

Claremont Colleges

Scholarship @ Claremont

HMC Senior Theses

HMC Student Scholarship

2001

The Approximate Inclusion of Triple Excitations in EOM-type Quantum Chemical Methods

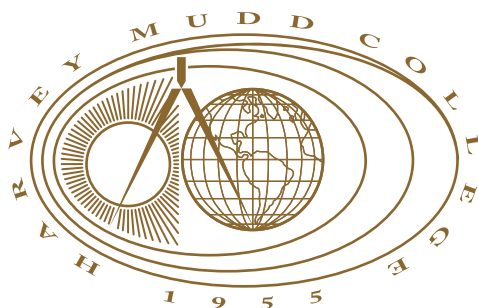
Mike Rust
Harvey Mudd College

Follow this and additional works at: https://scholarship.claremont.edu/hmc_theses

Recommended Citation

Rust, Mike, "The Approximate Inclusion of Triple Excitations in EOM-type Quantum Chemical Methods" (2001). *HMC Senior Theses*. 135.
https://scholarship.claremont.edu/hmc_theses/135

This Open Access Senior Thesis is brought to you for free and open access by the HMC Student Scholarship at Scholarship @ Claremont. It has been accepted for inclusion in HMC Senior Theses by an authorized administrator of Scholarship @ Claremont. For more information, please contact scholarship@claremont.edu.



Estimating the Error in Coupled-Cluster Calculations of Molecular Ground States

by
Michael Rust
Robert Cave, Advisor

Advisor: _____

Second Reader: _____

(Michael Moody)

May 2001

Department of Mathematics

HARVEY MUDD
COLLEGE

Abstract

Estimating the Error in Coupled-Cluster Calculations of Molecular Ground States

by Michael Rust

May 2001

In non-relativistic quantum mechanics, stationary states of molecules and atoms are described by eigenvectors of the Hamiltonian operator. For one-electron systems, such as the hydrogen atom, the solution of the eigenvalue problem (Schrödinger's equation) is straightforward, and the results show excellent agreement with experiment. Despite this success, the multi-electron problem corresponding to virtually every system of interest in chemistry has resisted attempts at exact solution.

Perhaps the most popular method for obtaining approximate, yet very accurate results for the ground states of molecules is the coupled-cluster approximation. Coupled-cluster methods move beyond the simple, average-field Hartree-Fock approximation by including the effects of excited configurations generated in a size-consistent manner. In this paper, the coupled-cluster approximation is developed from first principles. Diagrammatic methods are introduced which permit the rapid calculation of matrix elements appearing in the coupled-cluster equations, along with a systematic approach for unambiguously determining all necessary diagrams. A simple error-bound is obtained for the ground state energy by considering the coupled-cluster equations as entries in the first column of a matrix whose eigenvalues are the exact eigenvalues of the Hamiltonian. In addition, a strategy is considered for treating the error in the ground state energy perturbatively.

Table of Contents

Chapter 1:	Introduction	1
1.1	Dirac Notation	1
1.2	The Schrödinger Equation	2
1.3	The Hartree-Fock Approximation	3
1.3.1	The Born-Oppenheimer Approximation	3
1.3.2	The Pauli Exclusion Principle	4
1.3.3	Independent-Particle States	5
1.3.4	Sketch of Derivation of Hartree-Fock Equations	5
1.4	Unrestricted Hartree-Fock Equations	6
1.4.1	Correlation Energy	7
1.4.2	Performance of the Hartree-Fock Approximation	9
Chapter 2:	Configuration Interaction (CI)	10
2.1	The Hartree-Fock Basis	10
2.2	The Method of Second Quantization	11
2.2.1	Creation and Annihilation Operators	12
2.2.2	Schrödinger's Equation in Second-Quantized Form	12
2.3	Configuration Interaction	13
2.3.1	Linear Excitation Operator	14
2.4	Performance of CI	15
2.5	Size-Inconsistency	16

Chapter 3:	Coupled-Cluster Methods (CC)	18
3.1	Ensuring Size-Consistency	18
3.2	Exponential Excitation Operator	19
3.3	The Coupled-Cluster Equations	20
Chapter 4:	Diagrammatics	21
4.1	Evaluating Matrix Elements	21
4.1.1	Simple Example	21
4.2	Wick's Theorem	22
4.3	Linked Diagram Theorem	24
4.3.1	A Proof of the Linked Diagram Theorem	25
4.4	Diagrams	26
4.4.1	The H Sub-Diagram	27
4.4.2	The T Sub-Diagram	28
4.5	Algebraic Equivalence Rules	28
4.5.1	Closed Diagram Rules	29
4.5.2	Additional Open Diagram Rules	29
4.5.3	Example	30
Chapter 5:	Diagram Generation Algorithm	31
5.1	Excitation Numbers of H and T	31
5.2	Disconnection and Connection Numbers	32
5.3	Legal Diagram Classes	32
Chapter 6:	Error Estimates	35
6.1	The EOM matrix	36
6.2	Gershgorin's Circle Theorem	36
6.3	Matrix Representations of H	37

6.4	A Perturbative Approach	38
6.4.1	Perturbation Theory for Non-Hermitian Operators	39
6.4.2	Perturbative Correction to CCSD Ground State Energy	40
6.5	Discussion	44
Appendix A: Forms of the Hamiltonian Sub-Diagram		45
Appendix B: Matrix Elements in the First Column of $e^{-T}He^T$		46
B.1	Triples Diagrams	47
B.2	Quadruples Diagrams	48
B.3	Quintuples Diagrams	50
B.4	Hexatuples Diagrams	51
Appendix C: Numerical Implementation		52
Bibliography		65

Acknowledgments

I would like to thank Bob Cave for his guidance and support.

Chapter 1

Introduction

The object of this work is to study quantum mechanical descriptions of molecules. It is worthwhile to first give an overview of quantum mechanics. In quantum mechanics, physical systems are described by vectors in a complete vector space over \mathbb{C} with an inner product. The details of the vector space depend on the system under consideration. For molecular systems, the space will generally be infinite-dimensional. As infinite-dimensional vector spaces are not amenable to computational work, we will typically deal with finite-dimensional subspaces.

In accordance with the probabilistic interpretation of the state vector, it is additionally required that the state vectors are normalized. To every physically observable quantity there corresponds a Hermitian operator which maps the vector space into itself. The spectrum of the operator corresponds to the values of the observable which may be measured. In particular, suppose X is a self-adjoint operator corresponding to the observable x . If χ is an eigenvector of X with associated eigenvalue λ and ψ is an arbitrary state vector, then the probability of measuring x for the system described by ψ and obtaining the result λ is given by $|(\chi, \psi)|^2$.

1.1 Dirac Notation

In connection with these ideas, it is useful to introduce a notation due to Dirac. In this scheme, elements of the vector space are referred to as *kets* and are enclosed in the symbol $|\ \rangle$. Elements of the dual space are referred to as *bras* and are enclosed

in the symbol $\langle |$. The reasons for this slightly peculiar terminology are that the action of the dual space on the space itself may be denoted by simple adjacency. That is, the *bra* is next to the *ket* forming a *bra-ket* or “bracket”, which is reminiscent of the standard notation for the inner product.

The principal advantages of Dirac notation come from the importance of Hermitian operators in quantum mechanics. In the usual notation, an inner product involving a self-adjoint operator X may be written $(\phi, X\psi) = (X\phi, \psi)$. The fact that it is immaterial whether a self-adjoint operator acts on the dual space or on the space itself is made transparent in Dirac’s scheme where the inner product would be written as $\langle \phi | X | \psi \rangle$ and it is understood that X may act either “to the right” on the ket vector or “to the left” on the bra vector.

1.2 The Schrödinger Equation

Because of the importance of the energy in determining the dynamics of the system, the fundamental problem in quantum mechanics is determining the eigenvectors and eigenvalues of the Hamiltonian, H , the operator corresponding to the energy, E . The energy eigenvalue equation:

$$H|\psi\rangle = E|\psi\rangle$$

is also sometimes called the time-independent Schrödinger equation.

The form of the Hamiltonian in position space for a single particle is familiar to students of quantum mechanics as:

$$H = -\frac{\hbar^2}{2m}\nabla^2 + V$$

where m is the mass of the particle, V is an operator corresponding to the classical potential energy, and \hbar is Planck’s constant over 2π .

For the hydrogen atom in SI units, the Hamiltonian takes the form:

$$H = -\frac{\hbar^2}{2m_e}\nabla^2 - \frac{e^2}{4\pi\epsilon_0 r}$$

where e is the electron charge, ε_0 is the permittivity of free space, and m_e is the mass of the electron¹. In dealing with electron structure problems it is useful to work in atomic units which non-dimensionalize the above Hamiltonian. To do so, we express length in units of *Bohrs* where one Bohr is equal to the Bohr radius, $\frac{4\pi\varepsilon_0\hbar^2}{m_e e^2}$ in SI. Energy is expressed in *Hartrees* where one Hartree is given by $\frac{e^4 m_e}{(4\pi\varepsilon_0)^2 \hbar^2}$. In atomic units, Schrödinger's equation takes the form:

$$H|\psi\rangle = \left(-\frac{1}{2}\nabla^2 - \frac{1}{r}\right)|\psi\rangle = E|\psi\rangle$$

1.3 The Hartree-Fock Approximation

Let us now consider the general problem of quantum chemistry, the determination of the quantum mechanical description of a multi-electron molecule. Suppose we are given n electrons, and N nuclei. Let $\{Z_i\}$ be a set of natural numbers that specify the electric charge of each of the N nuclei, and let $\{M_i\}$ be a set of reals that specify that mass of each nucleus. We can naturally split the full Hamiltonian for this system into nuclear and electronic pieces as

$$H = H_{nuc} + H_{elec}$$

where, in SI units,

$$H_{nuc} = -\sum_i^N \frac{\hbar^2}{2M_i} \nabla_i^2 + \sum_{i<j}^N \frac{Z_i Z_j e^2}{4\pi\varepsilon_0} \frac{1}{|R_i - R_j|}$$

$$H_{elec} = -\sum_i^n \frac{\hbar^2}{2m_e} \nabla_i^2 - \sum_i^n \sum_j^N \frac{Z_j e^2}{4\pi\varepsilon_0} \frac{1}{|r_i - R_j|} + \sum_{i<j}^n \frac{e^2}{4\pi\varepsilon_0} \frac{1}{|r_i - r_j|}$$

1.3.1 The Born-Oppenheimer Approximation

As the nuclei are at least three orders of magnitude more massive than the electrons, to an excellent approximation it is possible to separate the electronic motion

¹In reality, m_e is the reduced mass of the electron-proton system, but this distinction is of little importance for us since the reduced mass approach is not useful for multi-electron systems

from the nuclear motion. Because of this mass difference, the period of motion of the electrons is much shorter than that of the nuclei[15], and we may regard the electrons as moving much faster than the nuclei so that the nuclear potential appears fixed to the electrons. Treating the nuclei as fixed from the point of view of the electrons is known as the Born-Oppenheimer approximation and is made quantitatively precise elsewhere [11].

Since within the Born-Oppenheimer approximation H_{nuc} does not depend on the electronic coordinates, we see that an eigenvector $|\psi\rangle$ of H_{elec} is the electronic part of an eigenvector of the full Hamiltonian with an eigenvalue different only by a constant corresponding to the nuclear energy for the given geometry. Thus, to an excellent approximation, the problem has been reduced to solving for the electronic eigenvectors which depend only parametrically on the nuclear coordinates.

1.3.2 The Pauli Exclusion Principle

In dealing with quantum mechanical many-body problems, it is vital to ensure that the state vectors have the appropriate symmetry under particle exchange. As electrons have spin- $\frac{1}{2}$, the state vector must be anti-symmetric under exchange. This requirement is often known as the Pauli exclusion principle, and arises naturally in a quantum field theory of the electron[8].

A straightforward method for the enforcement of the Pauli principle is the use of so-called Slater determinants. Given n single-particle states $|\psi_1\rangle, \dots, |\psi_n\rangle$ the n -particle state

$$|\Psi\rangle = \sqrt{\frac{1}{n!}} \begin{vmatrix} |\psi_1(1)\rangle & |\psi_2(1)\rangle & \cdots & |\psi_n(1)\rangle \\ |\psi_1(2)\rangle & |\psi_2(2)\rangle & \cdots & |\psi_n(2)\rangle \\ \vdots & \vdots & \ddots & \vdots \\ |\psi_1(n)\rangle & |\psi_2(n)\rangle & \cdots & |\psi_n(n)\rangle \end{vmatrix}$$

has the appropriate normalization and anti-symmetry (the arguments denote par-

title labels).

For convenience, the Slater determinant formed from $|\psi_1\rangle, \dots, |\psi_n\rangle$ is sometimes denoted simply $|\psi_1\psi_2 \dots \psi_n\rangle$. This convention will be used throughout this paper.

1.3.3 Independent-Particle States

Notice that if the electron-electron term $\sum_{i>j}^n \frac{1}{|r_i - r_j|}$ could be neglected in H_{elec} , the electronic eigenvalue problem would separate simply into pieces depending on the coordinates of only single electrons. Since each electron would in this case move independently of all the others, eigenvectors could be formed as an antisymmetrized direct product of single-particle states, $|\Psi\rangle = |\psi_1\psi_2 \dots \psi_n\rangle$. In the language of partial differential equations, the Schrödinger equation would be separable. For this reason, such a state $|\Psi\rangle$ is referred to as an *independent-particle* state.

Though the eigenvectors of the Hamiltonian are not, in general, independent particle states, these states have a particularly simple form and a transparent physical interpretation. Furthermore, the similarity of the structure of the periodic table to the hydrogenic orbitals should convince us that independent-particle states should be good approximations for at least atomic structure. It is natural, therefore, to seek the “best” independent-particle approximation to the true ground state. This notion leads immediately to the Hartree-Fock approximation.

1.3.4 Sketch of Derivation of Hartree-Fock Equations

It is outside of the focus of this work to present the derivation of the Hartree-Fock equations in detail. For a complete discussion, see a reference text such as Szabo & Ostlund[12]. To make the notion of “best” mentioned above quantitatively precise,

note that the ground state energy satisfies the variational principle ²

$$\langle \Phi | H | \Phi \rangle \leq E_{\text{ground}} \quad \forall |\Phi\rangle$$

The Hartree-Fock equations follow from minimizing the expectation value of the energy of an independent-particle state subject to the constraint that the single-particle states are orthonormal. The single-particle states that are incorporated in the n -electron independent-particle state are sometimes called *orbitals*. The standard picture of electronic structure presented in introductory chemistry courses with electrons occupying orbitals is in reality an inexact model corresponding to the Hartree-Fock approximation.

1.4 Unrestricted Hartree-Fock Equations

It is useful to display the Hartree-Fock equations obtained from the derivation only because they provide useful clues as to the origin of the weaknesses of the approximation. The following are known as the unrestricted Hartree-Fock equations as no steps have been taken to ensure that the ground state is an appropriate eigenstate of total spin[12].

Let $\{|\chi_i\rangle\}$ be a set of single-electron states and denote the coordinates of the n electrons by $1, 2, \dots, n$. Denote the position-space representation of electron j in orbital $\{|\chi_i\rangle\}$ by $\chi_i(j)$. The Hartree-Fock orbitals must satisfy[12]

$$\begin{aligned} \left(-\frac{1}{2}\nabla_1^2 - \sum_i^N \frac{Z_i}{|r_{1i}|} \right) |\chi_a(1)\rangle + \sum_{b \neq a} \left[\int d\vec{r}_2 |\chi_b(2)|^2 \frac{1}{|r_{12}|} \right] |\chi_a(1)\rangle \\ - \sum_{b \neq a} \left[\langle \chi_b | \chi_a \rangle_{\text{spin}} \int d\vec{r}_2 \chi_b(2)^* \chi_a(2) \frac{1}{|r_{12}|} \right] |\chi_b(1)\rangle = \varepsilon_a |\chi_a\rangle \end{aligned}$$

where the “spin” subscript denotes that the inner-product is to be taken over the spin components of the state vectors.

²This can easily be shown by expanding an arbitrary state in the complete basis of eigenstates of H .

It is convenient to define the *Coulomb operator*, $\mathfrak{J}_b(1)$, as[12]

$$\mathfrak{J}_b(1) = \int d\vec{r}_2 |\chi_b(2)|^2 \frac{1}{|r_{12}|}$$

and the *exchange operator*, $\mathfrak{K}_b(1)$ by its action on a state $|\chi_a(1)\rangle$ as

$$\mathfrak{K}_b(1)|\chi_a(1)\rangle = \langle\chi_b|\chi_a\rangle_{spin} \left[\int d\vec{r}_2 \chi_b(2)^* \chi_a(2) \frac{1}{|r_{12}|} \right] |\chi_b(1)\rangle$$

and the *one-electron operator*, $h(1)$ as

$$h(1) = -\frac{1}{2}\nabla_1^2 - \sum_i^N \frac{Z_i}{|r_{1i}|}$$

so that the unrestricted Hartree-Fock equations assume the simple form of an eigenvalue problem:

$$\left[h(1) + \sum_{b \neq a} \mathfrak{J}_b(1) - \sum_{b \neq a} \mathfrak{K}_b(1) \right] |\chi_a(1)\rangle = \varepsilon_a |\chi_a(1)\rangle$$

Although this has the formal appearance of an ordinary eigenvalue problem, it is important to note that it is not since the operators \mathfrak{J} and \mathfrak{K} depend implicitly on the set of orbitals $\{|\chi_i\rangle\}$ we wish to find. In practice, the equations must be projected onto a basis set yielding matrix representations for the above operators. The equation is then iterated until numerical convergence is achieved. The choice of basis set is a subtle and complex issue that will not be discussed here. For details see Szabo & Ostlund[12].

The Hartree-Fock ground state, hereafter denoted by $|\Phi_0\rangle$, is obtained by filling the n lowest-energy orbitals (that is, $|\Phi_0\rangle = |\chi_1\chi_2 \cdots \chi_n\rangle$).

1.4.1 Correlation Energy

We are now in a position to assign physical interpretations to the terms in the Hartree-Fock equation. The $h(1)$ operator is unchanged from the true Hamiltonian

and is simply the kinetic energy of the electrons and the electron-nucleus interaction. The $\hat{\mathcal{K}}$ operator results from the requirement that the state vector be antisymmetric under particle exchange. Note that if two orbitals have opposing spins the $\hat{\mathcal{K}}$ operator does not contribute while orbitals with identical spins feel a repulsive potential based on their spatial overlap, consistent with the Pauli exclusion principle. $\hat{\mathcal{K}}$ has no classical interpretation. The $\hat{\mathcal{J}}$ operator can be simply interpreted as the average Coulomb interaction between the electrons. In the Hartree-Fock approximation, each electron feels an average potential resulting from the electrons in the other $n - 1$ orbitals. For this reason, the Hartree-Fock approximation is said to be an average-field approximation.

Recall that the Hartree-Fock ground state was constructed to be the optimum independent-particle state vector in a variational sense. Of course, the true ground state is not an independent-particle state. In the language of quantum mechanics, we may say that the true eigenvectors are entangled states. A semi-classical argument can give us a clue as to the nature of the error made in insisting on an independent-particle ground state. If we imagine that the electrons possess definite trajectories, then it is clear that in the true ground state two electrons cannot spend much time on average near each other due to their Coulomb repulsion. In the Hartree-Fock Hamiltonian, however, the electrons do not experience an instantaneous Coulomb interaction with each other, but rather an average field obtained by averaging out the trajectories of the other electrons. Since there is nothing in the Hartree-Fock theory that prevents two electrons from being found near each other in space due to their Coulomb interaction, it is sometimes said that the electron trajectories are not correlated and that the Hartree-Fock ground state lacks correlation³. For this reason, the difference between the energy of the Hartree-Fock

³This is not strictly true since the non-local $\hat{\mathcal{K}}$ operator does correlate the motions of electrons with like spins due to the requirements of the Pauli exclusion principle. It is perhaps more accurate to say that the Hartree-Fock ground state lacks Coulombic correlation.

ground state and the exact (within the basis set) energy of the true ground state is known as the *correlation energy*.

$$E_{\text{corr}} = E_{\text{exact}} - \langle \Phi_0 | H | \Phi_0 \rangle$$

1.4.2 Performance of the Hartree-Fock Approximation

How significant are the effects of electron correlation? It is generally very difficult to answer this question. In many condensed-matter systems, such as superconductors, correlation effects are so important as to make the physics fundamentally different from an average-field description.

In an absolute sense, the Hartree-Fock approximation performs very well on molecular systems. Typical calculations yield the total energy of a molecule to better than 5% accuracy. While this kind of performance may bolster our faith in the correctness of quantum mechanics, it is insufficient to obtain an accurate description of chemical processes. This is because the total energy of a molecule is irrelevant in calculating chemical dynamics[12]. It is instead the differences in energy between different species or geometries that determine the chemistry.

Unfortunately, the small relative error in Hartree-Fock calculations of ground state energies is typically on the order of electron Volts, which is characteristic strength of a chemical bond. Because of this, there are cases where the Hartree-Fock description is qualitatively incorrect, despite its high accuracy relative to the total energy. An example of qualitatively incorrect chemistry is the incorrect ordering of the N_2 ionization potentials given by the Hartree-Fock approximation[12].

Since the error committed by the Hartree-Fock approximation is significant for important problems in quantum chemistry, a number of techniques have been developed for partially recovering the correlation energy. The rest of this work will be devoted to these so-called *correlated methods*.

Chapter 2

Configuration Interaction (CI)

Although the purpose of this work is to study coupled-cluster methods, it is useful to first study a less complex approach to electron correlation known as *configuration interaction*. Although the meaning of the term may not be initially transparent, configuration interaction (hereafter CI) is a conceptually very simple approach to electron correlation. The CI strategy is to diagonalize the Hamiltonian in a limited subspace that includes so-called “excited configurations”. This statement will be made more precise below.

2.1 The Hartree-Fock Basis

Recall that in forming the Hartree-Fock ground state, we took $|\Phi_0\rangle$ to be an antisymmetrized product of the n Hartree-Fock orbitals with lowest energy. As a self-adjoint operator, however, the Hartree-Fock operator possesses a complete set of eigenvectors. We can therefore express corrections to the Hartree-Fock ground state in terms of states formed using higher energy orbitals from the Hartree-Fock basis.

It is worthwhile to introduce some more notation and terminology. A Slater determinant formed from a given set of n Hartree-Fock orbitals is called a *configuration*. The Hartree-Fock ground state $|\Phi_0\rangle$ is sometimes called the *reference configuration*¹. For simplicity it is useful to label the n orbitals with the Roman

¹Some of the utility of this language is that it allows one to generalize correlated methods so that they refer to an arbitrary set of single-particle orbitals, rather than the Hartree-Fock basis in particular.

letters i, j, \dots and higher energy orbitals with a, b, \dots , so that the ground state is $|\Phi_0\rangle = |ij\dots\rangle$, ignoring the fact that for any calculation of interest the first part of the alphabet runs into the second².

Configurations that differ from the reference $|\Phi_0\rangle$ are known as *excited configurations*. These configurations are classified by the number of orbitals in which they differ from the reference state. Thus, a configuration which differs by only one orbital is said to be singly-excited, a configuration which differs by two orbitals is said to be doubly-excited, etc. . . The terminology of excited configurations is somewhat unfortunate as it immediately suggests the physical excited states of molecules which are the true eigenvectors of the Hamiltonian corresponding to higher energies. The concept of “excited” configuration is quite distinct from that of the true excited states³.

Finally, excited configurations are often indicated by simply listing the orbitals that are newly occupied in the excited configuration above those which are freshly unoccupied inside of the ket symbol. Thus a configuration with virtual orbital a filled in favor of orbital i is denoted $|_i^a\rangle$.

2.2 The Method of Second Quantization

We now present an operator formalism known as second quantization⁴ capable of handling the requirements of many-body quantum mechanics in a more convenient way. The true utility of this method will become apparent in the next chapter in the discussion of coupled-cluster theory.

²The orbitals which are not occupied in the Hartree-Fock ground state are sometimes called *virtual orbitals*[12]

³Though in some cases excited configurations can provide rough approximations to the excited state vectors.

⁴“First quantization” is simply the replacement $p \rightarrow \frac{\hbar}{i}\nabla$ of wave mechanics[8].

2.2.1 Creation and Annihilation Operators

As discussed previously, the Pauli principle requires that a quantum mechanical state vector for a many-body electron system be antisymmetric under exchange. Our goal is to recast the theory in terms of operators whose commutation properties naturally ensure antisymmetry. To do this, we introduce an operator x^\dagger whose action on the vacuum state $|\rangle$ is to create an electron in the orbital labelled by x . That is, $x^\dagger|\rangle = |x\rangle$. It can be verified that the action of the adjoint operator x is to annihilate an electron in the orbital labelled by x ⁵.

Define the *anticommutator* of operators A and B by $\{A, B\} = AB + BA$. It may be shown that $\{i, j^\dagger\} = \delta_{ij}$ and $\{i, j\} = \{i^\dagger, j^\dagger\} = 0$. These anticommutation relations encapsulate the requirement that electron state vectors be antisymmetric under exchange since

$$|ab\rangle = b^\dagger a^\dagger |\rangle = -a^\dagger b^\dagger |\rangle = -|ba\rangle$$

$$a^\dagger a^\dagger |\rangle = -a^\dagger a^\dagger |\rangle = 0$$

2.2.2 Schrödinger's Equation in Second-Quantized Form

In order to express the theory entirely in second quantization, we must express the Hamiltonian in terms of the creation and annihilation operators. As can be verified by comparing with the projections of the ordinary form of the Hamiltonian, the second-quantized form of the H is[3]

$$H = \sum_{pq} \langle p|h(1)|q\rangle p^\dagger q + \frac{1}{4} \sum_{pqrs} \left(\langle pq|\frac{1}{|r_1 - r_2|}|rs\rangle - \langle pq|\frac{1}{|r_1 - r_2|}|sr\rangle \right) p^\dagger q^\dagger sr$$

⁵This presentation is casual. It is not *a priori* obvious that the operators should have these properties. The formalism is motivated by the quantum mechanical treatment of the harmonic oscillator. For more details, see Szabo & Ostlund[12] or, for a rigorous discussion, Sakurai[8] or Hoffmann & Schaeffer[4].

Where $h(1)$ is the one-electron piece of the Hamiltonian as previously defined and the sums are over the entire single-particle basis.

It is customary to denote $\langle p|h(1)|q\rangle$, the so-called *one-electron integral*, by f_{pq} and, in an unfortunate choice of notation, the *antisymmetrized two-electron integral* appearing in the second term by $\langle pq||rs\rangle$. After performing a Hartree-Fock calculation it is a straightforward matter to determine the values of these expressions using the Slater-Condon rules[12]. The second-quantized Hamiltonian is typically written using this notation as

$$H = \sum_{pq} f_{pq} p^\dagger q + \frac{1}{4} \sum_{pqrs} \langle pq||rs\rangle p^\dagger q^\dagger sr$$

2.3 Configuration Interaction

Since the Hartree-Fock basis spans the space of single-particle states, the set of all excited configurations spans the full n -electron space⁶. Clearly, if we could obtain a diagonal representation of H in this basis, this would yield the exact eigenvectors and eigenvalues to within the accuracy of the underlying calculation basis. Simple combinatorial reasoning should convince the reader that performing this diagonalization is a practical impossibility for any sizable system. The exact results that would be obtained from such a full diagonalization are often known as the full CI results and are primarily useful as a standard for the success of inexact correlated methods.

The lesser CI methods perform a less demanding calculation by diagonalizing H within a subspace of excited configurations. The variants of CI are sometimes denoted by appending letters to the end of the acronym to indicate the class of excitations over which the diagonalization is performed. For example, diagonalization of H over the space of singly and doubly excited determinants would be denoted

⁶More precisely, it spans the n -electron space where each one-electron subspace is the incomplete subspace spanned by the basis set chosen at the Hartree-Fock level.

CISD, adding triple excitations into the subspace would yield CISDT, etc...

2.3.1 Linear Excitation Operator

Though it is conceptually simple to think of CI in terms of diagonalizing a matrix representation of H , the CI procedure can be recast in the language of second quantization. The advantages of taking this step may not be clear at this point, but its utility will become apparent when we turn our attention to coupled-cluster theory in the next chapter.

Define the excitation operator T_n , a linear operator which produces excitations of order n from the reference state, as[3]

$$T_n = \left(\frac{1}{n!}\right)^2 \sum_{ab\dots ij\dots} t_{ij\dots}^{ab\dots} a^\dagger b^\dagger \dots j i$$

where the indices a, b, \dots and i, j, \dots are over virtual and occupied orbitals, respectively. The factor of $\left(\frac{1}{n!}\right)^2$ occurs because of the multiple-counting caused by the unrestricted summation.

Using this notion of excitation operators, we can restate the CI method in operator language. For clarity of exposition, we will deal with CISD, though the generalization to any form of CI is straightforward. Introduce the linear excitation operator S which includes all relevant excitations for CISD as

$$S = I + T_1 + T_2$$

The CISD approximation amounts to writing the eigenvectors $|\psi\rangle$ as

$$|\psi\rangle = S|\Phi_0\rangle$$

Schrödinger's equation can then be written

$$(H - E_{\text{CI}})S|\Phi_0\rangle = 0$$

Diagonalizing H in the appropriate subspace amounts to projecting this equation with excited configurations and solving for the sets of t amplitudes that appear in the S operator. We can think of the CI approach as a “linear ansatz” for obtaining the true eigenvectors from the reference state. Again, this excitation operator formalism is not the most natural way to think about CI, but will become very important in coupled-cluster theory.

2.4 Performance of CI

One of the principal advantages of CI is that it provides a straightforward method for approximating excited states and excitation energies. Recall that the Hartree-Fock equations minimize the total energy under the independent-particle approximation, thus there is no obvious way to obtain good excited state descriptions⁷. CIS provides a relatively inexpensive approximation to the excited states. In fact, this is all that the inclusion of singly-excited states alone does since the matrix elements $\langle_i^a | H | \Phi_0 \rangle$ necessarily vanish. If they did not, this would violate the variational requirement of the Hartree-Fock equations⁸. Thus CID is the simplest level of configuration interaction that can introduce correlation to the ground state.

Carrying out CI calculations including highly-excited configurations is very computationally intensive because the dimension of the sub-block which must be diagonalized grows combinatorially with the excitation level. For example, in a calculation with n electrons and N_B basis functions there are $\binom{n}{2} \binom{N_B - n}{2}$ doubly-excited configurations.

⁷Actually, this is not entirely true. It is possible to converge the Hartree-Fock calculation to the lowest energy state of a given symmetry using group-theoretic methods. See Tinkham[14], for example.

⁸i.e. the Hartree-Fock orbitals are optimal in the sense that including other configurations that differ by only one orbital cannot improve the ground state energy. This result is known as Brillouin’s theorem[12].

For most molecules of reasonable size it is feasible to carry out calculations to the CISD level, but not beyond. Szabo & Ostlund[12] present the results of several illustrative calculations demonstrating the performance of CISD. As an example to compare to the Hartree-Fock approximation, a calculation of the ionization potentials of N_2 in CISD correctly predicts the ordering of the first two potentials but still shows noticeable disagreement with experiment (0.452 a.u. *vs.* 0.463 a.u.)⁹.

2.5 Size-Inconsistency

Performing CI calculations to high orders of excitation can be very computationally expensive, but this is not the most serious objection to CI. Consider a system of n molecules which are spatially separated to the point that they can be regarded as non-interacting. Physically, the energy of the entire system must be simply n times the energy one of the monomers. Any approximation which does not agree with this fact is said to be *size-inconsistent*. Since the full CI result is exact, it is certainly size-consistent. However, it turns out that any truncation of the S operator results in a method which is not size-consistent.

To see why this is true we will consider a model system treated with CID, though it should be immediately clear that the same argument will hold for any flavor of CI that does not include all excitations. In general, the ground state energy of a molecule treated with CID will be lower than the Hartree-Fock energy because the ground state description is improved by mixing in doubly-excited configurations. Denote the energy of the ground state in the CID approximation by E_{CID} and the improved ground state vector by $|\theta\rangle$. Now consider two such molecules that are well-separated so that they may be regarded as non-interacting. In order to achieve a total energy $2E_{CID}$ as a size-consistent theory must, we must be able to

⁹Calculation performed in the 39-STO basis set, reported by B. J. Rosenberg & I. Shavitt in *J. Chem. Phys.* 63:2162 (1975), cited in Szabo & Ostlund[12].

form a total state vector that is a antisymmetrized direct product involving the CID ground state for the single molecule system as $|\Theta\rangle = |\theta_1\theta_2\rangle$ (where notation is being used rather loosely so that the subscripts denote the vector space corresponding to the two different molecules.) Since $|\theta\rangle$ includes doubly-excited configurations, however, the product $|\theta_1\theta_2\rangle$ must include configurations that are doubly-excited on *both* molecules. But this corresponds to a quadruple excitation in the system that includes both molecules and can therefore not be included in a CID description.

Thus, truncated CI is not size-consistent. Worse, it can be shown[12] that the correlation energy recovered by truncated CI per particle goes to zero as the number of particles goes to infinity. For treating a system consisting of large number of subunits such as a crystal, truncated CI is nearly useless.

Chapter 3

Coupled-Cluster Methods (CC)

The focus of this work is on a set of approximate methods known as *coupled-cluster* methods. Coupled-cluster (hereafter, CC) theory permits superior accuracy in ground state calculations by explicitly correcting the principal flaw in CI, size-inconsistency.

3.1 Ensuring Size-Consistency

In the previous chapter, we saw that the linear excitation operator S introduced in CI failed to produce size-consistent results when truncated. We now seek an operator \mathfrak{S} which is explicitly constructed to ensure size-consistency, regardless of truncation.

One easy way to motivate the form of \mathfrak{S} is to consider a model system of n non-interacting one-electron atoms. Unfortunately, this approach requires some suspension of disbelief since the Hartree-Fock orbitals for a hydrogenic atom are exact. Suppose that we have obtained a perverse set of orbitals from a non-Hartree Fock calculation so that the true ground state for each atom is a linear combination involving the virtual orbitals. In this case CIS would provide the exact ground state for a single atom, although, as we have seen, CIS applied to the n -atom system would not produce the true ground state.

What would be required for a size-consistent description of this model system? Clearly, all of the physics¹ is included in the T_1 amplitudes for single excitations

¹or “all of the chemistry”, if you prefer.

since the atoms do not interact. However, \mathfrak{S} must contain terms which generate higher-order excited configurations to produce the true eigenvectors of the n -atom Hamiltonian. Denote by \mathfrak{S}_k the term in \mathfrak{S} that generates k -tuple excitations. For simplicity, consider the properties of \mathfrak{S}_2 . Producing a double excitation is exactly the same as producing two single excitations on distinct atoms since there is no inter-atomic interaction. Thus the amplitude to produce a double excitation must be the product of the T_1 amplitudes to produce the component excitations. It is tempting to write $\mathfrak{S}_2 = T_1^2$, but this is not right since it does not specify which T_1 operator generates which excitation and therefore over-counts by a factor of 2. The correct expression is

$$\mathfrak{S}_2 = \frac{T_1^2}{2!}$$

Identical reasoning for the general term yields

$$\mathfrak{S}_k = \frac{T_1^k}{k!}$$

Thus, to ensure size-consistency in this model system, \mathfrak{S} must take the form

$$\mathfrak{S} = \sum_k \mathfrak{S}_k = \sum_k \frac{T_1^k}{k!} = e^{T_1}$$

3.2 Exponential Excitation Operator

In general, systems of interest have more than one electron, so it is useful to consider a generalized excitation operator e^T where $T = \sum_m T_m$. Using this operator², the coupled-cluster ground state can be written in terms of the reference state as

$$|\Psi\rangle = e^T |\Phi_0\rangle$$

If T is not truncated, then the coupled-cluster ground state is exact and identical with the full CI result. The advantage of coupled-cluster theory is that, as the

²Choosing this form for an excitation operator is often referred to as the “coupled-cluster ansatz”.

above argument demonstrates, even if T is truncated the method is still manifestly size-consistent³.

As with CI, CC methods can be classified based on the terms they retain in T . For example, the coupled-cluster method with $T = T_1 + T_2$ is abbreviated CCSD.

3.3 The Coupled-Cluster Equations

The Schrödinger equation in the CC approximation takes the form

$$(H - E_{\text{CC}})e^T|\Phi_0\rangle = 0$$

In order to obtain equations which express the unknown t amplitudes and the CC energy in terms of one- and two-electron integrals, the Schrödinger equation is projected with excited configurations including excitations up to the same order as the terms retained in T . For CCSD, the case we will be concerned with, the so-called CCSD equations can be written

$$\langle\Phi_0|(H - E_{\text{CC}})e^T|\Phi_0\rangle = 0$$

$$\langle_i^a|(H - E_{\text{CC}})e^T|\Phi_0\rangle = 0$$

$$\langle_{ij}^{ab}|(H - E_{\text{CC}})e^T|\Phi_0\rangle = 0$$

Unlike the CI case, it is non-trivial to evaluate operator products such as

$$\langle_{ij}^{ab}|(H - E_{\text{CC}})e^T|\Phi_0\rangle$$

The next chapter will be spent developing machinery for quickly performing these calculations.

³Coupled-cluster theory is not the only size-consistent approach to electron correlation. A more familiar approach is to introduce the correlation perturbatively with the perturbing Hamiltonian $H' = H_{\text{exact}} - H_{\text{HF}}$. The development of this perturbative expansion is known as Møller-Plesset perturbation theory. While conceptually attractive, this approach is both computationally expensive, and, more seriously, the perturbative expansion diverges for many molecular systems of interest[10]

Chapter 4

Diagrammatics

4.1 Evaluating Matrix Elements

Fundamentally, there is nothing complicated behind the process of evaluating a matrix element involving an string of Fermi creation and annihilation operators. One simply applies the anticommutation relation $\{i, j^\dagger\} = \delta_{ij}$ to move all annihilation operators to the right. The resulting form of the operator string is sometimes said to be *normal-ordered*[3]. The more sophisticated methods discussed below are simply machinery for generalizing the results of this procedure of algebraic manipulation.

4.1.1 Simple Example

To see how this procedure operates, it is worth working out a detailed example of the kind of operator product which appears in the derivation of the CCSD equations. For simplicity, the matrix element will be evaluated against the true vacuum state $|\rangle$, although it is straightforward to extend the notion of normal-ordering to work with a Fermi vacuum defined by the reference state $|\Phi_0\rangle$ [3]. In such a case, the normal-ordering would position all operators which give zero when acting on the Fermi vacuum to the right.

We wish to evaluate

$$\langle |tpq^\dagger u^\dagger| \rangle$$

which is similar to a term appearing in the matrix element $\langle \Phi_0 | He^T | \Phi_0 \rangle$. Applying

the anticommutation relation gives

$$\begin{aligned}
tpq^\dagger u^\dagger &= \delta_{pq} tu^\dagger - tq^\dagger pu^\dagger \\
&= \delta_{pq} \delta_{tu} - \delta_{pq} u^\dagger t - tq^\dagger pu^\dagger \\
&= \delta_{pq} \delta_{tu} - \delta_{pq} u^\dagger t - \delta_{tq} pu^\dagger + q^\dagger tpu^\dagger \\
&= \delta_{pq} \delta_{tu} - \delta_{pq} u^\dagger t - \delta_{tq} \delta_{pu} + \delta_{tq} u^\dagger p + q^\dagger tpu^\dagger \\
&= \delta_{pq} \delta_{tu} - \delta_{pq} u^\dagger t - \delta_{tq} \delta_{pu} + \delta_{tq} u^\dagger p + q^\dagger t \delta_{pu} - q^\dagger t u^\dagger p \\
&= \delta_{pq} \delta_{tu} - \delta_{pq} u^\dagger t - \delta_{tq} \delta_{pu} + \delta_{tq} u^\dagger p + q^\dagger t \delta_{pu} - \delta_{tu} q^\dagger p + q^\dagger u^\dagger tp
\end{aligned}$$

At this point, all annihilation operators lie to the right and the operator is normal-ordered. Evaluation against the vacuum state yields only the constant terms:

$$\langle |tpq^\dagger u^\dagger| \rangle = \delta_{pq} \delta_{tu} - \delta_{tq} \delta_{pu}$$

Matrix elements that involve excited states (or in our case states excited relative to the reference) can be evaluated similarly by writing the excited states in terms of operators acting on the vacuum, thereby extending the length of the operator string[3]. Although this procedure is conceptually straightforward, it should be apparent that it would be both time-consuming and error-prone for the significantly longer operator strings involved in the derivation of the coupled-cluster equations. To simplify this process, we turn to a result known as Wick's theorem.

4.2 Wick's Theorem

Notice that in the previous example the only terms that contributed were those that arose from non-anticommutivity of creation and annihilation operators referring to the same orbitals. With this process in mind we define the *contraction* between two creation or annihilation operators P and Q , denoted by drawing an overhead line linking P and Q , as the difference between PQ and the normal-ordering of PQ . It

is straightforward to see from this definition that the contraction between P and Q will vanish as expected unless Q is a creation operator and P is an annihilation operator in which case the contraction is δ_{pq} .

Wick's theorem[17] allows us to use this notion of contraction to more efficiently evaluate operator products. According to Wick's theorem we may determine the value of a matrix element such as $\langle \Phi_0 | ABC \cdots | \Phi_0 \rangle$ merely by forming all possible contractions between creation and annihilation operators in the operator string, that is by drawing overhead lines between pairs of creation and annihilation operators. The sign of the contraction is determined by the parity of the number of line-crossings: an even number of crossings gives a positive contribution while an odd number of crossings introduces a minus sign.

Since any operators left over after the contraction would cause the matrix element to vanish, we need only consider *fully contracted* pairings of operators (i.e. pairings where every creation operator is contracted with an annihilation operator). Returning to the previous example, $\langle |tpq^\dagger u^\dagger| \rangle$, it is clear that there are only two possible full contractions of the operators. Pairing p with q and t with u results in no line-crossings, so this term contributes with a plus sign. In contrast, pairing p with u and t with q results in a crossing and therefore a minus sign.

Therefore, by Wick's theorem, the above matrix element is equal to $\delta_{pq}\delta_{tu} - \delta_{pu}\delta_{tq}$, in agreement with the more cumbersome previous derivation that relied on direct application of the anticommutators. Nevertheless, attempting to work out all of the matrix elements in, for example, the appendices using only Wick's theorem would be a Herculean feat. It is for this reason that we must develop still more mathematical technology to rapidly evaluate matrix elements.

4.3 Linked Diagram Theorem

In deriving the coupled-cluster equations, we must project Schrödinger's equation with the ground state and excited configurations to obtain algebraic equations specifying the t excitation amplitudes. Schrödinger's equation

$$He^T|\Phi_0\rangle = Ee^T|\Phi_0\rangle$$

can be recast by multiplying on the left by e^{-T} yielding

$$(e^{-T}He^T - E)|\Phi_0\rangle = 0$$

According to Wick's theorem, in order to evaluate the projections of this equation, we must include contributions from every contraction within the string of creation and annihilation operators appearing in such a product.

In reality the structure of the excitation operator e^T ensures that a fortuitous cancellation will occur. In particular, terms in which H does not share at least one contraction with every factor in the excitation operator will ultimately cancel out. Contracted terms for which this requirement is met are called *linked*. Terms which do not satisfy the property are called *unlinked*¹. To denote an expression whose Wick expansion is to be done using only linked contractions, the operator is sometimes given the subscript C ². This claim is stated succinctly in the following

Theorem 1 (Linked Diagram Theorem) *Let T be an excitation operator and H a second-quantized Hamiltonian. Then the only non-zero contributions to matrix elements of $e^{-T}He^T$ arise from terms in which H is contracted with each factor of T present. Thus $e^{-T}He^T = (He^T)_C$.*

¹Unlinked terms are related to size-inconsistency which is why they do not appear in the explicitly size-consistent coupled cluster methods. In fact, size-consistency is sometimes defined by the exclusion of these unlinked terms[12].

²"C", confusingly, is not the first letter of the word "linked". This notation reflects the inconsistent usage of the terms "linked" and "connected" in the literature. In this work, "linked" refers to diagrams in which the Hamiltonian connects to every piece of the excitation operator while "connected" refers to an excitation operator which has only one factor. Thus, T_3 is a *connected* triple excitation while T_1T_2 or T_1^3 are *disconnected* triple excitations.

4.3.1 A Proof of the Linked Diagram Theorem

We must establish the claim that in every non-vanishing contribution to $e^{-T}He^T$, H contracts with each copy of the T operator present in the relevant term in the expansion. The first step is to rewrite $e^{-T}He^T$ in a form involving nested commutators of H and T , the so-called Hausdorff expansion:

$$e^{-T}He^T = H + [H, T] + \frac{1}{2!}[[H, T], T] + \frac{1}{3!}[[[H, T], T], T] + \dots$$

as can be verified by multiplying out $e^{-T}He^T$ and gathering terms on the number of factors of T present in each.

Having recast the transformed Hamiltonian in this form, we see that the theorem will be proved if each nested commutator appearing in the Hausdorff expansion can be shown to vanish when H is not contracted with every excitation operator. The zeroth order term in the expansion, H , trivially satisfies the theorem. The first order commutator $[H, T]$ is also easily seen to satisfy the theorem since a failure to contract between H and T implies that H and T commute and the term vanishes identically.

To prove the claim for an arbitrary term in the Hausdorff expansion, it is useful to label the copies of T in a general nested commutator in order to verify that H must contract with each factor present. Departing briefly from the earlier notation³, the factors of T will be labelled with indices 1, 2, ... so that the third order term in the Hausdorff expansion is written $\frac{1}{3!}[[[H, T_1], T_2], T_3]$.

Consider a k th order nested commutator from the Hausdorff expansion:

$$[[\dots [[H, T_1], T_2] \dots, T_{k-1}], T_k]$$

If this term is not to vanish identically then at least the inner-most commutator $[H, T_1]$ must be non-zero, since every operator commutes with zero. This estab-

³This indexing of the T factors should not be confused with the earlier (and standard) notation in which $T = T_1 + T_2 + \dots$ reflecting the order of excitation of each term in T . Here, the indices label different copies of the complete operator T including all relevant orders of excitations

lishes that H must contract with T_1 . To see that H must contract with the other T_i operators, the crucial observation is that the excitation operators commute, that is $[T_i, T_j] = 0$. We will now make use of the following operator identity:

$$[B, C] = 0 \implies [[A, B], C] = [[A, C], B]$$

Note that applying this identity at any level of a nested commutator with $B = T_i$, $C = T_{i+1}$ and A a commutator involving H allows us to swap T_i and T_{i+1} . Repeated application of this identity allows any T_i to be moved into the innermost commutator. If the entire term is not to vanish, this implies that H cannot commute with any of the T_i . Thus, H contracts with every copy of the T operator.

There are some interesting corollaries to this theorem. Since H contains at most four creation and annihilation operators it can contract with at most four T operators. Therefore, the Hausdorff expansion appears to go out to infinite order, but in fact terminates after the fourth order commutator.

4.4 Diagrams

It is possible to derive the coupled-cluster equations by evaluating matrix elements using Wick's theorem and the linked diagram theorem, as is done in the detailed examples in Crawford & Schaeffer's review[3]. Although this is a clear improvement over the "bare-bones" anticommutator approach, it is still a quite tedious exercise in combinatorics and, more seriously, error-prone.

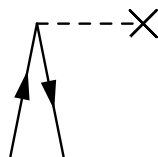
To evaluate matrix elements in a much more efficient way that is also more intuitively appealing, it is useful to develop a diagram formalism introduced to quantum chemistry by Čížek[16] and popularized by the Bartlett group. The rules for manipulating algebraic expressions (commutators, Wick's theorem, etc...) are replaced with a set of rules for drawing diagrams. A "dictionary" of rules is then used to write the algebraic expression corresponding to each legal diagram. The calculational appeal of the diagrammatic method is that each diagram stands for

a multiplicity of Wick contractions which it ultimately would have been necessary to combine in a purely algebraic derivation.

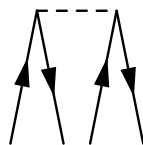
Generally speaking, diagrams are composed of directed *lines* and *vertices*. The vertices correspond to the factors of the operator product that occur in normal order. For example, each T_i operator is associated with a vertex. Lines are associated with creation and annihilation operators depending on their directionality and whether they are entering or leaving a vertex. An upward-going line corresponds to a virtual orbital and represents a creation operator if it is leaving a vertex and an annihilation operator if entering a vertex. Likewise, a downward-going line corresponds to an occupied orbital in the reference and represents a creation operator when leaving a vertex and an annihilation operator when entering⁴.

4.4.1 The H Sub-Diagram

The Hamiltonian operator consists of both one- and two-body terms, either of which may contribute in a matrix element. As the one-electron term contains one creation and one annihilation operator, the sub-diagram representing this operator must have one in-going and one out-going line. It is symbolized by



Likewise, the two-body term of the Hamiltonian must have two in-going and two out-going lines. It is symbolized by

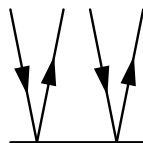


⁴Borrowing the language of the Dirac hole-theoretic treatment of the positron[8], upward-going lines are frequently called *particle* lines and downward-going lines are called *hole* lines.

It is important to note that the Hamiltonian sums over all orbitals, regardless of whether they are virtual or filled. In terms of diagrammatics, this means that the lines connected to the Hamiltonian sub-diagram could be any combination of upward- and downward-going (hole and particle). The different forms of the Hamiltonian sub-diagrams are presented in the appendices.

4.4.2 The T Sub-Diagram

The T_i operator term consists of a vertex to which i creation lines and i annihilation lines are connected. A product of T operators such as $T_1 T_2$ entails both sub-diagrams appearing in the final diagram. T_2 , for example, is represented as



Unlike the Hamiltonian, the creation and annihilation operators in the T operators are restricted so that only filled orbitals can be annihilated and only virtual orbitals can be created.

4.5 Algebraic Equivalence Rules

Generally speaking, a diagram consists of an H sub-diagram placed above and connected to a set of T sub-diagrams. A legal diagram for a matrix element involving $(He^T)_C$ is one that yields the correct final state⁵ and is linked in the same sense as the linked diagram theorem. To derive the coupled-cluster equations diagrammatically, the first step is to write down every such legal diagram. The following is a “dictionary” for writing down the proper algebraic equivalent for each such legal diagram.

⁵that is, the lines that remain unconnected are creation and annihilation operators that produce that state that $(He^T)_C$ is being projected with.

4.5.1 Closed Diagram Rules

Here we will recapitulate the rules for translating diagrams as given by Hoffman & Schaeffer[4]. It is simplest to start with closed diagram rules and then to add the slight complication that open diagrams (corresponding to excited bras) present. For examples of diagrams and their algebraic equivalents see the previous citation or the diagrams in the appendices of this work.

- I To each one-body Hamiltonian vertex assign the factor f_{pq} where p and q follow the mnemonic “out,in”. To each two-body Hamiltonian vertex assign the factor $\langle pq||rs \rangle$ where the indices follow the mnemonic “left out, right out, left in, right in”. To each excitation (T operator) vertex, assign the factor $t_{ij\dots}^{ab\dots}$.
- II Sum over all internal lines.
- III For each pair of hole lines or particle lines that begin at the same vertex and end at the same vertex, multiply by a factor of $\frac{1}{2}$.
- IV If simultaneous exchange of the particle lines and the hole lines yields a topologically equivalent diagram, and the hole and particle lines are not independently interchangeable, multiply by a factor of $\frac{1}{2}$.
- V The sign of the diagram is determined by $(-1)^{h+l}$ where h is the number of hole lines and l is the number of closed loops.

4.5.2 Additional Open Diagram Rules

The above rules must be slightly extended for open diagrams. Again following the treatment of Hoffman & Schaeffer[4]:

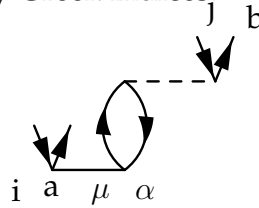
- I For the purposes of calculating the sign of a diagram, open lines should be treated as if they were closed by fictitious loops.
- II Antisymmetrically permute over all topologically distinct labellings of open line indices.

4.5.3 Example

To illustrate the application of these rules we will work out the algebraic equivalent of the following diagram which appears in the matrix element

$$\langle_{ij}^{ab} | e^{-T} H e^T | \Phi_0 \rangle$$

Internal lines are represented by Greek indices:



There are 3 hole lines and 1 real loop in the diagram. However, applying the first open diagram rule, we must add 2 more fictitious loops for the purposes of sign determination. Thus the overall sign is positive. Rules III and IV do not apply, so there is no factor of $\frac{1}{2}$ multiplying the diagram. Applying the second open diagram rule, we see that exchanging i and j yields a distinct arrangement of the indices, as does exchanging a and b . Thus, if $P(x|y)$ is understood to be an antisymmetric permutation operator which exchanges x and y , the algebraic equivalent is

$$P(i|j)P(a|b) \sum_{\mu} t_{i\mu}^{a\alpha} \langle \mu b | | \alpha j \rangle$$

For all diagrams and algebraic equivalents appearing in the CCSDT equations see Hoffmann & Schaeffer[4]. Diagrams and algebraic equivalents appearing in the triples through hexatuples projections of the CCSD effective Hamiltonian appear in the appendices of this work.

Chapter 5

Diagram Generation Algorithm

Although the above statement of the diagram rules is sufficient to properly derive the expression for an arbitrary matrix element of $(He^T)_C$, it is useful to have a systematic procedure for producing all legal diagrams without accidentally repeating topologically equivalent diagrams or missing diagrams that do contribute. The simplest form of these rules are for matrix elements in which the ket is the reference state. Since these are the only such elements that need to be evaluated for the CC methods, we will first present these rules and generalize to matrix elements with excited kets in the appendices.

5.1 Excitation Numbers of H and T

Let us formulate the general problem in calculating a reference-state-ket matrix element. A diagram will consist of an H sub-diagram and some product of T diagrams, $\prod_i T_i^{\alpha_i}$ where α_i is the number of T_i diagrams included in each factor.

Suppose the bra state is n -tuply excited. Clearly, a legal diagram must produce an overall excitation of level n . Assign an excitation number n_h to the Hamiltonian as

$$n_h = \frac{1}{2}(n_{down,in} + n_{up,out} - n_{down,out} - n_{up,in})$$

where the n terms on the right-hand side are the number of lines connected to the Hamiltonian sub-diagram with the subscripted properties. The excitation number of the T product, n_t is identical to the above formula, but can be written more naturally since T contains only in-going hole lines and out-going particle lines.

Exploiting this fact, we may write

$$n_t = \sum_i i\alpha_i$$

5.2 Disconnection and Connection Numbers

The other requirement for a legal diagram is given by the linked diagram theorem. The Hamiltonian sub-diagram must connect to each sub-diagram of the T product. However, because of the directionality of the lines attached to a T sub-diagram, only out-going hole lines and in-going particle lines from H can link to a T operator. Define κ , the connection number of H as

$$\kappa = n_{down, out} + n_{up, in}$$

It is usually easier to think of n_t and κ pictorially. The appendices present all the forms of the H sub-diagram listed with values of the excitation and connection number.

κ is the number of potential connections H can make. We must also know the number of pieces of the T product. Denote this by Δ , the disconnection number, defined as

$$\Delta = \sum_i \alpha_i$$

5.3 Legal Diagram Classes

Using the concepts of excitation, connection and disconnection number we can clearly state a systematic method for generating all legal diagrams for a given matrix element of $(He^T)_C$. We proceed by generating sequentially all T products that can be formed using whatever T_i operators are being included in a given approximation. We wish to ask which forms of the Hamiltonian sub-diagram can connect legally with each T product. Since we must produce the appropriate net excitation

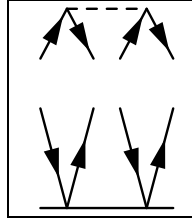
the *first diagram condition* must be satisfied

$$n_h + n_t = n$$

To satisfy the linked diagram requirement, we must have that the connection number exceeds the disconnection number. However, the connection number cannot exceed twice n_t for this would result in unwanted open lines in the diagram. These considerations yield the *second diagram condition*

$$\Delta \leq \kappa \leq 2n_t$$

Every pair of T product sub-diagram and H sub-diagram that satisfies these conditions will be called a *legal diagram class*. A diagram class may be denoted by enclosing the H and T sub-diagrams in a box without connecting them:



Proceeding through all possible T products easily produces all such diagram classes. An important corollary of the diagram conditions is an upper limit on the excitation level of non-vanishing projections. Denote by ν the highest order excitation operator retained in T . Since κ can be no larger than 4, Δ can be at most 4. Thus the maximal excitation level producible by a T operator product is 4ν . For the Hamiltonian with $\kappa = 4$, $n_h = -2$, thus no diagrams can contribute to projections with higher than $(4\nu - 2)$ -tuple excited configurations¹.

After obtaining all diagram classes, the classes may be expanded into diagrams to which the above Hoffmann & Schaeffer rules apply. For each diagram class,

¹This is another perspective on the termination of the Hausdorff commutator expansion. See Crawford & Schaeffer[3]. It is not possible to obtain higher excitations by choosing an H sub-diagram with a less negative n_h because increasing n_h by m decreases κ by at least m and therefore lowers the highest allowed excitation level of the T product by at least m .

form diagrams by connecting the H lines to the T lines in every unique way so that each T sub-diagram is linked to H by at least one line². This produces all diagrams that contribute to a general reference-state-ket matrix element of $e^{-T}He^T$.

²Two sets of connections are not unique if one may be obtained from the other by relabelling the internal lines.

Chapter 6

Error Estimates

Though coupled-cluster methods have enjoyed great success in a variety of difficult quantum chemical problems[9], it is desirable to obtain a reliable indicator of the success of the treatment of the electron correlation. One obvious approach is to perform the electronic structure calculation at a higher level of approximation to see if the results of the lower-level calculation change substantially. Indeed, many attempts have been made to extend CCSD to include some of the effects of connected triple excitations such as the so-called CCSDt method[2]. One disadvantage of such an approach is that it requires a more laborious and complex calculation than CCSD itself. It would be preferable to obtain an error estimate in the form of an internal consistency check that relied only on quantities available from the original CCSD calculation.

While it has been argued that the magnitude of the T_1 amplitudes can serve as an indicator of the success of the CC approximation[13], the error in the ground state energy seems to be an obvious candidate for a figure of merit¹. In this section we develop estimates for $|E_{\text{CC}} - E_{\text{Full CI}}|$. It is worth reemphasizing at this point that any such estimate does not directly relate approximate solutions to experimental values but rather relates approximate solutions to the exact solution of Schrödinger's equation within a finite basis set.

¹Furthermore, it is possible to work with so-called Brueckner orbitals where the reference orbitals are rotated to make the T_1 amplitudes vanish[1]. In such a case, a T_1 diagnostic would not be useful.

6.1 The **EOM** matrix

The so-called EOM matrix is simply the matrix representation of $e^{-T}He^T$ in the basis of excited configurations. The peculiar terminology comes from the EOM-CC or Equation-of-Motion Coupled-Cluster method[9] which was introduced to quantum chemistry in an attempt to calculate excited states based on a coupled-cluster description of the ground state². The calculation of excited states in EOM-CC is very like CI in that the matrix $e^{-T}He^T$ is diagonalized in an appropriate subspace of excited configurations.

6.2 Gershgorin's Circle Theorem

One of the cornerstone results of applied linear algebra and perturbation theory is Gershgorin's theorem. The theorem relates the location of eigenvalues of a matrix to the entries of the matrix as follows:

Theorem 1 (Gershgorin) *Given $A \in M_n(\mathbb{C})$, Let D be a diagonal matrix whose entries are equal to the diagonal elements of A . Form $F = A - D$.*

Let d_i be the i th diagonal entry of D and f_i be the i th row (column) of F . Define the disc $C_i = \{z \in \mathbb{C} : |z - d_i| \leq \|f_i\|_1\}$, where the 1-norm $\|f_i\|_1 = \sum_j |f_{ij}|$. Then every eigenvalue of A is in one of the C_i .

Further, if k such discs form a connected region then precisely k eigenvalues lie in this region, including multiplicity.

For the proof of Gershgorin's theorem, see a linear algebra text such as Lax[6].

²This approach is necessary because, unlike CI, it is problematic to converge the non-linear excitation operator in coupled-cluster theory to an excited state.

6.3 Matrix Representations of H

It is now useful to reconsider some of the quantum chemical approximations discussed in light of their effect on the H matrix. The determination of the self-consistent Hartree-Fock basis can be regarded as finding a particular similarity transformation which makes the matrix elements $\langle_a^i|H|\Phi_0\rangle$ vanish (Brillouin’s theorem).

The coupled-cluster approximation goes one step further. By applying the (non-orthogonal) similarity transformation $e^{-T}He^T$ to zero-out the remaining entries of the first column up to level of excitation retained in the T operator. Clearly, all of these approximations can be viewed as similarity transformations on the representation of H in the calculation basis, so that they all have the same energy eigenvalues as the full CI.

The presence of so many zero entries in the first column suggests that Gershgorin’s theorem may provide a useful bound for the true ground state energy³. Unfortunately, calculating the 1-norm of the first column is somewhat computationally intensive, even at the level of the simplest useful approximation CCSD. As demonstrated in the previous chapter, diagrams contribute to matrix elements out to $(4\nu - 2)$ -tuply excited configurations. Although very few diagrams contribute for highly excited matrix elements, the EOM-CC first column has strictly non-zero entries in the triple, quadruple, quintuple and hexatuple rows. All of the diagrams for the non-zero elements of the first column of the full EOM-CC matrix and their algebraic equivalents are presented in the appendices.

Unfortunately, we cannot make the claim that the 1-norm of the first column

³At first blush, it may seem that the Gershgorin row regions present a more advantageous error estimate, owing to the particularly simple structure of the first row of the transformed H matrix which vanishes after doubly excited elements. In fact, the only diagram which contributes for the doubles terms is algebraically equivalent to $\langle ij||ab\rangle$ which does not depend on T at all. Thus, the Gershgorin row estimator is essentially independent of the success of the coupled-cluster approximation. This reflects the choice to obtain T by projecting the right-hand Schrödinger equation with excited bras rather than *vice versa*.

is a rigorous upper bound on $|E_{\text{CCSD}} - E_{\text{fullCI}}|$ from Gershgorin’s theorem. Strictly speaking, this would only follow from the theorem if we could establish that the Gershgorin disc centered on the CCSD ground state energy did not intersect any of the other Gershgorin discs. It does not seem possible to obtain this kind of information about the topology of the Gershgorin regions without analyzing every matrix element in the full EOM-CC matrix, precisely the kind of computational chore we hoped to avoid. Nevertheless, speaking heuristically, we expect that the radii of other Gershgorin discs will only cross the ground state when the 1-norms of columns are on the order of the energy spacing between the ground state and the excited states. In this situation, one might expect the ground state description to have serious problems anyway. Thus, it should be possible to follow a given molecule from a geometry where the CCSD approximation, and therefore the Gershgorin estimator, works well into a regime where both begin to fail.

6.4 A Perturbative Approach

Another possible method for studying the disagreement between the CCSD ground state and the fullCI is to treat the difference between the two perturbatively. Unlike the analysis using Gershgorin’s theorem, we will not be able to make precise statements about the error since such rigorous results typically require detailed knowledge about the perturbative expansion[5]. Of course this is not an uncommon situation in applications of perturbation theory to quantum mechanics.

For simplicity, we will ignore the possibility of degeneracy in the singles and doubles block of the EOM matrix. It is important to note that the similarity transformation $e^{-T} H e^T$ is non-unitary so that the transformed Hamiltonian is not Hermitian. In this case the left and right eigenvectors (bras and kets) are distinct, and we must be more careful in formulating perturbation theory.

6.4.1 Perturbation Theory for Non-Hermitian Operators

We can construct a non-degenerate perturbation theory for the non-Hermitian transformed Hamiltonian as a straight-forward generalization of the familiar Rayleigh-Schrödinger perturbation theory. Although the left and right eigenvectors are not orthogonal among themselves, they can be chosen to satisfy a biorthogonality condition⁴ [9]:

$$\langle \tilde{\Psi}_i | \Psi_j \rangle = \delta_{ij}$$

We consider the operator-valued function $H(\lambda)$ of a parameter λ . With $H(\lambda) = H_0 + \lambda H_1$ so that as λ varies from 0 to 1, H switches smoothly from the unperturbed to the perturbed operator. We assume⁵ that the eigenvalues E_i and the left and right eigenvectors, $\langle \tilde{\Psi}_i |$ and $|\Psi_i\rangle$ can be expanded in convergent power series in λ . Then,

$$\begin{aligned} E_i &= E_i^{(0)} + \lambda E_i^{(1)} + \lambda^2 E_i^{(2)} + \dots \\ |\Psi_i\rangle &= |\Psi_i^{(0)}\rangle + \lambda |\Psi_i^{(1)}\rangle + \lambda^2 |\Psi_i^{(2)}\rangle + \dots \\ \langle \tilde{\Psi}_i | &= \langle \tilde{\Psi}_i^{(0)} | + \lambda \langle \tilde{\Psi}_i^{(1)} | + \lambda^2 \langle \tilde{\Psi}_i^{(2)} | + \dots \end{aligned}$$

To obtain expressions for the coefficients to each order in λ we make use of the eigenvalue equations (in this case the Schrödinger equation):

$$\begin{aligned} (H_0 + \lambda H_1)(|\Psi_i^{(0)}\rangle + \lambda |\Psi_i^{(1)}\rangle + \dots) &= (E_i^{(0)} + \lambda E_i^{(1)} + \dots)(|\Psi_i^{(0)}\rangle + \lambda |\Psi_i^{(1)}\rangle + \dots) \\ (\langle \tilde{\Psi}_i^{(0)} | + \lambda \langle \tilde{\Psi}_i^{(1)} | + \dots)(H_0 + \lambda H_1) &= (\langle \tilde{\Psi}_i^{(0)} | + \lambda \langle \tilde{\Psi}_i^{(1)} | + \dots)(E_i^{(0)} + \lambda E_i^{(1)} + \dots) \end{aligned}$$

By requiring that these series equalities hold identically, equating terms on powers of λ and projecting with the ground state eigenvectors of H_0 we obtain:

$$E_i^{(1)} = \langle \tilde{\Psi}_i^{(0)} | H_1 | \Psi_i^{(0)} \rangle$$

⁴The tilde on the bra vector is just to emphasize that $\langle \tilde{\Psi} | \neq (|\Psi\rangle)^\dagger$.

⁵It is possible to rigorously justify this assumption under certain conditions[6][5].

$$E_i^{(2)} = \langle \tilde{\Psi}_i^{(0)} | H_1 | \Psi_i^{(1)} \rangle$$

etc...

Likewise, the first-order shift in the eigenvectors can be found by expanding $|\Psi_i^{(1)}\rangle$ in the complete basis of eigenkets of H_0 using the biorthonormality condition as:

$$|\Psi_i^{(1)}\rangle = \sum_{n \neq i} |\Psi_n^{(0)}\rangle \langle \tilde{\Psi}_n^{(0)} | \Psi_i^{(1)} \rangle$$

Substituting this expansion into the equation obtained from the terms linear in λ gives the following closed form for $E_i^{(2)}$:

$$E_i^{(2)} = \sum_{n \neq i} \frac{\langle \tilde{\Psi}_i^{(0)} | H_1 | \Psi_n^{(0)} \rangle \langle \tilde{\Psi}_n^{(0)} | H_1 | \Psi_i^{(0)} \rangle}{E_i^{(0)} - E_n^{(0)}}$$

It is straightforward to continue this procedure for higher order terms in the expansion, though these results will be sufficient for our purposes. Szabo & Ostlund present a detailed derivation of the analogous Hermitian case[12].

6.4.2 Perturbative Correction to CCSD Ground State Energy

Suppose that we know a complete set of right and left eigenvectors for the singles and doubles block. This corresponds to having performed an EOM-CCSD excited state calculation[9]⁶ We may extend this set of eigenvectors in a natural way to obtain a basis of the same dimension as the full CI matrix by choosing the unperturbed Hamiltonian to agree with the EOM-CCSD matrix in the singles and doubles block while the remainder is diagonal with diagonal elements equal to the diagonal elements of the full EOM-CC matrix. Trivially, the extended set of eigenvectors and eigenvalues contains the standard basis vectors for the space beyond

⁶Although this discussion is couched in terms of knowing all the eigenvectors of the singles and doubles block, it will not be necessary to know the excited states to calculate the leading order energy shift.

the singles and doubles subspace and the associated eigenvalues are simply the diagonal elements. Symbolically⁷,

$$H_0 = \begin{pmatrix} & \Phi_0 & S & D & T & \text{Quad} & \text{Quint} & H \\ \hline \Phi_0 & E & H_{0S} & H_{0D} & 0 & 0 & 0 & 0 \\ S & 0 & H_{SS} & H_{SD} & 0 & 0 & 0 & 0 \\ D & 0 & H_{DS} & H_{DD} & 0 & 0 & 0 & 0 \\ T & 0 & 0 & 0 & \ddots & 0 & 0 & 0 \\ \text{Quad} & 0 & 0 & 0 & 0 & \ddots & 0 & 0 \\ \text{Quint} & 0 & 0 & 0 & 0 & 0 & \ddots & 0 \\ H & 0 & 0 & 0 & 0 & 0 & 0 & \ddots \end{pmatrix}$$

In the spirit of the Møller-Plesset approach, the perturbation H_1 is taken as the difference between the full EOM-CC matrix and the unperturbed Hamiltonian. Thus H_1 vanishes in the singles and doubles block and along the entire diagonal. Elsewhere, H_1 agrees with the full EOM-CC matrix:

$$H_1 = \begin{pmatrix} & \Phi_0 & S & D & T & \text{Quad} & \text{Quint} & H \\ \hline \Phi_0 & 0 & 0 & 0 & H_{0T} & H_{04} & H_{05} & H_{0H} \\ S & 0 & 0 & 0 & H_{ST} & H_{S4} & H_{S5} & H_{SH} \\ D & 0 & 0 & 0 & H_{DT} & H_{D4} & H_{D5} & H_{DH} \\ T & H_{T0} & H_{TS} & H_{TD} & H_{TT}^* & H_{T4} & H_{T5} & H_{TH} \\ \text{Quad} & H_{40} & H_{4S} & H_{4D} & H_{4T} & H_{44}^* & H_{45} & H_{4H} \\ \text{Quint} & H_{50} & H_{5S} & H_{5D} & H_{5T} & H_{54} & H_{55}^* & H_{5H} \\ H & H_{H0} & H_{HS} & H_{HD} & H_{HT} & H_{H4} & H_{H5} & H_{HH}^* \end{pmatrix}$$

where the \star indicates that the diagonal elements are zero.

We may then ask what the perturbative correction to the ground state energy is to leading order. Because the singles and doubles block of the perturbing Hamil-

⁷For clarity, the matrices below extend only out to hexatuple excitations since these are all that is relevant for our consideration of the ground state. In reality, the matrices extend to n -tuple excitations, where n is the number of electrons

tonian vanishes and the left and right ground state eigenvectors lie in the singles and doubles subspace, the first-order shift in the energy must vanish.

The second-order shift, however, can provide useful information about the relative importance of non-zero matrix elements in the first column. Recalling the previous expression for the second-order energy shift, it is clear that there will be no contribution from the singly- and doubly-excited configurations since the perturbing Hamiltonian vanishes in this subspace. We can write the second-order energy shift for the ground state in the simplified form:

$$\Delta E^{(2)} = \sum_{n \in \{T, 4, 5, H, \dots\}} \frac{\langle n | H_1 | \Phi_0 \rangle \langle \tilde{\Phi}_0 | H_1 | n \rangle}{E_{\text{CCSD}} - \langle n | H_1 | n \rangle}$$

where the $|n\rangle$ stand for the standard basis vectors, the eigenvectors of H_0 beyond the singles and doubles block⁸.

In fact, this can be further simplified by noting that the transformed Hamiltonian $e^{-T} H e^T$ can only de-excite an excited ket by two levels in evaluating a matrix element. Thus, any matrix element with the ket configuration excited more than two levels above the bra configuration must vanish. For example:

$$H_{0T} = H_{04} = H_{05} = \dots = 0$$

⁸We have written $\langle n |$ sans tilde since H_0 is diagonal beyond the singles and doubles sub-block so that the left and right eigenvectors in the triples and higher subspace are simple Hermitian conjugates.

We can now rewrite the perturbing matrix as

$$H_1 = \left(\begin{array}{c|ccccccc} & \Phi_0 & S & D & T & \text{Quad} & \text{Quint} & H \\ \hline \hline \Phi_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ S & 0 & 0 & 0 & H_{ST} & 0 & 0 & 0 \\ D & 0 & 0 & 0 & H_{DT} & H_{D4} & 0 & 0 \\ T & H_{T0} & H_{TS} & H_{TD} & H_{TT}^* & H_{T4} & H_{T5} & 0 \\ \text{Quad} & H_{40} & H_{4S} & H_{4D} & H_{4T} & H_{44}^* & H_{45} & H_{4H} \\ \text{Quint} & H_{50} & H_{5S} & H_{5D} & H_{5T} & H_{54} & H_{55}^* & H_{5H} \\ H & H_{H0} & H_{HS} & H_{HD} & H_{HT} & H_{H4} & H_{H5} & H_{HH}^* \end{array} \right)$$

Since the bra ground state vector $\langle \tilde{\Phi}_0 |$ lies in the singles and doubles subspace, the matrix element $\langle \tilde{\Phi}_0 | H_1 | n \rangle$ must vanish for n higher than quadruple excitations.

Thus,

$$\Delta E^{(2)} = \sum_{n \in \{T, 4\}} \frac{\langle n | H_1 | \Phi_0 \rangle \langle \tilde{\Phi}_0 | H_1 | n \rangle}{E_{\text{CCSD}} - \langle n | H_1 | n \rangle}$$

To leading order in perturbation theory, the ground state energy does not depend on the elements in the first column in the quintuples and hexatuples blocks. For this reason, we are led to propose the following approximate Gershgorin-like inequality which simply neglects contributions from elements beyond the quadruples:

$$|E_{\text{CCSD}} - E_{\text{fullCI}}| < \approx \|H_{0T}\|_1 + \|H_{04}\|_1$$

The elements of H_{0T} and H_{04} (and H_{05} and H_{0H} , for that matter) are presented in the appendices, both in diagrammatic form and in explicit algebraic form in terms of two-electron integrals of the molecular orbitals and the excitation amplitudes available from a CCSD calculation.

6.5 Discussion

We have presented an algebraic function of the excitation amplitudes and known two-electron integrals that, under certain assumptions, is an upper-bound to the difference between the CCSD correlation energy and the full CI correlation energy, the exact solution of Schrödinger’s equation within the given basis set. Furthermore, a perturbative argument has been given which suggests that only a smaller subset of terms in this upper-bound contribute substantially to this difference.

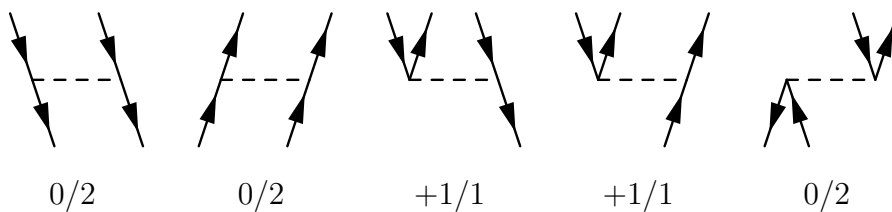
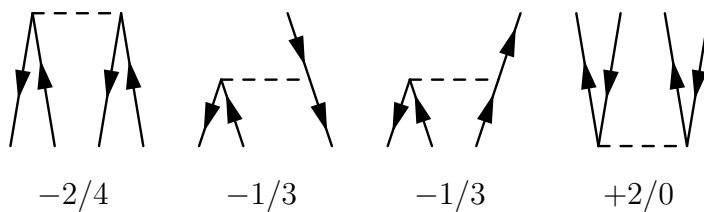
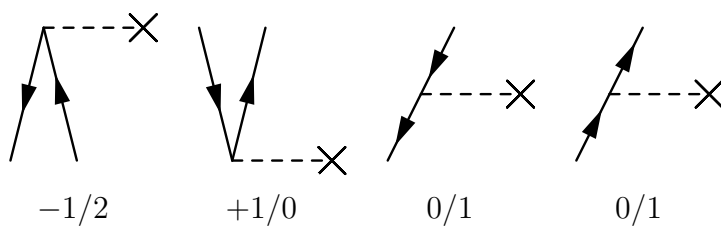
It is desirable, of course, to numerically evaluate this estimator for molecular systems of interest, particularly ones for which the CCSD approximation is known to encounter difficulties. The principal obstacle to such an implementation is obtaining the necessary information about the underlying CCSD calculation. One approach is to read the relevant data out of the output files of an existing implementation of the CCSD approximation such as the Bartlett group’s code ACES II. Preliminary progress in this direction has been made in the form of the incomplete code `gersh` presented in the appendices. Further study of the ACES II output is required to complete this implementation.

Another avenue of research is to consider the perturbative correction to the CCSD ground state energy more seriously. Although it would be somewhat computationally burdensome, it is possible to directly calculate the second-order shift in the ground state energy using the left-hand ground state which we have denoted $\langle \tilde{\Phi}_0 |$, which is available from a standard CCSD properties calculation[9]. Along these lines, it might also be fruitful to consider corrections to the excited state energies, or perhaps analyze the perturbative corrections to the eigenvectors themselves.

Appendix A

Forms of the Hamiltonian Sub-Diagram

The numbers beneath each sub-diagram indicate *excitation level / connection number*.



Appendix B

Matrix Elements in the First Column of $e^{-T}He^T$

In what follows we present all diagram classes and corresponding algebraic equivalents for matrix elements of the form $\langle\psi|e^{-T}He^T|\Phi_0\rangle$ where $\langle\psi|$ is a triply-through hexatuply-excited configuration. As stated in the main body of the text, higher excitation level matrix elements vanish owing to the linked diagram theorem. Diagrams and algebraic equivalents corresponding to singly- and doubly-excited configurations and the energy term $\langle\Phi_0|e^{-T}He^T|\Phi_0\rangle$ amount to the CCSD equations and are widely available elsewhere[3].

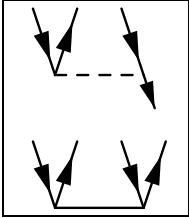
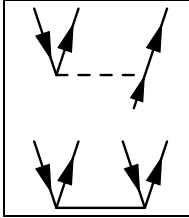
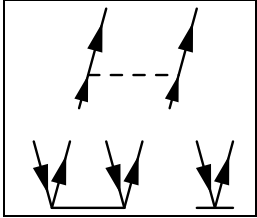
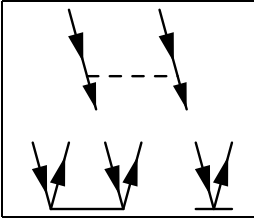
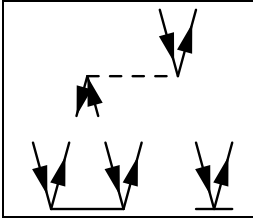
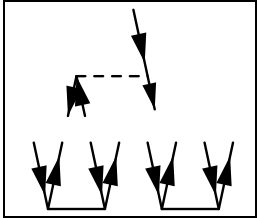
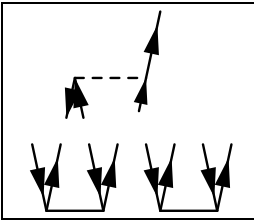
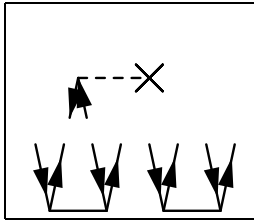
The following slightly unconventional notation is used for the algebraic equivalents to each diagram. Roman indices are used to label open lines with a, b, \dots referring to virtual orbitals and i, j, \dots referring to occupied orbitals. Greek indices are used to label internal lines with α, β, \dots referring to virtual orbitals and μ, ν, \dots referring to occupied orbitals. Greek indices are implicitly summed over. P is an anti-symmetrized permutation operator. The argument of P specifies which permutations are to be carried out according to the rule that indices which are exchanged must cross a vertical bar. Thus $P(ab|c)$ tells us to construct two permutations swapping a and c and then b and c , but not a permutation swapping a and b .

Above each diagram class is a parenthetical ID by which the diagrams are referred to in notes and the preliminary implementation of the Gershgorin estimator.

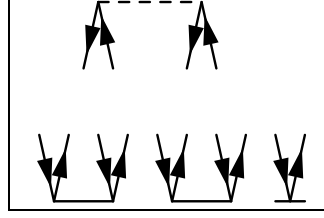
⁰All diagrams in this text were created using the open-source METAFONT Feynman diagram package, `feynmf`.

B.1 Triples Diagrams

$$\langle abc | e^{-T} H e^T | \Phi_0 \rangle =$$

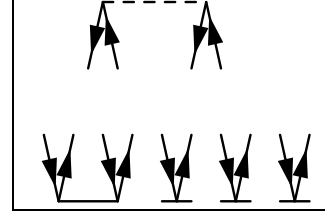
<p>(A1)</p>  <p>$-P(ij k)P(ab c)t_{\mu k}^{bc}\langle a\mu ij\rangle$</p>	<p>(B1)</p>  <p>$+P(ab c)P(i jk)t_{jk}^{ac}\langle ab i\alpha\rangle$</p>	<p>(C1)</p>  <p>$+P(a bc)P(ij k)t_{ij}^{aa}t_k^{\beta}\langle bc \alpha\beta\rangle$</p>
<p>(D1)</p>  <p>$+P(i jk)P(ab c)t_{i\mu}^{ab}t_{\nu}^{c}\langle \mu\nu jk\rangle$</p>	<p>(E1-2)</p>  <p>$-P(ij k)P(a b c)t_{ij}^{a\alpha}t_{\mu}^b\langle \mu c \alpha k\rangle$ $-P(i jk)P(ac b)t_{\mu k}^{ac}t_i^{\alpha}\langle \mu b \alpha j\rangle$</p>	<p>(F1-2)</p>  <p>$-P(i jk)P(a bc)t_{i\mu}^{a\alpha}t_{\nu k}^{bc}\langle \mu\nu \alpha j\rangle$ $+ \frac{1}{2}P(a bc)P(ij k)t_{ij}^{a\alpha}t_{\mu\nu}^{bc}\langle \mu\nu \alpha k\rangle$</p>
<p>(G1-2)</p>  <p>$+P(i jk)P(a b c)t_{i\mu}^{a\alpha}t_{jk}^{\beta c}\langle \mu b \alpha\beta\rangle$ $-\frac{1}{2}P(ij k)P(a bc)t_{ij}^{\alpha\beta}t_{\mu k}^{bc}\langle a\mu \alpha\beta\rangle$</p>	<p>(H1)</p>  <p>$-P(ij k)P(a bc)t_{ij}^{a\alpha}t_{\mu k}^{bc}f_{\mu\alpha}$</p>	

(I1-5)



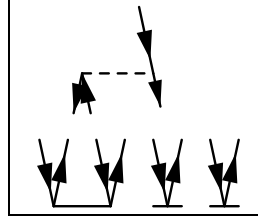
$$\begin{aligned}
& -P(a|b|c)P(i|j|k)t_{i\mu}^{a\alpha}t_{jk}^{b\beta}t_{\nu}^c\langle\nu\mu||\alpha\beta\rangle \\
& -P(a|bc)P(i|j|k)t_{i\mu}^{a\alpha}t_{\nu k}^{bc}t_j^{\beta}\langle\nu\mu||\alpha\beta\rangle \\
& -P(a|bc)P(ij|k)t_{ij}^{a\alpha}t_{\mu k}^{bc}t_{\nu}^{\beta}\langle\nu\mu||\alpha\beta\rangle \\
& +\frac{1}{2}P(ij|k)P(ab|c)t_{ij}^{\alpha\beta}t_{\mu k}^{ab}t_{\nu}^c\langle\nu\mu||\alpha\beta\rangle \\
& +\frac{1}{2}P(i|jk)P(a|bc)t_{jk}^{\beta a}t_{\mu\nu}^{bc}t_i^{\alpha}\langle\nu\mu||\alpha\beta\rangle
\end{aligned}$$

(J1-2)



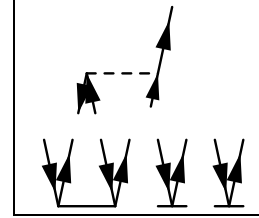
$$\begin{aligned}
& +P(a|b|c)P(ij|k)t_{ij}^{a\alpha}t_{\mu}^bt_k^{\beta}t_{\nu}^c\langle\nu\mu||\alpha\beta\rangle \\
& +P(i|j|k)P(a|bc)t_{\nu k}^{bc}t_i^{\alpha}t_{\mu}^at_j^{\beta}\langle\nu\mu||\alpha\beta\rangle
\end{aligned}$$

(K1-2)



$$\begin{aligned}
& +P(ij|k)P(a|b|c)t_{ij}^{a\alpha}t_{\mu}^bt_{\nu}^c\langle\mu\nu||\alpha k\rangle \\
& +P(i|j|k)P(a|bc)t_{\nu k}^{bc}t_i^{\alpha}t_{\mu}^a\langle\mu\nu||\alpha j\rangle
\end{aligned}$$

(L1-2)

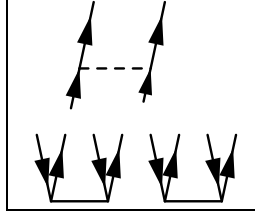


$$\begin{aligned}
& -P(ij|k)P(a|b|c)t_{ij}^{a\alpha}t_{\mu}^bt_k^{\beta}\langle\mu c||\alpha\beta\rangle \\
& -P(i|j|k)P(ab|c)t_{\mu j}^{ab}t_i^{\alpha}t_k^{\beta}\langle\mu c||\alpha\beta\rangle
\end{aligned}$$

B.2 Quadruples Diagrams

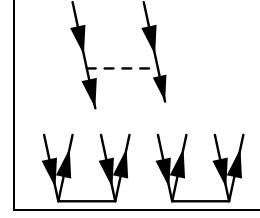
$$\langle abcd|e^{-T}He^T|\Phi_0\rangle =$$

(A1)



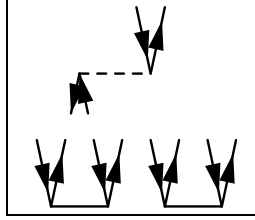
$$+P(ac|bd)t_{ij}^{\alpha b}t_{kl}^{\beta d}\langle ac||\alpha\beta\rangle$$

(B1)



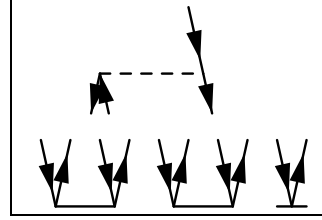
$$+P(ik|jl)t_{\mu j}^{ab}t_{\nu l}^{cd}\langle \nu\mu||ik\rangle$$

(C1)



$$-P(ij|k|l)P(a|c|bd)t_{ij}^{a\alpha}t_{\mu l}^{bd}\langle \mu c||jk\rangle$$

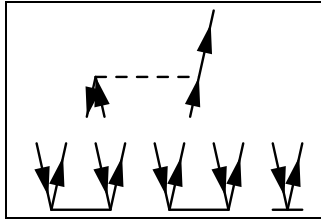
(D1-2)



$$+P(ij|k|l)P(a|bc|d)t_{ij}^{a\alpha}t_{\mu k}^{bc}t_{\nu}^d\langle \mu\nu||\alpha l\rangle$$

$$+P(i|j|k|l)P(ab|cd)t_{\mu j}^{ab}t_{kl}^{cd}t_i^{\alpha}\langle \mu\nu||\alpha k\rangle$$

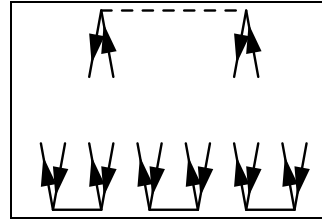
(E1-2)



$$-P(i|j|kl)P(ab|c|d)t_{\mu j}^{ab}t_{kl}^{\beta d}t_i^{\alpha}\langle \mu c||\alpha\beta\rangle$$

$$-P(ij|kl)P(a|b|c|d)t_{ij}^{a\alpha}t_{kl}^{\beta d}t_{\mu}^b\langle \mu c||\alpha\beta\rangle$$

(F1-3)

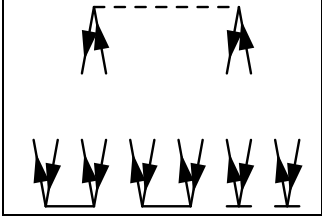


$$-P(i|jk|l)P(a|b|cd)t_{i\mu}^{a\alpha}t_{jk}^{\beta b}t_{\nu l}^{cd}\langle \mu\nu||\alpha\beta\rangle$$

$$+P(ij|kl)P(ab|cd)t_{ij}^{\alpha\beta}t_{\mu k}^{ab}t_{\nu l}^{cd}\langle \mu\nu||\alpha\beta\rangle$$

$$+P(ij|kl)P(ab|cd)t_{ij}^{a\alpha}t_{kl}^{b\beta}t_{\mu\nu}^{cd}\langle \mu\nu||\alpha\beta\rangle$$

(G1-3)

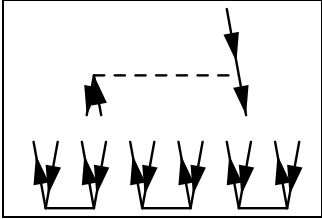


$$\begin{aligned}
& +P(ij|k|l)P(a|bc|d)t_{ij}^{a\alpha}t_{\mu k}^{bc}t_l^\beta t_\nu^d \langle \mu\nu || \alpha\beta \rangle \\
& +P(ac|bd)t_{ij}^{a\alpha}t_{kl}^{c\beta}t_\mu^b t_\nu^d \langle \mu\nu || \alpha\beta \rangle \\
& +P(ik|jl)t_{\mu j}^{ab}t_{\nu l}^{cd}t_i^\alpha t_k^\beta \langle \mu\nu || \alpha\beta \rangle
\end{aligned}$$

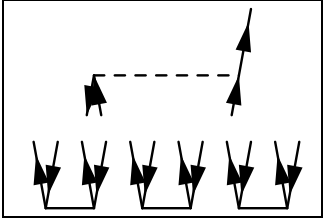
B.3 Quintuples Diagrams

$$\langle ijklm | e^{-T} H e^T | \Phi_0 \rangle =$$

(A1)



(B1)

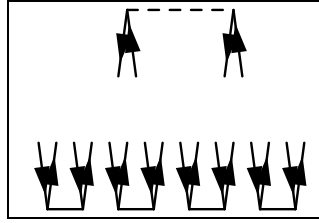


$$\begin{aligned}
& +P(ij|km|l)P(a|bcde)t_{ij}^{a\alpha}t_{\mu k}^{bc}t_{\nu m}^{de} \langle \mu\nu || \alpha\lambda \rangle \quad -P(ijlm|k)P(ae|bc|d)t_{ij}^{a\alpha}t_{\mu k}^{bc}t_{lm}^{\beta e} \langle \mu d || \alpha\beta \rangle \\
& \text{(C1-2)} \\
& +P(i|jm|kl)P(abde|c)t_{\mu j}^{ab}t_{kl}^{c\beta}t_{\nu m}^{de}t_i^\alpha \langle \mu\nu || \alpha\beta \rangle \\
& +P(ijlm|k)P(ad|bc|e)t_{ij}^{a\alpha}t_{\mu k}^{bc}t_{lm}^{d\beta}t_\nu^e \langle \mu\nu || \alpha\beta \rangle
\end{aligned}$$

B.4 Hexatuples Diagrams

$$\langle abcdef | e^{-T} H e^T | \Phi_0 \rangle =$$

(A1)



$$+ P(ijlm|kn) P(ad|bcef) t_{ij}^{a\alpha} t_{\mu k}^{bc} t_{lm}^{d\beta} t_{\nu n}^{ef} \langle \mu\nu || \alpha\beta \rangle$$

Appendix C

Numerical Implementation

The following code `gersh.cc` is a C++ program which reads information from the ACES II output files to compute the Gershgorin estimator derived in the body of this document. It is currently incomplete because of difficulties in reading certain integral lists from the ACES II output. In particular the two-electron integrals of the form $\langle ab||cd \rangle$ are stored in a file named MOABCD whose structure is unknown. Also, the $\langle ab||ci \rangle$ integrals do not appear in the location specified by the most recent version of the ACES II manual[7]. Any attempt at a general implementation will require more detailed knowledge about the integral storage of the ACES II code.

Currently the code is only designed to work with an RHF reference state and gets confused by molecular orbitals which are classified according to different irreducible representations of the symmetry point group of the molecule. It expects all orbitals to belong to the first irreducible representation in the ACES II output. As noted in the program comments below, this is designed to work with a (very) slightly modified version of ACES II which outputs all of the T_1 amplitudes to a large number of significant figures.

```
//////////////////////////////////////////////////////////////////
// gersh.cc      March 4, 2001      Mike Rust
// Last modified: April 14, 2001
//
// Usage: gersh <aces2 output file> <MOINTS file> [options]
// This code parses the output from an aces2 (CRAPS) CCSD calculation
// to determine the T1 and T2 amplitudes and the symmetry of the wavefunction.
// This information is used to load integral lists from the MOINTS file
// into core memory. This information is used to form the appropriate
// matrix elements of e^-T H e^T and calculate the Gershgorin estimator.
// NB: This code assumes that the modified version of aces2 has been used
// which outputs all of the symmetry allowed T amplitudes (see thesis
// notes). Further, the aces2 input file must specify a large enough
// value for PRINT (999 should do the trick) so that the MOINTS file
// structure is printed.
// NB: Currently works only for Hartree-Fock references. (i.e. assumes
// that Brillouin's theorem holds)
#include<stdlib.h>
#include<stdio.h>
```

```

#include<string.h>
#include<iostream.h>
#include<fstream.h>
#include<math.h>

#define RHF 1
#define UHF 2

//Here are some global constants that are relevant to lots of stuff
int calctype, nele, nocc, nvirt;

class fourIndex {
//The fourIndex class represents a generic four-index quantity
//e.g. a set of two-electron integrals
private:
    char ph[5]; //particle/hole string
    char ab[5]; //spin string
    char name[20]; //name of the list
    double *data;
    int dim[4];
    int restr_left, restr_right;
    //Which indices are restricted?

public:
    //constructor
    fourIndex(char *newName, char *newPH, char *newAB) {
    //Copy strings
        strcpy(ph,newPH);
        strcpy(ab,newAB);
        strcpy(name,newName);

    //Check for spin silliness
        int i;
        int checksum1=0, checksum2=0;
        for(i=0;i<2;i++)
            checksum1+=(ab[i]!='a');
        for(i=2;i<4;i++)
            checksum2+=(ab[i]!='a');
        if(checksum1!=checksum2)
            cout << "Warning! " << name << " list initialized with
            incompatible spins" << endl;

    //Check for failure to use notation properly
        for(i=0;i<4;i++) {
            if(ab[i]!='a'&&ab[i]!='b')
                cout << "Non-sensical spin string " << ab <<
                " in " << name << " initialization." << endl;
            if(ph[i]!='p'&&ph[i]!='h')
                cout << "Non-sensical particle/hole string " << ph <<
                " in " << name << " initialization." << endl;
        }

    //Now we have split up by calctype:
        if(calctype==RHF) {
        //Determine dimension of each index
            for(i=0;i<4;i++)
                if(ph[i]!='p')
                    dim[i]=nocc;
                else
                    dim[i]=nvirt;

            //This is fine, except that we have to account for
            //restricted lists
            restr_left=(ph[0]==ph[1] && ab[0]==ab[1]);
            restr_right=(ph[2]==ph[3] && ab[2]==ab[3]);

            int size=1;
            for(i=0;i<4;i++)
                size*=dim[i];

            //allocate the required space
            data = new double[size];
            //check if it's all good
            if(!data) {
                cout << "Failed to allocate " << size << " elements for "
                << name << "!" << endl;
                exit(-1);
            }
        }
    }

    ~fourIndex() {
        cout << "Destructor called for " << name << "!" << endl;
        if(data)
            delete data;
    }

    void getFirstIndices(int &i1, int &i2, int &i3, int &i4);
    void getNextIndices(int &i1, int &i2, int &i3, int &i4);
    int getSize(void);
    void set(int &i1, int &i2, int &i3, int &i4, double value);
    double get(int &i1, int &i2, int &i3, int &i4);
    void dump(void);
    char * getName(void) {
        return name;
    }
};

//Methods for four-index class
//The only complication here is whether the indices are
//restricted or not. In general the convention for restricted
//indices is left < right, following the aces2 convention.
i1=i3=0;

//Note: we are assuming the list is non-empty. If it is, don't
//blame this method.
i2=restr_left;

```

```

    i4=restr_right;
}
void fourIndex::getNextIndices(int &i1, int &i2, int &i3, int &i4) {
    //Rather than writing this in some clever way, we'll just do it
    //by hand.
    i4++;
    if(i4>=dim[3]) {
        i3++;
        if(restr_right)
            i4=i3+1;
        else
            i4=0;
    }
    if(i3>=dim[2]) {
        i3=0;
        if(restr_right)
            i4=1;
        i2++;
    }
    if(i2>=dim[1]) {
        i1++;
        if(restr_left)
            i2=i1+1;
        else
            i2=0;
    }
    if(i1>=dim[0])
        cout << "Error: Cannot increment (" << i1 << ", " << i2 << ", "
            << i3 << ", " << i4 << ") in " << name << endl;
}
int fourIndex::getSize(void) {
    //simply returns the number of elements in the list
    int leftsize, rightsize;
    if(restr_left)
        leftsize=(dim[0]*(dim[0]-1))/2;
    else
        leftsize=dim[0]*dim[1];
    if(restr_right)
        rightsize=(dim[2]*(dim[2]-1))/2;
    else
        rightsize=dim[2]*dim[3];
    return leftsize*rightsize;
}
void fourIndex::set(int &i1, int &i2, int &i3, int &i4, double value) {
    //Do range-checking
    if(i1>=dim[0]) {
        cout << "First index exceeds its dimension in set(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return;
    }
    if(i2>=dim[1]) {
        cout << "Second index exceeds its dimension in set(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return;
    }
    if(i3>=dim[2]) {
        cout << "Third index exceeds its dimension in set(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return;
    }
    if(i4>=dim[3]) {
        cout << "Fourth index exceeds its dimension in set(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return;
    }
    if(restr_left && i1>=i2) {
        cout << "set(" << i1 << ", " << i2 << ", " << i3 << ", " << i4 <<
            ") violates left index restrictions." << endl;
        return;
    }
    if(restr_right && i3>=i4) {
        cout << "set(" << i1 << ", " << i2 << ", " << i3 << ", " << i4 <<
            ") violates right index restrictions." << endl;
        return;
    }
    //If we made it this far, we're o.k.
    data[i1*dim[1]*dim[2]*dim[3] + i2*dim[2]*dim[3] + i3*dim[3] + i4]=value;
    //DEBUG
    // cout << "Setting element " << i1*dim[1]*dim[2]*dim[3] + i2*dim[2]*dim[3] + i3*dim[3] + i4 << " to " << value << endl;
}
double fourIndex::get(int &i1, int &i2, int &i3, int &i4) {
    //Do range-checking
    if(i1>=dim[0]) {
        cout << "First index exceeds its dimension in get(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return -1;
    }
    if(i2>=dim[1]) {
        cout << "Second index exceeds its dimension in get(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return -1;
    }
    if(i3>=dim[2]) {
        cout << "Third index exceeds its dimension in get(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return -1;
    }
    if(i4>=dim[3]) {
        cout << "Fourth index exceeds its dimension in get(" << i1 << ", "
            << i2 << ", " << i3 << ", " << i4 << ") in " << name << endl;
        return -1;
    }

```

```

    }
    if(restr_left && i1>=i2) {
        cout << "get(" << i1 << ", " << i2 << ", " << i3 << ", " << i4 <<
            ") violates left index restrictions." << endl;
        return -1;
    }
    if(restr_right && i3>=i4) {
        cout << "get(" << i1 << ", " << i2 << ", " << i3 << ", " << i4 <<
            ") violates right index restrictions." << endl;
        return -1;
    }
    //If we made it this far, we're o.k.

    return
        data[i1*dim[1]*dim[2]*dim[3] + i2*dim[2]*dim[3] + i3*dim[3] + i4];
}

void fourIndex::dump(void) {
    //This method displays the contents of the list
    cout << "Contents of " << name << ":" << endl;
    cout << "-----" << endl;
    //Isn't this cute?
    int minusone=-1;
    int *left, *right;
    int i1, i2, i3, i4;
    int iphys1, iphys2, iphys3, iphys4;
    if(restr_left)
        left=&i1;
    else
        left=&minusone;
    if(restr_right)
        right=&i3;
    else
        right=&minusone;
    //Begin the dump loop
    for(i1=0;i1<dim[0];i1++)
        for(i2=*left+1;i2<dim[0];i2++)
            for(i3=0;i3<dim[2];i3++)
                for(i4=*right+1;i4<dim[3];i4++) {
                    //Convert from internal to physical indices
                    if(ph[0]=='p')
                        iphys1=i1+1;
                    else
                        iphys1=i1+nocc+1;
                    if(ph[1]=='p')
                        iphys2=i2+1;
                    else
                        iphys2=i2+nocc+1;
                    if(ph[2]=='p')
                        iphys3=i3+1;
                    else
                        iphys3=i3+nocc+1;
                    if(ph[3]=='p')
                        iphys4=i4+1;
                    else
                        iphys4=i4+nocc+1;
                    cout << iphys1 << "\t" << iphys2 << "\t" << iphys3 << "\t"
                        << iphys4 << "\t" << get(i1,i2,i3,i4) << endl;
                }
}

///////////////////////////////////////////////////

class twoIndex {
    //The twoIndex class represents a generic two-index quantity
private:
    char ph[3]; //particle or hole string
    char ab[3]; //spin string
    char name[20]; //name of the list
    double *data; //the goods
    int dim1, dim2;
public:
    //constructor
    twoIndex(char *newName, char *newPH, char *newAB) {
        //Copy strings
        strcpy(ph,newPH);
        strcpy(ab,newAB);
        strcpy(name,newName);
        //Check for spin silliness
        if(ab[0]!=ab[1])
            cout << "Warning! " << name << " list initialized with
                non-matching spins" << endl;
        //Check for failure to use notation properly
        int i;
        for(i=0;i<2;i++) {
            if(ab[i]!='a'&&ab[i]!='b')
                cout << "Non-sensical spin string " << ab <<
                    " in " << name << " initialization." <<endl;
            if(ph[i]!='p'&&ph[i]!='h')
                cout << "Non-sensical particle/hole string " << ph <<
                    " in " << name << " initialization." <<endl;
        }
        //Now we have split up by calctype:
        if(calctype==RHF) {
            //Determine dimension of each index
            if(ph[0]=='p')
                dim1=nocc;
            else

```

```

        dim1=nvirt;
        if(ph[1]!='p')
            dim2=nocc;
        else
            dim2=nvirt;
        //allocate the required space
        data = new double[dim1*dim2];
        //check if it's all good
        if(!data) {
            cout << "Failed to allocate " << dim1*dim2 << " elements for "
                 << name << "!" << endl;
            exit(-1);
        }
    }
}

//Method prototypes
void set(int index1, int index2, double value);
double get(int index1, int index2);
void dump(void);

//Destructor
~twoIndex() {
    if(data)
        delete data;
}

};

/////////////////////////////////////////////////////////////////
//Methods for two-index class
/////////////////////////////////////////////////////////////////

void twoIndex::set(int index1, int index2, double value) {
    //Sanity check
    if(index1>=dim1||index2>=dim2) {
        cout << "Range check error: attempt to set ("
             << index1 << " , " << index2 << ") in " << name << endl;
    }
    else
        data[index1*dim2 + index2]=value;
}

double twoIndex::get(int index1, int index2) {
    //Sanity check
    if(index1>=dim1||index2>=dim2) {
        cout << "Range check error: attempt to get ("
             << index1 << " , " << index2 << ") in " << name << endl;
        return -1.0; //Error code
    }
    else
        return data[index1*dim2 + index2];
}

void twoIndex::dump(void) {
    //This method displays the contents of the list
    cout << "Contents of " << name << ":" << endl;
    cout << "-----" << endl;
    int i, j;
    int iphys, jphys;
    for(i=0;i<dim1;i++) {
        iphys=i+1;
        if(ph[0]=='h')
            iphys+=nocc;
        for(j=0;j<dim2;j++) {
            jphys=j+1;
            if(ph[1]=='h')
                jphys+=nocc;
            cout << iphys << "\t" << jphys << "\t" << get(i,j) << endl;
        }
    }
}

/////////////////////////////////////////////////////////////////
//Global instances of four-index and two-index to store
//the integral lists.
/////////////////////////////////////////////////////////////////

twoIndex *t1;
fourIndex *T2mixed;
fourIndex *T2same;
fourIndex *hhpp_aaaa;
fourIndex *hhpp_abab;
fourIndex *ppph_aaaa;
fourIndex *ppph_abab;
fourIndex *hhhp_aaaa;
fourIndex *hhhp_abab;

/////////////////////////////////////////////////////////////////
//Debugging functions
/////////////////////////////////////////////////////////////////

void printmoints(FILE *moints) {
    double dummy;
    int i=0;
    while(!feof(moints)) {
        i++;
        fread(&dummy,sizeof(double),1,moints);
        cout << i << " : " << dummy << endl;
    }
}

/////////////////////////////////////////////////////////////////
//int matchesHere(target, line)
//
// Returns a truth value according to whether target occurs
// starting at the first position of line.
/////////////////////////////////////////////////////////////////
int matchesHere(char *target, char *line) {
    if(*target=='\0')

```

```

        return 1;
    while(*target!='\0' && *line!='\0') {
        if(*target==*line)
            return 0; //Bad programmer, bad!
        target++;
        line++;
    }
    return (*target=='\0');
}

////////////////////////////////////
//int matches(target, line)
//
// Returns a truth value according to whether target occurs in
// line.
////////////////////////////////////
int matches(char *target, char *line) {
    int check;
    if(*target=='\0') //Base case, null target string
        return 1;
    if(*line=='\0') //Can't match an empty line
        return 0;
    while(*line!='\0') {
        if(matchesHere(target,line))
            return 1;
        line++;
    }
    //No match found
    return 0;
}

////////////////////////////////////
//long goToLine(file, scanString)
//
// Scans through file line by line until scanString is found or
// the end of file is reached. The number of lines read is returned
// or -1 if scanString is not found. This routine potentially
// suffers from a truncation error, since it assumes the file is
// nicely cut into lines with < 256 chars (no problem for normal
// aces2 output...)
////////////////////////////////////
long goToLine(FILE *file, char *scanString) {
    char buff[256];
    long lineCount=0;
    int found=0;
    do {
        fgets(buff,255,file);
        lineCount++;
        found=matches(scanString,buff);
    } while(!found && !feof(file));
    if(found)
        return lineCount;
    return -1;
}

////////////////////////////////////
//showHelp()
//
// Displays info on the command-line options.
////////////////////////////////////
void showHelp() {
    cout << endl << "Usage: gersh <aces2 output> <MOINTS file>"
        << " [options]" << endl;
    cout << "Options: " << endl;
    cout << "-debug      Show copious debugging output." << endl;
    cout << "-verbose    Show helpful(?) details." << endl;
    cout << "-elec <n>   Correlate n electrons." << endl;
    cout << "-energy     Recalculate CCSD energy." << endl;
}

////////////////////////////////////
//parseOptions(argc (number of arguments), argv (argument array),
// debug, verbose, nelec (number of electrons), energy)
//
// Scans through the argv list and processes any command-line
// options.
////////////////////////////////////
void parseOptions(int argc, char **argv, int &debug, int &verbose,
    int &elec, int &energy) {
    int n;
    for(n=3;n<argc;n++) {
        if(!strcmp(argv[n],"-help")||!strcmp(argv[n],"-?")) {
            showHelp();
            exit(-1); //multiple exit points.
            //deal with it, code fascists!
        }
        else
            if(!strcmp(argv[n],"-verbose"))
                verbose=1;
            else
                if(!strcmp(argv[n],"-debug"))
                    debug=1;
                else
                    if(!strcmp(argv[n],"-energy"))
                        energy=1;
                    else
                        if(!strcmp(argv[n],"-elec")) {
                            n++; //move to next argument
                            nelec=atoi(argv[n]);
                        }
            else {

```



```

        cout << "Option " << argv[n] << " ignored." << endl;
    }
}

//analyzeCalc(outfile, calctype, norbs)
//
// Extracts from outfile the calculation type (e.g. RHF, UHF) and
// the number of active orbitals.
//
void analyzeCalc(FILE *outfile, int &calctype, int &norbs) {
    char dummy[255]; //dummy buffer

    rewind(outfile);
    goToLine(outfile, "Parameters for SCF");
    fgets(dummy, 255, outfile);
    //Check for calculation type
    if (matches(" RHF", dummy)) {
        calctype=RHF;
        cout << "Detected RHF reference function." << endl;
    }
    else {
        //Unimplemented reference type
        cout << dummy << endl << "Not yet implemented!!" << endl;
        exit(-1); //More bad style
    }
    //Detection needs to be done by cases
    if (calctype==RHF) {
        goToLine(outfile, "Index      Eigenvalue      Symmetry");
        fgets(dummy, 255, outfile); //skip line of hyphens
        norbs=0;
        int good=0;
        float temp1, temp2, temp3;
        do {
            good=fscanf(outfile, "%f %f %f", &temp1, &temp2, &temp3);
            if (good>0)
                norbs++;
        }
        while (good>0);
        cout << "Read in " << norbs << " molecular orbitals." << endl;
    }
}

//readTlRHF(outfile (aces2 output), t1 (t1 class))
//
// Parses the aces2 output file to obtain the t1 amplitudes for
// the RHF case (again, ignoring symmetry).
//
void readTlRHF(FILE *outfile, twoindex *t1) {
    char buff[255];
    int i, j, n;
    double amp;

    rewind(outfile);
    //Skip to modified output portion
    goToLine(outfile, "Displaying T amplitudes");
    //Skip 3 lines
    for (n=0; n<3; n++)
        fgets(buff, 255, outfile);
    //loop over all T1 amplitudes
    for (n=0; n<nocc*nvirt; n++) {
        fscanf(outfile, "%i %i %lf", &i, &j, &amp);
        //convert to internal indices
        i--;
        j-=1+nocc;
        t1->set(i, j, amp);
    }
}

//int getListStart(outfile (aces2 output), listnum, irrep)
//
// Parses the aces2 output file to find where listnum for
// irreducible representation irrep starts on the MOINTS file
// Returns the number of words listnum is from the beginning
// of the MOINTS file.
//
// NB: Currently assumes 4096 words per record without checking.
//
int getListStart(FILE *outfile, int listnum, int irrep) {
    int n;
    char buff[255];

    rewind(outfile);
    //First, scan to the MOINTS directory listing
    goToLine(outfile, "List Subclass Distribution");
    //Skip the next three lines
    for (n=0; n<3; n++)
        fgets(buff, 255, outfile);

    //linear search for the proper entry
    int good;
    int record, word, list, subclass, dummy;
    do
        good=fscanf(outfile, "%i %i %i %i %i %i", &list, &subclass,
            &dummy, &dummy, &record, &word);
    while (good && (list != listnum || subclass != irrep));

    if (list != listnum || subclass != irrep)
        cout << "Danger! Failed to locate list (" << listnum
            << ", " << irrep << ")" << endl;

    return (record-1)*4096 + (word-1);
}

//loadListRHF(outfile (aces2 output), MOINTS, listnum, irrep,

```

```

//          fourList (4-index list to hold data))
//
// Locates the desired list on the MOINTS file and loads it into
// core memory in fourList (an instance of the fourIndex class
// structure).
//
// NB: The particle/hole string in fourList is used to determine
// the indices of each quantity. Thus, it must match the ordering
// used by aces2 in storing the lists. For details see the aces2
// user's manual. Note that the left-most index is the least
// significant.
//
//
//
//
//
void loadListRHF(FILE *outfile, FILE *MOINTS, int listnum,
                int irrep, fourIndex *fourList) {
    //First, obtain the location of the list
    int offset;
    offset=getListStart(outfile,listnum,irrep);
    //DEBUG
    //cout << "Found offset " << offset << " for list " << fourList->getName()
    // << endl;
    //DEBUG END
    //Now, seek to that location in MOINTS
    fseek(MOINTS,(offset/2)*sizeof(double),SEEK_SET);

    //Get size and index information from the fourList
    int size, n;
    double temp;
    int i1, i2, i3, i4;
    size=fourList->getSize();
    fourList->getFirstIndices(i1,i2,i3,i4);
    //Now, loop over the n data points
    for(n=0;n<size;n++) {
        fread(&temp,sizeof(double),1,MOINTS);
        fourList->set(i1,i2,i3,i4,temp);
        if(n<size-1) //avoid range-check error
            fourList->getNextIndices(i1,i2,i3,i4);
    }
    //Presumably, the list has now been read in
}

//
//
//
//
//double CCSDenergyRHF(T1, T2mix, T2same, aaaa, abab)
//
// Calculates and returns the CCSD correlation energy for RHF reference.
//
//
//
double CCSDenergyRHF(twoIndex *T1, fourIndex *T2mix, fourIndex *T2same,
                    fourIndex *aaaa, fourIndex *abab) {
    double tally=0;
    double term1=0, term2=0;
    double temp1, temp2, temp3, temp4;
    int i,j,a,b;
    //First, do aaaa sums
    for(i=0;i<nocc;i++)
        for(j=i+1;j<nocc;j++)
            for(a=0;a<nvirt;a++)
                for(b=a+1;b<nvirt;b++) {
                    temp1=T2same->get(a,b,i,j);
                    temp2=aaaa->get(a,b,i,j);
                    temp3=T1->get(i,a)*T1->get(j,b);
                    //temp4=T1->get(i,b)*T1->get(j,a);
                    tally+=temp2*(temp1 + 2*temp3);
                }
    //Now, do abab sums
    for(i=0;i<nocc;i++)
        for(j=0;j<nocc;j++)
            for(a=0;a<nvirt;a++)
                for(b=0;b<nvirt;b++) {
                    temp2=abab->get(a,b,i,j);
                    temp1=T2mix->get(a,b,i,j);
                    temp3=T1->get(i,a)*T1->get(j,b);
                    term2+=temp2*(temp1 + temp3);
                }
    term2/=2;
    return 2*(tally+term2);
}

//
//
//
//
//Low-level contraction functions
//
//
//
//
double tripAlss(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double tphase=1, intphase=-1;
    int temp, mu;
    if(b>c) {
        temp=b;
        b=c;
        c=temp;
        tphase*=-1;
    }
    if(i>j) {
        temp=i;
        i=j;
        j=temp;
        intphase*=-1;
    }
    for(mu=0;mu<nocc;mu++) {
        if(mu==k||mu==a)

```

```

        continue;
        if(mu<k)
            t=tphase*T2same->get(mu,k,b,c);
        else
            t=-tphase*T2same->get(k,mu,b,c);
        integral=intphase*ppph_aaaa->get(i,j,mu,a);
        result+=t*integral;
    }
    return result;
}

double tripBlss(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double tphase=1, intphase=-1;
    int temp, alpha;
    if(j>k) {
        temp=j;
        j=k;
        k=temp;
        tphase*=1.0;
    }
    if(a>b) {
        temp=a;
        a=b;
        b=temp;
        intphase*=1.0;
    }
    for(alpha=0;alpha<nvirt;alpha++) {
        if(alpha==c||alpha==i)
            continue;
        if(alpha<c)
            t=tphase*T2same->get(j,k,alpha,c);
        else
            t=-tphase*T2same->get(j,k,c,alpha);
        integral=intphase*hhhp_aaaa->get(a,b,alpha,i);
        result+=t*integral;
    }
    return result;
}

double tripAlms(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double intphase=-1;
    int temp, mu;
    if(i>j) {
        temp=i;
        i=j;
        j=temp;
        intphase*=-1;
    }
    for(mu=0;mu<nocc;mu++) {
        if(mu==a)
            continue;
        if(ks!=cs)
            t=-T2mixed->get(mu,k,c,b);
        else
            t=T2mixed->get(mu,k,b,c);
        integral=intphase*ppph_aaaa->get(i,j,mu,a);
        result+=t*integral;
    }
    return result;
}

double tripBlms(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double tphase=1, intphase=-1;
    int temp, alpha;
    if(a>b) {
        temp=a;
        a=b;
        b=temp;
        intphase*=1.0;
    }
    for(alpha=0;alpha<nvirt;alpha++) {
        if(alpha==i)
            continue;
        if(ks==cs)
            t=tphase*T2mixed->get(j,k,alpha,c);
        else
            t=-tphase*T2mixed->get(k,j,alpha,c);
        integral=intphase*hhhp_aaaa->get(a,b,alpha,i);
        result+=t*integral;
    }
    return result;
}

double tripAlsm(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double tphase=1, intphase=-1;
    int temp, mu;
    if(b>c) {
        temp=b;

```

```

        b=c;
        c=temp;
        tphase*=-1;
    }
    for(mu=0;mu<nocc;mu++) {
        if(mu==k)
            continue;
        if(mu<k)
            t=tphase*T2same->get(mu,k,b,c);
        else
            t=-tphase*T2same->get(k,mu,b,c);
        if(as!=js)
            integral=-intphase*ppph_abab->get(j,i,mu,a);
        else
            integral=intphase*ppph_abab->get(i,j,mu,a);
        result+=t*integral;
    }
    return result;
}

double tripBlsm(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double tphase=1, intphase=-1;
    int temp, alpha;
    if(j>k) {
        temp=j;
        j=k;
        k=temp;
        tphase*=1.0;
    }
    for(alpha=0;alpha<nvirt;alpha++) {
        if(alpha==c)
            continue;
        if(alpha<c)
            t=tphase*T2same->get(j,k,alpha,c);
        else
            t=-tphase*T2same->get(j,k,c,alpha);
        if(as!=is)
            integral=-intphase*hhhp_abab->get(a,b,alpha,i);
        else
            integral=intphase*hhhp_abab->get(b,a,alpha,i);
        result+=t*integral;
    }
    return result;
}

double tripAlmm(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double tphase=1, intphase=-1;
    int temp, mu;
    for(mu=0;mu<nocc;mu++) {
        if(ks!=cs)
            t=-tphase*T2mixed->get(mu,k,c,b);
        else
            t=tphase*T2mixed->get(mu,k,b,c);
        if(as!=js)
            integral=-intphase*ppph_abab->get(j,i,mu,a);
        else
            integral=intphase*ppph_abab->get(i,j,mu,a);
        result+=t*integral;
    }
    return result;
}

double tripBlmm(int i, int j, int k, int a, int b, int c,
                int is, int js, int ks, int as, int bs, int cs) {
    double result=0.0;
    double t, integral;
    double tphase=1, intphase=-1;
    int temp, alpha;
    for(alpha=0;alpha<nvirt;alpha++) {
        if(ks!=cs)
            t=-tphase*T2mixed->get(j,k,c,alpha);
        else
            t=tphase*T2mixed->get(j,k,alpha,c);
        if(as!=is)
            integral=-intphase*hhhp_abab->get(a,b,alpha,i);
        else
            integral=intphase*hhhp_abab->get(b,a,alpha,i);
        result+=t*integral;
    }
    return result;
}

//Diagram drivers:
//
//Each diagram has two levels of driver functions so that the
//implementation is as close to a literal rendering of the
//mathematics as possible. The top level driver generates all
//specified antisymmetrized permutations, while the second
//level driver determines the spin case and calls a final
//function to actually perform the contraction.
//second-level spin case drivers
double TripAld2(int i, int j, int k, int a, int b, int c) {
    //Unpack spin and orbital information
    int is, js, ks, as, bs, cs;
    int io, jo, ko, ao, bo, co;

```

```

is=i%2; io=i/2;
js=j%2; jo=j/2;
ks=k%2; ko=k/2;
as=a%2; ao=a/2;
bs=b%2; bo=b/2;
cs=c%2; co=c/2;

//Check to see if the element is spin-forbidden
if(bs==cs && ks!=bs)
    return 0.0;
if(is==js && as!=is)
    return 0.0;

//Sort according to spin-case
if(bs==cs) { //T2 same
    if(is==js) //same integral
        return tripAlss(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
    else
        return tripAlsm(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
}
else { //T2 mixed
    if(is==js)
        return tripAlms(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
    else
        return tripAlmm(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
}
}

double TripBld2(int i, int j, int k, int a, int b, int c) {
    //Unpack spin and orbital information
    int is, js, ks, as, bs, cs;
    int io, jo, ko, ao, bo, co;

    is=i%2; io=i/2;
    js=j%2; jo=j/2;
    ks=k%2; ko=k/2;
    as=a%2; ao=a/2;
    bs=b%2; bo=b/2;
    cs=c%2; co=c/2;

    if(as==bs && is != as)
        return 0.0;
    if(js==ks && cs != js)
        return 0.0;

    if(js==ks) { //T2 same
        if(as==bs)
            return tripBlss(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
        else
            return tripBlsm(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
    }
    else {
        if(as==bs)
            return tripBlms(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
        else
            return tripBlmm(io,jo,ko,ao,bo,co,is,js,ks,as,bs,cs);
    }
}

//top-level permutation drivers
double TripAldriver(int i, int j, int k, int a, int b, int c) {
    double result=0.0;
    result+= -TripAld2(i,j,k,a,b,c);
    result+= +TripAld2(i,j,k,a,c,b);
    result+= +TripAld2(i,j,k,c,b,a);
    result+= +TripAld2(i,k,j,a,b,c);
    result+= -TripAld2(i,k,j,a,c,b);
    result+= -TripAld2(i,k,j,c,b,a);
    result+= +TripAld2(k,j,i,a,b,c);
    result+= -TripAld2(k,j,i,a,c,b);
    result+= -TripAld2(k,j,i,c,b,a);
    return result;
}

double TripBldriver(int i, int j, int k, int a, int b, int c) {
    double result=0.0;
    result+= TripBld2(i,j,k,a,b,c);
    result+= -TripBld2(i,j,k,c,b,a);
    result+= -TripBld2(i,j,k,a,c,b);
    result+= -TripBld2(j,i,k,a,b,c);
    result+= TripBld2(j,i,k,c,b,a);
    result+= TripBld2(j,i,k,a,c,b);
    result+= -TripBld2(k,j,i,a,b,c);
    result+= TripBld2(k,j,i,c,b,a);
    result+= TripBld2(k,j,i,a,c,b);
    return result;
}

////////////////////////////////////
//Main function
////////////////////////////////////
int main(int argc, char **argv) {
    //Print banner
    cout << "gersh: Calculates Gershgorin estimator v0.3" << endl;
    //Program operation flags
    int debug=0;
    int verbose=0;
    int energy=0;
    //The first two arguments must be the aces2 output and MOINTS file
    //respectively. If argc is too small, display the help and quit

```

```

if(argc<3) {
    cout << "gersh: file names missing" << endl << endl;
    showHelp();
    return -1;
}
//Most importantly need the number of active electrons
int nelecs=-1; //set to minus 1 as a flag
//Try opening the files to be sure everything works
FILE *outfile=fopen(argv[1],"r");
FILE *intsfile=fopen(argv[2],"rb");
if(!outfile)
    cout << "Couldn't open aces2 output file: " << argv[1] << endl;
if(!intsfile)
    cout << "Couldn't open integral file: " << argv[2] << endl;
if(!outfile||!intsfile)
    return -1;
//Parse any special options
parseOptions(argc,argv,debug,verbose,nelec,energy);
if(nelec==-1) {
    nelecs=2;
    cout << "Warning! Number of electrons not specified" << endl
    << "Defaulting to 2 electrons." << endl;
}
//If this is debugging mode, examine MOINTS
if(debug)
    printmoints(intsfile);
//We must read the MOINTS file structure
int norbs; //number of orbitals
//For now, we will make no attempt to exploit the
//symmetry group of the molecule under consideration
if(debug||verbose)
    cout << "Scanning " << argv[1] << "..." << endl;
analyzeCalc(outfile,calctype,norbs);
if(calctype==RHF) {
    //Currently only closed-shell RHF reference works
    //Then occupied orbitals are just n/2, etc...
    nocc=nelec/2;
    nvirt=norbs-nocc;
    cout << "Correlating " << nelecs << " electrons..." << endl;
    if(verbose) {
        cout << nocc << " occupied orbitals." << endl;
        cout << nvirt << " virtual orbitals." << endl;
    }
    //O.K., now we have to load all the relevant coupled-cluster info
    //We'll start with T1
    //Sadly, T1 does not appear in MOINTS, so we have to parse it
    //out of the aces2 output file.
    int numt1=nocc*nvirt;
    t1 = new twoIndex("T1 Amps","ph","aa");
    //Now, read in the T1 amplitudes...
    readTLRHF(outfile,t1);
    if(debug)
        t1->dump();
    //Load the mixed T2 amplitudes
    T2mixed = new fourIndex("T2 IjAb", "pphh", "abab");
    loadListRHF(outfile,intsfile,46,1,T2mixed);
    if(debug)
        T2mixed->dump();
    //Load the alpha-alpha T2 amplitudes
    T2same = new fourIndex("T2 IJAB", "pphh", "aaaa");
    loadListRHF(outfile,intsfile,44,1,T2same);
    if(debug)
        T2same->dump();
    //For coupled-cluster energy, only integrals
    //of the form <pp|hh> appear.
    //Load the hhhh, aaaa integrals (list 14)
    hhhh_aaaa = new fourIndex("<AB|IJ>", "hhpp", "aaaa");
    loadListRHF(outfile,intsfile,14,1,hhhh_aaaa);
    if(debug)
        hhhh_aaaa->dump();
    //Load the hhhh, abab integrals (list 16)
    hhhh_abab = new fourIndex("<Ab|Ij>", "hhpp", "abab");
    loadListRHF(outfile,intsfile,16,1,hhhh_abab);
    if(debug)
        hhhh_abab->dump();
    //consistency test
    if(energy)
        cout << "CCSD correlation energy: " <<
        CCSDenergyRHF(t1,T2mixed,T2same,hhhh_aaaa,hhhh_abab) << endl;
    //For the excited diagrams, we need more integral lists
    ppph_aaaa = new fourIndex("<IJ|KA>", "ppph", "aaaa");
    loadListRHF(outfile,intsfile,7,1,ppph_aaaa);
    ppph_abab = new fourIndex("<Ij|Ka>", "ppph", "abab");
    loadListRHF(outfile,intsfile,10,1,ppph_abab);
    hhhh_aaaa = new fourIndex("<AB|CI>", "hhhp", "aaaa");
    loadListRHF(outfile,intsfile,27,1,hhhh_aaaa);
    hhhh_abab = new fourIndex("<Ab|Ci>", "hhhp", "abab");
    loadListRHF(outfile,intsfile,30,1,hhhh_abab);

    double G_3 = 0.0; //this is supposed to be the triples piece...

```

```

//Now, loop over all triples
//I'll try to follow to notation used in my thesis (latin indices to
//specify the matrix elements, greek indices to specify summation)
//I guess that's the Sommerfeld convention.
int i, j, k; //occupied indices
int a, b, c; //virtual indices
double element; //tally current element
for(k=2;k<2*nocc;k++)
  for(j=1;j<k;j++)
    for(i=0;i<j;i++)
      for(c=2;c<2*nvirt;c++)
        for(b=1;b<c;b++)
          for(a=0;a<b;a++) { //Whew!
            if(debug)
              cout << "Calculating matrix element for: " << endl
                << a << "\t" << b << "\t" << c << endl
                << i << "\t" << j << "\t" << k << endl;
            //The convention here for RHF reference is that
            //even numbers (0, 2, 4,...) are alpha spin (0, 1, 2)
            //odd numbers (1, 3, 5,...) are beta spin (0, 1, 2)
            //Begin calling diagrams (see thesis)
            element=0;
            element+=TripAldriver(i,j,k,a,b,c);
            element+=TripBldriver(i,j,k,a,b,c);
          }
    }
  }
}

```

Bibliography

- [1] T.J. Lee R. Kobayashi N.C. Hardy R.D. Amos. Comparison of the brueckner and coupled-cluster approaches to electron correlation. *J. Chem. Phys.*, 96(12):8931–8937, 1992.
- [2] P. Piecuch S.A. Kucharski R.J. Bartlett. Coupled-cluster methods with internal and semi-internal triply and quadruply excited clusters: Ccsdt and ccsdtq approaches. *J. Chem. Phys.*, 110(13):6103–6122, 1999.
- [3] T.D. Crawford and H.F. Schaefer. An introduction to coupled cluster theory for computational chemists. In K.B. Lipkowitz and D.B. Boyd, editors, *Reviews in Computational Chemistry*, volume 14. John Wiley and Sons, Inc., 2000.
- [4] M.R. Hoffmann and H.F. Schaefer. A full coupled-cluster singles, doubles, and triples model for the description of electron correlation. In *Advances in Quantum Chemistry*, volume 18. Academic Press, Inc., 1986.
- [5] T. Kato. *A Short Introduction to Perturbation Theory for Linear Operators*. Springer-Verlag New York, Inc., 1982.
- [6] P.D. Lax. *Linear Algebra*. John Wiley and Sons, Inc., 1997.
- [7] Quantum Theory Project. Aces 2 users manual. Available electronically as a Postscript file.
- [8] J.J. Sakurai. *Advanced Quantum Mechanics*. Addison-Wesley, 1967.
- [9] J.F. Stanton and R.J. Bartlett. The equation of motion coupled-cluster method. a systematic biorthogonal approach to molecular excitation energies, transition probabilities, and excited state properties. *J. Chem. Phys.*, 98(9):7029–7039, 1993.
- [10] F. H. Stillinger. Møller-plesset convergence issues in computational quantum chemistry. *J. Chem. Phys.*, 112(22):9711–9715, 2000.
- [11] B.T. Sutcliffe. Fundamental of computational quantum chemistry. In G.H.F. Diercksen B.T. Sutcliffe and A. Veillard, editors, *Computational Techniques in Quantum Chemistry*. McGraw-Hill Publishing Company, 1975.

- [12] A. Szabo and N.S. Ostlund. *Modern Quantum Chemistry*. McGraw-Hill Publishing Company, 1982.
- [13] T.J. Lee P.R. Taylor. *Int. J. Quantum Chem. Symp.*, 23(199), 1989.
- [14] M. Tinkham. *Group Theory and Quantum Mechanics*. McGraw-Hill Publishing Company, 1964.
- [15] J. Townsend. *A Modern Approach to Quantum Mechanics*. McGraw-Hill Publishing Company, 1992.
- [16] J. Čížek. On the correlation problem in atomic and molecular systems. calculation of wavefunction components in ursell-type expansion using quantum-field theoretical methods. *J. Chem. Phys.*, 45(11):4256–4266, 1966.
- [17] G.C. Wick. The evaluation of the collision matrix. *Phys. Rev.*, 80(268), 1950.