrams represent the second-order correlation effects and are the frequency-dependent part of the self-energy.

time τ and imaginary frequency $i\omega_n$ axes. The interacting Green's function is built using the Dyson equation, namely

$$G(i\omega_n) = [G_0(i\omega_n)^{-1} - \Sigma(i\omega_n)]^{-1}, \tag{4.1}$$

where G_0 is the non-interaction Green's function defined as

$$G(i\omega_n) = [(i\omega_n + \mu)S - F-]^{-1}, \tag{4.2}$$

where F and S are the Fock and overlap matrices in a given orbital basis and μ is the chemical potential. The chemical potential is a necessary to guarantee a proper particle number. In this work, the self-energy, $\Sigma(i\omega_n)$, is truncated at the level of second-order perturbation. The second-order self-energy is described diagrammatically in figure 4.1. Algebraically, the second-order self-energy is given as

$$\Sigma_{ij}(\tau) = -\sum_{klmnpq} G_{kl}(\tau)G_{mn}(\tau)G_{pq}(-\tau)v_{imqk}(2v_{lpnj} - v_{nplj}), \tag{4.3}$$

where $G(\tau)$ is the Green's function on the imaginary time τ axis and v are two body Coulomb integrals.

A general implementation of GF2 can be described as follows:

1. Run either Hartree-Fock or DFT reference solution

2. Build the HF (non-interacting) Green's Function: $G_{HF}(\omega) = [(\mu + \omega)S - F]^{-1}$

3. Build second-order self-energy in the time domain from the Fourier transformed $G_{HF}(\omega)$: $\Sigma_{ij}(\tau) = -\sum_{klmnpq} G_{kl}(\tau)G_{mn}(\tau)G_{pq}(-\tau)v_{imqk}(2v_{lpnj} - v_{nplj})$

4. Fast Fourier transform (FFT) $\Sigma(\tau)$ to the ω domain

5. Rebuild interacting Green's function: $G(\omega) = [(\mu + \omega)S - F - \Sigma(\omega)]^{-1}$

6. Update the chemical potential, μ, and build correlated single-particle density matrix, $\gamma = -2G(\tau = \beta)$ with the correct particle number

7. Update the Fock matrix $_{ij} = \sum_{kl} \gamma_{kl}(v_{ijkl} - 0.5v_{ilkj})$

8. Rebuild interacting Green's function using the new Fock matrix.

9. Begin the self-consistent scheme by repeating steps 3-8 using the new interacting $G(\omega)$ and iterate until convergence of both one-body and two-body energies

A simplified bird's eye view of this scheme can be seen in figure 2.

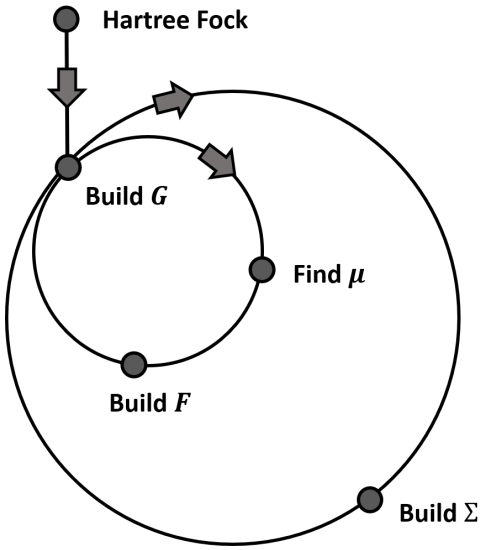


Figure 4.2: The second order Green's function can be solved via an iterative self-consistent scheme. Further description of the implementation and unique benefits of GF2 have been previously studied.[1]

GF2 has several promising aspects such as possible evaluation within a non-orthogonal atomic orbital basis. Additionally, the method has been proven to give qualitatively correct descriptions

of correlated systems such as stretched hydrogen lattices. Further more, while its computational expense, $O(\tau_n N^5)$, is similar to MP2, in the strongly correlated limit it does not suffer from divergences known to affect MP2. While GF2 displays these benefits, its computational scaling of $O(\tau_n N^5)$ results in quite costly algorithm limiting its applicability to rather small systems. The following sections will explore a new stochastic method to reduce the cost of these calculations.

4.3 Phi Functional and self-energy

Evaluated stochastically, the GF2 algorithm will now be called, stochastic GF2 (sGF2). To begin, we must choose an object to sample. The most obvious objects to choose from are either the Green's function itself or the self-energy in the imaginary time domain where they behave smoothly. However, there is a related object called the Luttinger-Ward functional, Φ, which depends both of the self-energy and Green's function and is defined as

$$\Phi = \sum_{m=1}^{\infty} \frac{1}{2m} \mathrm{Tr}\left[\sum_{n} \Sigma^{(m)}(i\omega_n) G(i\omega_n)\right], \tag{4.4}$$

where m signifies the order of perturbation. Unlike the $N \times N \times n_\tau$ matrices of Green's function and self-energy, where N is the number of orbitals and n_τ is the number of imaginary time grid points, the Luttinger-Ward functional is a scalar.

Given that Φ is a scalar it is convenient to sample for its value whenever possible. At the second-order, the Luttinger-Ward functional is given algebraically as

$$\Phi = \sum_{ijklmnpq} \sum_{\tau} v_{imqk}(2v_{jnlp} - v_{jlnp}) G_{ij}(-\tau) G_{kl}(\tau) G_{mn}(\tau) G_{pq}(-\tau). \tag{4.5}$$

The diagrammatic representation of Φ is illustrated in Fig. 4.3.

The Luttinger-Ward functional is especially useful because of its direct relationship to the self-energy. The second order self-energy is evaluated from the second-order Luttinger-Ward functional

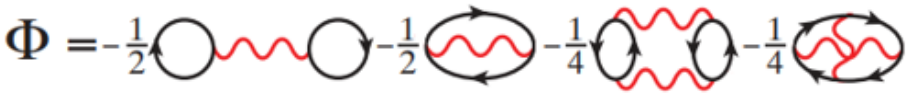


Figure 4.3: Diagrammatic expansion of the Luttinger-Ward functional, truncated at the second order.

as a functional derivative over Green's function, namely

$$\frac{d\Phi^{(2)}}{dG_{ji}(\tau)} = \Sigma^{(2)}_{ij}(\tau). \tag{4.6}$$

Thus, while $\Phi^{(2)}$ is stochastically sampled, the self-energy quantity is simply and directly measured based on the sampled object $\Phi^{(2)}$.

As discussed, evaluating Σ at the second order level scales as $O(\tau N^5)$ thus it is important to to devise a method even at the deterministic level which could help with the reduction of the computer time. When matrices in a numerical calculation are sparse, a useful mathematical trick is to reduce the size of the dimensions using the Cholesky or eigen decomposition. In this case, each $G(\tau)$ from Eq. 4.5 decay along the diagonal and contain many zero elements. Each $G(\tau)$ can be decomposed such that

$$G_{ij}(\tau) = G^{\lambda}_{i}(\tau) D^{\lambda}(\tau) G^{\lambda}_{j}(\tau) \tag{4.7}$$

where λ represents the number of Cholesky eigen vectors vectors from the given decomposition. The two-electron integrals v_{imqk} can be similarily decomposed

$$v_{imqk} = v^{\alpha}_{ik} D^{\alpha} v^{\alpha}_{mq}, \tag{4.8}$$

where α represents the number of Cholesky or eigen vectors. Under these assumptions, contractions over the orbital indices i and k in v^{α}_{ik} with the decomposed Green's functions G^{λ}_{i} and G^{μ}_{k} to further define

$$x^{\alpha}_{\lambda\mu} = G^{\mu}_{k} v^{\alpha}_{ik} G^{\lambda}_{i}. \tag{4.9}$$

After contracting all subexpressions for v_{imqk} and G_{ij}, the overall expression for Φ then becomes

$$\Phi = \int d\tau \sum_{\alpha\beta}^{n_d^v} \sum_{\lambda\mu\nu\sigma}^{n_d^g} x^{\alpha}_{\lambda\mu}(\tau) x^{\alpha}_{\sigma\nu}(\tau) D^{\alpha} D^{\lambda}(-\tau) D^{\mu}(\tau) D^{\nu}(\tau) D^{\sigma}(-\tau) \times (2x^{\beta}_{\lambda\mu}(\tau) x^{\beta}_{\sigma\nu}(\tau) - x^{\beta}_{\lambda\nu}(\tau) x^{\beta}_{\sigma\mu}(\tau)) D^{\beta}, \tag{4.10}$$

where n_d^v and n_d^g denote number of Choleskly or eigen vectors from the decompositions or integrals and Green's functions, respectively. These numbers can be radically truncated without loosing any accuracy if either $D^{\lambda}(\tau)$ from the Green's functions or D^{α} from the interaction matrix decomposition decay quickly.

Now that Φ has been built in a more manageable form amenable to stochastic and importance sampling, in Sec. **??** we will discuss how Monte Carlo integration can be applied to it.

4.4 Configurations and Weights

In section 4.3 we found a useful way to begin reducing the complexity of the problem. While the main object of interest is the self-energy, Σ, it is more convenient to focus sampling over configurations of Φ, especially in the truncation scheme proposed. When possible, typically one should avoid sampling for a multidimensional object, such as Σ. Instead, Φ, which is a scalar value, is computationally more favorable for a sampling procedure. Again, looking at Φ

$$\Phi = \int d\tau \sum_{\alpha\beta}^{n_d^v} \sum_{\lambda\mu\nu\sigma}^{n_d^g} x^{\alpha}_{\lambda\mu}(\tau) x^{\alpha}_{\sigma\nu}(\tau) D^{\alpha} D^{\lambda}(-\tau) D^{\mu}(\tau) D^{\nu}(\tau) D^{\sigma}(-\tau) \times (2x^{\beta}_{\lambda\mu}(\tau) x^{\beta}_{\sigma\nu}(\tau) - x^{\beta}_{\lambda\nu}(\tau) x^{\beta}_{\sigma\mu}(\tau)) D^{\beta}, \tag{4.11}$$

the possible configuration space contains four discrete Green's functions indices, two interaction indices, and one τ index :

$$C = \{\lambda\mu\nu\sigma : \alpha\beta; \tau_r\}$$

In the same manner as GF2, τ is prebuilt on a non-uniform grid.[14] In the proposed sampling scheme, these τ points will have discrete integration weights we will denote as w_r.

It can be assumed that to good approximation, the analytical calculation of Φ can by carried out by summing all orbital indices, $\{\lambda\mu\nu\sigma : \alpha\beta\}$, from 0 to all orbitals within the *truncation* limit. Additionally, the *tau* index goes from $0 < r < N_\tau$. This means that Φ becomes a simple summation

$$\Phi = \int d\tau \sum_{\alpha\beta}^{n_d^v} \sum_{\lambda\mu\nu\sigma}^{n_d^g} x^{\alpha}_{\lambda\mu}(\tau) x^{\alpha}_{\sigma\nu}(\tau) D^{\alpha} D^{\lambda}(-\tau) D^{\mu}(\tau) D^{\nu}(\tau) D^{\sigma}(-\tau) \times (2x^{\beta}_{\lambda\mu}(\tau) x^{\beta}_{\sigma\nu}(\tau) - x^{\beta}_{\lambda\nu}(\tau) x^{\beta}_{\sigma\mu}(\tau)) D^{\beta} w_r, \tag{4.12}$$

While the equation itself looks daunting, the process of sampling such an object can be simple. Taking a look at individual terms, it is clear that the sampling scheme must access individual precalculated scalars

$$w(c) = x^{\alpha}_{\lambda\mu}(\tau) x^{\alpha}_{\sigma\nu}(\tau) D^{\alpha} D^{\lambda}(-\tau) D^{\mu}(\tau) D^{\nu}(\tau) D^{\sigma}(-\tau) \times (2x^{\beta}_{\lambda\mu}(\tau) x^{\beta}_{\sigma\nu}(\tau) - x^{\beta}_{\lambda\nu}(\tau) x^{\beta}_{\sigma\mu}(\tau)) D^{\beta} w_r, \tag{4.13}$$

However, upon observation of these scalars it is apparent that individual terms may be positive or negative. In the processes of building the overall summation, this means the Monte Carlo mean could likely sum to zero. Instead, a simple fix is to redefine the terms as absolute values

$$p(c) = |w(c)|$$

This process of taking the absolute value seamlessly avoids the ever-worrisome numerical sign problem often cumbersome to stochastic calculations.

4.5 Importance Sampling

While the method could be done with standard Monte Carlo sampling, in many chemical systems this does not take full advantage of the physical behavior of the objects to be sampled. For example, it is known that the Green's function and integrals have many 0 values. If an algorithm treats particular indices of the Green's function with 0 value and and other indexes with greater values equally, excessive sampling time will be wasted choosing configurations that have little to no influence on the final result. Instead, an importance sampling scheme can be used. In the current work, a Metropolis algorithm will be employed. In order to use the Metropolis algorithm, the procedure must sample a configuration with the weight with which it will contribute to the integral. This means a normalization factor, proportional to the entire integral must be chosen. While this choice of normalization factor could be a number of choices, a simple first choice is as follows. A subset of the integral is chosen to calculate analytically - this is the normalization factor. For the first choice of implementation, a subset of the possible range of orbitals $1 < \lambda\mu\nu\sigma < Cholesky$ is chosen and can be renamed *norm*.

$$\Phi = \int d\tau \sum_{\alpha\beta}^{n_d^v} \sum_{\lambda\mu\nu\sigma}^{norm} |x^{\alpha}_{\lambda\mu}(\tau) x^{\alpha}_{\sigma\nu}(\tau) D^{\alpha} D^{\lambda}(-\tau) D^{\mu}(\tau) D^{\nu}(\tau) D^{\sigma}(-\tau) \times (2x^{\beta}_{\lambda\mu}(\tau) x^{\beta}_{\sigma\nu}(\tau) - x^{\beta}_{\lambda\nu}(\tau) x^{\beta}_{\sigma\mu}(\tau)) D^{\beta} w_r|, \tag{4.14}$$

This subset of orbitals can be as small as 1 term, but as the subset chosen gets larger and a greater portion of Φ is summed up deterministically, the closer the result should be to the true deterministic value. While other normalization factors could easily be chosen, the proposed normalization factor is a good first choice.

4.6 Metropolis Sampling Scheme

In the last sections, it was mentioned that the possible configuration space contains four discrete Green's functions indices, two interaction indices, and one τ index :

$$C = \{\lambda\mu\nu\sigma : \alpha\beta; \tau_r\}$$

We will call C the configuratin space. Within an importance sampling scheme, it is important that no important contributions to the overall calculation are missed. This takes careful management of the walkers over the entire configuration space. The following will describe the metropolis conditions chosen as well as exactly how walkers were allowed to move through each subset of the configuration space. In diagrammatic Monte Carlo schemes, it is particularly important to devise a scheme where walkers can move throughout the diagrams and not get "stuck" sampling in only one vertice. This could be done in a few different ways, but we will mention only one specific method.

The configuration space, C, can be broken into three pieces - C_1, C_2, and C_3. Where,

$$C_1 = \{\tau_r\}$$

$$C_2 = \{\lambda\mu\nu\sigma\}$$

$$C_3 = \{\alpha\beta\}$$

We must create a unique metropolis condition for each piece. Beginning with C_1, we can document within a specific calculation how many times certain tau points are sampled. These tau points should be weighted so that places where a cross-section of the Green's function, say $G_{xx}(\tau)$, has high curvature there will be lots of samples taken. Places where the cross-section has almost no curvature or is zero should have no sampling. The method that sGF2 has shown most success with is choosing a value $tau_{jump} << n\tau$ and allowing each new metropolis opportunity to jump along

the *tau* grid according to the value of tau_{jump}. If the grid reaches one end of the grid, the procedure automatically resets the new point to be sampled at the other end of the grid. Figure 4 displays that the metropolis condition for $C_1 = \{\tau\}$ successfully samples only for those important contributions.

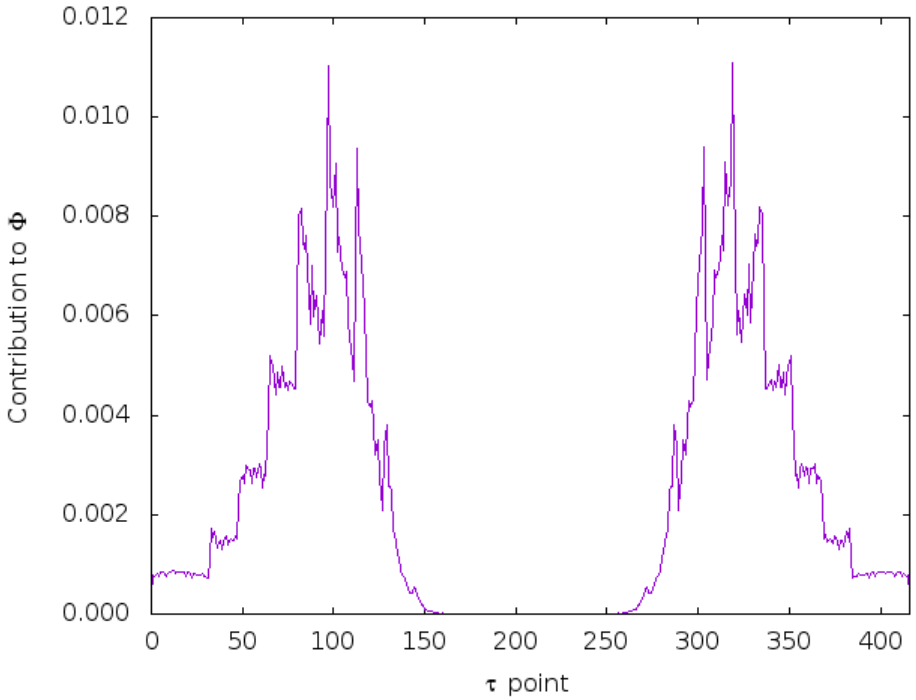


Figure 4.4: The figure shows that places of high curvature in the Green's function have higher frequency of sampling, thus more contribution to the phi functional. On the other hand, places where the curvature is almost vertical or the value of the Green's function is 0, there is 0 contribution to the phi functional.

Similarly, the indexes from the cholesky decomposed Green's functions must be efficiently sampled. These indices again should have larger contributions coming from the lower valued indices within C_2 and smaller contributions from the larger value indices within C_2. One successful way to ensure proper sampling of all important contributions of the space is to simply choose a random number between zero and one at each update. This random number can be multiplied by the predetermined truncated Cholesky space for λ, μ, ν, or σ. New configurations are chosen each time a new update is required for this space. The quality of this algorithmic choice can be visualized for any system. Here we show that the metropolis condition chosen efficiently chooses

orbitals for both a chain of 10 Hydrogens and a four by four square lattice of Hydrogens in minimal basis sets. In each case, Hydrogens are spaced 1 Angstrom apart. Both systems are important to consider because while in a chain it is generally easy to choose walkers that will sample the system properly, adding another dimension - such as in the square lattice - is often more difficult for walkers to efficiently evaluate the whole space. Figure 5 displays how effectively the Metropolis algorithm ensures to promote sampling at indices of the Green's Functions where values are non-zero.

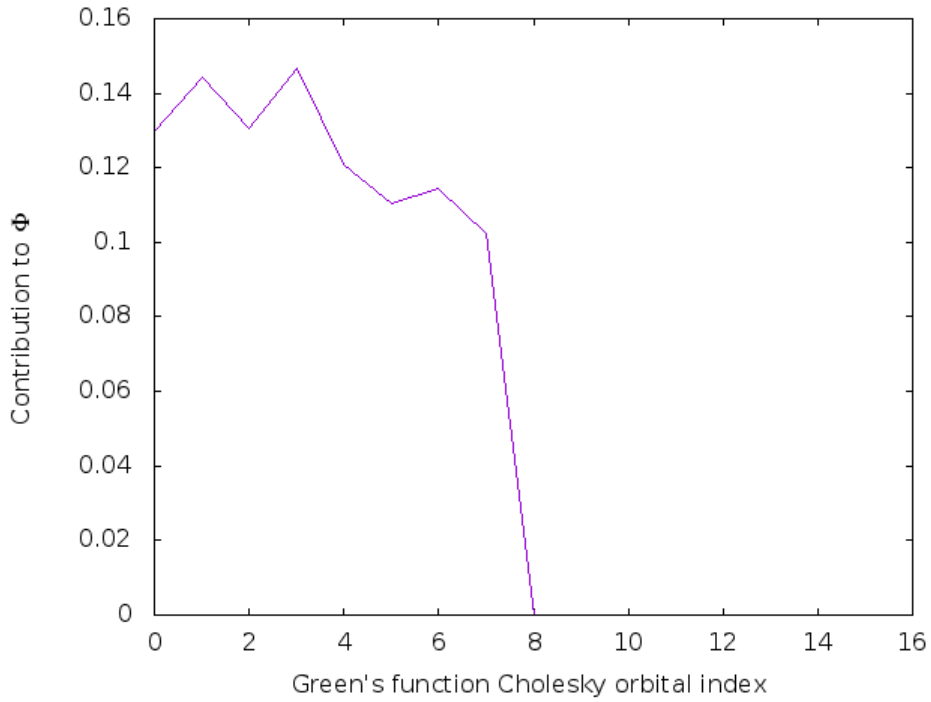


Figure 4.5: The figure shows that where the elements of the decomposed Green's function are large, there is the most contribution to Φ. Because each Green's function decays quickly, after only a few Cholesky orbitals indices the contribution to Φ decays to 0.

Finally, the third piece we call C_3 must be considered in a similar fashion to C_2. The only difference is that the predetermined truncated Cholesky space for α and β from the two-electron integrals is a different value than the predetermined truncated Cholseky space for the green's function Cholesky indices λ, μ, ν, and σ. Again a random number between 0 and 1 is chosen which is multiplied by α or β. Again, figure 6 shows the success of the metropolis algorithm.

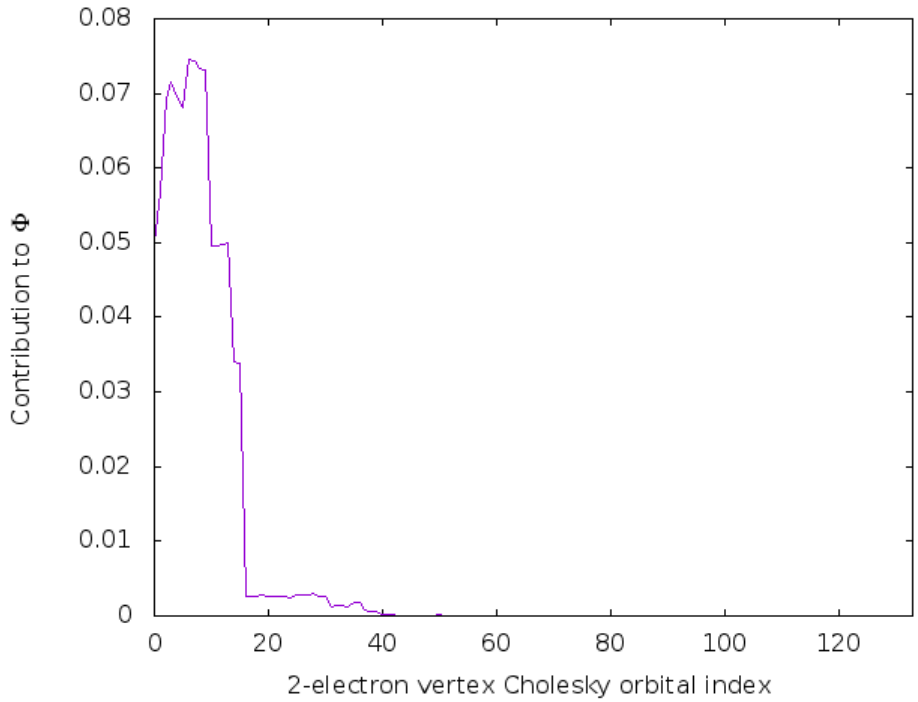


Figure 4.6: The figure shows that where the elements of the decomposed 2-electron vertex function are large, there is the most contribution to Φ. Because each 2-electron vertex function decays quickly, after only a few Cholesky orbitals indices the contribution to Φ decays to 0.

4.7 Other Choices for Metropolis Sampling Schemes

While the method described has proven to work well, other methods of implementing the metropolis conditions are available. Of those methods, one other has been considered. A cumbersome aspect of this method is that the contractions involving the two-electron integrals must be stored in memory for each tau point, $x^{\alpha}_{\lambda\mu}(\tau)$. While storing this decomposed form is favorable to storing the entirety of the two-electron integrals, the size can still easily surpass the size of a standard computer processing unit (4GB). Another option is to fully parallelize each sampling procedure on individual τ points. Contrary to how Φ was previously evaluated

$$\Phi = \int d\tau \sum_{\alpha\beta}^{n_d^v} \sum_{\lambda\mu\nu\sigma}^{norm} |x^{\alpha}_{\lambda\mu}(\tau) x^{\alpha}_{\sigma\nu}(\tau) D^{\alpha} D^{\lambda}(-\tau) D^{\mu}(\tau) D^{\nu}(\tau) D^{\sigma}(-\tau) \times (2x^{\beta}_{\lambda\mu}(\tau) x^{\beta}_{\sigma\nu}(\tau) - x^{\beta}_{\lambda\nu}(\tau) x^{\beta}_{\sigma\mu}(\tau)) D^{\beta} w_r|, \tag{4.15}$$

the Monte Carlo scheme can now fully evaluate every point of the τ space

$$\Phi_\tau = \int d\tau \sum_{\alpha\beta}^{n_d^v} \sum_{\lambda\mu\nu\sigma}^{norm} |x^\alpha_{\lambda\mu} x^\alpha_{\sigma\nu} D^\alpha D^\lambda D^\mu D^\nu D^\sigma \times (2x^\beta_{\lambda\mu} x^\beta_{\sigma\nu} - x^\beta_{\lambda\nu} x^\beta_{\sigma\mu}) D^\beta w_r|, \tag{4.16}$$

This method clearly becomes embarassingly parallel with each Φ_τ being evaluated on a different core. While this method technically loses mathematical optimization, in most cases studied the tau grid is only a few hundred points and if parallelized the time of simulation is not greatly effected. Results to summarize this sampling scheme will be included in the results section.

CHAPTER 5

Error Control and Validation

Many body perturbation theory by way of the second-order Green's function has shown many successes. Given the expense of the problem, evaluation of the self-energy via a stochastic algorithm has clear motivation. In the second section, the methods for sampling in a stochastic evaluation were discussed. An important portion of analysis of a stochastic calculation is the error control and validation. Typically, error is controlled by running the same calculation with many different seeds and taking an average of the resulting values. In the case of the second-order Green's function, this is not as straight forward for a few reasons. First, in this work the values of primary interest are the one-body, $E = \frac{1}{2}\,\mathrm{Tr}[(H + F)\gamma]$, and two-body, $\frac{2}{\beta}\sum Re(\mathrm{Tr}[G\Sigma])$, energy. The sum of these two values should equal the total energy of the system. These values are obtained via both the Green's function and self-energy. This leads into the second difficulty. Baym showed that when the self-energy is calculated via a subset of diagrams, such as in the case of the second-order Green's function, conservation laws may be violated.[15] When the Green's function is obtained from a fully self-consistent solution of the Dyson equation, the approximation is then called Φ derivable and is automatically conserving. While self-consistency is clearly the most physically rigorous way to evaluate a system via diagrammatic perturbation theory, many formalisms forgo self-consistency due to the immense computational cost. However, this work will follow the rigorous path involving self-consistency. The stochastic second-order Green's function can be calculated self-consistently in a similar scheme to the deterministic second-order Green's function. In the deterministic GF2 scheme, each iteration can generate a single one-body and two-body energy. In the stochastic

scheme, while the object obtained from the stochastic procedure is still the imaginary time self-energy, there are actually a population of imaginary time self-energies produced from different starting seeds. In order to generate an error bar, M starting seeds must be used and each resulting self-energy must be stored. To return the Green's function from the self-energy, an inverse of the matrix must take place. This inversion is given as follows,

$$G(\omega) = [(\mu + \omega)S - F - \Sigma(\omega)]^{-1}. \tag{5.1}$$

An aspect of inversion well known in the statistical community is that an inversion of a normal set of data is considered to be a non-linear transformation. This means that data that was initially normal and then put through a non-linear transformation, may lose its normality. In this case, standard parametric statistics such as the simple method of taking a mean ($\bar{a}$),

$$\bar{a} = \frac{1}{n}\sum_{i=1}^{n} a_i = \frac{a_1 + a_2 + \cdots + a_n}{n} \tag{5.2}$$

are no longer applicable. Instead, bias has been introduced and must be monitored. This new set of non-parametric data must be adjusted in order to study statistical objects such as the mean and standard error. One way to make this adjustment is via a non-parametric procedure called resampling. Resampling can be done via many methods, most commonly the jackknife and bootstrap procedures. In this work, a jackknife analysis is the resampling method of choice. This section will explore whole jackknife analysis can be employed within a slightly altered GF2 scheme which includes evaluation of the stochastic self-energy.

5.1 Review of the Jackknife Procedure

In statistics, the jackknife method is a type of resampling technique useful for variance and bias estimation. The jackknife estimator of a parameter is found by leaving out a single observation from a dataset and calculating the estimate, $\bar{x}_i$, then repeating for each observation in the dataset.

This is typically done in systematic and ordered way. After each observation has been dropped and the corresponding estimate of these calculations has been stored, the average,$\bar{X}$ of each of these estimates is taken. If a sample dataset has size n, the jackknife estimate is found by systematically accumulating the estimates of each $(n-1)$-sized sub-sample, $\bar{x}_i$.

5.2 Jackknife Analysis Within sGF2 Scheme

Implementing the Jackknife Analysis within the sGF2 scheme is not completely straight forward. One may assume that the one-body and two-body energies at the end of the first iteration can simply be resampled to find the jackknife average; however, the noise from the inversion of the stochastically obtained self-energy would create too expensive of a chemical potential search for this to be a viable solution. Instead, the jackknife must be built into steps 4-8 of the self consistent scheme - these steps are described in the first section.

The self consistent scheme remains exactly the same for steps 1 and 2. The method must be adjusted to accommodate the stochastic self-energy and the jackknife analysis starting in step 3. The steps are summarized here:

1. Run restricted Hartree-Fock reference solution
2. Build HF (non-interacting) Green's Function: $G_{HF}(\omega) = [(\mu+\omega)S - F]^{-1}$
3. Build a sample of stochastic second-order self-energies in the time domain via the method described in chapter 4 from the Fourier transformed $G_{HF}(\omega)$: $\Sigma_{ij}^{seed}(\tau) = -\sum_{klmnpq} G_{kl}(\tau)G_{mn}(\tau)G_{pq}(-\tau)v_{imqk}($ $v_{nplj})$
4. Using the "leave-one-out" method described in the previous section, obtain a sub-estimate, $\bar{x}_i$ for each $\Sigma_{ij}^{seed}(\tau)$
5. Fast Fourier transform (FFT) $\Sigma(\tau)$ sub-estimate to the ω domain
6. Rebuild an interacting Green's function: $G(\omega) = [(\mu+\omega)S - F - \Sigma(\omega)]^{-1}$ for each sub-estimate.

7. Update the chemical potential, μ, and build correlated single-particle density matrix, $\gamma = \frac{1}{2\pi i}\oint G(\omega)d\omega$ with correct particle number for each sub-estimate

8. Update the Fock matrix for each sub-estimate.

9. Find the jackknife estimate/average of each interacting Green's function $G(\omega)$ and Fock matrix. Repeat steps 3-8 using the newly jackknifed correlated $G(\omega)$.

10. Iterate until both the jackknifed one-body and two-body energies converge independently. Convergence can be deemed where the one-body/two-body energy changes by less than the error bar for two iterations.

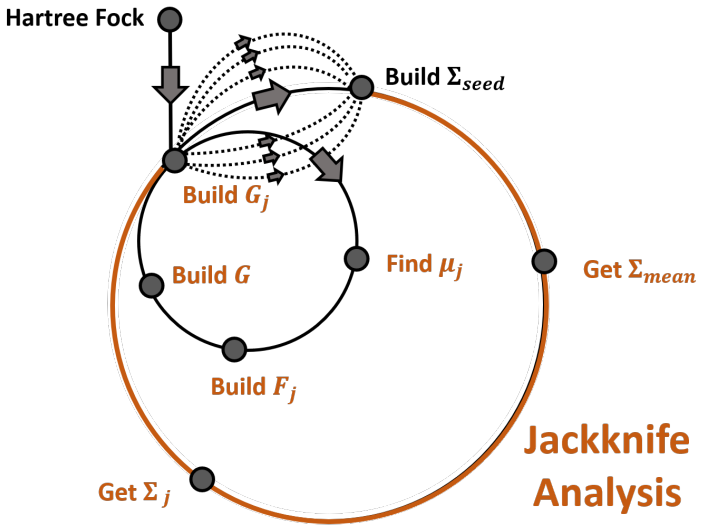


Figure 5.1: The stochastic second order Green's function can be solved via an iterative self-consistent scheme. The scheme in many ways looks similar to the GF2 scheme given in figure 1. However, due to the inversion of data taken from transforming the self-energy to the Green's function, the scheme needs to be altered to account for non-parametric bias. This is carefully done through a resampling scheme called the jackknife analysis.

This method very clearly mimics the deterministic GF2 self-consistent procedure with a few alterations. While this method of self-consistency is clearly more costly than the typical GF2 procedure, the sGF2 jackknife analysis can again easily be parallelized so that each seed, or instance of

”take-one-out”, is processed on a different node. The following results will discuss the success of this method and proof of its viability to properly study the error associated with the Monte Carlo evaluation.

CHAPTER 6

Data

In this subsection, we will provide a calibration of the one-body, two-body, and subsequent total energies for a few simple toy systems. The values determined from the sGF2 procedure will be compared to those at the deterministic GF2. While the goal of sGF2 is to obtain physical information about systems too large or expensive to reach by deterministic GF2, to analyze the quality of the method it must be first compared to deterministic GF2 results for small systems. Additionally, some larger real chemical systems will be explored. In these systems, the efficiency of the metropolis algorithm will be shown and discussed. While the energies could be given, these values are far less interesting than observing the efficiency of using the metropolis algorithm for even increasingly large systems.

The overall sGF2 algorithm can be considered in two main parts. Part one is the Monte Carlo sampling scheme. This procedure is built on a version of the core libraries of ALPS (Applications and Libraries for Physics Simulations libraries) [ALPSCore libraries].[16] Further the Cholesky decompositions for all Green's functions are done via the Eigen3 library. The Cholesky decomposition for all two-electron integrals are done via Dalton. The second part is the overall self-consistent scheme and required jackknife analysis. The results will analyze the error following one iteration of sGF2 for a chain of 10 Hydrogens as well as a 4x4 plaquette of Hydrogens in a minimal basis set. Following, full self-consistent convergence of a water molecule in a cc-pVDZ basis will be displayed. Finally, the efficiency of the metropolis algorithm will be shown with a water trimer being stretched as well as a series of Glycine peptide chains ranging from 1-5 Glycine residues.

6.1 Error Following One Iteration of sGF2

To monitor the error of the sGF2 method, it is important to start by comparing small toy chemical systems such as a chain of Hydrogens or a plaquette of Hydrogens.
In the first iteration of the sGF2 scheme, the two-electron integrals, Green's function and Fock matrix used to calculate the first interacting Green's function matrix could be identical to those in the first iteration of deterministic GF2. This makes it convenient to compare the one-body and two-body energies after the first iteration of sGF2 to those after the first iteration of deterministic GF2.

If the error is well controlled, a two conditions must be met. First, after the first iteration the resulting one-body and two-body deterministic energies must fall within two error bars of the stochastic one-body and two-body energies. Second, as the the numbers of samples considered increases, the error bars on the resulting stochastic energies should systematically decrease by $1/\sqrt{N}$.

The following results show the validation of error control for two toy systems.

6.2 Chain of 10 Hydrogens Spaced 1 Angstrom Apart

First, we chose a system of 10 Hydrogens placed 1Å apart. A minimal sto-3G basis set is used. τ_{jump} is set at 16, and a thermalization of 800,000 is used. The number of Monte Carlo sampling steps is 10^7, with 10 measurements taken at each step. The normalization subspace is only one orbital. Finally the matsubara frequency grid is built with 6,000 points and the non-uniform *tau* grid contains 416 points. The jackknife results on the one-body and two-body energies are given for calculations with 8, 16, 32, 64 and 128 seeds in the plots in figures 1 and 2. It is clear that the error decreases by $1/\sqrt{N}$ as the number of seeds increases. Additionally the energies stay within two error bars of the deterministic result.

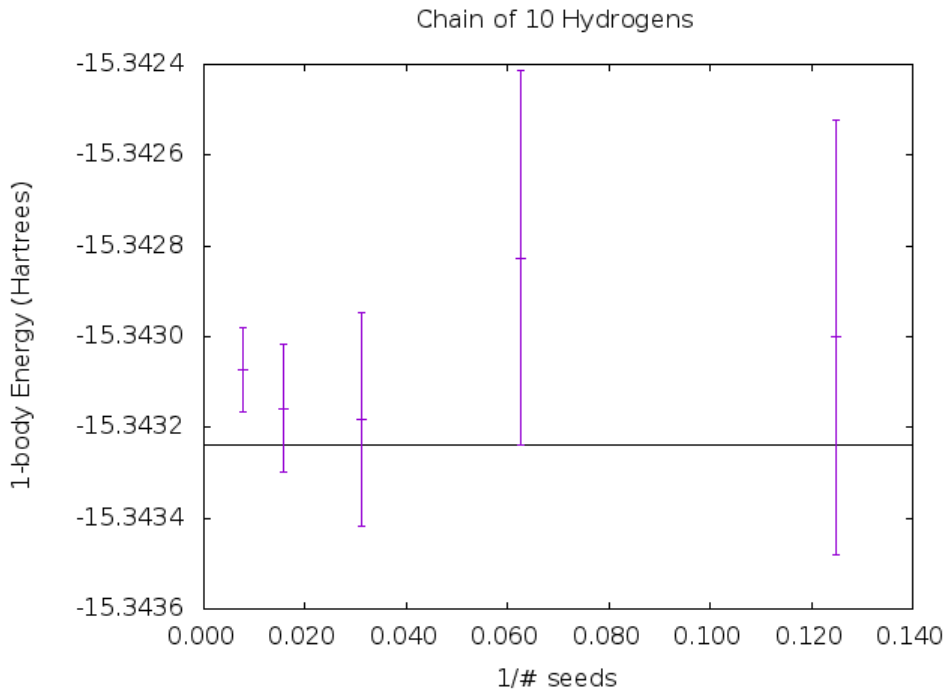


Figure 6.1: Here we show that with increase number of starting seeds on the x axis, the deterministic one-body and two-body energies fall within two error bars of the stochastic one-body energy of a chain of 10 Hydrogens in a sto-3G basis. The black horizontal line represents the deterministic result for the one-body energy. The purple vertical lines represent the 1-σ error bars.

6.3 4x4 Square Lattice of Hydrogens Spaced 1 Angstrom Apart

Second, we chose a system of 16 Hydrogens organized into a 4x4 square lattice in a minimal sto-3G basis. τ_{jump} is set at 16, and a thermalization of 800,000 is used. The number of Monte Carlo sampling steps is 10^7, with 10 measurements taken at each step. The normalization subspace is chosen as three orbitals. Finally the matsubara frequency grid is built with 6,000 points and the non-uniform *tau* grid contains 416 points. The jackknife results on the one-body and two-body energies are given for calculations with 8, 16, 32, 64 and 128 seeds in the plots in figures 3 and 4.

It is clear that both conditions required for error validation are met.

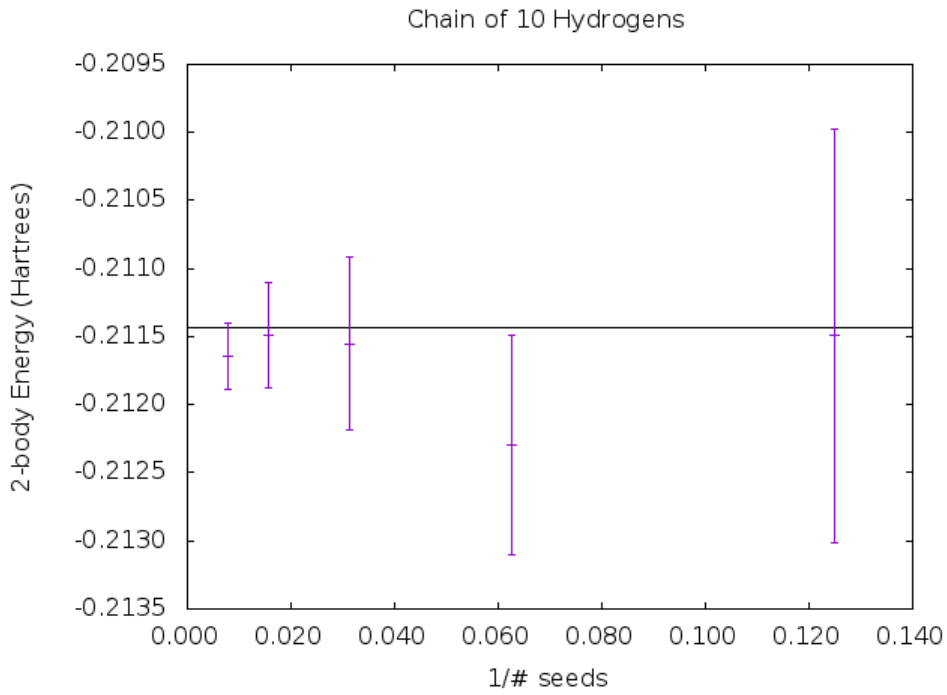


Figure 6.2: Here we show that with increase number of starting seeds on the x axis, the deterministic one-body and two-body energies fall within two error bars of the stochastic two-body energy of a chain of 10 Hydrogens in a sto-3G basis. The black horizontal line represents the deterministic result for the one-body energy. The pruple vertical lines represent the 1-σ error bars.

6.4 Error Following Upon Convergence of Fully Self-Consistent sGF2

As discussed the sGF2 procedure is self-consistent like deterministic GF2. Unlike the first iteration where the same two-electron integrals, Green's function, and Fock matrix are read into the procedure, in later iterations a jackknife average Green's function and Fock matrix will be fed into the following iterations. These objects have associated errors. Due to this, convergence cannot be assumed behave identically to the deterministic procedure; however, if the error is controlled properly between iterations, once at convergence the one-body and two-body deterministic converged energy should still lie within two stochastic one-body and two-body error bars. Additionally, the error bar should stay stable in that it will not grow or shrink arbitrarily.

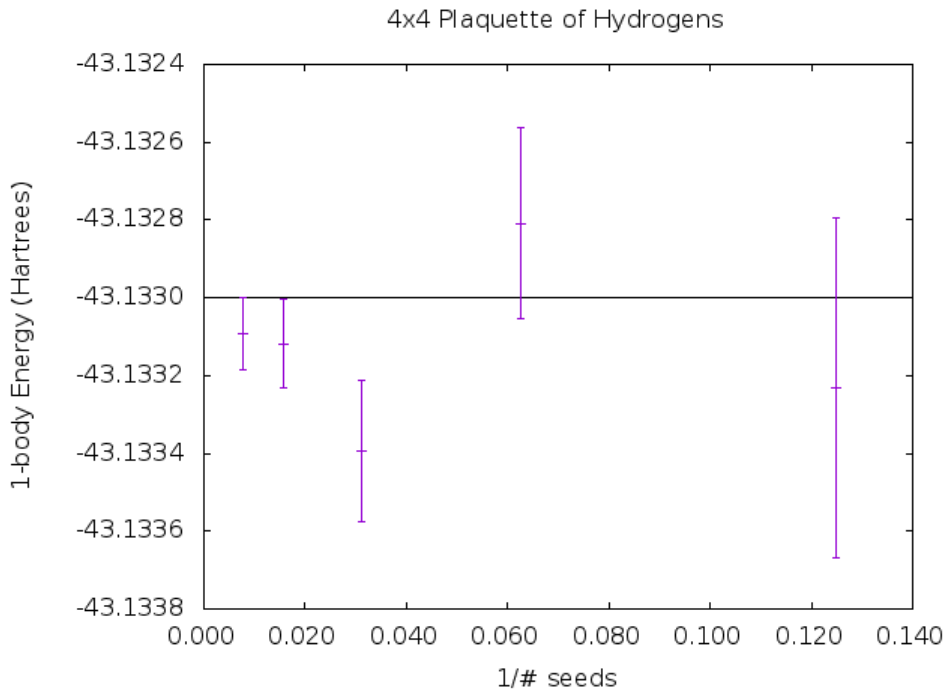


Figure 6.3: Here we show that with increase number of starting seeds on the x axis, the deterministic one-body energy fall within two error bars of the stochastic one-body energy of a square lattice of 16 Hydrogens in an sto-3G basis. The black horizontal line represents the deterministic result for the one-body energy. The pruple vertical lines represent the 1-σ error bars.

6.5 Water Molecule

To study the full self-consistent scheme, we chose a slightly less homogenous system and consider the error between each iteration until convergence is reached. The system is a water molecule in a optimized geometry at the restricted Hartree Fock level with a 6-31G(d,p) basis set. In this case a cc-pVDZ basis set is used for the deterministic and stochastic GF2 calculations. τ_{jump} is set at 16, and a thermalization of 800,000 is used. The number of Monte Carlo sampling steps is 10^8, with 10 measurements taken at each step. The normalization subspace is chosen as six orbitals. Finally the matsubara frequency grid is built with 10,000 points and the non-uniform *tau* grid contains 416 points. The jackknife results on the totaled one-body and two-body energies are given for each iteration of the calculation. Each iteration is evaluated with 128 seeds. The results are given in figure 5.

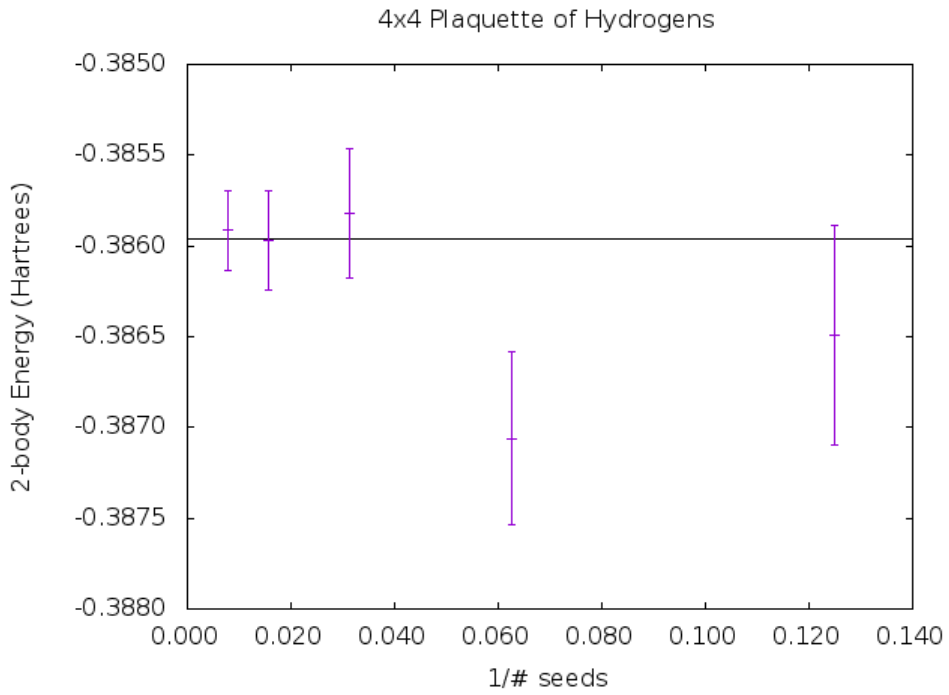


Figure 6.4: Here we show that with increase number of starting seeds on the x axis, the deterministic two-body energy fall within two error bars of the stochastic two-body energy of a square lattice of 16 Hydrogens in an sto-3G basis. The black horizontal line represents the deterministic result for the one-body energy.The pruple vertical lines represent the 1-σ error bars.

The purpose of the sGF2 method is to reach insight into systems that deterministic GF2 cannot yet reach. If we are studying a system where we don't know the resulting deterministic value, an important question to ask is: what convergence criteria can be set so we know convergence is met? We propose that this criteria should be that the change in the jackknife estimate of the one-body and two-body energies, individually, should not change by more than one error bar for two iterations.

6.6 Metropolis Efficiency in a Stretched Water Trimer

Next, we consider how efficient the metropolis condition is in the sGF2 procedure. For this we studied a water trimer in the cc-pVDZ basis. The geometries studied were based off an optimized

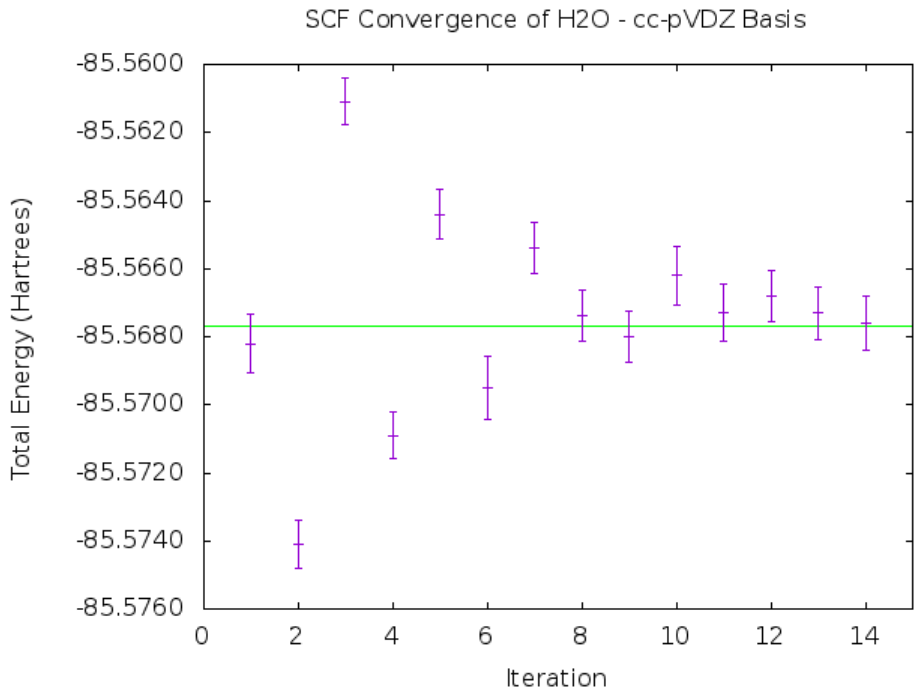


Figure 6.5: Here we show that upon iterating the self-consistent sGF2 procedure, the converged deterministic GF2 total energy will fall within two error bars of the stochastic converged total energy of a water molecule in the cc-pVDZ basis. The x-axis displays which iteration the procedure is in. Overall, the procedure converges at 14 iterations.The pruple vertical lines represent the 1-σ error bars of the total energy, and the black horizontal line represents the deterministic GF2 result for the converged total energy.

geometry at the restricted Hartree Fock level with a 6-31G(d,p) basis set. Each stretched geometry took the individual water molecules from the midpoint and placed them 1 Angstrom further from the midpoint. τ_{jump} is set at 16, and a thermalization of 800,000 is used. The number of Monte Carlo sampling steps is 10^8, with 10 measurements taken at each step. The normalization subspace is chosen as eight orbitals. Finally the matsubara frequency grid is built with 10,000 points and the non-uniform *tau* grid contains 416 points. As expected, we show that as the distance between individual water monomers increases, a threshhold is met where the number of accepted orbital and Green's function updates reaches a plateau.

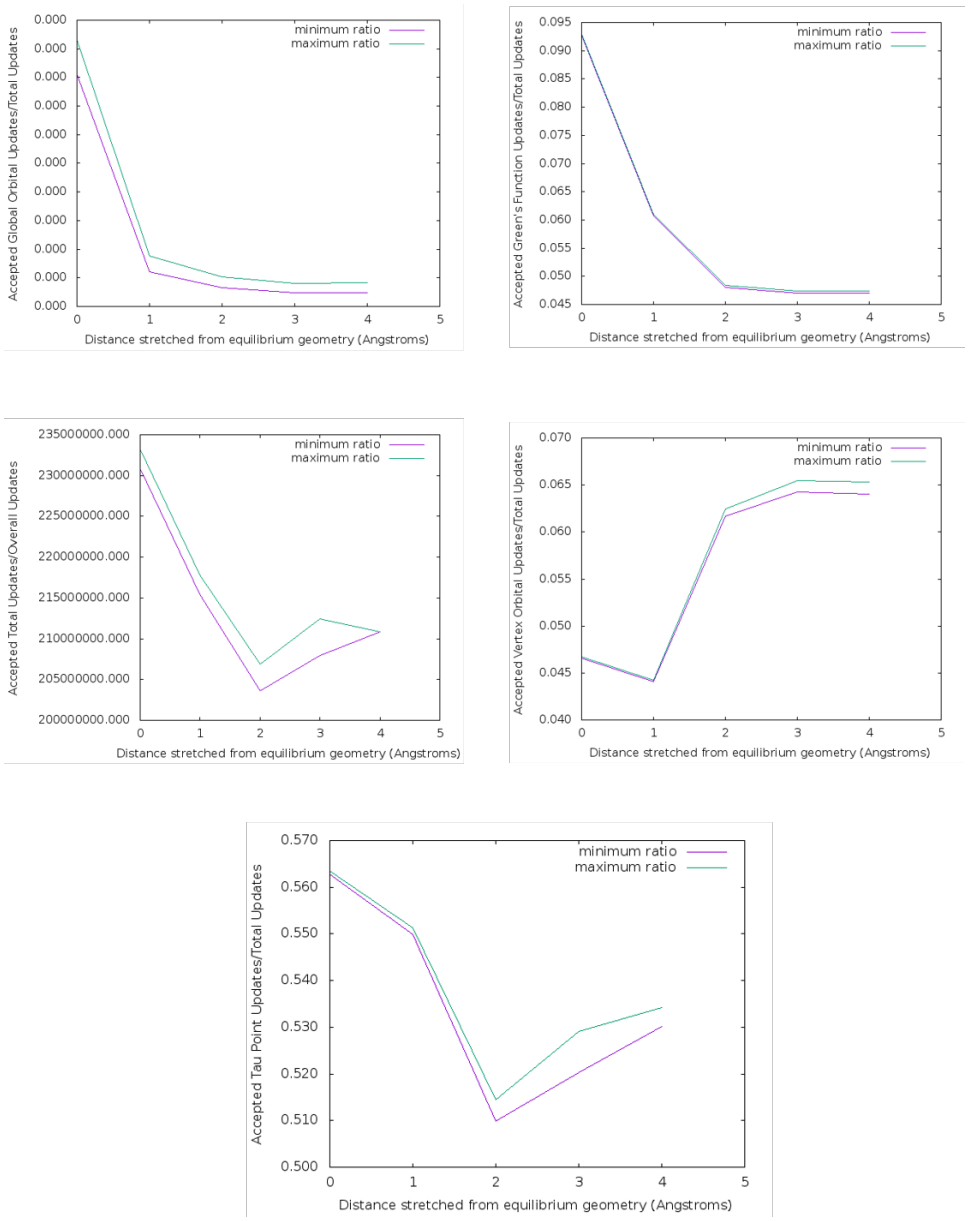


Figure 6.6: Here we show that with a Water trimer that is stretched, the the acceptance ratios for each type of metropolis condition behave as expected.

6.7 Metropolis Efficiency for Growing Glycine Peptide Chain

Finally, we consider how efficient the metropolis condition is in the sGF2 procedure as the size of the system, or number of orbitals, increases. For this we studied a series of peptide chains containing Glycine residues in the 6-31G basis. We studied peptides ranging from 1 to 5 residues. The geometries studied are given in the supplementary information. τ_{jump} is set at 16, and a thermaliza-

tion of 800,000 is used. The number of Monte Carlo sampling steps is 10^7, with 10 measurements taken at each step. The normalization subspace is chosen as fifteen orbitals. Finally the matsubara frequency grid is built with 10,000 points and the non-uniform *tau* grid contains 416 points. While the cost of a deterministic calculation would be expected to grow like $O(n_\tau N^5)$, we expect the computational cost of the Monte Carlo simulation to grow at slower rate. In this example, the time of simulation seems to develop a linear relationship with the number of orbitals in the Glycine chain studied. This result is exactly as desired from the Monte Carlo simulation.

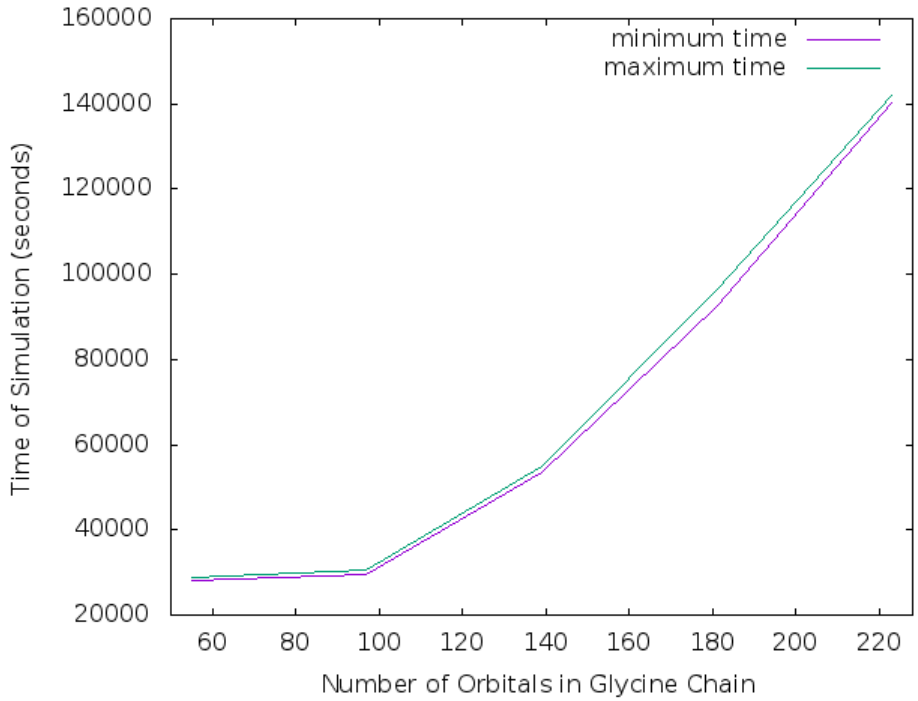


Figure 6.7: Here we show that with a five-fold increase in the number of orbitals, the timing of simulation also increases five-fold.

CHAPTER 7

Compute-to-Learn: Authentic Learning via Development of Interactive Computer Demonstrations within a Peer-Led Studio Environment

7.1 Introduction

Learning is widely acknowledged as being situational[17,18,19,20,21] meaning it is inherently dependent upon the environment in which it occurs. "Where" and "how" something is learned is just as important as "what" is learned, because knowledge is dependent on the context in which it was developed and applied. Within this framework of situated cognition, instructors can provide students with opportunities to reconstruct a concept for themselves through engagement in activities that are typical of the discipline's practices,[22] thus providing a more meaningful and authentic learning environment.[23,24] This can be accomplished via the use of real-world problems that are investigated in a collaborative environment over a sustained period of time.[25] It can be difficult

to incorporate such authentic learning activities into the traditional lecture-based undergraduate classroom, which is often explanatory in nature and relies on activities structured around established knowledge, generally in an environment that is not conducive to collaboration. In an effort to address these issues in the physical sciences we have developed a compute-to-learn pedagogy, which incorporates important features of scholarly research into a collaborative studio environment for undergraduate students.

The compute-to-learn pedagogy was developed as an extension and expansion of the writing-to-teach pedagogy of Vázquez et al.,[ázquez$_m$cloughlin$_s$abbagh$_r$unkle$_s$imon$_c$oppola$_p$azicni$_2$012] which focused on utilizing explanatory writing and peer learning activities to engage students in authentic and meaningful learning, and on creation of textbook-like sections that could be utilized by future students in the course as learning resources. While writing-to-teach effectively enhanced students' ability to generate explanations of scientific concepts, incorporating the textbook-like sections written by the students was not straightforward due to the likelihood of mis-statements and misinformation. Furthermore, reviewing the documents written by students for accuracy, providing feedback, and repeating this cycle to achieve a correct description of the physical concept was a time-consuming and cumbersome process. Due to the time commitment of reviewing said documents, the writing-to-teach pedagogy could only be applied to small groups of students. The compute-to-learn pedagogy follows a guided inquiry approach,[26,27] in which students design and develop an interactive, computer-based demonstration that illustrates a physical chemistry concept of their choice. Students work on this semester-long project in a studio environment that borrows elements from the arts, where students work in a studio and receive feedback from one another regarding their artistic creations.[ázquez$_m$cloughlin$_s$abbagh$_r$unkle$_s$imon$_c$oppola$_p$azicni$_2$012, ?, 29] The pedagogy focuses on active learning strategies and encourages cooperation and collaboration among peers, with undergraduate peer leaders and graduate student instructors (GSIs) serving as mentors.[28,29] Students investigate a challenging, open-ended physical chemistry concept rather than a well-defined problem, similar to a researcher in a scientific discipline. After they have completed a computer-programming tutorial and studied their topic, students design an interactive visual representation that they present to their

peers for review in a storyboard process. The students proceed to develop their demonstrations using the programming skills obtained during the tutorial. Utilizing feedback from several cycles of development and peer review, students continue to program and refine their demonstrations. Finally, students submit their work to the Wolfram Demonstrations Project,[30] an open-source library of interactive demonstrations from a variety of fields, for external review and publication. Thus, students incorporate physical chemistry concepts into teaching tools that could be utilized by others.

This framework preserves the advantageous components of writing-to-teach, such as enhancing the ability of students to generate explanations of scientific concepts, and overcomes the cumbersome review process of writing-to-teach. The introduction of computer programming into the pedagogy of compute-to-learn corresponds to an additional benefit for students who are interested in acquiring that skill. The compute-to-learn pedagogy immerses students in a research environment that closely simulates the real-world scientific research process, and thus helps to apprentice students as members of the scientific community.[19,22] Through the process of creating the demonstration, students learn new concepts in physical chemistry, mathematics, writing, and programming and are given the opportunity to hone various skills, such as communication, collaboration, and task management.

The paper is organized as follows. In the next (second) section, we present our first implementation of the compute-to-learn pedagogy and reflect on the shortcomings that we found. In the third section, we provide an overview of why we utilized Wolfram Mathematica[31] to develop the demonstrations and provide an example of a demonstration developed in the studio. In the fourth section, we detail the modifications we made in subsequent iterations of the compute-to-learn studio and the motivations behind these improvements. In the fifth section, we discuss assessment of the compute-to-learn pedagogy on student learning utilizing interviews of participants from the Fall 2016 and Winter 2017 semesters.

7.2 Initial Implementation of Compute-to-Learn

Compute-to-learn[32] was first implemented at the University of Michigan in the 13-week-long Fall semester of 2015 as the Honors option affiliated with the "Chemical Principles" (CHEM 260) course. CHEM 260 provides a survey of quantum mechanics, thermodynamics, and kinetics. It serves as an introductory physical chemistry course in place of a second-semester general chemistry course for those students majoring in chemistry or biochemistry. Students could take the course for honors credit (CHEM 260H) by participating in the compute-to-learn studio, which included additional weekly 2 hour meetings. This option was available to all students enrolled in the primary course. Of the 60 students that were enrolled in CHEM 260 during the Fall 2015 semester, six students chose to participate in the compute-to-learn studio. This subset of six students met for 12 weekly 2 hour studio sessions. During those sessions the students learned to program in Mathematica, design and create an interactive visualization of a physical chemistry concept, and prepare their work for publication.

A team consisting of faculty members, graduate students, and a pair of undergraduate peer leaders met regularly throughout the semester to develop course materials, review progress of students, and prepare for studio sessions. The undergraduate peer leaders served as advisors during the studio sessions and were the primary source of guidance for the students, while two GSIs were also present during the sessions to assist the peer leaders when necessary. The weekly meeting schedule for the Fall 2015 studio is shown in Table 1.

Compute-to-learn was designed with the expectation that students would have no prior experience with Wolfram Mathematica or computer programming in general. To this end, we designed a four-week tutorial to introduce students to the Wolfram Language. The tutorial focuses on core Wolfram syntax, with an emphasis on components of the language that are necessary for developing interactive Mathematica demonstrations. A detailed description of Mathematica demonstra-

Week Number	Agenda	Review	Assignment
1	Engage in an interactive tutorial to learn essential Mathematica programming skills; Choose and research a scientific concept for demonstration	No	Research prompt topic
2			Implement new features into existing demonstration
3			Prepare storyboard
4	Complete tutorial; Present "storyboard" depiction of demonstration	Peer Review	Start programming demonstration
5	Fall Break – No Classes	No	Continue programming
6-8	Program demonstration in Mathematica	No	Continue programming
9	Informally present progress; Receive feedback from peers	Peer Review	Continue programming; Complete Wolfram Demonstration page's publication checklist
10	Modify demonstration to incorporate peer input; Write succinct one paragraph description	No	Continue programming; Complete final edits
11	Submit to Wolfram Demonstration Project for further review and publication	External Review by Wolfram	Write one-page reflection on peer review process
12	Incorporate edits offered by the Wolfram Mathematica reviewers	External Review by Wolfram	Incorporate external review edits into demonstrations
13	Formally present work to peers	Peer Review	None

Table 7.1: This table gives a week-by-week overview of the curriculum used in the third iteration of the compute-to-learn curriculum.

tions is provided in the next section. Scaffolding was incorporated into the tutorial to provide initial support to students, and is systematically removed as they progress through it.[33,34] The full tutorial, which is formatted as a Mathematica notebook file, is provided in the Supporting Information. While the peer leaders guided students through the tutorial in the studio, the tutorial itself is entirely self-contained and was designed to make it possible for one to complete it with minimal supervision. To keep students engaged in the tutorial, they were given short homework assignments after each session and were encouraged to collaborate with peers on these programming exercises as they completed the tutorial.

Along with small, practical coding assignments, students were provided with a number of what we refer to as "prompts" (see Supporting Information), which were meant to serve as open-ended introductions to scientific concepts that they could potentially use in their visualizations. This is similar to how a research mentor would pitch a research idea to an incoming graduate student. Each prompt included the following: background information about a physical chemistry concept, resources to find more information about that topic, and a few open-ended questions to help the

students think about how they might want to design their demonstration. The prompts were based on concepts from thermodynamics, such as heat flow, heat capacities, heat engines, ideal solutions, and redox reactions. Students were asked to read the prompts or brainstorm other physical chemistry concepts of interest and then select and research a topic for their project. Each student was tasked with creating a storyboard visualization of his or her Mathematica demonstration, to be used as a guide for completing the project. These storyboards were presented to the class as part of a peer-review process during which the other students, peer leaders, and GSIs were free to ask questions and make suggestions.

Following the storyboarding process, students began a three-week period of programming their demonstrations. They were encouraged to work collaboratively to solve any programming or design problems, with the peer leaders and graduate students available to provide guidance and advice as necessary. After completing the first 3 weeks of programming, students presented drafts of their Mathematica demonstrations to the rest of the class in a second peer-review session. During this second peer review, students were required to discuss how the input from the previous peer-review session impacted the development of their demonstrations. Following this session, students had an additional 1–2 weeks to implement any changes deemed necessary from the review session and to finish programming their demonstrations. At the end of the semester, students were asked to submit a brief paragraph describing their Mathematica demonstration. After the descriptive paragraph was reviewed by peer leaders and edited by students, the demonstrations were submitted to the Wolfram Demonstration Project to undergo official review. Finally, students were asked to submit a one-page reflection describing how they dealt with the comments given in each peer-review process.

By the final week of the inaugural semester of the compute-to-learn studio, every student submitted their individually coded Mathematica demonstration to the Wolfram Demonstrations Web site for a total of six submitted demonstrations; however, only one of the six demonstrations was published. Students received official review from the Wolfram Demonstrations Project at or after the end of the semester. This made it difficult to continue working with the students to address

reviewer comments and incorporate requested changes to complete the publication process. This was clearly a problem that we needed to address in future iterations of the compute-to-learn studio in order to provide a more complete research experience. We discuss the implementation of these and other changes made in two subsequent iterations of the studio later in this article.

7.3 Mathematica Demonstrations

Wolfram Mathematica is a powerful computing environment commonly used in academia, industry, and education.[35] It was chosen as the development tool for the compute-to-learn studio primarily based on its fairly straightforward syntax, availability of tools for creating interactive demonstrations, and the existence of the Wolfram Demonstrations Project, which is a peer-reviewed, open-source web catalog of interactive demonstrations that provides students with a platform for publishing their newly created demonstrations. Students can also take advantage of the existing catalog of 10,000+ demonstrations to survey the capabilities of Mathematica and adopt ideas for their own visualizations, similar to how researchers can review and be inspired by the scientific literature.

The user does not need to have any knowledge of programming in order to interact with the dynamic output of Mathematica, which generally consists of graphics, sliders, and simple button presses. Creating an original demonstration from the ground up, however, requires a basic understanding of the Wolfram Language used in Mathematica, so this is taught in the first few weeks of the studio using the tutorial developed by our team. The code for many published demonstrations is short and concise, making construction of a demonstration an achievable goal in a single semester. Additionally, the external review process of the Wolfram Demonstrations Project serves to simulate the real-world peer-review process involved in publishing a paper in a peer-reviewed scientific journal, thus contributing to an authentic research experience for the students.

An example of a Mathematica demonstration and the corresponding code are shown in Figures 1 and 2, respectively, and can be found on the Wolfram Demonstration Project webpage.[2] This par-

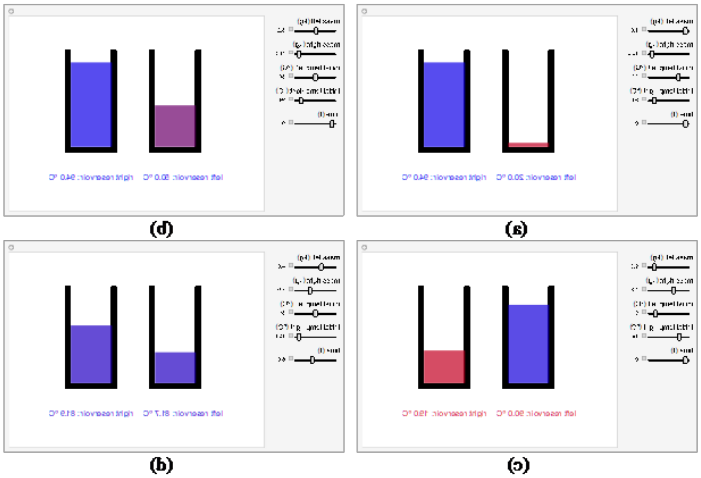


Figure 7.1: Snapshots of a Mathematica demonstration illustrating heat flow between two reservoirs of water. There are two sets of sliders for mass and temperature of each reservoir, which are varied between snapshots a, b, and c. Adjusting the slider for time will update the temperatures of the reservoirs until equilibrium has been reached, as has nearly been achieved in snapshot d. This demonstration is available on the Wolfram Demonstrations Project webpage[2]

ticular example was developed by a student who participated in CHEM 260H in the Fall of 2015. It demonstrates heat flow between a hot and a cold vessel, each containing water. The user is able to interact with the sliders to vary the mass and temperature of both the hot and cold reservoirs, while the visual demonstration calculates the equilibrium temperature in real time. Additionally, there is a slider to show how the system approaches equilibrium with exponential time dependence. The heat capacity is chosen to be a constant. The code in Figure 2 contains several lines (ca. 20) of Mathematica's built-in functions, which produce all of the visual and interactive components of the demonstration.

```
In[1]:= Manipulate[
         Graphics[{
           Polygon[{{0, 0}, {0.2, 0}, {0.2, 0.4}, {0.18, 0.4}, {0.18, 0.02}, {0.02, 0.02}, {0.02, 0.4}, {0, 0.4}}],
           Polygon[
            {{0.33, 0}, {0.53, 0}, {0.53, 0.4}, {0.51, 0.4}, {0.51, 0.02}, {0.35, 0.02}, {0.35, 0.4}, {0.33, 0.4}}],
           RGBColor[1 - .01 θl[Tl, Tr, ml, mr, t], 0, .01 θl[Tl, Tr, ml, mr, t], 0.7],
           Rectangle[{.021, .019}, {0.181, .035 ml}],
           RGBColor[1 - .01 θr[Tl, Tr, ml, mr, t], 0, .01 θr[Tl, Tr, ml, mr, t], 0.7],
           Rectangle[{.35, .019}, {0.511, .035 mr}], Text[
            Row[{"left reservoir: ", NumberForm[N@θl[Tl, Tr, ml, mr, t], {5, 2}], " ºC"}], {.1, -.1}],

        Text[Row[{"right reservoir: ", NumberForm[N@θr[Tl, Tr, ml, mr, t], {5, 2}], " ºC"}], {.44, -.1}]
          }, PlotRange → {{-.1, .6}, {-.2, .5}},
          ImageSize → {400, 300}],

         Row[{"mass left (kg) "}],
         Control[{{ml, 4, ""}, 1, 10, .1, Appearance → "Labeled", ImageSize → Tiny}],
         "",
         Row[{"mass right (kg)"}],
         Control[{{mr, 8, ""}, 1, 10, .1, Appearance → "Labeled", ImageSize → Tiny}],
         "",
         Row[{"initial temp left (ºC) "}],
         Control[{{Tl, 20, ""}, 0, 100, 1, Appearance → "Labeled", ImageSize → Tiny}],
         "",
         Row[{"initial temp right (ºC)"}],
         Control[{{Tr, 80, ""}, 0, 100, 1, Appearance → "Labeled", ImageSize → Tiny}],
         "",
         Row[{"time ", "(t)"}],
         Control[{{t, 0, ""}, 0, 10, Appearance → "Labeled", ImageSize → Tiny}],
         TrackedSymbols ↦ {ml, mr, Tl, Tr, t},
         ControlPlacement → Left,

         Initialization ↦ (

           Te[Tl_, Tr_, ml_, mr_] := (ml Tl + mr Tr)/(ml + mr);
           θl[Tl_, Tr_, ml_, mr_, t_] := Te[Tl, Tr, ml, mr] (1 - e^-t) + Tl e^-t;
           θr[Tl_, Tr_, ml_, mr_, t_] := Te[Tl, Tr, ml, mr] (1 - e^-t) + Tr e^-t;
          )
        ]
```

Figure 7.2: Code used for creating the heat flow demonstration[2] shown in Figure 1.

7.4 Subsequent Implementations of Compute-to-Learn

Since the initial pilot run, the compute-to-learn honors studio has been offered two additional times in the Fall 2016 and Winter 2017 semesters. These subsequent iterations presented opportunities to ameliorate problems that became evident after the first semester (Fall 2015). At the end of the first iteration of compute-to-learn, the students, peer leaders, and GSIs had an informal discussion about the studio in order to generate a set of modifications for improving on the studio experience and execution. For the following iterations, we developed and conducted formal interviews with individual students, which are discussed in more detail in the next section. We sought Institutional Review Board (IRB) approval for these interviews and surveys involving students, which were determined to be exempt from IRB oversight. Most of the concerns that students mentioned were centered on time management and project management, as well as misleading visualizations stemming from misunderstandings of physical chemistry concepts. Common concerns from students revolved around not knowing where to start with their code or experiencing frustration during extended time periods where it was felt that little progress was being made. As a way to help address these issues, as well as further promote collaboration, we encouraged students to work in groups of two or three in the Fall 2016 and Winter 2017 semesters.

Similar to the issues found in the writing-to-teach implementation, we found that students were prone to creating confusing visualizations of concepts. For example, a student who was showing the difference between temperature and heat chose to show heat change on a thermometer. For this reason, we decided to increase the level of GSI involvement to a role similar to that of a research advisor, with the purpose of addressing misconceptions and ensuring physically accurate representations earlier in the semester. Additionally, the GSIs began holding weekly office hours during the Winter 2017 semester so that students had a forum between studio sessions to ask questions and discuss problems they were having.

In the first iteration of the studio, many students indicated that they would have liked to have additional time in the studio allocated to programming their projects, and would rather have spent

less time on the Mathematica tutorial. To accommodate this, we reduced the length of the tutorial from 4 weeks to 3 weeks for both the Fall 2016 and Winter 2017 iterations of the studio. While this presented the students with less formal training in programming with Mathematica, most students expressed their appreciation for having more time to program. With an extra week available for programming, we instituted an additional peer-review session the week prior to submission of the demonstrations, as we found the peer-review sessions and the discussions that took place during them to be very beneficial during the first iteration. This provided the students, peer leaders, and GSIs with another chance to identify mistakes. With this additional peer-review session, students were able to discuss and compare challenges they were facing with their projects, and to provide suggestions or solutions to their peers. Because most of the students in the first iteration did not follow through with publication once the semester had ended, in subsequent iterations we asked all students to submit their demonstrations for publication 3 weeks prior to the end of the semester. This allowed time to receive official review from the Wolfram Demonstrations Project and implement any changes that were required before the end of the semester.

In addition to format changes, we have expanded the studio to include students from the second-semester general chemistry/physical chemistry course for nonchemistry majors (CHEM 230), in order to enhance diversity and increase communication between the two groups of students. CHEM 230 covers many of the same topics as the CHEM 260 course, including thermodynamics, kinetics, electrochemistry, and nuclear chemistry. It excludes calculus-based formulations and quantum mechanics. The CHEM 230H and CHEM 260H students met together within the same studio environment as a single class, enabling collaboration between students with different levels of physical chemistry and mathematics knowledge. Opening the honors studio to CHEM 230 students increased enrollment, and had the benefit of allowing for more collaboration and a greater diversity of projects among students.

Many of the students enrolled in CHEM 230 are interested in health professions and medicine. To tailor the honors studio experience to better fit their needs, we invited a guest speaker from the University of Michigan Medical School during the Winter 2017 iteration of the studio. The invited

speaker was a medical student who uses programming with his own medical studies. He develops demonstrations that visualize and simulate medical topics, such as anesthesia administration rates, that are not easy for students or patients to understand. He also discussed how his skills were used at clinics to help parse through and find patterns in large backlogs of data. The speaker presented computer programming in a context that many of the students could relate to. As one student put it:

He used programming a lot using medical concepts and I thought that was really cool, and right now I'm trying to choose between one or the other and it just gave me the idea, "Why not do both?"

The students were extremely interested in his talk; it further motivated them to learn more about programming and complete their projects.

A comparison of the three iterations of the compute-to-learn honors studio is shown in Table 2. With the modifications made in each successive iteration, the number of publications increased significantly. In the first semester, one of six students published, while in the second iteration of the studio three of four groups published. In the most recent semester, where we had the most students and strictly enforced the submission deadline, all nine groups submitted and published their demonstrations. It should be noted that publishing a demo was not a requirement for completing the studio (see Table 2). Every student who completed the studio acquired the level of mastery necessary for creating an educational demo. Thus, a student who submitted all work, put in a strong display of effort, and created an educationally effective demo could still complete the studio without publishing the demo. Students who dropped out of the studio did so for a variety of reasons, most of which related to health issues and time constraints. Students were not penalized for dropping out of the studio. Surveying the semesters, the studio retained the most students in the Winter 2017 semester, with only about 20% deciding to drop out of the studio compared to 50% in the prior semesters.

Parameters	Fall 2015	Fall 2016	Winter 2017
Courses Involved	CHEM 260	CHEM 260 + CHEM 230	CHEM 260 + CHEM 230
Students Initially Enrolled in Studio	12	16	24
Students Completing Studio	6	8	19
Tutorial Length	4 weeks	3 weeks	3 weeks
Group/Individual Work	Individual	Groups of 1-3	Groups of 1-3
Guest Speaker	No	No	Yes
GSI Involvement	Very little	Biweekly meetings with groups	1 GSI per 3 groups; biweekly meetings
Office Hours	No	No	Yes

Table 7.2: This table summarizes the general enrollment statistics as well as interventions that took place in each iteration of the compute-to-learn curriculum.

7.5 Student Outcomes from the Compute-to-Learn Studio

We conducted formal interviews at the end of the Fall 2016 and Winter 2017 iterations of the compute-to-learn studio to assess student experiences and outcomes. Our hypothesis was that the students enrolled in the honors studio would gain meaningful experience and familiarity with the scientific research process, as well as new skills in programming and problem solving. The interviews were voluntary for the students and were composed of 10 general questions about their experience in the studio (see Supporting Information). The interviews were conducted by studio peer leaders from the previous semester. The students were not acquainted with their interviewer so that they could speak more freely about their experiences. Eight students were interviewed in the Fall 2016 semester and nine students were interviewed in the Winter 2017 semester. Audio recordings of the interviews were transcribed and then coded to identify commonalities and overarching themes. While many of the themes that we identified fit well with our hypothesis, others came as a surprise.

The themes most commonly mentioned by five or more students can be placed into the following seven categories:

1. Desire to learn programming (mentioned by all 17 students)
2. Appreciation for peer review

3. Independent learning

4. Learning to ask questions

5. Benefits of group work

6. Appreciation for hands-on activities

7. Need for patience and practice when faced with a new challenge

The most universal theme we discovered was that every student was either motivated to join the studio because of the opportunity to learn to code or finished the studio with a sense of accomplishment having gained programming skills. As one student described the reason they joined the compute-to-learn studio, "I thought it would be really interesting because I don't have any coding experience, and I don't think I would have had the opportunity to take any classes on coding." After working closely with the students and analyzing the interviews, we found that many of the students shared this sentiment of not having the opportunity or the ability to learn computer programming within the confines of their curricula. The compute-to-learn studio provided students in CHEM 230 and 260 with an environment in which they could learn how to code within the context of a required chemistry course. Many students felt that the greatest skill they learned in the studio was how to code; as one student put it, "Just learning what code really is, and I guess how it works, because I had no interaction with that before."

The remaining six categories fit well into the main purpose of the studio: to apprentice students into a scientific community via an authentic research experience. To identify this, we looked for an impact on the following student abilities:

1. The ability to learn independently while collaborating with a broader community

2. The ability to ask substantive, pertinent questions and to expediently ask a colleague or mentor for assistance when necessary

3. The ability to manipulate or analyze data or information into a presentable form

Many of the students commented on the sense of accomplishment associated with being able to independently learn complex topics or navigate various resources to resolve programming issues, without relying on formal instruction from a teacher. In one student's words, "I think I learned

how to teach myself how to use the online guides and play with them, and kind of just figure it out without much official lecture", while another student described it as, "Alright, this is an individual project and you're going to have to figure it out on your own." Other students even mentioned that they found it more "fun" and fulfilling to research their problems independently and come to a solution without help.

Many students also mentioned that they learned the benefits of knowing when and how to seek help, rather than spend an exorbitant amount of time trying to solve problems on their own. Additionally, they found the peer-review process generally helpful, both from the perspective of the reviewer and as the presenter. As a researcher, the process of official peer review prior to publication in a journal is integral to the scientific process. While we expected students to underappreciate the importance of this part of their project, a sizable number mentioned how important peer review was to submitting a high-quality demonstration. For example, one student offered the following observations:

It was really useful when we would present our storyboard and then everyone else would tell us what they thought about it, because we already have our own opinions on it. I was like, "Oh, I think it's perfect already, the way it is," and the audience members, our peers, were like "Oh, I think you should add that and then you'll clarify this one point for other people," and I was like "Oh, I didn't know that's another way to view my demo." So, be open to other comments that people might have.

In addition to the benefit of peer review, many students discovered the interdisciplinary aspect of research through their participation in the studio. Students learned that they could utilize computer skills, which they originally did not relate with chemistry, to help guide their chemistry research. As one student phrased it, "I definitely think that's a skill, to be able to analyze something and then be able to use your intuition to translate that to another field. Chemistry into computer science, I thought that was a good skill to pick up." Overall, the feedback from the interviews

was positive. In addition, their comments were consistent with our original hypothesis that the compute-to-learn studio constitutes an experience for the students that resembles that of scientific research.

7.6 Future Directions

After completion of three full semesters of the compute-to-learn studio, it is clear that the studio provides a suitable environment for students interested in scientific programming to learn how to write code in Mathematica and to utilize this skill to explain physical chemistry concepts. While we recognize that the low ratio of students to peer leaders and GSIs would make scaling up the studio to a traditional large lecture setting challenging, we would like to emphasize the fact that the Wolfram Mathematica software is incredibly versatile, as the Wolfram Demonstration Project includes demonstrations from a wide variety of fields. As such, the compute-to-learn pedagogy that we implemented within the CHEM 260H and CHEM 230H studio can easily be generalized and extended to many other courses and disciplines. For example, the studio could be easily extended to quantitative courses in Physics, Statistics, Economics, Math, and Engineering. The software and pedagogy is also suitable for more qualitative topics, such as those found in Biology, Medicine, Art, and Architecture.

Going forward, several ways of expanding and strengthening the pedagogy will be investigated. For example, we intend for the peer leaders to take stronger leadership roles in the forthcoming Fall 2017 iteration. We are also considering shortening the tutorial to 2 weeks as we have seen the students' confidence in self-guided learning of programming increase semester to semester. This will give the students more time to research their chosen prompts before presenting their storyboards and beginning to code. Overall, we wish to make note of the positive impact the studio had on the students' ability to problem solve, develop an appreciation for research, and acquire confidence in learning new and useful skills such as programming.

We also plan on expanding our assessment efforts. In Fall 2016 and Winter 2017, we collected

some preliminary data regarding student attitudes about chemistry using the "Colorado Learning Attitudes about Science Survey" (CLASS survey).[36] The survey was given to all students enrolled in CHEM 230 and CHEM 260, including the students enrolled in CHEM 230H and CHEM 260H, at the beginning and end of the semester. The CLASS survey is meant to model the distinctive differences between novice and expert learners' beliefs about learning science and science as a discipline. Ideally, a person who takes part in research would be defined as a more expert learner while a nonresearcher would be a novice learner. At the end of each semester, the surveys showed neither a positive nor a negative relationship for the students enrolled in either the normal CHEM 230/260 course or the CHEM 230H/260H compute-to-learn studio. We will continue to expand and improve the compute-to-learn pedagogy by expanding our assessment and continuing to improve the studio, which we hope will benefit anyone in the scientific community who will consider adopting this pedagogy.

Part III:

Conclusions

CHAPTER 8

Conclusion

This work has made progress toward understanding the full quantitative potential of the second-order Green's function to study real solids. In chapter four of this thesis, I proposed a method to use importance sampling in the Monte Carlo integration of the second-order Luttinger-Ward functional in a way that can easily provide us with the second-order self-energy. It was shown that this sampling scheme was behaving as expected. In Chapter five, I have shown that the newly devised method can be used within the fully self-consistent second-order Green's function scheme with the proper statistical analysis. I described how the use of jackknife resampling provides a rigorous way to complete self-consistency. In chapter six, I have shown that the use of jackknife resampling does indeed control the stochastic error to behave as expected with a $1/\sqrt{N}$ relationship for a chain of 10 Hydrogens and a square plaquette of 16 Hydrogens both in minimal basis sets. Additionally, I showed with a series of results ranging from one dimensional and two dimensional toy systems to more realistic systems like stretched water trimers and amino acid chains, that the metropolis algorithm is extremely beneficial at a large enough system size. Overall, these sections of the thesis showed that by using the method in a slightly altered self-consistent second-order Green's function scheme, we were able to accurately evaluate the energetics of several systems of interest. We also were able to investigate the bias associated with the necessary inversion of data required in the sGF2 iterative procedure, and by way of jackknife resampling showed good control of the bias through iterations of multiple systems. This gives us insight into what can be achieved with the GF2 method. Lastly, in chapter seven I discussed an investigation of the importance to include

coding, such as the coding needed to develop the worked presented in chapters four through six, curriculum into upper level physical chemistry courses.

Overall, this work opens many avenues for future research. To begin, recently an interesting extension of the GF2 formalism has been extensively studied. It is an embedding procedure called Self-Energy Embedding Theory (SEET).[37] A very clear next direction for the sGF2 algorithm is for it to be used within an embedding procedures such as SEET for molecules and solid-state problems. The major concern in this field of study would be how to incorporate the error of the stochastically evaluated bath with the deterministically calculated system. Furthermore, stochastic procedures and importance sampling could be used to make a generic study on the Dyson equation to observe what order of perturbation is necessary to retrieve all the physical information necessary from a particular system. There is still much to be explored in the area of Monte Carlo studies of electronic structure methods. Like the Second-Order Green's function, other many body Green's function formalisms may also benefit from use the importance sampling procedures described. The GW method is one that could seamlessly introduce a nearly identical algorithm as sGF2 to study the results of GW for increasingly large systems. Additionally, it is my hope that this work provides a foundation for using self-consistent methods in electronic structure theory, even when the computational cost seems steep. From this work, it should be clear that much is to be done in the area of not only Green's function research, but stochastic diagrammatic perturbation theory within the field of chemistry.

Electronic structure is a growing field with immense importance to the chemistry, physics, and materials community. Considering the importance of this field, high performance computing and technology in general, I would like to comment on the importance of enhancing and reconstructing the educational program that undergraduate chemists in the United States receive. Most programs do not including a programming requirement, and it is clear that our world in the 21st century is dominated by computer technology. This is important for even a synthetic chemist. Many pharmaceutical companies are obtaining data at a scale never before seen. Chemists entering industrial positions will need to deal with this data in an efficient way. Generally, this should involve using

many of the new sophisticated programming languages meant to deal with the statistics of large amounts of data such as Python or R. Even more, open source software packages are becoming a standard for computational chemists. Learning the standard coding practices is of upmost importance for incoming (and continuing) theoretical chemists. Lastly, I argue that most chemists who use programming languages in their work would agree that they have obtained a deeper understanding of the chemistry behind the quantitative aspect of the field by implementing new codes. Further work must be done to include programming into, not only the undergraduate, but also the graduate level coursework for the chemistry degree at the institutions in the United States. This addition will not only strengthen the educational background of our chemists, but it will also drive forward the field of theoretical chemistry.

Appendix

Appendix A

Interview Questions

1. What were the reasons that made you want to join the honors studio?

2. What was the biggest challenge that you encountered through your participation in the honors studio, and how did you overcome it?

3. What was your biggest accomplishment in the honors studio?

4. What are the most important skills that you learned through your participation in the honors studio?

5. What are the most important lessons that you learned through your participation in the honors studio?

6. What advice would you give to future participants regarding how to be successful at producing a publishable demo?

7. Did participation in the studio have an impact on your view of scientific research, and if so in what way?

8. Did participation in the studio change in any way your view of your own strengths and weaknesses as a scientist, and if so how?

9. Did participation in the studio have an impact on your future professional plans, and if so in what way?

10. In your opinion, how did your participation in the studio impact your performance in CHEM 230/260?

BIBLIOGRAPHY

[1] F. J. Dyson. The radiation theories of tomonaga, schwinger, and feynman. *Phys. Rev.*, 75:486–502, Feb 1949.

[2] Heat flow between two reservoirs. https://demonstrations.wolfram.com/HeatFlowBetweenTwoReservoirs.

[3] Jean-Louis Calais. Density-functional theory of atoms and molecules. r.g. parr and w. yang, oxford university press, new york, oxford, 1989. ix 333 pp. price £45.00. *International Journal of Quantum Chemistry*, 47(1):101–101, May 1993.

[4] M. Fuchs, Y.-M. Niquet, X. Gonze, and K. Burke. Describing static correlation in bond dissociation by kohn–sham density functional theory. *The Journal of Chemical Physics*, 122(9):094116, 2005.

[5] Aron J. Cohen, Paula Mori-Sánchez, and Weitao Yang. Challenges for density functional theory. *Chemical Reviews*, 112(1):289–320, 2012. PMID: 22191548.

[6] Attila Szabo and Neil S. Ostlund. *Modern quantum chemistry: introduction to advanced electronic structure theory*. Dover Publications, 2006.

[7] Gianluca Stefanucci and Robert J. van. Leeuwen. *Nonequilibrium many-body theory of quantum systems: a modern introduction*. 2013.

[8] Eleftherios N. Economou. *Greens functions in quantum physics*. Springer, 1983.

[9] Potthoff. Non-perturbative construction of the luttinger-ward functional. *Condensed Matter Physics*, 9(3):557, 2006.

[10] Alicia Rae Welden, Alexander A. Rusakov, and Dominika Zgid. Exploring connections between statistical mechanics and green's functions for realistic systems: Temperature dependent electronic entropy and internal energy from a self-consistent second-order green's function. *The Journal of Chemical Physics*, 145(20):204106, 2016.

[11] Jordan J. Phillips and Dominika Zgid. Communication: The description of strong correlation within self-consistent greens function second-order perturbation theory. *The Journal of Chemical Physics*, 140(24):241101, 2014.

[12] Gordon Baym and Leo P. Kadanoff. Conservation laws and correlation functions. *Phys. Rev.*, 124:287–299, Oct 1961.

[13] Daniel Neuhauser, Roi Baer, and Dominika Zgid. Stochastic self-consistent green's function second-order perturbation theory (sgf2). 03 2016.

[14] Alexei A. Kananenka, Jordan J. Phillips, and Dominika Zgid. Efficient temperature-dependent green's functions methods for realistic systems: Compact grids for orthogonal polynomial transforms. *Journal of Chemical Theory and Computation*, 12(2):564–571, 2016. PMID: 26735685.

[15] Gordon Baym. Self-consistent approximations in many-body systems. *Phys. Rev.*, 127:1391–1401, Aug 1962.

[16] B Bauer, L D Carr, H G Evertz, A Feiguin, J Freire, S Fuchs, L Gamper, J Gukelberger, E Gull, S Guertler, A Hehn, R Igarashi, S V Isakov, D Koop, P N Ma, P Mates, H Matsuo, O Parcollet, G Pawłowski, J D Picon, L Pollet, E Santos, V W Scarola, U Schollwöck, C Silva, B Surer, S Todo, S Trebst, M Troyer, M L Wall, P Werner, and S Wessel. The ALPS project release 2.0: open source software for strongly correlated systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(05):P05001, may 2011.

[17] John Seely Brown, Allan Collins, and Paul Duguid. Situated cognition and the culture of learning. *Educational Researcher*, 18(1):32–42, 1989.

[18] M Orgill. *Theoretical Frameworks for Research in Chemistry/Science Education*, pages 187–203. Prentice Hall, Upper Saddle River, N.J., 2007.

[19] Jean Lave. *Cognition in practice: mind, mathematics and culture in everyday life*. Cambridge University Press, 2003.

[20] John Seely Brown and Paul Duguid. *The social life of information*. Harvard Business Review Press, 2017.

[21] Arthur L. Wilson. The promise of situated cognition. *New Directions for Adult and Continuing Education*, 1993(57):71–79, 1993.

[22] Troy D. Sadler. Situated learning in science education: socio-scientific issues as contexts for practice. *Studies in Science Education*, 45(1):1–42, 2009.

[23] David Paul. Ausubel. *The psychology of meaningful verbal learning: an introduction to school learning*. Grune Stratton, 1968.

[24] Stacey Lowery Bretz. Novaks theory of education: Human constructivism and meaningful learning. *Journal of Chemical Education*, 78(8):1107, 2001.

[25] M. M. Lombardi. Authentic learning for the 21st century: An overview. *Educause Learning Initiative*, pages 1–12, 2007.

[26] Michael J. Pavelich and Michael R. Abraham. Guided inquiry laboratories for general chemistry students. *Journal of College Science Teaching*, 7, 01 1977.

[27] John J. Farrell, Richard S. Moog, and James N. Spencer. A guided-inquiry general chemistry course. *Journal of Chemical Education*, 76(4):570, 1999.

[28] Michael Prince. Does active learning work? a review of the research. *Journal of Engineering Education*, 93(3):223–231, 2004.

[29] B. P. Coppola P. Varma-Nelson. *Chemist's Guide to Effective Teaching*, pages 155–169. Pearson, Upper Saddle River, N.J., 2005.

[30] Wolfram demonstrations project. http://demonstrations.wolfram.com/.

[31] Wolfram Research, Inc. Mathematica, Version 12.0. Champaign, IL, 2019.

[32] Compute-to-learn: Learn physical chemistry concepts. http://umich.edu/~pchem/compute-to-learn.html.

[33] Jeong-Im Choi and Michael Hannafin. Situated cognition and learning environments: Roles, structures, and implications for design. *Educational Technology Research and Development*, 43(2):53–69, 1995.

[34] *Innovating to learn, learning to innovate*. OECD, 2008.

[35] Customer stories: Using wolfram technologies in technical computing. https://www.wolfram.com/customer-stories/).

[36] Wendy K. Adams, Carl E. Wieman, Katherine K. Perkins, and Jack Barbera. Modifying and validating the colorado learning attitudes about science survey for use in chemistry. *Journal of Chemical Education*, 85(10):1435, 2008.

[37] Tran Nguyen Lan and Dominika Zgid. Generalized self-energy embedding theory. *The Journal of Physical Chemistry Letters*, 8(10):2200–2205, 2017.