Ground state energy of quantum dots using the coupled cluster method

Winther-Larsen, Sebastian Gregorius[1,*] and Schøyen, Øyvind Sigmundson[1,*]

[1] *University of Oslo*

(Dated: June 13, 2018)

The coupled cluster method for estimating the ground state energy of a many-body system is arguably the most important post-Hartree-Fock method. We have implemented the coupled cluster method with double excitations (CCD). We find that it does not perform well with a simple harmonic oscillator basis, but the results vastly improve when it is provided a basis from a restricted Hatree-Fock self-consistent field method run as a preliminary computation. For a system consisting of $N = 2$ interacting particles, using 12 orbital shells we get a ground stated energy of 3.005970, very close to the analytical "true answer" of 3[1].

CONTENTS

I. INTRODUCTION

In this project we will compute the ground state energy of quantum dots, a construct that has become a popular area of study in recent decades, both from a theoretical and an experimental point of view. We achieve the ground state computation by implementing a coupled cluster amplitude equation solver which is used in succession with a restricted Hartree-Fock self-consistent field method.

Regardless of how sophisticated the method used, the many-body problem remains quite intractable. The true wavefunction of any system will be a linear combination of all possible slater-type orbitals one can model. This means that in order to represent a system correctly, one would need a basis set of infinite size. This is an obvious problem when we solve problems numerically, because we obviously need to limit the size of the basis we use because we have limitations in memeory and computating power. Luckily, great minds have bestowed a wide selection of very god numerical methods upon the world, which the coupled cluster is a part of.

First, we provide a thorough theoretical outline of how we model a quantum dot, the mathematical formalism behind many-body problems and outline how the computational methods we employ both turn into iterative schemes that are easy to represent numerically. Second,

* Project code: https://github.com/Schoyen/FYS4411

we provide a brief overview of the program we have constructed which is in the form an installable python package. Third, we have employed the methods on a large parameter space presenting results for systems consisting of $N = \{2, 6, 12, 20\}$ particles. Fourth, we discuss the most interesting aspects of the results and try to shed some light over the shortcomings as well as the beneficial features of the different metods. Lastly, we end on some summary remarks.

II. THEORY

In this project we will study a system of N interacting electrons. We will be looking at a Hamiltonian consisting of a one-body and a two-body part. The one-body part is given by

$$h(\mathbf{r}_i) = -\frac{1}{2}\nabla_i^2 + \frac{1}{2}\omega^2\mathbf{r}_i^2, \tag{1}$$

where we use natural units $\hbar = c = e = 1$ and set the mass to unity. The two-body part is the Coulomb interaction potential,

$$u(\mathbf{r}_i, \mathbf{r}_j) = \frac{1}{\left|\mathbf{r}_i - \mathbf{r}_j\right|}. \tag{2}$$

We thus get the total Hamiltonian

$$H = h + u = \sum_{i=1}^{N} h(\mathbf{r}_i) + \sum_{i<j}^{N} u(\mathbf{r}_i, \mathbf{r}_j), \tag{3}$$

where h is the full one-body operator and u the full two-body operator, i.e., over the entire system. Working in a basis of L single particle functions, $\{|p\rangle\}_{p=1}^{L}$. We define the reference Slater determinant as

$$|\Phi_0\rangle \equiv |1, 2, \ldots, N\rangle, \tag{4}$$

i.e., a tensorproduct of the N first single particle functions, $|i\rangle$, of the system. We call these single particle functions *occupied* as they are contained in the Slater determinant. We will denote the occupied indices with $i, j, k, l, \cdots \in \{1, \ldots, N\}$, the *virtual* states with $a, b, c, d, \cdots \in \{N+1, \ldots, L\}$ and general indices with $p, q, r, s, \cdots \in \{1, \ldots, L\}$. We define $M = L - N$ as the number of virtual states. In terms of sets of basis functions we can write this as

$$\{|p\rangle\}_{p=1}^{L} = \{|i\rangle\}_{i=1}^{N} \cup \{|a\rangle\}_{a=N+1}^{L}, \tag{5}$$

i.e., the general indexed states consists of both occupied and virtual states. Note that the single particle functions are orthonormal, i.e.,

$$\langle p|q\rangle = \delta_{pq}. \tag{6}$$

We can construct other Slater determinants in this basis by exciting or relaxing the reference determinant. A general excitation is labeled $|\Phi_{ij\ldots}^{ab\cdots}\rangle$ which means that we have removed the single particle functions with indices $i, j, \ldots$ from the reference and added $a, b, \ldots$. Note that

$$\langle \Phi_{ij\ldots}^{ab\cdots} | \Phi_0 \rangle = 0, \tag{7}$$

for any excitation.

A. *Motivational Interlude*: Quantum Dots

The system described above is a theoretical approximation of a quantum dot. Some consideration should be given to an explanation of why the system described above is a quantum dot, and why we should care about quantum dots. There are at least two things we refer to as quantum dots. The first one is a man-made device, also known as artificial atoms, and the other is the simplest theoretical construct of such a device.

Aritifical atom quantum dots are semiconducting crystals, typically GaAs, and they are relatively easy to build in a laboratory. Some free electrons are confined to a volume, either by a physical barrier, like an insulator, or an electromagnetic field[2]. Due to the confinement of the electrons, the energy levels of a quantum dot becomes quantized, and they therefore hold many properties similar to naturally occuring quantized systems, which is the reason why quantum dots have attracted much interest.

The theoretical approximation of a quantum dot is impossible to make perfectly accurate, yet simple. There are some important effects that should be included such the potential arising as a result of the confinement, and the interaction between particles. We therefore model our system with an electron-electron interaction, given by the two-body operator in Equation 2, and with we approximate the confining potential by the harmonic oscillator potential, given by the second term in the one-body Hamiltonian in Equation 1. The last term in the one-body Hamiltonian is the kinetic energy of particle in question, essential in any system.

B. Second quantization

Employing the creation operators, $a_p^\dagger$, and the destruction operators, a_p, we can write the Hamiltonian as

$$H = h_q^p a_p^\dagger a_q + \frac{1}{4} u_{rs}^{pq} a_p^\dagger a_q^\dagger a_s a_r, \tag{8}$$

where we have used the Einstein summation convention with repeated indices having an implicit sum. The matrix elements are defined as

$$h_q^p \equiv \langle p|h|q\rangle, \tag{9}$$

$$u_{rs}^{pq} \equiv \langle pq|u|rs\rangle - \langle pq|u|sr\rangle, \tag{10}$$

where the u-matrix consists of the the antisymmetric two-body elements.

C. The coupled cluster approximation

We approximate the true wavefunction, $|\Psi\rangle$, of the system by the coupled cluster wavefunction, $|\Psi_{\mathrm{CC}}\rangle$, defined by

$$|\Psi_{\mathrm{CC}}\rangle \equiv e^T|\Phi_0\rangle = \left(\sum_{i=0}^{n}\frac{1}{n!}T^n\right)|\Phi_0\rangle, \tag{11}$$

where the *cluster operator*, T, is given by a sum of p-excitation operators labeled T_p. They consist of *cluster amplitudes*, $t_{i\ldots}^{a\cdots}$, and creation and annihilation operators.

$$T = T_1 + T_2 + \cdots + T_p \tag{12}$$

$$= t_i^a a_a^\dagger a_i + \left(\frac{1}{2!}\right)^2 t_{ij}^{ab} a_a^\dagger a_b^\dagger a_i a_j + \ldots. \tag{13}$$

In our approximation we limit the cluster operator to only include double excitations,

$$T \equiv T_2 = \frac{1}{4}t_{ij}^{ab} a_a^\dagger a_b^\dagger a_j a_i. \tag{14}$$

The first part of the coupled cluster method consists of constructing the cluster amplitudes using the *amplitude equations*. After we have found the amplitudes we can compute the energy.

D. Energy of the coupled cluster approximation

When we're going to compute the energy of a system using the coupled cluster approximation we would ideally want to find the expectation value of the energy using the coupled cluster wavefunction.

$$E_{\mathrm{CC}} = \langle\Psi_{\mathrm{CC}}|H|\Psi_{\mathrm{CC}}\rangle. \tag{15}$$

As it turns out, this is an uncomfortable way of finding the energy as $T \neq T^\dagger$. Instead we will define what we call the *similarity transformed Hamiltonian*. We plug the coupled cluster wavefunction into the Schrödinger equation.

$$H|\Psi_{\mathrm{CC}}\rangle = E_{\mathrm{CC}}|\Psi_{\mathrm{CC}}\rangle. \tag{16}$$

Next, we left multiply with the inverse of the cluster expansion, i.e.,

$$e^{-T}H|\Psi_{\mathrm{CC}}\rangle = e^{-T}E_{\mathrm{CC}}|\Psi_{\mathrm{CC}}\rangle = E_{\mathrm{CC}}|\Phi_0\rangle. \tag{17}$$

Projecting this equation onto the reference state we get

$$E_{\mathrm{CC}} = \langle\Phi_0|e^{-T}H|\Psi_{\mathrm{CC}}\rangle = \langle\Phi_0|e^{-T}He^T|\Phi_0\rangle, \tag{18}$$

where in the latter inner-product we have located the similarity transformed Hamiltonian defined by

$$\bar{H} \equiv e^{-T}He^T. \tag{19}$$

To simplify the energy equation and the amplitude equations we use the normal ordered Hamiltonian.

$$H = H_N + \langle\Phi_0|H|\Phi_0\rangle. \tag{20}$$

The energy equation thus becomes

$$E_{\mathrm{CC}} = \langle\Phi_0|\bar{H}|\Phi_0\rangle = E_0 + \langle\Phi_0|e^{-T}H_Ne^T|\Phi_0\rangle, \tag{21}$$

where the reference energy is given by

$$E_0 = \langle\Phi_0|H|\Phi_0\rangle. \tag{22}$$

We now define the normal ordered similiarity transformed Hamiltonian as

$$\bar{H}_N \equiv e^{-T}H_Ne^T. \tag{23}$$

By expanding the exponentials of this Hamiltonian and recognizing the commutators we get the Baker-Campbell-Hausdorff expansion.

$$\bar{H}_N = H_N + [H_N, T] + \frac{1}{2!}\left[[H_N, T], T\right] + \ldots. \tag{24}$$

From the connected cluster theorem we know that the only nonzero terms in the Baker-Campbell-Hausdorff expansion will be the terms where the normal ordered Hamiltonian has at least one contraction[1] with every cluster operator on its right. This lets us write the expansion as

$$\bar{H}_N = H_N + (H_NT)_c + \frac{1}{2!}\left(H_NT^2\right)_c + \ldots, \tag{25}$$

where the subscript c signifies that only contributions where at least one contraction between H_N and T has been performed will be included.

1. Coupled cluster doubles energy equation

Using the doubles approximation with the cluster operator T_2 defined in Equation 14 the energy equation becomes

$$E_{\mathrm{CCD}} = E_0 + \langle\Phi_0|e^{-T_2}H_Ne^{T_2}|\Phi_0\rangle. \tag{26}$$

The doubles cluster operator will excite a pair of single particle functions in the reference state. As the single particle functions are orthonormal we have that

$$\langle\Phi^X|\Phi_0\rangle = 0, \tag{27}$$

where X is an excitation different from zero. This means that for a term in the Baker-Campbell-Hausdorff expansion contribute with a nonzero value to the energy equation it must leave the reference state in its original state.

---

[1] In the Wick's theorem sense.

The two-body part of the Hamiltonian is only able to relax a pair of single particle functions in the reference state whereas the doubles cluster operator will excite a pair. This means that we are only left with the following contributions to the energy equation.

$$E_{\text{CCD}} = E_0 + \langle\Phi_0|H_N|\Phi_0\rangle + \langle\Phi_0|(H_N T_2)_c|\Phi_0\rangle. \quad (28)$$

Further, by construction we have that

$$\langle\Phi_0|H_N|\Phi_0\rangle = 0. \quad (29)$$

In the second term only the normal ordered two-body operator can contribute as the cluster operator gives a total excitation of +2. As we are projecting onto the reference we have to relax to zero again. The normal ordered Fock operator is at most able to excite and relax by 1 and does therefore not contribute to the overall expression.

$$\langle\Phi_0|(W_N T_2)_c|\Phi_0\rangle = \frac{1}{4}u^{ij}_{ab}t^{ab}_{ij}. \quad (30)$$

In total the energy equation reduces to

$$E_{\text{CCD}} = h^i_i + \frac{1}{2}u^{ij}_{ij} + \frac{1}{4}u^{ij}_{ab}t^{ab}_{ij}, \quad (31)$$

where the first two terms come from the reference energy. See Appendix B for the calculation of this expression.

E. Coupled cluster amplitude equations

In order for us to solve the energy equation using the coupled cluster approximation we need to figure out what the cluster amplitudes, $t^{ab\cdots}_{ij\ldots}$, are. This is done by projecting Equation 17 onto an excited Slater determinant, i.e.,

$$\langle\Phi^{ab\cdots}_{ij\ldots}|e^{-T}He^{T}|\Phi_0\rangle = 0. \quad (32)$$

Note that in the amplitude equations we can use both the regular and the normal ordered Hamiltonian. They are equal as the reference energy term disappears due to Equation 7. The order of the excitation in the projection determines the order of the amplitudes you will find. As in the case of the energy equation we use the Baker-Campbell-Hausdorff expansion when expanding the exponentials in the similarity transformed Hamiltonian. We only keep the terms which excites the reference state to the same degree as the amplitudes we are trying to find.

1. Coupled cluster doubles amplitude equations

In our case we are only interested in the second order ampltiudes found in the doubles approximation, hence we will solve the equation

$$\langle\Phi^{ab}_{ij}|e^{-T_2}H_N e^{T_2}|\Phi_0\rangle = 0, \quad (33)$$

to find an expression that can be used to solve for the amplitudes t^{ab}_{ij}. As the state we are projecting onto is doubly excited we will only keep terms from the Baker-Campbell-Hausdorff expansions which leaves the reference doubly excited.

$$\bar{H} = \left(H_N + H_N T_2 + \frac{1}{2}H_N T_2^2\right)_c. \quad (34)$$

Higher order terms will leave the reference in a too high excitation state.

Now comes the rather tedious task of evaluating all the terms that arises from inserting Equation 34 into Equation 33. This can be done by applying Wick's generalised theorem, but the task is a daunting and streneous one. A few example computations of how this can be done is included in Appendix C. Instead of doing it in this manner, we employ the second quantization library from SymPy [2]. The CCD amplitude equation, from SymPy[3], is

$$\begin{aligned} 0 = {} & u^{ab}_{ij} + f^b_c t^{ac}_{ij}P(ab) - f^k_j t^{ab}_{ik}P(ij) \\ & + \frac{1}{4}t^{cd}_{ij}t^{ab}_{mn}u^{mn}_{cd} + \frac{1}{2}t^{cd}_{ij}u^{ab}_{cd} \\ & + \frac{1}{2}t^{cd}_{jm}t^{ab}_{in}u^{mn}_{cd}P(ij) - \frac{1}{2}t^{ac}_{nm}t^{bd}_{ij}u^{nm}_{cd}P(ab) \\ & + t^{ac}_{im}t^{bd}_{jn}u^{mn}_{cd}P(ij) + t^{ac}_{im}u^{bm}_{jc}P(ab)P(ij) \\ & - \frac{1}{2}t^{ab}_{im}u^{mn}_{jn}. \end{aligned} \quad (35)$$

Here $P(ij)$ is an exchange operator which interchanges two particles with indices i and j.

2. The Iterative Scheme

In order to find the amplitude t we have to start with an initial guess and use an iterative procedure to improve on the initial guess. We start by picking the diagonal elements of f to be a part of the unperturbed Hamiltonian and consider the rest of the terms a perturbation. The second and third terms in Equation 35 can now be rewritten,

$$\begin{aligned} & f^b_c t^{ab}_{ij}P(ab) - f^k_j t^{ab}_{ik}P(ij) \\ & \quad \to f^b_b t^{ab}_{ij} - f^a_a t^{ba}_{ij} - f^j_j t^{ab}_{ij} + f^i_i t^{ab}_{ji} \\ & \quad = (f^a_a + f^b_b - f^i_i - f^j_j)t^{ab}_{ij} \\ & \quad = (\epsilon_a + \epsilon_b - \epsilon_i - \epsilon_j)t^{ab}_{ij} \\ & \quad = -(\epsilon_i + \epsilon_j - \epsilon_a - \epsilon_b)t^{ab}_{ij} \\ & \quad = -D^{ab}_{ij}t^{ab}_{ij}, \end{aligned} \quad (36)$$

[2] This is also more in the spirit of this project, as it is within the realm of *Computational* Physics.

[3] Note that this is one of many ways to write the amplitude equation as the ordering of the indices can change drastically for every run of the SymPy-script.

where the arrow signifies that we only look at the diagonal elements of f. We now define the right hand side of Equation 35 to $g(f,u,t)$ where we make sure that we only use the off-diagonal elements of f in the computations.

By moving $D_{ij}^{ab}t_{ij}^{ab}$ to the left hand side we we get

$$D_{ij}^{ab}t_{ij}^{ab} = g(u,t). \tag{37}$$

This allows us to define an iterative scheme,

$$t^{(k+1)} = \frac{g(u,t^{(k)})}{D_{ij}^{ab}}, \tag{38}$$

with the initial guess

$$t^{(0)} = \frac{u_{ij}^{ab}}{D_{ij}^{ab}}. \tag{39}$$

3. Intermediate computations

Looking closely at the tensors in Equation 35 one might come to realize that many of the terms share a common structure. This warrants the search for an algebraic transformation of the amplitude equation that has the potential to reduce the amount of floating point operations needed to compute it. As it turns out, such terms exist and they will decrease the computing time necessary by two orders of magnitude for the price of higher storage costs. We will define the following "intermediates"[4],

$$\chi_{cd}^{ab} = \frac{1}{4}t_{mn}^{ab}u_{cd}^{mn} + \frac{1}{2}u_{cd}^{ab} \tag{40}$$

$$\chi_{j}^{n} = \frac{1}{2}t_{jm}^{cd}u_{cd}^{mn} \tag{41}$$

$$\chi_{d}^{a} = \frac{1}{2}t_{nm}^{ac}u_{cd}^{nm} \tag{42}$$

$$\chi_{jc}^{bm} = u_{jc}^{bm} + \frac{1}{2}t_{jn}^{bd}u_{cd}^{mn} \tag{43}$$

See Appendix D for the derivation. These intermediate structures will allow us to rewrite Equation 35 to,

$$\begin{aligned} 0 = u_{ij}^{ab} &+ f_c^b t_{ij}^{ac} P(ab) - f_j^k t_{ik}^{ab} P(ij) \\ &+ t_{ij}^{cd}\chi_{cd}^{ab} + t_{in}^{ab}\chi_j^n P(ij) - t_{ij}^{bd}\chi_d^a P(ab) \\ &+ t_{im}^{ac}\chi_{jc}^{bm} P(ab)P(ij) + \frac{1}{2}t_{im}^{ab}u_{jn}^{mn}. \end{aligned} \tag{44}$$

Doing this reduces the number of FLOPS from $\mathcal{O}(M^4N^4)$ for the heaviest contraction, i.e., the contraction

$$g(f,u,t) \leftarrow \frac{1}{4}t_{ij}^{cd}t_{nm}^{ab}u_{cd}^{mn}, \tag{45}$$

to $\mathcal{O}(M^4N^2)$ when using the largest intermediate to do the contraction

$$g(f,u,t) \leftarrow t_{ij}^{cd}\chi_{cd}^{ab}, \tag{46}$$

for the price of storing M^4 more matrix elements and pre-computing the intermediate.

[4] Note that are more than one way of defining the intermediates.

F. Constructing the matrix elements

Having found the equations needed in order to find an estimate to the ground state energy using the coupled cluster doubles approximation is all well and dandy. But, we need basis functions to create the matrix elements needed to feed into the coupled cluster code. Often these basis functions are not known and we have to use an approximation or utilize Hartree-Fock to create more optimized basis functions.

1. Harmonic oscillator basis

We will be looking at a system of two-dimensional quantum dots with a Coulomb repulsion. If we assume, or make it so, that the repulsive two-body part is small we can use the eigenfunctions of the one-body part as our basis of single particle functions. In this case we have two-dimensional harmonic oscillator functions as eigenfunctions. We can then pre-compute the matrix elements, h_q^p and u_{rs}^{pq}, before feeding these into the coupled cluster code.

In polar coordinates we can write the harmonic oscillator wavefunction for a single particle in two dimensions as[5].

$$\phi_{nm}(r,\theta) = N_{nm}(ar)^{|m|}L_n^{|m|}(a^2r^2)e^{-a^2r^2/2}e^{im\theta}, \tag{47}$$

where $a = \sqrt{m\omega/\hbar}$ is the Bohr radius, $L_n^{|m|}$ is the associated Laguerre polynomials, n and m are the principal and azimuthal quantum numbers respectively and N_{nm} is a normalization constant given by

$$N_{nm} = a\sqrt{\frac{n!}{\pi(n+|m|)!}}. \tag{48}$$

Included is also the spin, σ, of the wavefunction, which can be either up or down. This means each level, (n,m), is doubly occupied. We also have that the wavefunctions are orthonormal

$$\langle n_1 m_1 \sigma_1 | n_2 m_2 \sigma_2 \rangle = \delta_{n_1 n_2}\delta_{m_1 m_2}\delta_{\sigma_1 \sigma_2}. \tag{49}$$

The eigenenergy of a single harmonic oscillator is given by

$$\epsilon_{nm} = \hbar\omega(2n+|m|+1). \tag{50}$$

Our next job is now to create a mapping from the three quantum numbers n, m and σ to a single quantum number α as the matrices h and u use single indices for each

[5] Note that this is without spin. As we are looking at fermions this means that each mode of the harmonic oscillator function will be repeated twice.

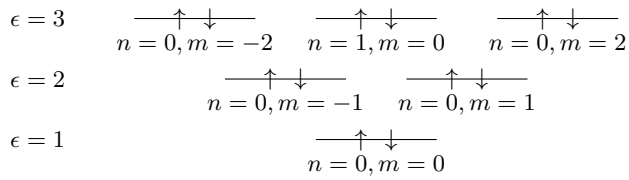

FIG. 1: In this plot we can see the energy degeneracy of the lowest three energy levels in the two-dimensional quantum dot. Each arrow representes a spin up or a spin down state with the quantum numbers n and m as listed below. This pattern goes on indefinitly with the addition of one bar (two oscillators) per level.

wavefunction. In Figure 1 we can see the lowest three energy levels that needs to be mapped.

Starting from the bottom and working our way upwards from left to right we can label each line from 0 to n in increasing order. This enumeration will serve as our common quantum number α[6]. We will only work with full *shells*, i.e., we restrict our views to systems of N particles where N will be a magic number which we get by counting all spin states for each energy level and the energy levels below. In Figure 1 we can see the magic numbers $N \in [2, 6, 12]$ by counting the levels in increasing order.

The normalization condition now reads

$$\langle \alpha | \beta \rangle = \delta_{\alpha\beta}\delta_{\sigma_\alpha\sigma_\beta}, \tag{51}$$

The one-body matrix will be a diagonal matrix with the eigenenergies of the single particle harmonic oscillator functions as elements.

$$\langle \alpha | h | \beta \rangle = \epsilon_\beta \delta_{\alpha\beta}\delta_{\sigma_\alpha\sigma_\beta}. \tag{52}$$

The two-body matrix elements are a little harder to work out as the harmonic oscillator wavefunctions are not eigenfunctions to the correlation operator. Luckily, from E. Anisimovas and A. Matulis (Equation A2)[3] we can get an analytical expression for the two-body matrix elements. We can then construct the antisymmetric two-body matrix elements in the harmonic oscillator basis.

$$u^{\alpha\beta}_{\gamma\delta} = \langle \alpha\beta | u | \gamma\delta \rangle - \langle \alpha\beta | u | \delta\gamma \rangle. \tag{53}$$

Computing the Coulomb elements is an extremely costly operation and we will thus only compute the non-zero orbital integrals, i.e., neglecting spin and antisymmetry, before saving these to a file which we can read several times.

[6] Note that we use greek letters $\alpha, \beta, \ldots$ for the harmonic oscillator wavefunctions as opposed to latin letters for the general indices.

2. Constructing the Hartree-Fock basis

Having found the matrix elements of h and u we can now use the *self-consistent field* (SCF) iteration method to construct h and u in a *restricted Hartree-Fock* (RHF) basis. This will yield a better estimate to the actual, unknown basis functions of the system.

The neatest, yet arguably the most abstract, way to write the Hartree-Fock equations for electron i is

$$f\varphi_p = \varepsilon_p \varphi_p, \tag{54}$$

where f is the Fock operator, φ_p are eigenstates of the Fock operator consisting of a set of one-electron wave functions called the Hartree-Fock molecular orbitals, and ε_p are the eigenenergies of the Fock operator. The Fock operator, in matrix notation, is given by

$$f^p_q = h^p_q + J(D)^p_q - \frac{1}{2}K(D)^p_q, \tag{55}$$

where h^p_q is the one-body operator, while the two-body operator is divided into what we call a direct part,

$$J(D)^p_q = \langle pr | u | qs \rangle D_{sr}, \tag{56}$$

and an exchange part,

$$K(D)^p_q = \langle pr | u | sq \rangle D_{sr}. \tag{57}$$

Pay special attention to the non-antisymmetric two-body elements in the direct and exchange part. The direct part of the two-body operator is comparable to classical Coloumb repulsion, while the exchange does not have a classical analog as it arises from the antisymmetry requirement of the wavefunction. The matrix D_{sr} is the density matrix of the system and will be defined shortly.

Introducing a basis set transforms the Hartree-Fock equations into the Rothaan equations

$$FC = SC\varepsilon, \tag{58}$$

where F is the Fock matrix constructed from f^p_q. This is a generalised eigenvalue problem where

$$S^p_q = \langle p | q \rangle, \tag{59}$$

i.e., the *overlap matrix*. We now define the density matrix D by

$$D^p_q \equiv C^p_\alpha C^{\alpha *}_q, \tag{60}$$

Since the Fock matrix F depends on it's own solution through the orbitals, the eigenvalue problem must be solved iteratively[7]

[7] This is also the reason why the Roothan-Hall equations are often called the self-consistent-field procedure.

Because the RHF SCF method is a variational method, the initial wavefunction made up of the single-particle wavefunctions φ_i is varied until the minimal energy is found.

$$E_{\text{RHF}} = \left(f_q^p - \frac{1}{2} J(D)_q^p + \frac{1}{4} K(D)_q^p \right) D_p^q. \tag{61}$$

The transformations necessary to make all these variations will now be contained in C. So if we start from a harmonic oscillator (HO) basis, $|\alpha\rangle$ we can transform into a better basis, called a Hartree-Fock (HF) basis, $|p\rangle$.

$$|p\rangle = \big(|\alpha\rangle\langle\alpha|\big)|p\rangle = C_p^\alpha |\alpha\rangle. \tag{62}$$

We will see how the coupled cluster method with double excitations works with both the HO basis and the HF basis, but not equally well.

III. IMPLEMENTATION

We have costructed a flexible framework for performing Hartree-Fock self-consistent field iterations and coupled cluster with double excitations computations. The entire code base consists of package for Python that is easy to install globally on any computer.

Because we have employed a mixture of Python and C++ with Cython interfaces, the code sturcture may be difficult to understand for someone without Cython experience, but we have tried to make the actual usage of the software as painless as possible.

Most of the computations that are done in a coupled cluster algorithm is linear algebra and we found that NumPy, relying on BLAS and LAPACK, is quite fast and at the very least, efficient enough for our purposes. The quicker and easier scripting capabilities Python provides is worth what little extra speed-up writing the entire code base in C++ could have provided.

A. Installation and Usage

The software is easy to install by first cloning the GitHub reopsitory,

```
git clone git@github.com:Schoyen/FYS4411.git
```

Then all you need to do is change directory to the project code folder where a Makefile is included so you only need to build and install,

```
cd FYS4411/project_2/coupled-cluster
make build
make install
```

Now everything should be able to run from anywhere on your computer. To ensure that all requirements of the program are satisfied you can install with `make installr` instead. Running `py.test` on the command-line in the top-directory[8] will run the test scripts located in the `tests`-directory to make sure that the installation passed successfully.

The script that is used to generate most of the data is located in `https://github.com/Schoyen/FYS4411/blob/master/project_2/scripts/comparison.py` and provides a good example on how to use the code base.

B. Program Structure

There are three main subsections within our program structure, given in the directory tree below.

```
coupled_cluster
├── matrix_elements
│   ├── generate_matrices
│   └── index_map
├── hartree_fock
│   ├── scf_rhf
│   └── basis_transformation
└── schemes
    ├── ccd
    ├── ccd_sparse
    └── ccd_optimized
```

The first subsection, `matrix_elements`, contains methods for computing matrix elements in harmonic oscillator basis from an analytical expression[3]. Computing the matrix elements is a very intensive task, and the central functions are therefore implemented in C++ with a Cython interface to enable use in Python[9].

The `hartree_fock` subsection contains methods for changing from harmonic oscillator basis to Hartree-Fock basis as well as an implementation of the self-consistent-field algorithm to make this transaction possible.

The `schemes` subsection is arguably the most important part of this project. The subsection has three different classes that perform the exact same computations, but in different and increasingly intelligent ways. First, `CoupledClusterDoubles` is the most straightforward and naïve way to solve the CCD amplitude equations. Second, because an overbearing amount of the elements in the operator matrices in this problem are zero, we have implemented a sparse matrix CCD solver in `CoupledClusterDoublesSparse`. Third, `CoupledClusterDoublesOptimized` is parallellized and optimized with memory use and number of floating point operations in mind.

---

[8] That is, in `FYS4411/project_2/coupled-cluster`.

[9] The C++ code is inherited from Alocias Mariadason from his project in FYS4411 last year with slight modifications. The original source code can be found here `https://github.com/0o1Insane1o0/FYS4411/blob/master/project1/src/Coulomb_Functions.cpp` and `https://github.com/0o1Insane1o0/FYS4411/blob/master/project1/src/Coulomb_Functions.hpp`.

C. Index mapping

The index mapping generating a $(n, m) \mapsto p$ and $p \mapsto (n, m)$ is a Python adaption of the code found here: `https://github.com/ManyBodyPhysics/LectureNotesPhysics/blob/master/Programs/Chapter8-programs/cpp/CCD/src/qDots2DSPBasis.cpp#L18-L68`.

D. Generating the antisymmetric two-body elements

Constructing the Coulomb elements is by far the most intensive task that is done. Luckily, we only need to do this once for each basis size as the elements do not change and can be reused. We therefore compute the elements once and store all the non-zero elements as Python pickle-objects. If we wish to do a new run we first look for the relevant Coulomb elements in the `dat`-directory before attempting to create them anew.

After the elements are created/read we have to scale them with the current frequency, i.e.,

$$\langle \alpha\beta|u|\gamma\delta\rangle = \sqrt{\omega}\langle 12|u|34\rangle, \tag{63}$$

where $\langle 12|u|34\rangle$ is found from E. Anisimovas and A. Matulis (Equation A2)[3]. Next we antisymmetrize the two-body elements and include spin which doubles each dimension of the orbital integrals.

$$\begin{aligned} u^{\alpha\beta}_{\gamma\delta} = &\delta_{\sigma_\alpha\sigma_\gamma}\delta_{\sigma_\beta\sigma_\delta}\langle \alpha\beta|u|\gamma\delta\rangle \\ &- \delta_{\sigma_\alpha\sigma_\delta}\delta_{\sigma_\beta\sigma_\gamma}\langle \alpha\beta|u|\delta\gamma\rangle. \end{aligned} \tag{64}$$

The antisymmetrization process uses four nested loops and is therefore implemented in Cython to reduce the computation time.

E. Reducing the complexity of the basis transformation

Having computed the RHF energy and the basis transformation matrix C we can transform from the HO basis to the HF basis. For the one-body matrix elements this becomes

$$\langle p|h|q\rangle = C^{p\dagger}_{\alpha} C^{\beta}_{q} \langle \alpha|h|\beta\rangle. \tag{65}$$

Note the usage of the Hermitian conjugate and complex conjugate. We define them in such a manner

$$C^{p\dagger}_{\alpha} = C^{\alpha *}_{p}. \tag{66}$$

For the two-body elements this is a more complicated task[10].

$$\langle pq|u|rs\rangle = C^{p\dagger}_{\alpha} C^{q\dagger}_{\beta} C^{\gamma}_{r} C^{\delta}_{s} \langle \alpha\beta|u|\gamma\delta\rangle, \tag{67}$$

which translates into eight nested loops when programming the expression as it stands. Luckily, by noticing that no indices are repeated more than twice we can do each loop separately viz.

$$\langle pq|u|rs\rangle = C^{p\dagger}_{\alpha}\left[C^{q\dagger}_{\beta}\left[C^{\gamma}_{r}\left[C^{\delta}_{s}\langle \alpha\beta|u|\gamma\delta\rangle\right]\right]\right]. \tag{68}$$

This reduces the computations to four quintuply nested loops[11].

F. Implementation details

Both in `CoupledClusterDoublesSparse` and `CoupledClusterDoublesOptimized` we make use of the intermediates in equations 40 to 43. In the optimized class special care was taken to ensure that every contraction was computed in an optimal way, with the minimization of memory use and CPU time as the overall goal. Moreover, several methods in NumPy for doing linear algebra was tested, including but not limited to `numpy.einsum`, `numpy.tensordot`, and reshaping datastructures to two dimensions and using `numpy.matmul`. Additionally, we tried to make use of the python package `dask` which allows for arrays that are larger than the computer's memory buffer as well as automated concurrency. We found that the dask arrays was a bad fit for our code base, but we ended up using `numba` for paralellization of some iterative loops as well as just-in-time compilation of some functions.

G. Scaling of the coupled cluster doubles method

When talking about the scaling of the coupled cluster method there are two aspects to consider.

1. The number of FLOPS to compute $g(f, u, t)$.
2. The number of bytes needed to store the arrays of matrix elements.

For the CCD method the naïve solution using the form of $g(f, u, t)$ without the intermediates results in $\mathcal{O}(M^4N^4)$ FLOPS each iteration. Using the form of the intermediates we have set up this reduces to $\mathcal{O}(M^4N^2)$, i.e., two order of magnitude less. The effect of this is *dramatic* and will be shown in the results section in due time.

A downside of using the intermediates[12] is the higher storage costs. Lucky enough for us, the largest storage cost comes from storing u^{pq}_{rs} and overshadows all other

[10] Imagine the poor fellows that needs to deal with three-body elements...

[11] This trick was made aware to us through `http://sirius.chem.vt.edu/wiki/doku.php?id=crawdad:programming:project4`.

[12] Particularly our form for the intermediates as we store χ^{ab}_{cd} instead of a rewriting this intermediate as χ^{ij}_{kl} which is also possible.

TABLE I: Storage costs of u_{rs}^{pq} in gigabytes for a given shell R and the corresponding number of basis functions L assuming 8 B (`double`) precision.

R	L	u_{rs}^{pq} [GB]
8	72	0.2
9	90	0.49
10	110	1.1
11	132	2.3
12	156	4.4
13	182	8.2
14	210	14.5

terms as it scales as L^4. See Table I for some example storage costs. For a normal desktop computer we quickly reach a threshold where u_{rs}^{pq} will eat up all memory and activate disk storage thus triggering a massive slowdown in computational speed.

For larger shell values we have to be particularly careful in terms of the memory usage. This can be semi-problematic when using a programming language such as Python as memory usage is camouflaged. Using NumPy allows us to reuse memory buffers quite efficiently, but we have to be careful what type of operations we do and if they incur copying.

1. *Sparse implementation*

It turns out that most of the elements in both $u_{\gamma\delta}^{\alpha\beta}$ (the HO basis) and u_{rs}^{pq} (the HF basis) are zero. Roughly 0.014% and 0.082% of the elements are non-zero respectively (for $R = 12$). This inspired us to look for a way to do sparse CCD. Using the `sparse`-library in Python[13] we were able to set up a sparse implementation. Unfortunately being efficient in terms of re-using memory is quite complicated and there are definitely improvements to be made. Nonetheless the performance of the method is comparable to the optimized code.

2. *Parallelizing the contractions*

The easiest way to parallelize the coupled cluster code is to use OpenMP on the contractions. In particular the heavy contractions. We solved this using Numba with the `prange` function which essentially translates into an OpenMP directive. This turned out to be slower than the built-in NumPy calls, especially since NumPy comes with the option of running in parallel if the code is built with BLAS by setting the environment variable `OMP_NUM_THREADS`. For example:

```
export OMP_NUM_THREADS=4
```

[13] Found here: `https://github.com/pydata/sparse`.

Therefore, our most optimized code uses the matrix multiplication code (`np.matmul`) from NumPy which translates into the BLAS call `DGEMM` for matrix-matrix multiplication. This requires us to reshape the four-dimensional tensors as matrices and write them in such a way that we can do the contraction using matrix multiplication. Here NumPy comes to our rescue as a reshape is done inplace and all memory is stored as a contiguous array.

H. Convergence Problems and Mixing

Iterative many-body methods are prone to convergence problems for some configurations. Since Hartree-Fock is a variational method, SCF convergence is found when the energy is stationary with respect to inifitesimal variations in the orbitals. Unfortunately, the SCF iteration scheme does not always converge. Luckily, numerous techniques exist for controlling and accelerating convergence[4]. The same kind of methods have proven useful to ensure convergence of coupled cluster methods [5].

The simplest way to try to "massage" convergence out of the CCD-method is to use *damping* where you include a part of the result from the previous iteration i.e.,

$$\tilde{t}^{k+1} = (1-\theta)t^{k+1} + \theta t^k, \tag{69}$$

where t^{k+1} is the current value computed using Equation 38 and t^k is the previous value for the amplitude. Choosing $\theta \in [0, 1]$ we can tune how much of the previous amplitude we wish to include in the new state. This allows for a more gradual transition between the iterations. We now use $\tilde{t}^{k+1}$ as our estimate of the new amplitude.

A more sophisticated mixing method is the direct inversion in the iterative subspace (DIIS), also known as Pulay mixing. While the common mixing method is used for many other applications [14], the DIIS method is developed with the sole intent of accelerating convergense in Hartree-Fock methods. In DIIS one would construct a linear combinations of approximate errors from previous iterations, analogous to a very clever weighted moving average. We have not implemented this scheme, but it nevertheless warrants mention.

IV. RESULTS

A. Ground state energies for two-dimensional quantum dots

Here we show the ground state energies for the two-dimensional quantum dots using restriced Hartree-Fock (RHF), coupled cluster doubles using both the harmonic

[14] It is, for instance, called an alpha filter in data acquisition.

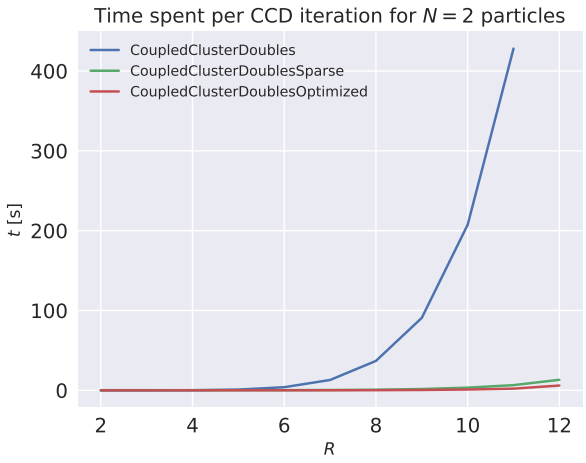

FIG. 2: In this figure we can see the running time per iteration for $N = 2$ particles as a function of the shell number R. We did not run the naïve implementation for $R = 12$.

oscillator basis and the Hartree-Fock basis which we construct after running the RHF-method. We have chosen the convergence criteria to be 1×10^{-6} in our results for the sake of comparison with M. P. Lohne[6].

All the results from our simulations are displayed in tables II, III, IV and V. The tables show the results for systems of increasing size, $N = 2$, $N = 6$, $N = 12$ and $N = 20$ particles respectively. The results are thoroughly discussed in the next section.

B. Running time

Here we show how the three implementations of the CCD-method scales as a function of the shell number R for $N = 2$ and $N = 6$, in figures 2 and 3 respectively. The same behaviour is shown for higher number of particles. In Figure 4 we have plotted the computation time for $N = 6$ particles again, but we have excluded the naïve method for better comparison of the sparse and optimized method.

V. DISCUSSION

A. Performance comparison

Somewhat surprisingly, the Hartree-Fock method outperforms the coupled cluster doubles when the harmonic oscillator basis is used. This happens for almost any configuration except the smallest system with $N = 2$ particles. This comes from the fact that the HO basis is not very good at representing the states in the system. Because the Hartree-Fock SCF iterations is a variational method, the basis is varied in order to find the minimal energy. The new basis states will be linear combinations of the old basis states. When the iterations are finished, the new transformed HF basis will then be much better at representing the possible quantum configurations compared to the HO basis. This effect is only apparent in systems with a larger number of particles and/or low

TABLE II: Results for $N = 2$ particles, where convergence was achieved for all parameters. Taut's[1] analytic result for $\omega = 1.0$ a.u. is 3.

ω	R	RHF	CCD(HO)	CCD(HF)
0.1	1	0.596333	0.596333	0.596333
	2	0.596333	0.512520	0.512520
	3	0.526903	0.505972	0.442235
	4	0.526903	0.499216	0.442011
	5	0.525666	0.497172	0.443293
	6	0.525666	0.494232	0.443145
	7	0.525635	0.493142	0.443056
	8	0.525635	0.491895	0.442981
	9	0.525635	0.491262	0.442927
	10	0.525635	0.490649	0.442886
	11	0.525635	0.490262	0.442853
	12	0.525635	0.489918	0.442827
0.5	1	1.886227	1.886227	1.886227
	2	1.886227	1.786914	1.786914
	3	1.799856	1.778903	1.681979
	4	1.799856	1.760117	1.673881
	5	1.799748	1.754385	1.670053
	6	1.799748	1.748232	1.667804
	7	1.799745	1.745231	1.666474
	8	1.799745	1.742548	1.665494
	9	1.799743	1.740860	1.664805
	10	1.799743	1.739444	1.664270
	11	1.799742	1.738416	1.663856
	12	1.799742	1.737562	1.663522
1.0	1	3.253314	3.253314	3.253314
	2	3.253314	3.152328	3.152328
	3	3.162691	3.141827	3.039048
	4	3.162691	3.118679	3.025273
	5	3.161921	3.110967	3.017944
	6	3.161921	3.103338	3.013923
	7	3.161909	3.099324	3.011405
	8	3.161909	3.095916	3.009621
	9	3.161909	3.093662	3.008343
	10	3.161909	3.091818	3.007357
	11	3.161909	3.090436	3.006590
	12	3.161909	3.089299	3.005970
2.0	1	5.772454	5.772454	5.772454
	2	5.772454	5.671234	5.671234
	3	5.679048	5.658272	5.553528
	4	5.679048	5.631669	5.534333
	5	5.677282	5.622092	5.523490
	6	5.677282	5.613118	5.517552
	7	5.677206	5.608130	5.513709
	8	5.677206	5.604026	5.511012
	9	5.677204	5.601216	5.509050
	10	5.677204	5.598946	5.507540
	11	5.677204	5.597213	5.506356
	12	5.677204	5.595791	5.505399

TABLE III: Results for $N = 6$ particles. No convergence is marked as 😱 in the table. For low frequencies the CCD-method using the HO basis has a hard time achieving convergence.

ω	R	RHF	CCD(HO)	CCD(HF)
	2	4.864244	4.864244	4.864244
	3	4.435740	4.446235	4.319901
	4	4.019787	4.383692	3.829962
	5	3.963149	😱	3.666722
	6	3.870617	😱	3.597876
0.1	7	3.863135	😱	3.590388
	8	3.852880	😱	3.587711
	9	3.852591	😱	3.587291
	10	3.852393	😱	3.587136
	11	3.852391	😱	3.586821
	12	3.852382	😱	3.586582
	2	13.640713	13.640713	13.640713
	3	13.051620	13.385987	12.901520
	4	12.357471	13.261097	12.057345
	5	12.325128	13.138572	11.934988
	6	12.271499	13.084158	11.864098
0.5	7	12.271375	13.068399	11.849762
	8	12.271361	13.055561	11.841330
	9	12.271337	13.045386	11.835470
	10	12.271326	13.037878	11.831351
	11	12.271324	😱	11.828242
	12	12.271320	😱	11.825835
	2	22.219813	22.219813	22.219813
	3	21.593198	21.974675	21.423811
	4	20.766919	21.854191	20.429265
	5	20.748402	21.793624	20.332454
	6	20.720257	21.750091	20.274013
1.0	7	20.720132	21.718843	20.249849
	8	20.719248	21.695224	20.234705
	9	20.719248	21.675931	20.224389
	10	20.719217	21.661830	20.217075
	11	20.719215	21.649812	20.211541
	12	20.719215	21.640765	20.207258
	2	37.281425	37.281425	37.281425
	3	36.637217	37.042127	36.450634
	4	35.689555	36.925664	35.328432
	5	35.681729	36.864367	35.250185
	6	35.672333	36.812895	35.200308
2.0	7	35.671851	36.775986	35.168245
	8	35.670358	36.747864	35.147097
	9	35.670333	36.725261	35.131953
	10	35.670144	36.708362	35.121033
	11	35.670143	36.694188	35.112680
	12	35.670127	36.683281	35.106188

TABLE IV: Here we look at $N = 12$ particles. We did not achieve convergence using the harmonic oscillator basis for the lower frequency values and large number of shells. No convergence is marked as 😱 in the table.

ω	R	RHF	CCD(HO)	CCD(HF)
	3	46.361130	46.361130	46.361130
	4	43.663267	45.837079	43.309845
	5	41.108851	45.456883	40.654710
	6	40.750512	😱	40.068340
0.5	7	40.302719	😱	39.508500
	8	40.263752	😱	39.399128
	9	40.216688	😱	39.329311
	10	40.216252	😱	39.309409
	11	40.216195	😱	39.296007
	12	40.216165	😱	39.285968
	3	73.765549	73.765549	73.765549
	4	70.673849	73.314476	70.324250
	5	67.569930	72.990679	67.031096
	6	67.296869	😱	66.526677
1.0	7	66.934745	😱	66.049564
	8	66.923094	😱	65.972157
	9	66.912244	😱	65.921205
	10	66.912035	😱	65.889281
	11	66.911365	😱	65.866715
	12	66.911364	😱	65.849776
	3	120.722260	120.722260	120.722260
	4	117.339642	120.296556	116.995036
	5	113.660396	120.007146	113.048934
	6	113.484866	119.759037	112.658821
2.0	7	113.247601	119.662199	112.309482
	8	113.246579	119.584733	112.235521
	9	113.246303	119.524394	112.181828
	10	113.245854	119.472283	112.140661
	11	113.245256	119.430353	112.109973
	12	113.245183	119.394712	112.085683
	3	242.334879	242.334879	242.334879
	4	238.739591	241.927593	238.394598
	5	234.352741	241.663950	233.680649
	6	234.282331	241.507595	233.425243
5.0	7	234.194820	241.390525	233.226529
	8	234.194059	241.293417	233.137933
	9	234.190797	241.221056	233.070205
	10	234.190714	241.158896	233.020009
	11	234.190665	241.109926	232.980634
	12	234.190553	241.068091	232.948528

frequency. This is because in a system with a large number of particles relative to the frequency will experience that the particle-particle interaction will be more prominent than the harmonic oscillator potential the particles are subject to. The trapped particles in the harmonic oscillator potential will be pushed out of the well unless we increase the frequency thus increasing the strength of the trap.

In other words, our first guess using the harmonic oscillator basis functions as an approximation to the exact eigenfunction of the quantum dot Hamiltonian is only a good first attempt when the number of particles is low and/or the frequency is very high. In those situations we can treat the particle-particle interaction as a small perturbation.

B. Validity of results

The analytical computed energy for $\omega = 1.0$ a.u. and $N = 2$ is 3[1]. From our tables we see that RHF is least successful in reaching this value, CCD in HF basis is

TABLE V: In this table we look at $N = 20$ particles. We did not achieve convergence using the harmonic oscillator basis for the lower frequency values and large number of shells. No convergence is marked as 😲 in the table.

ω	R	RHF	CCD(HO)	CCD(HF)
1.0	4	177.963297	177.963297	177.963297
	5	168.792442	177.206536	168.459124
	6	161.339721	😲	160.594507
	7	159.958722	😲	158.841120
	8	158.400172	😲	157.038330
	9	158.226030	😲	156.676039
	10	158.017667	😲	156.367930
	11	158.010276	😲	156.292422
	12	158.004951	😲	156.238258
2.0	4	286.825295	286.825295	286.825295
	5	276.898196	286.159148	276.381708
	6	267.269712	285.614958	266.413122
	7	266.213200	😲	264.969415
	8	264.933622	😲	263.434546
	9	264.874009	😲	263.215451
	10	264.809954	😲	263.046195
	11	264.809901	😲	262.963703
	12	264.809306	😲	262.899698
5.0	4	563.773952	563.773952	563.773952
	5	552.630093	563.160136	552.118708
	6	540.804720	562.692231	539.824400
	7	540.227793	562.306123	538.886074
	8	539.499326	562.114279	537.925127
	9	539.495941	😲	537.769045
	10	539.494611	😲	537.646668
	11	539.493513	😲	537.548828
	12	539.491764	😲	537.470616
10.0	4	973.032700	973.032700	973.032700
	5	961.371081	972.439478	960.862053
	6	948.057077	972.002302	947.019789
	7	947.765474	971.716015	946.399546
	8	947.410305	971.508584	945.827806
	9	947.409440	971.332133	945.663820
	10	947.404930	971.193592	945.528489
	11	947.404361	971.076617	945.424175
	12	947.403875	970.978571	945.339080

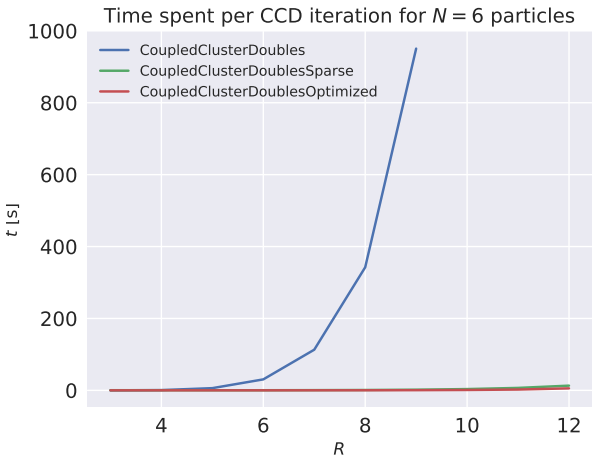

FIG. 3: Here we see the running time per iteration for $N = 6$ particles as a function of the shell number R. We did not run the naïve implementation for $R > 9$ in this figure.

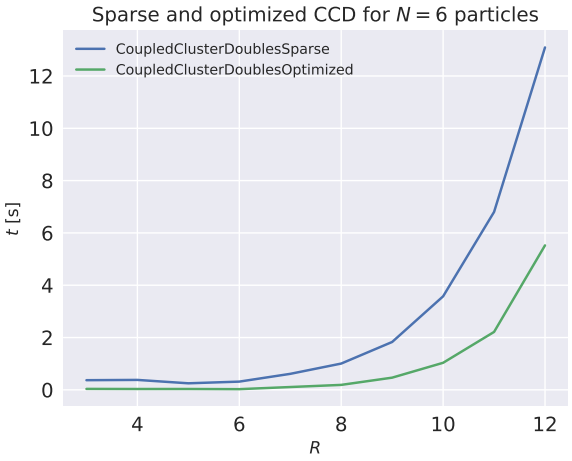

FIG. 4: Run time for $N = 6$ particles, only the faster methods are included for better comparison.

best, while CCD with HO basis is generally somewhere between the two. We see that we get even closer to the analytical "truth" by increasing the number of shells.

The rest of the results have been compared with the master thesis of M. P. Lohne[7] for up to $R = 10$ shells. Lohne gets two sets of energies, one set from RHF and one set from CCSD code with harmonic oscillator basis functions. Our CCD code with Hartree-Fock basis will for some configurations[15] beat CCSD with plain harmonic oscillator basis. For $R = 12$ shells we can compare with the results from M. P. Lohne et al.[6], but in this article an *effective interaction* for the Hamiltonian with a CCSD code using the HF basis has been used. These results are therefore significantly better than ours, but we can get a "ball-park" idea to benchmark against.

[15] By configurations we mean frequency ω and number of particles N.

C. Running time

Looking at the running times shown in Figure 2 and Figure 3 we can see how the naïve implementation blows up very fast. As both the sparse and the optimized version uses intermediate calculations these running times scales on the order of $\mathcal{O}(l^6)$ whereas the naïve implementation goes as $\mathcal{O}(l^8)$. Another important factor is that the naïve implementation uses `np.einsum` to calcu-

late the tensor contractions whereas the sparse and optimized version uses `sparse.tensordot` and `np.matmul` respectively. The two latter function calls uses BLAS's implementation of the matrix dot which is way faster than Numpys `np.einsum`.

D. Convergence trouble

For a large number of particles and a low frequency the CCD-method get trouble with convergence. This happens as the confining potential $v \propto \omega^2$ will not be able to confine the particles when the interaction gets too strong. The CCD-method will in particular get a hard time keep the particles together as the excitation operator t only excites pairs of particles leading to a too strong increase in the interaction when the particles increase their energy level. This effect can potentially be somewhat alleviated by including the singles excitation, e.g., using the CCSD-method.

We can see this behaviour in our results. For $N = 2$ we get convergence for all methods for $\omega = 0.1$ shown in Table II. By increasing to $N = 6$ particles we immediately see how CCD with the HO basis will get in trouble for small values of ω. Even by increasing the strength of the confinment this method does converge.

For $N = 6$ none of the methods converged when $\omega = 0.1$[16]. By increasing the frequency we restored the convergence. For larger number of shells we had to start increasing the value of $\theta \to 0.9$ in order to get convergence within the threshold.

CCD with HO basis experiences the same convergence problems also for, $N = 12$ particles and for $N = 20$ particles.

VI. SUMMARY REMARKS

Our main product from this is a product that can do both computations with restricted Hartree-Fock (RHF) self-consistent field and coupled cluster doubles (CCD). We have tried to make a package in python that should be easy enough for anyone to test. Moreover we hav demonstrated that one can speed up the computation time with a relatively simple algebraic transformation and finding the "intermediates". The computational speed was further reduced using sparse matrices. It should be noted, as a warning, that there are relatively few systems adequately sparese for this to work. We added upon the program to increase it further with paralellization and optimal use of numerical functions.

In order to perform the computations we have constructed a harmonic oscillator basis. We saw how the CCD method was outperformed RHF in most cases, and often exhibited convergence problems. The remedy for these shortcomings is to use the Hartree-Fock basis which is provided after successful convergence of SCF iterations. The results are improved markedly and classification of coupled cluster methods as a post-Harte-Fock method is justified.

The coupled cluster method provides surprisingly good results even though one only includes double esctiations. The reason for this is that particle-particle interactions is the amongt the most important thing that happens in a ferminon system, second only to no interaction at all, i.e what particles do on their own. The way to improve the computations is to include other types of excitations like singles (CCSD) and triples (CCSDT), add more basis functions, and/or supplement the coupled cluster method with another method like a perturbative one. All of this will, needless to say, increase the compute time, which must be balanced against the accuracy needed for the situation at hand.

[16] The RHF-method would probably have converged if we had started tuning its mixing parameter, but as we have focused on CCD we did not think it relevant to include these results.

Appendix A: The normal ordered Hamiltonian

When constructing the normal ordered Hamiltonian we use Wick's theorem to write the one-body, h, and the two-body, u, operators onto a normal ordered form. Specifically we define the normal ordered form in terms of the *Fermi vacuum*[17]. That is, an operator on normal ordered form destroys the reference Slater determinant.

We start by writing the one-body operator, h, to its normal-ordered form.

$$h = \sum_{pq} h_q^p a_p^\dagger a_q = \sum_{pq} h_q^p \left(\{a_p^\dagger a_q\} + \{\overbrace{a_p^\dagger a_q}\} \right) \tag{A1}$$

$$= \sum_{pq} h_q^p \{a_p^\dagger a_q\} + \sum_{pq} h_q^p \delta_{p\in i}\delta_{pq} \tag{A2}$$

$$= h_N + \sum_i h_i^i, \tag{A3}$$

where we have used $\delta_{p\in i}$ to mean that p must be an occupied index. Doing the same for the two-body operator is a slightly more tedious endeavor. For brevity we will only write out the operator strings and only keep the non-zero contributions.

$$\begin{aligned} a_p^\dagger a_q^\dagger a_s a_r = &\{a_p^\dagger a_q^\dagger a_s a_r\} + \{\overbrace{a_p^\dagger a_q^\dagger} a_s a_r\} + \{\overbrace{a_p^\dagger a_q^\dagger a_s a_r}\} \\ &+ \{a_p^\dagger \overbrace{a_q^\dagger a_s} a_r\} + \{a_p^\dagger \overbrace{a_q^\dagger a_s a_r}\} \\ &+ \{\overbrace{a_p^\dagger a_q^\dagger a_s} a_r\} + \{\overbrace{a_p^\dagger a_q^\dagger a_s a_r}\} \end{aligned} \tag{A4}$$

$$\begin{aligned} = &\{a_p^\dagger a_q^\dagger a_s a_r\} - \delta_{p\in i}\delta_{ps}\{a_q^\dagger a_r\} + \delta_{p\in i}\delta_{pr}\{a_q^\dagger a_s\} \\ &+ \delta_{q\in i}\delta_{qs}\{a_p^\dagger a_r\} - \delta_{q\in i}\delta_{qr}\{a_p^\dagger a_s\} \\ &- \delta_{p\in i}\delta_{ps}\delta_{q\in j}\delta_{qr} + \delta_{p\in i}\delta_{pr}\delta_{q\in j}\delta_{qs}. \end{aligned} \tag{A5}$$

Inserted into the full two-body operator and sorting out the sums we get

$$\begin{aligned} u = &\frac{1}{4}\sum_{pqrs}\langle pq||rs\rangle\{a_p^\dagger a_q^\dagger a_s a_r\} - \frac{1}{4}\sum_{iqr}\langle iq||ri\rangle\{a_q^\dagger a_r\} \\ &+ \frac{1}{4}\sum_{iqs}\langle iq||is\rangle\{a_q^\dagger a_s\} + \frac{1}{4}\sum_{pir}\langle pi||ri\rangle\{a_p^\dagger a_r\} \\ &- \frac{1}{4}\sum_{pis}\langle pi||is\rangle\{a_p^\dagger a_s\} - \frac{1}{4}\sum_{ij}\langle ij||ji\rangle \\ &+ \frac{1}{4}\sum_{ij}\langle ij||ij\rangle. \end{aligned} \tag{A6}$$

Using the antisymmetric properties of the two-body matrix elements,

$$\langle pq||rs\rangle = -\langle pq||sr\rangle = -\langle qp||rs\rangle = \langle qp||sr\rangle, \tag{A7}$$

and relabeling of the indices we can rearrange and collect some terms.

$$u = W_N + \sum_{pir}\langle pi||ri\rangle\{a_p^\dagger a_r\} + \frac{1}{2}\sum_{ij}\langle ij||ij\rangle, \tag{A8}$$

where the normal ordered two-body operator is

$$W_N = \frac{1}{4}\sum_{pqrs}\langle pq||rs\rangle\{a_p^\dagger a_q^\dagger a_s a_r\}. \tag{A9}$$

When we now construct the full Hamiltonian we can collect some terms. The constants in both the one-body and the two-body operator in total constitues the reference energy.

$$E_0 \equiv \langle\Phi_0|H|\Phi_0\rangle = \sum_i h_i^i + \frac{1}{2}\sum_{ij}\langle ij||ij\rangle. \tag{A10}$$

Combining the normal ordered one-body operator and the second term in the two-body operator, i.e., the term with a single creation and annihilation operator pair, we get the normal ordered Fock-operator.

$$F_N = \sum_{pq} h_q^p \{a_p^\dagger a_q\} + \sum_{pqi}\langle pi||qi\rangle\{a_p^\dagger a_q\} \tag{A11}$$

$$= \sum_{pq} f_q^p \{a_p^\dagger a_q\}, \tag{A12}$$

where we have defined the Fock matrix elements as

$$f_q^p = h_q^p + \sum_i \langle pi||qi\rangle. \tag{A13}$$

In total we get the full Hamiltonian

$$H = F_N + W_N + \langle\Phi_0|H|\Phi_0\rangle \tag{A14}$$

$$= H_N + \langle\Phi_0|H|\Phi_0\rangle, \tag{A15}$$

which is what we wanted to show.[8]

Appendix B: Energy equation

Here we show how to compute the coupled cluster doubles energy from Equation 31. The first two terms come from the reference energy shown in Equation A10. The last term we get by using Wick's theorem.

$$\langle\Phi_0|(H_N T_2)_c|\Phi_0\rangle = \langle\Phi_0|(W_N T_2)_c|\Phi_0\rangle, \tag{B1}$$

as the normal ordered Fock operator can at most relax a single state, i.e.,

$$\langle\Phi_0|(F_N T_2)_c|\Phi_0\rangle = 0. \tag{B2}$$

We are thus left with computing the contraction of the doubles cluster operator and the normal ordered two-body operator. Note that we must keep only the fully contracted terms.

$$\langle\Phi_0|(W_N T_2)_c|\Phi_0\rangle = \frac{1}{16}u_{rs}^{pq}t_{ij}^{ab}\langle\Phi_0|A|\Phi_0\rangle, \tag{B3}$$

[17] Fermi vacuum defines the reference state, i.e., $|\Phi_0\rangle$, as the vacuum.

where A is the create and destruction operator strings. By using generalized Wick's theorem we get

$$\begin{aligned}
A &= \{a_p^\dagger a_q^\dagger a_s a_r\}\{a_a^\dagger a_b^\dagger a_j a_i\} \\
&= \{a_p^\dagger a_q^\dagger a_s a_r a_a^\dagger a_b^\dagger a_j a_i\} + \{a_p^\dagger a_q^\dagger a_s a_r a_a^\dagger a_b^\dagger a_j a_i\} \\
&\quad + \{a_p^\dagger a_q^\dagger a_s a_r a_a^\dagger a_b^\dagger a_j a_i\} + \{a_p^\dagger a_q^\dagger a_s a_r a_a^\dagger a_b^\dagger a_j a_i\} \\
&= \delta_{pi}\delta_{qj}\delta_{sb}\delta_{ra} - \delta_{pi}\delta_{qj}\delta_{sa}\delta_{rb} \\
&\quad - \delta_{pj}\delta_{qi}\delta_{sb}\delta_{ra} + \delta_{pj}\delta_{qi}\delta_{sa}\delta_{rb}.
\end{aligned} \tag{B4}$$

Inserted back into Equation B3 and summing the indices in the Kronecker-Delta's we get

$$\begin{aligned}
\langle\Phi_0|(W_N T_2)_c|\Phi_0\rangle &= \frac{1}{16}\left(u_{ab}^{ij} - u_{ba}^{ij} - u_{ab}^{ji} + u_{ba}^{ji}\right) t_{ij}^{ab} \\
&= \frac{1}{4} u_{ab}^{ij} t_{ij}^{ab},
\end{aligned} \tag{B5}$$

where we have used the antisymmetry of the two-body matrix elements, thus completing what we wanted to show.

Appendix C: Amplitude equations

In this section we have provided a few sample computations of how one would evaluate the amplitude equations using wicks theorem. Starting with the simplest term including only the normal-ordered Hamiltonian,

$$\begin{aligned}
&\langle\Phi_{ij}^{ab}|(F_N + V_N)|\Phi_0\rangle \\
&\quad = \sum_{pq} f_{pq} \langle\Phi_0|\{a_i^\dagger a_j^\dagger a_b a_a\}\{a_p^\dagger a_q\}|\Phi_0\rangle \\
&\qquad + \frac{1}{4}\sum_{pqrs} \langle pq||rs\rangle \langle\Phi_0|\{a_i^\dagger a_j^\dagger a_b a_a\}\{a_p^\dagger a_q^\dagger a_s a_r\}|\Phi_0\rangle.
\end{aligned} \tag{C1}$$

The one-electron component does not have any full contractions, while the two-electron component produces one contributing integral,

$$\begin{aligned}
&\langle\Phi_{ij}^{ab}|(V_N)|\Phi_0\rangle \\
&= \frac{1}{4}\sum_{pqrs} \langle pq||rs\rangle \langle\Phi_0|\{a_i^\dagger a_j^\dagger a_b a_a\}\{a_p^\dagger a_q^\dagger a_s a_r\}|\Phi_0\rangle \\
&= \frac{1}{4}\sum_{pqrs} \langle pq||rs\rangle \\
&\quad \times \Big(\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q^\dagger a_s a_r\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q^\dagger a_s a_r\} \\
&\quad + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q^\dagger a_s a_r\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q^\dagger a_s a_r\}\Big) \\
&= \frac{1}{4}\sum_{pqrs} \langle pq||rs\rangle (\delta_{ap}\delta_{bq}\delta_{js}\delta_{ir} - \delta_{aq}\delta_{bp}\delta_{js}\delta_{ir} \\
&\quad - \delta_{ap}\delta_{bq}\delta_{jr}\delta_{is} + \delta_{aq}\delta_{bp}\delta_{jr}\delta_{is}) \\
&= \langle ab||ij\rangle = u_{ij}^{ab}.
\end{aligned} \tag{C2}$$

Next we would like to evaluate $\langle\Phi_{ij}^{ab}|(F_N + V_N)T_2|\Phi_0\rangle$. Starting with the term involving the Fock operator F_N,

$$\begin{aligned}
&\langle\Phi_{ij}^{ab}|F_N T_2|\Phi_0\rangle \\
&= \frac{1}{4}\sum_{pq}\sum_{klcd} f_{pq} t_{kl}^{cd} \langle\Phi_0|\{a_i^\dagger a_j^\dagger a_b a_a\}(\{a_p^\dagger a_q\}\{a_c^\dagger a_d^\dagger a_l a_k\})_c|\Phi_0\rangle \\
&= \frac{1}{4}\sum_{pq}\sum_{klcd} f_{pq} t_{kl}^{cd} \\
&\times\Big(\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} \\
&+\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} \\
&+\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} \\
&+\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} \\
&+\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} \\
&+\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} \\
&+\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} \\
&+\{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\} + \{a_i^\dagger a_j^\dagger a_b a_a a_p^\dagger a_q a_c^\dagger a_d^\dagger a_l a_k\}\Big)
\end{aligned} \tag{C3}$$

$$
\begin{aligned}
&= \sum_{pq} \sum_{klcd} f_{pq} t_{kl}^{cd} \\
&\times (-\delta_{ac}\delta_{bd}\delta_{ik}\delta_{jq}\delta_{lp} + \delta_{ac}\delta_{bd}\delta_{il}\delta_{jq}\delta_{kp} \\
&\quad + \delta_{ac}\delta_{bd}\delta_{iq}\delta_{jk}\delta_{lp} - \delta_{ac}\delta_{bd}\delta_{iq}\delta_{jl}\delta_{kp} \\
&\quad + \delta_{ac}\delta_{bp}\delta_{dq}\delta_{ik}\delta_{jl} - \delta_{ac}\delta_{bp}\delta_{dq}\delta_{il}\delta_{jk} \\
&\quad + \delta_{ad}\delta_{bc}\delta_{ik}\delta_{jq}\delta_{lp} - \delta_{ad}\delta_{bc}\delta_{il}\delta_{jq}\delta_{kp} \\
&\quad - \delta_{ad}\delta_{bc}\delta_{iq}\delta_{jk}\delta_{lp} + \delta_{ad}\delta_{bc}\delta_{iq}\delta_{jl}\delta_{kp} \\
&\quad - \delta_{ad}\delta_{bp}\delta_{cq}\delta_{ik}\delta_{jl} + \delta_{ad}\delta_{bp}\delta_{cq}\delta_{il}\delta_{jk} \\
&\quad - \delta_{ap}\delta_{bc}\delta_{dq}\delta_{ik}\delta_{jl} + \delta_{ap}\delta_{bc}\delta_{dq}\delta_{il}\delta_{jk} \\
&\quad + \delta_{ap}\delta_{bd}\delta_{cq}\delta_{ik}\delta_{jl} - \delta_{ap}\delta_{bd}\delta_{cq}\delta_{il}\delta_{jk}) \\
&= (f_{bc}t_{ij}^{ac} - f_{ac}t_{ij}^{bc}) - (f_{jk}t_{ik}^{ab} - f_{ik}t_{jk}^{ab}) \\
&= f_{bc}t_{ij}^{ac}P(ab) - f_{kj}t_{ik}^{ab}P(ij)
\end{aligned} \tag{C4}
$$

Where in the last steps we have consigned the sums by Einstein summation notation.

Appendix D: Finding Amplitude Equation Intermediates

Starting from the CCD amplitude equation we factor out terms that will have the same indices if we contract them,

$$
\begin{aligned}
0 =& u_{ij}^{ab} + f_c^b t_{ij}^{ac} P(ab) - f_j^k t_{ik}^{ab} P(ij) \\
&+ \frac{1}{4} t_{ij}^{cd} t_{mn}^{ab} u_{cd}^{mn} + \frac{1}{2} t_{ij}^{cd} u_{cd}^{ab} \\
&+ \frac{1}{2} t_{jm}^{cd} t_{in}^{ab} u_{cd}^{mn} P(ij) - \frac{1}{2} t_{nm}^{ac} t_{ij}^{bd} u_{cd}^{nm} P(ab) \\
&+ t_{im}^{ac} t_{jn}^{bd} u_{cd}^{mn} P(ij) + t_{im}^{ac} u_{jc}^{bm} P(ab) P(ij) \\
&+ \frac{1}{2} t_{im}^{ab} u_{jn}^{mn},
\end{aligned} \tag{D1}
$$

$$
\begin{aligned}
0 =& u_{ij}^{ab} + f_c^b t_{ij}^{ac} P(ab) - f_j^k t_{ik}^{ab} P(ij) \\
&+ t_{ij}^{cd} \left(\frac{1}{4} t_{mn}^{ab} u_{cd}^{mn} + \frac{1}{2} u_{cd}^{ab} \right) \\
&+ t_{in}^{ab} \left(\frac{1}{2} t_{jm}^{cd} u_{cd}^{mn} \right) P(ij) \\
&- t_{ij}^{bd} \left(\frac{1}{2} t_{nm}^{ac} u_{cd}^{nm} \right) P(ab) \\
&+ \frac{1}{2} t_{im}^{ac} t_{jn}^{bd} u_{cd}^{mn} P(ij) - \frac{1}{2} t_{im}^{bc} t_{jn}^{ad} u_{cd}^{mn} P(ij) \\
&+ t_{im}^{ac} u_{jc}^{bm} P(ab) P(ij) \\
&+ \frac{1}{2} t_{im}^{ab} u_{jn}^{mn}
\end{aligned} \tag{D2}
$$

$$
\begin{aligned}
0 =& u_{ij}^{ab} + f_c^b t_{ij}^{ac} P(ab) - f_j^k t_{ik}^{ab} P(ij) \\
&+ t_{ij}^{cd} \left(\frac{1}{4} t_{mn}^{ab} u_{cd}^{mn} + \frac{1}{2} u_{cd}^{ab} \right) \\
&+ t_{in}^{ab} \left(\frac{1}{2} t_{jm}^{cd} u_{cd}^{mn} \right) P(ij) \\
&- t_{ij}^{bd} \left(\frac{1}{2} t_{nm}^{ac} u_{cd}^{nm} \right) P(ab) \\
&+ \frac{1}{2} t_{im}^{ac} t_{jn}^{bd} u_{cd}^{mn} P(ij) \\
&+ t_{im}^{ac} u_{jc}^{bm} P(ab) P(ij) \\
&+ \frac{1}{2} t_{im}^{ab} u_{jn}^{mn}
\end{aligned} \tag{D3}
$$

$$
\begin{aligned}
0 =& u_{ij}^{ab} + f_c^b t_{ij}^{ac} P(ab) - f_j^k t_{ik}^{ab} P(ij) \\
&+ t_{ij}^{cd} \left(\frac{1}{4} t_{mn}^{ab} u_{cd}^{mn} + \frac{1}{2} u_{cd}^{ab} \right) \\
&+ t_{in}^{ab} \left(\frac{1}{2} t_{jm}^{cd} u_{cd}^{mn} \right) P(ij) \\
&- t_{ij}^{bd} \left(\frac{1}{2} t_{nm}^{ac} u_{cd}^{nm} \right) P(ab) \\
&+ t_{im}^{ac} \left(\frac{1}{2} t_{jn}^{bd} u_{cd}^{mn} + u_{jc}^{bm} \right) P(ab) P(ij) \\
&+ \frac{1}{2} t_{im}^{ab} u_{jn}^{mn}.
\end{aligned} \tag{D4}
$$

Now we can introduce the intermediate χ-terms,

$$\chi_{cd}^{ab} = \frac{1}{4} t_{mn}^{ab} u_{cd}^{mn} + \frac{1}{2} u_{cd}^{ab} \tag{D5}$$

$$\chi_j^n = \frac{1}{2} t_{jm}^{cd} u_{cd}^{mn} \tag{D6}$$

$$\chi_d^a = \frac{1}{2} t_{nm}^{ac} u_{cd}^{nm} \tag{D7}$$

$$\chi_{jc}^{bm} = \frac{1}{2} t_{jn}^{bd} u_{cd}^{mn} + u_{jc}^{bm}, \tag{D8}$$

this gives us,

$$
\begin{aligned}
0 =& u_{ij}^{ab} + \tilde{f}_c^b t_{ij}^{ac} P(ab) - \tilde{f}_j^k t_{ik}^{ab} P(ij) \\
&+ t_{ij}^{cd} \chi_{cd}^{ab} + t_{in}^{ab} \chi_j^n P(ij) - t_{ij}^{bd} \chi_d^a P(ab) \\
&+ t_{im}^{ac} \chi_{jc}^{bm} P(ab) P(ij) + \frac{1}{2} t_{im}^{ab} u_{jn}^{mn}.
\end{aligned} \tag{D9}
$$

[1] M. Taut, Journal of Physics A: Mathematical and General **27**, 1045 (1994).
[2] S. M. Reimann and M. Manninen, Reviews of modern physics **74**, 1283 (2002).
[3] E. Anisimovas and A. Matulis, Journal of Physics: Condensed Matter **10**, 601 (1998).
[4] H. B. Schlegel and J. McDouall, in *Computational Advances in Organic Chemistry: Molecular Structure and Reactivity* (Springer, 1991) pp. 167–185.
[5] G. E. Scuseria, T. J. Lee, and H. F. Schaefer III, Chemical physics letters **130**, 236 (1986).
[6] M. P. Lohne, G. Hagen, M. Hjorth-Jensen, S. Kvaal, and F. Pederiva, Physical Review B **84**, 115302 (2011).
[7] M. P. Lohne, *Coupled-cluster studies of quantum dots*, Master's thesis (2010).
[8] T. D. Crawford and H. F. Schaefer, Reviews in Computational Chemistry, Volume 14 , 33 (2007).