Work in progress

by

Alfred Alocias Mariadason

THESIS

for the degree of

MASTER OF SCIENCE

Faculty of Mathematics and Natural Sciences
University of Oslo

May 2018

TODO: write creative commons licence

Abstract

Work in progress

Acknowledgements

Work in progress

Contents

Symbols List

Work in progress

Symbol	Meaning
ψ	Spin-orbital
ϕ	Spacial part of spin-orbital
χ	Basis functions used in basis-expansion
Ψ	Trial wavefunction
Φ	Slater determinant
g	Hermite-Gaussian
G	Gaussian-Type-Orbital (GTO)
Ω	Overlap Distribution
ξ	Recurrence relation for Coulomb integral

Table 1: List of symbols used with explaination.

Source Code

Work in progress

Chapter 1

Introduction

This chapter will give a brief introduction to the structure of the thesis and the goals laid forth beforehand.

1.1 Structure and Goals

The main goal(at least at some point into the semester) was to make a code-base for large-scale quantum many-body calculations from scratch. Of course there exists many such code-bases and competing with those in terms of scalability and computational efficiency is beyond the scope of this thesis, the choice for a from-scratch approach was made in order to gain some insight into the methods and a better understanding for the programming aspects which would follow. With this in mind, we did however get much inspiration from previous theses and implementations from other sources as well.
The main goals for the thesis was

- Make a general C++ code for the *Hartree-Fock method* and the *Variational Monte Carlo method* which could take any basis into play with minimal effort.

- Use the C++ code on different *quantum mechanical many-body systems* and use this as validation of the code and benchmark the code.

- Build an optimal one-body basis with the Hartree-Fock method and make improvements on this basis with the variational Monte-Carlo method with a *Slater-Jastrow* wavefunction.

With this in mind a general open-source code for the restricted Hartree-Fock method was developed and is open for use in https://github.com/Oo1Insane1oO/HartreeFock. The variational Monte-Carlo method is also open-source and for use in » URL THIS «. Both repositories each have a directory with tests implemented, see section 6.6 for more information on these tests.
Extending the code to other systems is made easier with python scripts given in the mentioned repositories. A better description of these are given in the repositories themselves and involve auto-generation of abstract wavefunction classes which only need to be filled in with analytical expressions and are otherwise already integrated with the existing code.

Chapter 2

Chapter 3

Many-Body Quantum Theory

This chapter takes forth the theory regarding the basics of identical particles and *many-body quantum mechanics*. The reader is referred to [14] for an introductory text on quantum mechanics(for single particles) and also the so-called *Dirac-notation* used throughout the entire chapter. We will address » LIST METHODS « regarding computational quantum mechanics and further deepen into Hartree-Fock methods and Variational Monte Carlo method. Optimization of calculation is also given while structure of program is given in chapter 6.

3.1 The Hamiltonian and the Born-Oppenheimer Approximation

The task at hand is to solve the many-body system described by *Schrödinger's* equation

$$H \left| \Psi_i \right\rangle = E_i \left| \Psi_i \right\rangle \tag{3.1}$$

for some state $\left| \Psi_i \right\rangle$ with energy E_i. Usually the desired state is the ground-state energy E_0 of the system meaning we are primarily interested in the *ground-state* $\left| \Psi_0 \right\rangle$.
With the goal determined we can define the system to consist of N identical particles[1] with positions $\{ \boldsymbol{r}_i \}_{i=0}^{N-1}$ and A nuclei with positions $\{ \boldsymbol{R}_k \}_{k=0}^{A-1}$. The Hamiltonian H is then

$$H = -\frac{1}{2} \sum_i \nabla_i^2 + \sum_{i<j} f\left(\boldsymbol{r}_j, \boldsymbol{r}_j \right) - \frac{1}{2} \sum_k \frac{\nabla_k^2}{M_k} + \sum_{k<l} g\left(\boldsymbol{R}_k, \boldsymbol{R}_l \right) + V\left(\boldsymbol{R}, \boldsymbol{r} \right) \tag{3.2}$$

The first and second terms represent the kinetic- and inter-particle interaction terms[2] for the N identical particles while the latter three represent kinetic- and interaction terms for the nuclei(with the last one being the nuclei-particle interaction). The constant M_k is the mass of nucleon k and Z_k is the corresponding atomic number.

   We assume the nuclei to be much heavier than the identical particles, meaning they move much slower, at which the system can be viewed as electrons moving around the vicinity of stationary nuclei. This means the kinetic term for the nuclei vanish and the nuclei-nuclei interaction becomes a constant[3]. The approximation we end up with is the so-called *Born-Oppenheimer approximation* and the Hamiltonian is now

$$H = H_0 + H_I \tag{3.3}$$

where we have split the Hamiltonian in a *one-body* part and a *two-body* or *interaction* parts defined as

$$H_0 \equiv -\frac{1}{2} \sum_i \nabla_i^2 + V\left(\boldsymbol{R}, \boldsymbol{r} \right) \tag{3.4}$$

and

$$H_I \equiv \sum_{i<j} f\left(\boldsymbol{r}_i, \boldsymbol{r}_j \right) \tag{3.5}$$

---

[1] These are in both atomic physics and in the quantum dot case always fermions or bosons.
[2] This is usually the well-known Coulomb interaction.
[3] Adding a constant to an operator does not alter the eigenvector, only the eigenvalues by the constant factor[25].

3.2   Slater Determinant and Permanent

Throughout section 3.1 we only referred to the wavefunction Ψ as a state, a function closely connected to the probabilistic nature of the quantum particles. However, we have not given it a form. One possible solution is the *Hartree product* Ψ_H defined as

$$\Psi_\mathrm{H} = \prod_i \psi_i(\boldsymbol{r}_i) \tag{3.6}$$

with $\{\psi\}_{i=0}^N$ being the orbitals which solve the single-particle Schrödinger equation for H_0. The Hartree-product is unfortunately a poor choice since it does not solve the H_I part meaning it is not a physically valid solution. This comes from the fact that the Hartree-product does not take into account the fact that the particles in question are *identical particles*. Since the particles are identical, switching the labels on the particles shouldn't change the expectation value of some observable. If we run this remark through we end up with the conclusion that the state $|\Psi\rangle$ must be either symmetric or antisymmetric with the symmetric part being the *bosonic state* and antisymmetric being the *fermionic state*. The connection between antisymmetric states and fermions is called the *Pauli exclusion principle*.

   The problem with the Hartree-product is, with the above sentiment, that it is not symmetric nor antisymmetric. However we can transform it with an operator

$$\mathcal{B} \equiv \frac{1}{N!} \sum_P \sigma_b P \tag{3.7}$$

where σ_b is defined as

$$\sigma_b \equiv \begin{cases} 1 & b \text{ represents bosonic system} \\ (-1)^p & b \text{ represents fermionic system} \end{cases} \tag{3.8}$$

P is a permutation operator that switches the labels on particles [4] and p is the parity of permutations. The operator $\mathcal{B}$ has the following properties

- Applying $\mathcal{B}$ to itself doesn't change the operator meaning $\mathcal{B}^2 = \mathcal{B}$.

- The Hamiltonian H and $\mathcal{B}$ *commute*, that is $[\mathcal{B}, H] = [H, \mathcal{B}]$.

- $\mathcal{B}$ is *unitary*, which means $\mathcal{B}^\dagger \mathcal{B} = \mathcal{I}$.

The solution Ψ_T to the Schrödinger equation can now be written as

$$\Psi_T(\boldsymbol{r}) = \sqrt{N!}\mathcal{B}\Psi_\mathrm{H}(\boldsymbol{r}) \tag{3.9}$$

The antisymmetric case of $\mathcal{B}$ results in a *Slater determinant*

$$\Psi_T^\mathrm{AS} = \frac{1}{\sqrt{N!}} \sum_P (-1)^p P \prod_i \psi_i \tag{3.10}$$

while the symmetric case gives the so-called *permanent*[5].

$$\Psi_T^\mathrm{S} = \sqrt{\frac{\prod\limits_{i=1}^N n_i!}{N!}} \sum_P P \prod_i \psi_i \tag{3.11}$$

Notice that the coordinated $\boldsymbol{r}_{i=0}^N$ is a bit of a sloppy notation as it also implicitly includes the spin orbitals. The extra factor in the symmetric case is to preserve the normalization due to number of particles, the n_i factor is determined by the the quantum-number for state i.

---

[4] $P_{ij}\Psi(\boldsymbol{r}_1,\ldots,\boldsymbol{r}_i,\ldots,\boldsymbol{r}_j,;\boldsymbol{r}_N) = \Psi(\boldsymbol{r}_1,\ldots,\boldsymbol{r}_j,\ldots,\boldsymbol{r}_i,;\boldsymbol{r}_N)$ [38].
[5] The permanent is basically just a determinant with all the negative signs replaced by positive ones.

3.3   Variational Principle

One important remark is that the Slater determinant and the permanent do not solve the interaction part, but only serves as a so-called *ansatz* or guess on the true ground-state wavefunction. This is quite useful due to the *variational principle.*

The Variational principle states that for any normalized function Ψ in Hilbert Space » REF HILBERT « with a Hermitian operator H the minimum eigenvalue E_0 for H has an upper-bound given by the expectation value of H in the function Ψ. That is

$$E_0 \leq \langle H \rangle = \langle \Psi | H | \Psi \rangle = \int \Psi^* H \Psi \mathrm{d}r \tag{3.12}$$

See [14] for proof and more.

The mentioned ansatz is thereby guaranteed to give energies larger than or equal the true ground state energy meaning a minimization method is sufficient in order to get closer to this minimum.

3.4   Energy Functional

We can find a more convenient expression for this energy by using equations 3.9 and 3.12. This gives us

$$E[\Psi] = N! \langle \Psi_{\mathrm{H}} | H\mathcal{B} | \Psi_{\mathrm{H}} \rangle \tag{3.13}$$

where the hermitian and unitary property of $\mathcal{B}$ as well as the fact that $\mathcal{B}$ and H commute have been used. This energy functional(functional in the sense that it is dependant on the wave function). Applying the $\mathcal{B}$ operator to the Hartree-product, pulling the sum out of the integrals and relabeling with the following definitions

$$\langle p| h | q \rangle \equiv \langle \psi_p(\boldsymbol{r}) | h(\boldsymbol{r}) | \psi_q(\boldsymbol{r}) \rangle = \int \psi_p^*(x) h(\boldsymbol{r}) \psi_q(\boldsymbol{r}) \mathrm{d}r$$

$$\langle pq| f | rs \rangle \equiv \langle \psi_p(\boldsymbol{r}_1)\psi_q(\boldsymbol{r}_2) | f(\boldsymbol{r}_1, \boldsymbol{r}_2) | \psi_r(\boldsymbol{r}_1)\psi_s(\boldsymbol{r}_2) \rangle = \int \psi_p(\boldsymbol{r}_1)\psi_q(\boldsymbol{r}_2) f(\boldsymbol{r}_1, \boldsymbol{r}_2)\psi_r(\boldsymbol{r}_1)\psi_s(\boldsymbol{r}_2) \mathrm{d}r$$

$$\tag{3.14}$$

yields in

$$E[\Psi] = \langle p| H_0 | p \rangle + \frac{1}{2} \sum_{p,q} \left[\langle pq| f_{pq} | pq \rangle \pm \langle pq| f_{pq} | qp \rangle \right] \tag{3.15}$$

The first part is written with the assumption that the single-particle wave functions $\{\psi\}$ are orthogonal and the 1/2 factor in front of the so-called *direct* and *exchange* terms[6] is due to the fact that we count the permutations twice in the sum when applying the $\mathcal{B}$ operator. The sign in the interaction term are chosen as positive for bosonic systems and negative for fermionic systems.

The expression given in equation 3.15 is the functional form we will use to derive the Hartree-fock equations in the following section.

3.5   Hartree-Fock Theory

Hartree-Fock method is a many-body method for approximating the wavefunction of a stationary many-body quantum state and thereby also obtain an estimate for the energy in this state. In this section we will derive the Hartree-Fock equations from scratch, following closely the literature by J.M Thjissen[38].

3.5.1   Assumptions

Hartree-Fock method makes the following assumptions of the system

- *The Born-Oppenheimer approximation*, see » REF BO «.

- All relativistic effects are negligible.

- The wavefunction can be described by a single *Slater determinant.*

- The *Mean Field Approximation* holds.

---

[6]The direct term is just due to inherent behaviour of the charge of the particles (known as the Coulomb repulsion). The exchange term is a direct consequence of the probabilistic nature of the identical particles.

With these inherent approximations the last one is the most important to take into account as it can cause large deviations from test solutions (analytic solutions, experimental data etc.) since the electron correlations is in reality, for many cases, not negligible. There exists many methods that try to fix this problem » LIST METHODS «. The *Variational Monte Carlo* (or VMC) is the method for deeper explorations in this Thesis, see section 3.7 for more details.

3.5.2 The $\mathcal{J}$ and $\mathcal{K}$ Operators

Before we begin with the Hartree-Fock equations it is desirable to rewrite the energy function obtained in section 3.4 (form given in equation 3.15) with two operators $\mathcal{J}$ and $\mathcal{K}$ defined as

$$\mathcal{J} \equiv \sum_k \langle \psi_k^* | f_{12} | \psi_k \rangle = \int \psi_k^*(\boldsymbol{r}) f_{12} \psi_k(\boldsymbol{r}) \mathrm{d}r$$
$$\mathcal{K} \equiv \sum_k \langle \psi_k^* | f_{12} | \psi \rangle = \int \psi_k^*(\boldsymbol{r}) f_{12} \psi(\boldsymbol{r}) \mathrm{d}r$$
(3.16)

The $\mathcal{J}$ operator just gives the simple interaction-term while the $\mathcal{K}$ operator gives the exchange term with the arbitrary (notice no index) $\psi(\boldsymbol{r})$. The energy functional is thus rewritten to

$$E\left[\Psi\right] = \sum_i \left\langle \psi_i \left| h + \frac{1}{2}\left(\mathcal{J} \pm \mathcal{K}\right) \right| \psi_i \right\rangle$$
(3.17)

where the one-body Hamiltonian is split into a sum of single particle functions as $H_0 = \sum_i h(\boldsymbol{r}_i)$.

3.5.3 Hartree-Fock Equations

As a reminder. The wavefunctions $\{\psi\}$ in equation 3.17 are spin-orbitals with both a spacial part and a spin part. In order to obtain the Hartree-Fock equations we try to minimize the energy functional in order to obtain the ground-state energy for a many-body system. This is done by a variational method.

The first observation to notice is the fact that variations in the spin-orbitals $\{\psi\}$ need to respect the spin-orthogonality relation

$$\langle \psi_i | \psi_j \rangle = \delta_{ij}$$
(3.18)

with δ_{ij} being the well-known Kronecker-delta. This property is essentially a constraint to the minimization problem and the method to be used is the *Lagrange multiplier method*[25], with the following *Lagrangian*

$$\mathcal{L} = \delta E\left[\Psi\right] - \sum_{ij} \Lambda_{ij} \left[\langle \psi_i | \psi_j \rangle - \delta_{ij}\right]$$
(3.19)

We know then that the minimum is reached when a displacement on the spin-orbitals $\psi_i \to \psi_i + \delta\psi_i$ results in an energy variation of zero meaning $\delta E\left[\Psi\right] = 0$ in the minimum. Which giving the variational problem

$$\sum_i \langle \delta\psi_i | h + \mathcal{J} \pm \mathcal{K} | \psi_i \rangle - \sum_{ij} \Lambda_{ij} \langle \delta\psi_i | \psi_j \rangle + \text{c.c} = 0$$
(3.20)

where c.c is a notation for the complex conjugate of the inner-products on its left-hand side.

The shift in the spin orbitals $\{\delta\psi\}$ is arbitrary and the constraints are symmetric[7] meaning we can with the *Fock-operator*

$$\mathcal{F} \equiv h + \mathcal{J} \pm \mathcal{K}$$
(3.21)

define the following eigenvalue problem

$$\mathcal{F}\psi_i = \sum_j \Lambda_{ij} \psi_j$$
(3.22)

Choosing the Lagrange parameter Λ_{ij} such that $\{\psi\}_{k=1}^N$ forms an orthonormal set for $\mathcal{F}$ with eigenvalues $\{\varepsilon\}_{k=1}^N$. This reduces the eigenvalue equation to

$$\mathcal{F}|\psi\rangle = \varepsilon|\psi\rangle$$
(3.23)

with $\boldsymbol{\varepsilon} = (\varepsilon_0, \dots, \varepsilon_N)$ being the set of eigenvalues of $\mathcal{F}$ meaning we have $N+1$ equations to be solved.

---

[7] $\langle \psi_i | \psi_j \rangle = \langle \psi_j | \psi_i \rangle^* \Rightarrow \Lambda_{ij} = \Lambda_{ji}^*$

If we only take the N lowest eigenfunctions into the Slater the corresponding eigenenergy is referred to as the *Hartree-Fock energy* and is the estimated ground-state energy which the Hartree-Fock method gives. We can rewrite the energy functional with the eigenenergies to

$$E\left[\Psi\right] = \sum_i \left\langle \psi_i \left| \varepsilon_i - \frac{1}{2}\left(\mathcal{J} \pm \mathcal{K}\right) \right| \psi_i \right\rangle \tag{3.24}$$

In the derivation of the Hartree-Fock equations we only worked with spin-orbital functions $\{\psi\}$. However it is much more convenient to rewrite these in terms of spatial orbitals $\{\phi\}$ and integrate the spin-dependant part out. There are two ways of doing this and the two different approaches give the so-called *restricted Hartree-Fock* and *unrestricted Hartree-Fock* methods.

3.6　Restricted Hartree-Fock and Roothan-Equations

The restricted spin-orbitals are paired as[8]

$$\{\psi_{2l-1}, \psi_{2l}\} = \{\phi_l(\boldsymbol{r})\alpha(s), \phi_l(\boldsymbol{r})\beta(s)\} \tag{3.25}$$

with $\alpha(s)$ and $\beta(s)$ being different spin-states (up and down). This pairing of spin-states with same and same spacial-orbitals means we can pull the spin degrees of freedom out from the $\mathcal{J}$ and $\mathcal{K}$ operators, reduce the sum to only run over half the states and multiply the entire sum by 2. The result is that the restricted energy-functional reads

$$E\left[\Psi\right] = \sum_{i=1}^{N} \varepsilon_i - \sum_{i=1}^{\frac{N}{2}} \langle i \, | \, 2\mathcal{J} \pm \mathcal{K} \, | \, i \rangle \tag{3.26}$$

Notice that the $\mathcal{K}$ operators sum only runs up to half the number of states.

As the title suggests we are going to end up with a set of equations referred to as the *Roothan-equations*. We start by first expanding the spacial part $\{\phi\}$ of the spin orbitals $\{\psi\}$ in some known orthonormal basis $\{\chi\}_{i=1}^{L}$

$$\phi_i(\boldsymbol{r}) = \sum_{p=1}^{L} C_{pi}\chi_p(\boldsymbol{r}) \tag{3.27}$$

and introduce the *Fock-matrix* F (associated with the Fock-operator) with elements

$$F_{pq} = h_{pq} + \sum_{pq} \rho_{pq}\left(2D_{prqs} \pm D_{prsq}\right) \tag{3.28}$$

We have here introduced a one-body matrix defined as

$$h_{pq} \equiv \langle p \, | \, h \, | \, q \rangle \tag{3.29}$$

a *density matrix* defined as[9]

$$\rho_{pq} \equiv \sum_{i=1}^{\frac{N}{2}} C_{pi}C_{qi}^* \tag{3.31}$$

and an interaction-matrix D with elements

$$D_{pqrs} \equiv \langle pq \, | \, f_{12} \, | \, rs \rangle \tag{3.32}$$

for convenience. The implicit relabeling of $\chi_p(\boldsymbol{r}) \to p$ is also present in the above expression for the Fock-matrix. The Hartree-Fock equations (equation 3.23) are then for the restricted case written as

$$F\boldsymbol{C}_i = \varepsilon S\boldsymbol{C}_i \tag{3.33}$$

with S being the overlap matrix with elements

$$S_{pq} \equiv \langle p \, | \, q \rangle \tag{3.34}$$

---

[8] This is specialised for a two-spin system. For a system with more spin-states one needs to either choose different spacial-orbitals or add more such orbitals which effectively changes the energy-levels.

[9] This is just the matrix formed by

$$\sum_i |\phi_i\rangle \langle \psi_i| \tag{3.30}$$

which is in quantum mechanics defined as the so-called *density matrix*.

3.7   Quantum Monte Carlo

Quantum Monte Carlo, or QMC is a method for solving Schrödinger's equation by a statistical approach using so-called *Markov Chain* simulations (also called random walk). The nature of the wave function at hand is fundamentally a statistical model defined on a large configuration space with small areas of densities. The Monte Carlo method is perfect for solving such a system because of the non-homogeneous distribution of calculation across the space. An standard approach with equal distribution of calculation would then be a waste of computation time.

   We will in this chapter address the Metropolis algorithm which is used to create a Markov chain and derive the equations used in the variational method.

   The chapter will use *Dirac Notation* [14] and all equations stated assume atomic units ($\hbar = m_e = e = 4\pi\varepsilon_0$) » REF HERE ATOMIC UNITS «.

3.7.1   The Variational Principle and Expectation Value of Energy

Given a Hamiltonian $\hat{H}$ and a trial wave function $\Psi_T(\boldsymbol{R}; \boldsymbol{\alpha})$, the variational principle [14, 30] states that the expectation value of $\hat{H}$

$$E[\psi_T] = \left\langle \hat{H} \right\rangle = \frac{\langle \psi_T | \hat{H} | \psi_T \rangle}{\langle \Psi_T | \Psi_T \rangle} \tag{3.35}$$

is an upper bound to the ground state energy

$$E_0 \leq \left\langle \hat{H} \right\rangle \tag{3.36}$$

Now we can define our PDF as(see section 3.7.3 for a more detailed reasoning)

$$P(\boldsymbol{R}) \equiv \frac{|\psi_T|^2}{\langle \Psi_T | \Psi_T \rangle} \tag{3.37}$$

and with a new quantity

$$E_L(\boldsymbol{R}; \boldsymbol{\alpha}) \equiv \frac{1}{\Psi_T(\boldsymbol{R}; \boldsymbol{\alpha})} \hat{H} \Psi_T(\boldsymbol{R}; \boldsymbol{\alpha}) \tag{3.38}$$

the so-called local energy, we can rewrite equation 3.35 as

$$E[\Psi_T(\boldsymbol{R}; \boldsymbol{\alpha})] = \langle E_L \rangle \tag{3.39}$$

The idea now is to find the lowest possible energy by varying a set of parameters $\boldsymbol{\alpha}$. The expectation value itself is found with the Metropolis algorithm, see section 3.7.2.

   An important property of the local energy is when we differentiate it with respect to one of the variational parameters $\{\alpha\}$ within the context of an expectation value. The result in this case would be zero. This is easily seen by direct calculation

$$\begin{aligned}
\left\langle \frac{\partial E_L}{\partial \alpha} \right\rangle &= \int \frac{|\psi|^2 \frac{\partial}{\partial \alpha} \left[\frac{1}{\psi} H \psi \right]}{\int |\psi|^2 \mathrm{d}r} \mathrm{d}r \\
&= \int \frac{|\psi|^2 \frac{\psi^* \frac{\partial}{\partial \alpha}(H\psi) - (H\psi^*) \frac{\partial \psi}{\partial \alpha}}{|\psi|^2}}{\int |\psi|^2 \mathrm{d}r} \mathrm{d}r \\
&= \int \frac{\psi^* H \frac{\partial \psi}{\partial \alpha} - \psi^* H \frac{\partial \psi}{\partial \alpha}}{\int |\psi|^2 \mathrm{d}r} \mathrm{d}r \\
&= 0
\end{aligned} \tag{3.40}$$

We have used the fact that H is not dependant on any variational parameter and used the hermitian properties[14] of H to justify the movement of H within the integral.

   This neat result presented in equation 3.40 will show its usefulness in the minimization when derivatives of the expectation value and variance comes into play since finding the derivative of the local energy would be much more of a hassle.[10]

---

[10] The differentiation of the wavefunction is enough to reduce the quality of life on its own!

3.7.2   Metropolis-Hastings Algorithm

The Metropolis algorithm bases itself on moves (also called transitions) as given in a Markov process. »
REF THIS HERE «. This process is given by

$$w_i(t + \varepsilon) = \sum_j w_{i \to j} w_j(t) \tag{3.41}$$

where $w(j \to i)$ is just a transition from state j to state i. In order for the transition chain to reach a desired
convergence while reversibility is kept, the well known condition for detailed balance must be fulfilled[35].
If detailed balance is true, then the following relation is true

$$w_i T_{i \to j} A_{i \to j} = w_j T_{j \to i} A_{j \to i} \Rightarrow \frac{w_i}{w_j} = \frac{T_{j \to i} A_{j \to i}}{T_{i \to j} A_{i \to j}} \tag{3.42}$$

We have here introduced two scenarios, the transition from configuration i to configuration j and the reverse
process j to i. Solving the acceptance A for the two cases where the ratio in 3.42 is either 1(in which case
the proposed state j is accepted and transitions is made) and when the ratio is less then 1. The Metropolis
algorithm would in this case not automatically reject the latter case, but rather reject it with a proposed
uniform probability. Introducing now a probability distribution function(PDF) P the acceptance A can be
expressed as

$$A_{i \to j} = \min\left(\frac{P_{i \to j}}{P_{j \to i}} \frac{T_{i \to j}}{T_{j \to i}}, 1\right) \tag{3.43}$$

The so-called selection probability T is defined specifically for each problem. For our case the PDF in
question is the absolute square of the wave function and the selection T is a Green's function derived in
section 3.7.4. The algorithm itself would then be

   (i) Pick initial state i at random.

   (ii) Pick proposed state at random in accordance to $T_{j \to i}$.

   (iii) Accept state according to $A_{j \to i}$.

   (iv) Jump to step (ii) until a specified number of states have been generated.

   (v) Save the state i and jump to step (ii).

3.7.3   Diffusion Theory and the PDF

The motivation for the use of diffusion theory is described very well in the Masters thesis of Jørgen Høgber-
get[22]. The essential results were a Green's function propagator arising from the *short time approximation*
and the observation of interest here which has to do with the statistics describing the expectation value, it
states that *any* distribution may be applied in calculation, however if we take a close look at the local en-
ergy(equation 3.38) we see that for all distributions the local energy is not defined at the zeros of $\Psi_T(\boldsymbol{R}; \boldsymbol{\alpha})$.
This means that an arbitrary PDF does not guarantee generation of points which makes $\psi_T = 0$. This can
be overcome by introducing the square of the wave function to be defined as the distribution function as
given in equation 3.37. This basically means that when using Quantum Monte-Carlo methods, the incorpo-
ration of the fact that when an energy is more undefined(meaning $\psi_T \to 0$ the less probable that point is
actually means the generation of states in which ψ_T is removed. The next chapter will derive the method
that includes this observation into play, the so-called *importance sampling*.

3.7.4   Importance Sampling

Using the selection probability mentioned in section 3.7.2, the Metropolis-Hastings algorithm is called an
Importance sampling because it essentially makes the sampling more concentrated around areas where the
PDF has large values.

   Because of the intrinsic statistical property of the wave function Quantum Mechanics can be modelled
as a diffusion process, or more specifically, an *Isotropic Diffusion Process* which is essentially just a random
walk model. Such a process is described by the Langevin equation with the corresponding Fokker-Planck
equation describing the motion of the walkers(particles). See [21, 33, 38] for details. The full derivations
presented here follows closely the descriptions in [21, 38]. Let us start off with the Langevin equation

$$\frac{\partial r}{\partial t} = DF(r(t)) + \eta \tag{3.44}$$

and apply Euler's method and obtain the new positions

$$r^{\text{new}} = r^{\text{old}} + DF^{\text{old}}\Delta t + \xi \tag{3.45}$$

with the r's being the new and old positions in the Markov chain respectively and $F^{\text{old}} = F(r^{\text{old}})$. The quantity D is a diffusion therm equal to $1/2$ due to the kinetic energy(remind of natural units) and ξ is a Gaussian distributed random number with 0 mean and $\sqrt{\Delta t}$ variance.

As mentioned a particle is described by the Fokker-Planck equation

$$\frac{\partial P}{\partial t} = \sum_i D \frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial x_i} - \boldsymbol{F_i}\right)P \tag{3.46}$$

With P being the PDF(in current case the selection probability) and F being the drift therm. In order to achieve convergence, that is a stationary probability density, we need the left hand side to be zero in equation 3.46 giving the following equation

$$\frac{\partial^2 P}{\partial x_i{}^2} = P\frac{\partial \boldsymbol{F_i}}{\partial x_i} + \boldsymbol{F_i}\frac{\partial P}{\partial x_i} \tag{3.47}$$

with the drift-therm being on the form $\boldsymbol{F} = g(x)\partial P/\partial x$ we finally have that

$$\boldsymbol{F} = \frac{2}{\psi_T}\nabla \psi_T \tag{3.48}$$

This is the so-called *Quantum Force* which pushes the walkers towards regions where the wave function is large.

The missing part now is to model the selection probability in equation 3.43. Inserting the quantum force into the Focker-Planck equation(equation 3.46 the following diffusion equation appears

$$\frac{\partial P}{\partial t} = -D\nabla^2 P \tag{3.49}$$

Applying the *Fourier Transform* to spatial coordinate r in equation 3.49, the equation is transformed to

$$\frac{\partial P(\boldsymbol{s},t)}{\partial t} = -Ds^2 P(\boldsymbol{s},t) \tag{3.50}$$

which has solution

$$P(\boldsymbol{s},\Delta t) = P(\boldsymbol{s},0)e^{Ds^2\Delta t} \tag{3.51}$$

Now we need to find the constant $P(\boldsymbol{s},0)$, and as is apparent with $t = 0$, we will make use of an initial condition. The initial positions are spread out from origin, that is $D\Delta t\boldsymbol{F}_j$. We can express this with a *Dirac-delta function*[2, 14] giving

$$P(s,0) = \delta\left(\boldsymbol{r}_i - D\Delta t\boldsymbol{F}_j\right) \tag{3.52}$$

Inserting this into equation 3.51 and making the inverse Fourier transform yields the following Green's function as solution

$$P\left(a,b,\Delta t\right) = \frac{1}{\sqrt{4\pi D\Delta t}}\exp\left(-\frac{(\boldsymbol{r}_a - \boldsymbol{r}_b - D\Delta t\boldsymbol{F}_b)^2}{4D\Delta t}\right) \tag{3.53}$$

This expression is precisely the selection probability T, Notice also that the indices a and b label a state transition $a \to b$ and not particle indices. The full transition probability needs to be summed over for all particles since we only solved the Focker-Planck equation for 1 particle(since the other solutions are found in the exact same manner). For clarity the full selection probability ratio is

$$\frac{T(b,a,\Delta t)}{T(a,b,\Delta t)} = \sum_i \exp\left(-\frac{\left(\boldsymbol{r}_i^{(b)} - \boldsymbol{r}_i^{(a)} - D\Delta t\boldsymbol{F}_i^{(a)}\right)^2}{4D\Delta t} + \frac{\left(\boldsymbol{r}_i^{(a)} - \boldsymbol{r}_i^{(b)} - D\Delta t\boldsymbol{F}_i^{(b)}\right)^2}{4D\Delta t}\right) \tag{3.54}$$

3.7.5 The Trial Wavefunction: One-Body

The trial wave function is generally an arbitrary choice specific for the problem at hand, however it is in most cases favorable to expand the wave function in the eigenbasis (eigenstates) of the Hamiltonian since they form a complete set. This can be expressed as

$$\Psi_T(\boldsymbol{R};\boldsymbol{\alpha}) = \sum_k C_i\psi_k(\boldsymbol{R};\boldsymbol{\alpha}) \tag{3.55}$$

where the ψ_i's are the eigenstates of the Hamiltonian. The coefficients can be found by any method preferable and is the usual procedure is to use a set of basis functions and then minimize to find the coefficients $\{C\}_{k=1}^L$. We use the Hartree-Fock method to minimize in this thesis. The trial wavefunction is also generally expressed as a *Slater determinant* for the fermionic case and a general product for bosonic systems[8, 10, 14, 30] We will explain the fermionic case shortly since it is the main focus here and since the bosonic wavefunction is simple to express. The Slater is expressed as

$$\Phi_T(\boldsymbol{R};\boldsymbol{\alpha}) = \det(\boldsymbol{\Phi}(\boldsymbol{R};\boldsymbol{\alpha}))\xi(s) \tag{3.56}$$

where the *Slater matrix* $\boldsymbol{\Phi}$ has elements

$$\Phi_{ij} = \phi_{n_j}(\boldsymbol{r}_i;\boldsymbol{\alpha}) \tag{3.57}$$

such that each row is evaluated for particle i and each column is for a quantum number n_j dependant on the basis used. The $\xi(s)$ is the spin-dependant part. Notice also that we switched the labeling from Ψ_T to Φ_T. This is to make a distinction between *one-body* and *correlation* terms. The latter will be introduced later in section 3.7.7. In this case the *single-particle* functions $\phi_j(r)$ are expanded in some basis(i.e Hartree-Fock).

3.7.6   The Trial Wavefunction: Splitting the Slater Determinant

An important part of the trial-wavefunction presented here is that the one-body term is *independent* of spin, that is, the Hamiltonian is not explicitly dependant on the spin degrees of freedom. For the case of a Hamiltonian with an inherent spin part The following splitting of the Slater determinant is not valid! In that case the expectation value(presented in the variational principal in equation 3.12) would be a product of the expectation value over the spin-independent part of the Hamiltonian and the expectation value over the remaining spin-dependant parts[32]. The results presented here is however for systems of spin-independent systems, and in those cases the spin-part is essentially just another label which can be integrated out(similar to the procedure with the restricted Hartree-Fock method). For the splitting with a *spin-dependent* Hamiltonian see [29] and [20].

The procedure is simply to arrange the basis functions in such a way that we have $N/2$ single-particle functions in spin-up and the same basis functions for spin-down. This means that the Slater is

$$\frac{1}{N!}\begin{pmatrix} \phi_1(\boldsymbol{r}_1)\xi_\uparrow & \dots & \phi_{N/2}(\boldsymbol{r}_1)\xi_\uparrow & \phi_1(\boldsymbol{r}_1)\xi_\downarrow & \dots & \phi_{N/2}(\boldsymbol{r}_1)\xi_\downarrow \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_1(\boldsymbol{r}_N)\xi_\uparrow & \dots & \phi_{N/2}(\boldsymbol{r}_N)\xi_\uparrow & \phi_1(\boldsymbol{r}_N)\xi_\downarrow & \dots & \phi_{N/2}(\boldsymbol{r}_N)\xi_\downarrow \end{pmatrix} \tag{3.58}$$

This restructuring of the single-particle states implies that

$$\det(\boldsymbol{\Phi}) \propto \det(\boldsymbol{\Phi}_\uparrow)\det(\boldsymbol{\Phi}_\downarrow) \tag{3.59}$$

where we have defined

$$\begin{aligned} \boldsymbol{\Phi}_\uparrow &= \begin{pmatrix} \phi_1(\boldsymbol{r}_1) & \dots & \phi_{N/2}(\boldsymbol{r}_1) \\ \vdots & \ddots & \vdots \\ \phi_1(\boldsymbol{r}_{N/2}) & \dots & \phi_{N/2}(\boldsymbol{r}_{N/2}) \end{pmatrix} \\ \boldsymbol{\Phi}_\downarrow &= \begin{pmatrix} \phi_1(\boldsymbol{r}_{N/2+1}) & \dots & \phi_{N/2}(\boldsymbol{r}_{N/2+1}) \\ \vdots & \ddots & \vdots \\ \phi_1(\boldsymbol{r}_N) & \dots & \phi_N(\boldsymbol{r}_N) \end{pmatrix} \end{aligned} \tag{3.60}$$

Which in this sense means to put the first $N/2$ particles in spin-up configuration and the remaining in spin-down and use the same single-particle functions. On a technical note, this rewriting is an approximation. However it can be shown(again see [29]) that the expectation value is still the same.
The one-body term is now rewritten to

$$\det(\boldsymbol{\Phi}) = \det(\boldsymbol{\Phi}_\uparrow)\det(\boldsymbol{\Phi}_\downarrow) \tag{3.61}$$

3.7.7   The Trial Wavefunction: Jastrow Factor

As mentioned, we model the wavefunction as a product of the one-body Slater and a correlation part known as a *Jastrow factor*. The Jastrow can have many forms, but is build to exert some key features. It should[23].

- Be dependent on the inter-particle distances.

- Approach unity at large distances.

- Vanish when particles are close to one another.

In this thesis we explore two form, the popular *Padé Function* and a more resent one build with a specific neural network known as a *Boltzmann Machine*.

Cusp Condition

Padé Function

A popular form is the *Padé-Jastrow* function. It is defined as[16, 20]

$$J_{\text{Padé}} = \exp\left(\sum_{i<j} \frac{\sum_l a_{ij}^{(l)} r_{ij}^l}{1 + \sum_l \beta_l r_{ij}^l}\right) \tag{3.62}$$

meaning the trial wavefunction is

$$\Psi_T(\boldsymbol{R}; \boldsymbol{\alpha}) = \det(\boldsymbol{\Phi})J \tag{3.63}$$

where the β_l's are variational parameters. This function follows the key features mentioned for a correlation factor, but a keen observation is that we can choose the a parameters to follow a so-called *cusp condition* which cancels the divergence of the Slater for particle positions close to one another. Following [9, 27] the conditions are

$$a_{ij}^{\text{2D}} = \begin{cases} \dfrac{1}{3}, & \text{parallel spin} \\ 1, & \text{anti-parallel spin} \end{cases} \tag{3.64}$$

$$a_{ij}^{\text{3D}} = \begin{cases} \dfrac{1}{4}, & \text{parallel spin} \\ \dfrac{1}{2}, & \text{anti-parallel spin} \end{cases} \tag{3.65}$$

with $a_{kj}^{(l)} = 0$ for $l > 1$, meaning we can relabel $a_{ij}^{(l)} \to a_{ij}$ and $\beta_l \to \beta$ and reduce the Padé-Jastrow factor to

$$J_{\text{Padé}} = \exp\left(\sum_{i<j} \frac{a_{ij} r_{ij}}{1 + \beta r_{ij}}\right) \tag{3.66}$$

For expressions concerning the gradient and Laplacian, see Appendix B.

Simple Exponential

» REF THIS « With the desirable features for a correlation function in mind, one of the simplest forms for a correlation factor is a scaled exponential evaluated with the inter-particle distance, this is[11]

$$J = \exp\left(\sum_{i<j} a_{ij} r_{ij}\right) \tag{3.67}$$

The a would be defined the same as with the Padeé function. In itself this function is fairly useless as it doesn't model correlations very well for different systems or even the same system with different parameters. However we will present a new type of function in the next section which does not give raise to any cusp conditions, but has a more flexible form and may in connection with this simple exponential give us all the necessary properties desired in a correlation function.

---

[11] The a's can also be variational parameters.

The Trial Wavefunction: NQS Wavefunction

A more resent and completely different approach is to model the wavefunction with a *neural network* as presented by Carleo and Troyer[5]. The approach used here is based on the *Restricted Boltzmann Machine* as described by Hinton[19]. The form is

$$J_{\text{NQS}} = \exp\left(-\sum_{i=1}^{N} \frac{(\boldsymbol{r}_i - \boldsymbol{a}_i)^2}{2\sigma^2}\right) \prod_{j}^{M} \left(1 + \exp\left(b_j + \sum_{i=1}^{N}\sum_{d=1}^{D} \frac{x_i^{(d)} w_{i+d,j}}{\sigma^2}\right)\right) \tag{3.68}$$

The derivatives of this are given in Appendix B.

3.7.8　Connect the Jastrows

In the previous section we presented four functions that can be used to model correlations in quantum systems, however they all had some limitations as well as advantageous properties. The question then remains, can we create a function which exhibits all the nice properties mentioned and none of the limitations? The answer is a bit ambivalent. We don't really know for sure, but a good start is to multiply the functions having the cusp conditions in order (Padé and exponential) with the NQS wavefunction and use this product as the new jastrow. The motivations for this is simple, we want the cusp conditions introduced by the old popular functions, but also want a more flexible type of function which can take care of other(possible) correlations in the system. How well this actually performed is presented in » REF RESULTS WHEN YOU HAVE SOME (GO GET THEM!) «. We will write out the exact form for clarity.

$$J = \exp\left(\sum_{i<j} a_{ij} r_{ij}\right) \exp\left(-\sum_{i=1}^{N} \frac{(\boldsymbol{r}_i - \boldsymbol{a}_i)^2}{2\sigma^2}\right) \prod_{j}^{M} \left(1 + \exp\left(b_j + \sum_{i=1}^{N}\sum_{d=1}^{D} \frac{x_i^{(d)} w_{i+d,j}}{\sigma^2}\right)\right) \tag{3.69}$$

$$J = \exp\left(\sum_{i<j} \frac{a_{ij} r_{ij}}{1 + \beta r_{ij}}\right) \exp\left(-\sum_{i=1}^{N} \frac{(\boldsymbol{r}_i - \boldsymbol{a}_i)^2}{2\sigma^2}\right) \prod_{j}^{M} \left(1 + \exp\left(b_j + \sum_{i=1}^{N}\sum_{d=1}^{D} \frac{x_i^{(d)} w_{i+d,j}}{\sigma^2}\right)\right) \tag{3.70}$$

It might not be completely clear as to why this function still makes it so that the total wavefunction satisfies the cusp conditions. The reason lies in the derivation of the conditions » REF (NOT REALLY) derivate this maybe?) «

Chapter 4

Basis Functions

We will in this chapter mention some popular basis-sets used in atomic physics and deepen into a particular set of functions called *Gaussian Type orbitals* and use them with the well known *Hermite functions*.

4.1   Hermite Functions

Hermite functions are functions of the following form

$$\phi_n^a(\boldsymbol{r}) \equiv \prod_d N_d H_{n_d}(\sqrt{a}x_d)\exp\left(-\frac{a}{2}x_d^2\right) \tag{4.1}$$

with $\boldsymbol{r} = \sum_d \mathbf{e}_d x_d$ and the sum over d being the sum over the number of dimensions and the H_n is the Hermite polynomial of order n. The integer n_d is the order of the function[1] while the parameter a is a scaling factor and N_d is a normalization factor. These functions show up as eigenfunctions for the *quantum harmonic oscillator system*[14] with the scaling parameter a equal to the oscillator frequency (ω) of the system.

The Hermite functions are orthogonal and give a good ansatz for the VMC method, see section 3.7, with the scaling parameter transformed with an additional variational parameter. The problem with these are however that the matrix-elements introduced in the Hartree-Fock method (section 3.5) are not solvable directly with the Hermite functions as basis functions. However, we can write the Hermite functions in terms of *Hermite-Gaussians*. See section 4.2.1.

4.2   Gaussian Type Orbitals

Gaussian Type Orbitals or GTO's are functions of the following form[37]

$$G_n(\boldsymbol{\alpha};\boldsymbol{r},\boldsymbol{A}) \equiv \prod_d (x_d - A_d)^{n_d}\mathrm{e}^{-\alpha_d(x_d-A_d)^2} \tag{4.2}$$

We call α for the scaling parameter and i for the order of the GTO. The variable A is where the function is centered. These are in many literatures referred to as *primitive Gaussians* and they alone make a poor approximation to the true wave function.

In atomic physics these functions are used directly as a linear combination referred to as *contracted Gaussian functions*. These are written as

$$G_k(x, A) \equiv \sum_{a_k=0}^{P} C_{a_k} G_{a_k}(\alpha_{a_k}; x, A) \tag{4.3}$$

and are fitted[2] to *Slater-type orbitals*, which are functions with decaying properties(present in atomic systems), or found by some variational method before-hand.

These functions are unfortunately not orthogonal, but they behave nicely in integrals and actually give an analytic expression for the interaction-elements mentioned in section 3.5. For this reason we will go forth and use the Gaussian contracted functions and actually fit them to Hermite functions.

---

[1] In quantum mechanics the number n is referred to as the principal quantum number and is associated with the energy of a given orbital(energy-level) of the system.
[2] Meaning we find the parameters C_{a_k}. and α_{a_k}

4.2.1 Hermite-Gaussian Functions

The GTO's described can be explicitly expressed in terms of so-called *Hermite-Gaussian functions*[3] defined as

$$g_n(\boldsymbol{\alpha}; \boldsymbol{r}, \boldsymbol{A}) = \prod_d \left(\frac{\partial}{\partial A_d} \right)^{n_d} \mathrm{e}^{-\alpha_d(x_d - A_d)^2} \tag{4.4}$$

meaning

$$G_n(\boldsymbol{\alpha}; \boldsymbol{r}, \boldsymbol{A}) = \prod_d (2\alpha_d)^{-n_d} \left(\frac{\partial}{\partial A_d} \right)^{n_d} \mathrm{e}^{-\alpha_d(x_d - A_d)^2} \tag{4.5}$$

Some properties of the one-dimensional Hermite-Gaussians are as follows

$$\begin{aligned}
\frac{\partial g_t}{\partial A_x} &= g_{t+1} \\
g_{t+1} &= \left(\frac{\partial}{\partial A_x} \right)^t \frac{\partial g_0}{\partial A_x} = 2\alpha(x - A_x) \left(\frac{\partial}{\partial A_x} \right)^t g_0 \\
g_{t+1} &= 2\alpha((x - A_x)g_t - tg_{t-1}) \\
(x - A_x)g_t &= \frac{1}{2\alpha} g_{t+1} + t g_{t-1}
\end{aligned} \tag{4.6}$$

The mentioned rewriting of the Hermite functions in terms of the Hermite-Gaussians is

$$\phi_n^a(\boldsymbol{r}) = \prod_d N_d \sum_{l=0}^{n_d} C_{n_d l} g_l \left(\frac{\boldsymbol{\alpha}}{2}, \boldsymbol{r}, \boldsymbol{A} \right) \tag{4.7}$$

with $C_{n_d l}$ being the $l'th$ Hermite-coefficient for the Hermite polynomial of order n_d. This means that the matrix-elements in Hartree-Fock is just a linear combination over integrals over Hermite-Gaussians. The following section will tackle this in detail for the two-dimensional case. The three-dimensional case is given by an excellent text by Trygve Helgaker and Peter R. Taylor[18], see also [17].

4.3 Integral Elements

In the Hartree-Fock scheme described in chapter 3 we need to calculate the integrals which define the different matrix elements. The integrals to be found are of the following form

$$\begin{aligned}
\langle i \,|\, j \rangle &= \int\limits_{-\infty}^{\infty} g_i(\alpha_i; r, A) g_j(\alpha_j; r, B) \mathrm{d}\boldsymbol{r} \\
\langle i \,|\, x_d^k \,|\, j \rangle &= \int\limits_{-\infty}^{\infty} g_i(\alpha_i; r, A) r^k g_j(\alpha_j; r, B) \mathrm{d}\boldsymbol{r} \\
\langle i \,|\, \nabla^2 \,|\, j \rangle &= \int\limits_{-\infty}^{\infty} g_i(\alpha_i; r, A) \nabla^2 g_j(\alpha_j; r, B) \mathrm{d}\boldsymbol{r} \\
\left\langle ij \,\middle|\, \frac{1}{r} \,\middle|\, kl \right\rangle &= \int\limits_{-\infty}^{\infty} \int\limits_{-\infty}^{\infty} g_i(\alpha_i; r_1, A) g_j(\alpha_j; r_2, B) \frac{1}{r_{12}} g_k(\alpha_k; r_1, C) g_l(\alpha_l; r_2, D) \mathrm{d}\boldsymbol{r}_1 \mathrm{d}\boldsymbol{r}_2
\end{aligned} \tag{4.8}$$

where $\mathrm{d}\boldsymbol{r}$ means integration over all dimensions and with the g's being the usual *Hermite-Gaussians* defined as

$$g_n(\alpha; \boldsymbol{r}, \boldsymbol{A}) = \prod_d (x_d - A_d)^{n_d} \mathrm{e}^{-\alpha(x_d - A_d)^2} \tag{4.9}$$

We will in this chapter limit ourselves to work with isotropic Gaussians (meaning α_d is the same for all dimensions) as this will yield a simpler closed-form solution to the integrals. For a calculation of integral elements using non-isotropic Gaussian functions see this wonderful article [6], this article takes gives a detailed calculations of integrals over s-type non-isotropic gaussians. The extension to general Gauss-Hermites

---

[3] The reason for the name is that the polynomial factors generated by the differentiation are precisely the Hermite polynomials.

can be done in the exact same manner as presented here, however the proportionality factors involved in the recursive relation is different.

Before we throw ourselves out into the integrals, let us first express the Hermite-Gaussians in a more convenient way(again see [18] and [17]. The Gaussians can be expressed as

$$g_n(\alpha; \boldsymbol{r}, \boldsymbol{A}) = \prod_d \left(\frac{\partial}{\partial A_{x_d}} \right)^{n_d} e^{-\alpha(x_d - A_d)^2} = \prod_d \left(\frac{\partial}{\partial A_{x_d}} \right)^{n_d} g_0(\alpha; \boldsymbol{r}, \boldsymbol{A}) \tag{4.10}$$

Since the derivatives are with respect to the center variables we may pull them out of the integration meaning the integrals will only be over s-type Gaussians, greatly simplifying the calculations. With the mentioned simplification in mind, the problem is to find a closed-form expression for the integrals over s-type Gaussians.

We also introduce the *Gaussian product rule*[4] which basically states that the product of two Gaussian functions is just a third Gaussian centered between the center of the two. The expressions give

$$g_0(\alpha; \boldsymbol{r}, \boldsymbol{A}) g_0(\beta; \boldsymbol{r}, \boldsymbol{B}) = K_{AB} \exp\left(-(\alpha + \beta)\boldsymbol{r}_s^2\right) \tag{4.11}$$

where

$$\begin{aligned} K_{AB} &\equiv \exp\left(-\frac{\alpha\beta}{\alpha + \beta} R_{AB}^2 \right) \\ R_{AB} &= |\boldsymbol{A} - \boldsymbol{B}| \\ \boldsymbol{r_s} &= \boldsymbol{r} - \boldsymbol{P} \\ \boldsymbol{P} &= \frac{\alpha \boldsymbol{A} + \beta \boldsymbol{B}}{\alpha + \beta} \end{aligned} \tag{4.12}$$

the vector $\boldsymbol{r}_S$ is just somewhere between $\boldsymbol{A}$ and $\boldsymbol{B}$(We will see that r_S disappears when the integration is done). The Gaussian product rule greatly simplifies the integral over two Gaussian functions since we can just pull K_{AB} out of the integration since it is a constant.

4.3.1   Overlap Distribution

An *overlap distribution* is defined as the product between two Hermite-Gaussian functions, that is

$$\Omega_{ij} = \prod_d g_{i_d}(x_d, \alpha, A_d) g_{j_d}(x_d, \beta, B_d) = K_{A_d B_d} x_A^{i_d} x_B^{j_d} e^{-(\alpha+\beta)x_P^2} \tag{4.13}$$

with the Gaussian product rule, which is just another Gaussian function centered in P, but with the extra *monomial* factors in $\boldsymbol{r} - \boldsymbol{A}$ and $\boldsymbol{r} - \boldsymbol{B}$. These factors are troublesome when integrating. With the motivation that Hermite-Gaussians make life simpler, we expand the overlap distribution in a Hermite-Gaussian basis. Following T. Helgaker[18] and working in one dimension(since Hermite-Gaussians can be split in each respective dimension) we have[5]

$$\Omega_{ij}(\alpha, \beta, \boldsymbol{r}, \boldsymbol{A}, \boldsymbol{B}) = \sum_{t=0}^{i+j} E_t^{ij} g_t(\alpha, \beta, \boldsymbol{r}, \boldsymbol{P}) \tag{4.14}$$

Again, we stress that the indices in equation 4.14 and the calculations further are in 1 dimension. Explicit expressions for the coefficients E_t^{ij} are difficult to derive, however a set of recurrence relations are possible to

---

[4]Still in the isotropic case.
[5]The indices i and j are now in 1 dimension!

find using the properties of the Hermite-Gaussian functions. Consider firstly the incremented distribution

$$
\begin{aligned}
\Omega_{i+1,j} &= \sum_{t=0}^{i+1+j} E_t^{i+1,j} g_t \\
&= \left(x_P - \frac{\beta}{\alpha+\beta}(A_x - B_x) \right) \Omega_{ij} \\
&= \sum_{t=0}^{i+j} E_t^{ij} \left(x_P - \frac{\beta}{\alpha+\beta}(A_x - B_x) \right) g_t \\
&= \sum_{t=0}^{i+j} E_t^{ij} \left(\left(t g_{t-1} + \frac{1}{2(\alpha+\beta)} g_{t+1} \right) - \frac{\beta}{\alpha+\beta}(A_x - B_x) g_t \right) \\
&= \sum_{t=0}^{i+j} \left((t+1) E_{t+1}^{ij} + \frac{1}{2(\alpha+\beta)} E_{j-1}^{ij} - \frac{\beta}{\alpha+\beta}(A_x - B_x) \right) g_t
\end{aligned}
\tag{4.15}
$$

Using the properties listed in equation 4.6(mainly the recurrence) and the expansion equation 4.14. The incrementation of j follows the exact same derivation. The starting coefficient is thus

$$
E_0^{00} = K_{AB}
\tag{4.16}
$$

This is found by inserting in $i = j = 0$ into equation 4.15, realizing the exponental is the same for all i and j and using the orthogonality between the Hermite-Gaussians[6]. The recurrent coupled relations for the E's are

$$
\begin{aligned}
E_t^{i+1,j} &= \frac{1}{2(\alpha+\beta)} E_{t-1}^{ij} - \frac{\beta}{\alpha+\beta}(A_x - B_x) E_t^{ij} + (t+1) E_{t+1}^{ij} \\
E_t^{i,j+1} &= \frac{1}{2(\alpha+\beta)} E_{t-1}^{ij} - \frac{\alpha}{\alpha+\beta}(A_x - B_x) E_t^{ij} + (t+1) E_{t+1}^{ij}
\end{aligned}
\tag{4.17}
$$

The overlap distribution can with this be expanded in Hermite-Gaussian functions.

As mentioned, the whole point of using Hermite-Gaussian functions is because of the inherent definition with the derivative with respect to the centering(remember s-types). This means that for attaining the final expression we must in the end differentiate the expansion coefficients. We state here the coefficients differentiated with respect to the difference variable $Q_x = A_x - B_x$

$$
\begin{aligned}
E_0^{00;n+1} &= -\frac{2\alpha\beta}{\alpha+\beta} \left(Q_x E_0^{00;n} + n E_0^{00;n-1} \right) \\
E_t^{i+1,j;n} &= \frac{1}{2(\alpha+\beta)} E_{t-1}^{ij;n} - \frac{\beta}{\alpha+\beta} \left(Q_x E_t^{ij;n} + n E_t^{ij;n-1} \right) + (t+1) E_{t+1}^{ij;n} \\
E_t^{i,j+1;n} &= \frac{1}{2(\alpha+\beta)} E_{t-1}^{ij;n} - \frac{\alpha}{\alpha+\beta} \left(Q_x E_t^{ij;n} + n E_t^{ij;n-1} \right) + (t+1) E_{t+1}^{ij;n} \\
E_t^{ij;n} &\equiv \frac{\partial^n E_t^{ij}}{\partial Q_x^n}
\end{aligned}
\tag{4.18}
$$

Notice that these expressions are just the same relations as for the coefficients, but with an extra factor in the middle.

---

[6]Another way of expressing this statement is to say that each index t in the sum corresponds to an equation for E_t^{ij}.

4.3.2 Overlap Integral

With The simplification to s-types and the product rule, the integration may begin. Starting with the overlap $\langle i \,|\, j \rangle$ and using equation 4.14[7]

$$\langle i \,|\, j \rangle = \int\limits_{-\infty}^{\infty} \Omega_{ij}(\alpha_p, \beta_p, \boldsymbol{r}, \boldsymbol{A}, \boldsymbol{B}) \mathrm{d}\boldsymbol{r}$$

$$= \sum_{p}^{i+j} E_p^{ij} \int\limits_{-\infty}^{\infty} g_p(\alpha, \beta, \boldsymbol{r}, \boldsymbol{P}) \mathrm{d}\boldsymbol{r}$$

$$= \sum_{p}^{i+j} E_p^{ij} \int\limits_{-\infty}^{\infty} (\boldsymbol{r} - \boldsymbol{P})^p \mathrm{e}^{-(\alpha_p + \beta_p)(\boldsymbol{r} - \boldsymbol{P})^2} \mathrm{d}\boldsymbol{r}$$

$$= \sum_{p}^{i+j} E_p^{ij} \left(\frac{((-1)^p - 1)\, \Gamma\left(\frac{p+1}{2}\right)}{2(\alpha_p + \beta_p)^{\frac{p+1}{2}}} \right)^{d} \tag{4.19}$$

The power d comes from splitting the integral into the d dimensions. Also using the *multi-index notation*(section A.3)[8] and expanding

$$E_n^{ab} = \prod_{d} E_{n_d}^{a_d b_d} \tag{4.20}$$

Such that the coefficients are all just products over coefficients in each dimension. A substitution in each dimension(i.e $u = x - P_x$) is also used. Notice in addition that the scaling factors α and β are specific for each p because of the overlap expansion.

4.3.3 Potential Integral

The second integral with the x_d^k part shows up in the external potential part of the Hamiltonian and again with the Gaussian product rule the expression gives

$$\langle i \,|\, x_d^k \,|\, j \rangle = \int\limits_{-\infty}^{\infty} x_d^k \Omega_{ij}(\alpha_p, \beta_p, \boldsymbol{r}, \boldsymbol{A}, \boldsymbol{B}) \mathrm{d}\boldsymbol{r}$$

$$= \sum_{p}^{i+j} E_p^{ij} \int\limits_{-\infty}^{\infty} x_d^k (\boldsymbol{r} - \boldsymbol{P})^p \mathrm{e}^{-(\alpha_p + \beta_p)(\boldsymbol{r} - \boldsymbol{P}_p)^2} \mathrm{d}\boldsymbol{r}$$

$$= \sum_{p}^{i+j} E_p^{ij} \left(\frac{((-1)^p - 1)\, \Gamma\left(\frac{p+1}{2}\right)}{2(\alpha_p + \beta_p)^{\frac{p+1}{2}}} \right)^{D-1} \int\limits_{-\infty}^{\infty} (u + P_d)^k \exp\!\left(-(\alpha_p + \beta_p)u^2\right) \mathrm{d}u$$

$$= \sum_{p}^{i+j} E_p^{ij} \left(\frac{((-1)^p - 1)\, \Gamma\left(\frac{p+1}{2}\right)}{2(\alpha_p + \beta_p)^{\frac{p+1}{2}}} \right)^{D-1} \sum_{l=0}^{k} \binom{k}{l} P_d^{k-l} \int\limits_{-\infty}^{\infty} u^l \exp\!\left(-(\alpha_p + \beta_p)u^2\right) \mathrm{d}u$$

$$= \sum_{p}^{i+j} E_p^{ij} \left(\frac{((-1)^p - 1)\, \Gamma\left(\frac{p+1}{2}\right)}{2(\alpha_p + \beta_p)^{\frac{p+1}{2}}} \right)^{D-1} \sum_{l=0}^{k} \binom{k}{l} \frac{P_d^{k-l}}{2(\alpha_p + \beta_p)^{\frac{l}{2}}} \left((-1)^l + 1\right) \Gamma\left(\frac{l+1}{2}\right) \tag{4.21}$$

The integrals are split in each dimension and the dimensions not equal to d(in x_d^k) are pulled out and the approach in equation 4.19 is applied. The integral over dimension d is then substituted with $u = x_d + P_d$. In line four $(u + P_d)^k$ is rewritten with the *binomial expansion*[9].

---

[7] Also using the following integral $\int\limits_{-\infty}^{\infty} \mathrm{e}^{-\lambda x^2} = \sqrt{\frac{\pi}{\lambda}}, \quad \lambda > 0$. See [24].

[8] The power d also means that with the multi-index notation the entire expression in the paranthesis are to be calculated for each dimension in p and then multiplied together.

[9] The integral $\int\limits_{-\infty}^{\infty} x^n \mathrm{e}^{-ax^2} \mathrm{d}x = \frac{1}{2} a^{-\frac{n}{2}} \Gamma\left(\frac{k+1}{2}\right), \quad n > -1, n$ even, see [24].

4.3.4 Laplacian Integral

The third integral with the Laplacian operator arises in the kinetic part of the Hamiltonian. This integral can be expressed in terms of equation 4.19, the overlap integral, however the Laplacian applied to a Hermite-Gaussian has to be calculated first

$$\nabla^2 g_i(\alpha; \boldsymbol{r}, \boldsymbol{A}) = \sum_d \frac{\partial^2}{\partial x_d{}^2} \left(\prod_{d'} (x - A_{d'})^{i_{d'}}_{d'} \exp\left(-\alpha (x_{d'} - A_{d'})^2\right) \right)$$

$$= \sum_d \prod_{d' \neq d} g_{i,d'} \frac{\partial^2}{\partial x_d{}^2} \left((x_d - A_d)^{i_d} \exp\left(-\alpha (x_d - A_d)^2\right) \right)$$

$$= \sum_d \prod_{d' \neq d} g_{i,d'} g_{i,d} \left(4\alpha^2 (x_d - A_d)^{i_d+2} - 2\alpha (2i_d + 1)(x_d - A_d)^{i_d} + i_d (i_d - 1)(x_d - A_d)^{i_d-2} \right)$$

$$= g_i \sum_d \left(4\alpha^2 (x_d - A_d)^{i_d+2} - 2\alpha (2i_d + 1)(x_d - A_d)^{i_d} + i_d (i_d - 1)(x_d - A_d)^{i_d-2} \right) \qquad (4.22)$$

Now for the integral we have

$$\langle i \,|\, \nabla^2 \,|\, j \rangle = \int_{-\infty}^{\infty} g_i(\alpha; \boldsymbol{r}, \boldsymbol{A}) \nabla^2 g_j(\beta; \boldsymbol{r}, \boldsymbol{B}) \mathrm{d}\boldsymbol{r}$$

$$= \sum_d \prod_{d' \neq d} \langle i_{d'} \,|\, \sigma_{d'}(S_d(\beta; x - B_d)) \,|\, j_{d'} \rangle \qquad (4.23)$$

with

$$S_d(\alpha; x_d - A_d) \equiv \left(4\alpha^2 (x_d - A_d)^{i_d+2} - 2\alpha (2i_d + 1)(x_d - A_d)^{i_d} + i_d (i_d - 1)(x_d - A_d)^{i_d-2} \right)$$

$$\sigma_d(S_d) \equiv \begin{cases} 1, & d' \neq d \\ S_d, & d' = d \end{cases} \qquad (4.24)$$

meaning the Laplacian integral can be expressed in terms of the overlap integrals $\langle i \,|\, j + 2 \rangle$, $\langle i \,|\, j \rangle$ and $\langle i \,|\, i - 2 \rangle$[10].

4.3.5 Coulomb Potential Integral

Lastly, the troublesome[11] Coulomb integral needs to be calculated. Due to the $1/r$ term we cannot split the integral in each respective dimension as previously. Before we approach the full Coulomb integral, lets calculate a simpler integral over a so-called *Coulomb Potential distribution*

$$\int_{-\infty}^{\infty} \mathrm{e}^{-\alpha(\boldsymbol{r}-\boldsymbol{A})^2} \frac{1}{|\boldsymbol{r} - \boldsymbol{B}|} \mathrm{d}\boldsymbol{r} \qquad (4.25)$$

The calculation of this integral will be beneficial for the calculation of the Coulomb integral as we can reuse most of the tricks used. With equation 4.11, the Gaussian product rule, in mind. We rewrite the inverse term with

$$\int_{-\infty}^{\infty} \mathrm{e}^{r_B^2 t^2} \mathrm{d}t = \frac{\sqrt{\pi}}{r_B} \Rightarrow \frac{1}{r_B} = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \mathrm{e}^{r_B^2 t^2} \mathrm{d}t \qquad (4.26)$$

The Coulomb potential integral is thus, with equation 4.11(again the product rule)

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mathrm{e}^{-\alpha(\boldsymbol{r}-\boldsymbol{A})^2} \frac{1}{|\boldsymbol{r} - \boldsymbol{B}|} \mathrm{d}\boldsymbol{r} = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \mathrm{e}^{-\alpha(\boldsymbol{r}-\boldsymbol{A})^2} \mathrm{e}^{t^2(\boldsymbol{r}-\boldsymbol{B})^2} \mathrm{d}\boldsymbol{r}\mathrm{d}t$$

$$= \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mathrm{e}^{-\frac{\alpha t^2}{\alpha+t^2}(\boldsymbol{A}-\boldsymbol{B})^2} \mathrm{e}^{-(\alpha+t^2)r_S^2} \mathrm{d}\boldsymbol{r}\mathrm{d}t$$

$$= \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \left(\frac{\pi}{\alpha + t^2} \right)^{\frac{d}{2}} \mathrm{e}^{-\frac{\alpha t^2}{\alpha+t^2}(\boldsymbol{A}-\boldsymbol{B})^2} \mathrm{d}t \qquad (4.27)$$

---

[10]Since $xg_i = g_{i+1}$ and $\frac{g_i}{x} = g_{i-1}$.

[11]Damn inverse term prevents dimensional decomposition.

The integral over t has to be addressed separately for two- and three dimensions. For the three-dimensional case the reader is referred to [18]. Here we will derive a closed-form form expression for the two-dimensional case. First let us use the following substitution

$$
\begin{aligned}
u &= \frac{t}{\sqrt{\alpha + t^2}} \\
t &= u\sqrt{\frac{\alpha}{1 - u^2}} \\
\frac{\mathrm{d}u}{\mathrm{d}t} &= \frac{\alpha}{(\alpha + t^2)^{\frac{3}{2}}} \\
\lim_{t \to -\infty} u(t) &= -1 \\
\lim_{t \to \infty} u(t) &= 1
\end{aligned}
\tag{4.28}
$$

The integrand(ignoring the exponential part) is then

$$
\begin{aligned}
\frac{\mathrm{d}t}{\alpha + t^2} &= \frac{1}{\alpha + t^2} \frac{(\alpha + t^2)^{\frac{3}{2}}}{\alpha} \mathrm{d}u \\
&= \frac{\sqrt{\alpha + t^2}}{\alpha} \mathrm{d}u \\
&= \frac{t}{\alpha u} \mathrm{d}u \\
&= \frac{1}{\alpha u} u\sqrt{\frac{\alpha}{1 - u^2}} \mathrm{d}u \\
&= \frac{1}{\sqrt{\alpha}} \sqrt{\frac{1}{1 - u^2}} \mathrm{d}u
\end{aligned}
\tag{4.29}
$$

giving

$$
I_{\mathrm{2D}} = \sqrt{\frac{\pi}{\alpha}} \int_{-1}^{1} \frac{1}{\sqrt{1 - u^2}} \mathrm{e}^{-\alpha u^2 |\boldsymbol{A} - \boldsymbol{B}|^2} \mathrm{d}u
\tag{4.30}
$$

and for the three-dimensional case we have a simpler form(easily seen with the same substitution)

$$
I_{\mathrm{3D}} = \pi \int_{-1}^{1} \mathrm{e}^{-\alpha u^2 |\boldsymbol{A} - \boldsymbol{B}|^2} \mathrm{d}u
\tag{4.31}
$$

These integrals must be solved numerically using *Chebyshev-Gauss Quadrature*[40]. One can also rewrite the 2D-integral in terms of the *Modified Bessel function of first kind* by using $u^2 = 1/2(1 - \cos(\theta)$[41]. [18]. A take on this resulted nowhere as the closed form expanded itself in ever-increasing order of polynomial factors with the first and second order of the modified Bessel function of first kind. The 3D-integral can be rewritten with an *incomplete Gamma function* From equation 4.10, the integrals have to be differentiated in order to get the final closed form expressions, see section 4.3.7.

4.3.6 Coulomb Interaction Integral

In the previous section an expression for integral over a Coulomb potential was derived. Before we embark into handling the full Coulomb interaction integral, another exercise with simpler interaction integral is worthwhile. The integral in question is

$$
I' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mathrm{e}^{-\alpha(\boldsymbol{r}' - \boldsymbol{A})^2} \mathrm{e}^{-\beta(\boldsymbol{r} - \boldsymbol{B})^2} \frac{1}{|\boldsymbol{r}' - \boldsymbol{r}|} \mathrm{d}\boldsymbol{r}\mathrm{d}\boldsymbol{r}'
\tag{4.32}
$$

This is an interaction between two distributions. Firstly, notice that we can rewrite the distribution centered in $\boldsymbol{A}$ and the Coulomb interaction with the previously calculated Coulomb potential integral given in

equation 4.27. Using I as a general label for equations 4.30 and 4.31 we have

$$I' = \int\limits_{-\infty}^{\infty} \int\limits_{-\infty}^{\infty} \mathrm{e}^{-\alpha(\boldsymbol{r'}-\boldsymbol{A})^2} \mathrm{e}^{-\beta(\boldsymbol{r}-\boldsymbol{B})^2} \frac{1}{|\boldsymbol{r'}-\boldsymbol{r}|} \mathrm{d}\boldsymbol{r}\mathrm{d}\boldsymbol{r'}$$

$$= \int\limits_{-\infty}^{\infty} I_D(\alpha; |\boldsymbol{r}-\boldsymbol{A}|)\mathrm{e}^{-\beta(\boldsymbol{r}-\boldsymbol{B})^2} \mathrm{d}\boldsymbol{r} \tag{4.33}$$

Inserting in the definition for u(the substitution in equation 4.28) and using the extremely useful Gaussian product rule for the product between the distribution centered in $\boldsymbol{B}$ and the exponential factor in I(which is labelled the same for both the two- and three dimensional case) is

$$\mathrm{e}^{-\alpha u^2(\boldsymbol{r}-\boldsymbol{A})^2} \mathrm{e}^{-\beta(\boldsymbol{r}-\boldsymbol{B})^2} = \mathrm{e}^{-(\alpha u^2+\beta)\boldsymbol{r}\boldsymbol{s}^2} \mathrm{e}^{-\frac{\alpha u^2 \beta}{\alpha u^2+\beta}(\boldsymbol{A}-\boldsymbol{B})^2} \tag{4.34}$$

Inserting this into equation 4.33 with

$$v \equiv \begin{cases} \sqrt{\frac{\pi}{\alpha}}\sqrt{\frac{1}{1-u^2}}, & \text{2D} \\ \pi, & \text{3D} \end{cases} \tag{4.35}$$

we have

$$I' = \int\limits_{-\infty}^{\infty} \int\limits_{-1}^{1} v\mathrm{e}^{-(\alpha u^2+\beta)\boldsymbol{r}\boldsymbol{s}^2} \mathrm{e}^{-\frac{\alpha u^2 \beta}{\alpha u^2+\beta}(\boldsymbol{A}-\boldsymbol{B})^2} \mathrm{d}\boldsymbol{r}\mathrm{d}u$$

$$= \int\limits_{-1}^{1} v\left(\frac{\pi}{\alpha u^2+\beta}\right)^{\frac{d}{2}} \mathrm{e}^{-\frac{\alpha u^2 \beta}{\alpha u^2+\beta}(\boldsymbol{A}-\boldsymbol{B})^2} \mathrm{d}u \tag{4.36}$$

Specializing to the two-dimensional case and substituting

$$v = u\sqrt{\frac{\alpha+\beta}{\alpha u^2+\beta}}$$

$$\frac{\mathrm{d}v}{\mathrm{d}u} = \frac{\beta\sqrt{\alpha+\beta}}{(\alpha u^2+\beta)^{3/2}}$$

$$u = v\sqrt{\frac{\beta}{\alpha+\beta-\alpha v^2}}$$

$$v(-1) = -1$$

$$v(1) = 1 \tag{4.37}$$

The integrand is

$$\frac{1}{\sqrt{1-u^2}}\frac{1}{\alpha u^2+\beta}\mathrm{d}u = \frac{1}{\sqrt{1-u^2}}\frac{1}{\alpha u^2+\beta}\frac{(\alpha u^2+\beta)^{3/2}}{\beta\sqrt{\alpha+\beta}}\mathrm{d}v$$

$$= \frac{1}{\sqrt{1-u^2}}\frac{u}{\beta v}\mathrm{d}v$$

$$= \sqrt{\frac{\alpha+\beta-\alpha v^2}{(\alpha+\beta)(1-v^2)}}\frac{1}{\beta v}v\sqrt{\frac{\beta}{\alpha+\beta-\alpha v^2}}\mathrm{d}v$$

$$= \frac{1}{\sqrt{\beta(\alpha+\beta)}}\frac{1}{\sqrt{1-v^2}}\mathrm{d}v \tag{4.38}$$

we have

$$I'_{2\mathrm{D}} = \frac{\pi^{\frac{3}{2}}}{\sqrt{\alpha\beta(\alpha+\beta)}}\int\limits_{-1}^{1} \frac{1}{\sqrt{1-v^2}}\mathrm{e}^{-\frac{\alpha\beta}{(\alpha+\beta)}v^2(\boldsymbol{A}-\boldsymbol{B})^2}\mathrm{d}v \tag{4.39}$$

This expression will be of great use when calculating the final full interaction integral over the Coulomb distribution. The next section will derive the mentioned recurrence relation before the full Coulomb integral is calculated

4.3.7   Recurrence Relation

Following directly from [18], we proceed with finding a similar recurrence relation for the derivatives. We define a function containing the integral which needs to be solved numerically

$$\zeta_n(x) \equiv \int\limits_{-1}^{1} \frac{u^{2n}}{\sqrt{1-u^2}} e^{-u^2 x} du$$

$$\frac{\partial \zeta_n}{\partial x} = -\zeta_{n+1}$$

(4.40)

The Coulomb potential integral is then, in terms of $\zeta_n(x)$

$$I_{2D} = \sqrt{\frac{\pi}{\alpha}} \zeta_0 \left(\alpha R_{AB}^2 \right)$$

(4.41)

and the first derivative with respect to A_x is

$$\frac{\partial I_{2D}}{\partial A_x} = \sqrt{\frac{\pi}{\alpha}} \frac{\partial}{\partial A_x} \zeta_0 \left(\alpha R_{AB}^2 \right)$$
$$= -2\sqrt{\alpha\pi} X_{AB} \zeta_1 \left(\alpha R_{AB}^2 \right)$$

(4.42)

With this we define an auxiliary function

$$\xi_{tu}^n = \left(\frac{\partial}{\partial A_x} \right)^t \left(\frac{\partial}{\partial A_y} \right)^u \xi_{00}^n$$

$$\xi_{00}^n = (-2)^n \alpha^{n-\frac{1}{2}} \zeta_n \left(\alpha R_{AB}^2 \right)$$

(4.43)

and take a look at the incrementation of t

$$\xi_{t+1,u}^n = \left(\frac{\partial}{\partial A_x} \right)^t \left(\frac{\partial}{\partial A_y} \right)^u \frac{\partial \xi_{00}^n}{\partial A_x}$$

$$= \left(\frac{\partial}{\partial A_x} \right)^t X_{AB} \xi_{0u}^{n+1}$$

(4.44)

Using the commutator between $\partial_{A_x}^t$ [12]

$$\frac{\partial^t}{\partial A_x^t} X_{AB} = \left[\frac{\partial^t}{\partial A_x^t}, X_{AB} \right] + X_{AB} \frac{\partial^t}{\partial A_x^t}$$

$$= t \frac{\partial^{t-1}}{\partial A_x^{t-1}} + X_{AB} \frac{\partial^t}{\partial A_x^t}$$

(4.45)

the final form of equation 4.44 is [13]

$$\xi_{t+1,u}^n = t\xi_{t-1,u}^{n+1} + X_{AB} \xi_{t,u}^{n+1}$$
$$\xi_{t,u+1}^n = u\xi_{t,u-1}^{n+1} + Y_{AB} \xi_{t,u}^{n+1}$$

(4.46)

With this all Hermite integrals of order $t+u \leq N$ can be calculated from ζ of order $n \leq N$, the only difference being X_{AB} and Y_{AB}. The Coulomb interaction integral (equation 4.39) follows this exact recurrence, but with a different proportionality factor $\alpha\beta/(\alpha + \beta)$. We will write it out for the sake of clarity

$$\frac{\partial I_{2D}'}{\partial A_x} = -\frac{2\alpha\beta}{\alpha + \beta} X_{AB} \zeta_1 \left(\frac{\alpha\beta}{\alpha + \beta} R_{AB}^2 \right)$$

$$\xi_{00}^n = \left(\frac{-2\alpha\beta}{\alpha + \beta} \right)^n \zeta_n \left(\frac{\alpha\beta}{\alpha + \beta} R_{AB}^2 \right)$$

(4.47)

Notice that the only difference between the obtained recurrence relations and the ones obtained by Helgaker[17] is in equation 4.43 and equation 4.47. Other than this the incrementation of ζ_n gives the exact same X_{AB} (and similar for the other directions) as with the incomplete gamma function.

---

[12] $\partial_x^t = \frac{\partial^t}{\partial x^t}$
[13] The incrementation of u is derived in the exact same manner as with t.

4.3.8   Coulomb Distribution Integral

With the derived expressions for the Coulomb potential integral the full two-body distribution can be treated. The expression with the simplification in equation 4.10 gives

$$
\left\langle ij \left| \frac{1}{r_{12}} \right| kl \right\rangle = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \Omega_{ik}(\boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{r_1}, \boldsymbol{A}, \boldsymbol{C}) \frac{1}{|\boldsymbol{r_1} - \boldsymbol{r_2}|} \Omega_{jl}(\boldsymbol{\beta}, \boldsymbol{\delta}, \boldsymbol{r_2}, \boldsymbol{B}, \boldsymbol{D}) \mathrm{d}\boldsymbol{r_1} \mathrm{d}\boldsymbol{r_2}
$$

$$
= \sum_{pq}^{i+k,j+l} E_p^{ik} E_q^{jl} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{g_p(\alpha + \gamma, \boldsymbol{r_1}, \boldsymbol{P}) g_q(\beta + \delta, \boldsymbol{r_2}, \boldsymbol{Q})}{|\boldsymbol{r_1} - \boldsymbol{r_2}|} \mathrm{d}\boldsymbol{r_1} \mathrm{d}\boldsymbol{r_2}
$$

$$
= \frac{a}{\sqrt{(\alpha + \gamma + \beta + \delta)}} \sum_{pq}^{i+k,j+l} E_p^{ik} E_q^{jl} (-1)^q \xi_{p+q} \left(\frac{(\alpha + \gamma)(\beta + \delta)}{\alpha + \gamma + \beta + \delta}, \boldsymbol{R_{S_1 S_2}} \right) \tag{4.48}
$$

Where we have used the multi-index[14] notation for p, q, i, k, j, and l equation 4.20 and used equation 4.39 to arrive at the final step. An additional simplification due to the fact that ζ_n is only dependant on the relative distance of the centers is also used, for the x-coordinate it is stated as

$$
\left(\frac{\partial}{\partial P_x} \right)^{p_x} \left(\frac{\partial}{\partial Q_x} \right)^{q_x} = (-1)^{p_x + q_x} \left(\frac{\partial}{\partial P_x} \right)^{p_x + q_x} \tag{4.49}
$$

and the same for the other directions. The factor a is

$$
a \equiv \begin{cases} \dfrac{\pi^{\frac{3}{2}}}{\sqrt{(\alpha + \gamma)(\beta + \delta)}}, & \text{2D} \\[3ex] \dfrac{\pi^{\frac{5}{2}}}{(\alpha + \gamma)(\beta + \delta)}, & \text{3D} \end{cases} \tag{4.50}
$$

4.4   Double-Well Functions

This section will explain the building of a basis for the *double-well potential*. We will expand them in a linear combination of harmonicoscillator functions and find the coefficients of this expansion by solving the arising eigenvalue-problem. Let us first express the potential

$$
U^{\mathrm{DW}}(r) = V^{\mathrm{HO}}(r) + V_n^{\mathrm{DW}}(r) \tag{4.51}
$$

A double well potential is essentially just a perturbation of the usual Harmonicoscillator potential(which is a single-well). The V_n^{DW} part is assumed to a polynomial of degree n. This means that the integral over such a potential can be calculated using equation 4.21.

4.4.1   The Eigenvalue problem

The mentioned eigenvalue problem comes from the basis expansion of the spacial part and from the trick of projecting with a single function from left. We will explain briefly. First let us express the expansion,

$$
\left| \psi_p^{\mathrm{DW}} \right\rangle = \sum_l C_{lp} \left| \psi_l^{\mathrm{HO}} \right\rangle \tag{4.52}
$$

and then project from left the bra $\left\langle \psi_k^{\mathrm{HO}} \right|$ in the inner-product space of h^{DW}

$$
\left\langle \psi_k^{\mathrm{HO}} \left| h^{\mathrm{DW}} \right| \psi_p^{\mathrm{DW}} \right\rangle = \sum_l C_{pl} \left\langle \psi_k^{\mathrm{HO}} \left| h^{\mathrm{DW}} \right| \psi_l^{\mathrm{HO}} \right\rangle = \sum_l C_{pl} \varepsilon_l^{\mathrm{DW}} \tag{4.53}
$$

This gives us an eigenvalue equation

$$
\boldsymbol{HC} = \varepsilon^{\mathrm{DW}} \boldsymbol{C} \tag{4.54}
$$

with

$$
H_{ij} = \left\langle \psi_i^{\mathrm{HO}} \left| h^{\mathrm{DW}} \right| \psi_j^{\mathrm{HO}} \right\rangle \tag{4.55}
$$

---

[14] Essentially just expanding an index in each dimension, for instance $i = (i_x, i_y, i_z)$ with corresponding $p = (p_x, p_y, p_z)$ with each index inside the tuple running to each respective index, meaning for instance $p_x = 0$ to i_x and so on.

Using equation 4.51 we can write H_{ij} as

$$H_{ij} = \varepsilon_i^{\mathrm{HO}} \delta_{ij} + \left\langle \psi_i^{\mathrm{HO}} \,\middle|\, V_n^{\mathrm{DW}} \,\middle|\, \psi_j^{\mathrm{HO}} \right\rangle \tag{4.56}$$

by using the solution to Schrödinger's equation for the harmonic oscillator system.

We are now in a position to build a basis for the double-well system by reusing all the results and expressions concerning the single-well system. The only difference is the extra integral over V_n^{DW} where n would be larger than 2(as is the case with the harmonic oscillator potential).

For clarity let us also write out the expression for the resulting Hartree-Fock basis to be used with the Variational Monte-Carlo method.

$$\psi_p^{\mathrm{HF}} = \sum_{kl} C_{pk}^{\mathrm{HF}} C_{kl} \psi_l^{\mathrm{HO}} \tag{4.57}$$

The procedure of diagonalizing H_{ij} also gives an additional set of energies we can use. The full form of the integral-elements involved in Hartree-Fock is thus

$$
\begin{aligned}
\left\langle \psi_p^{\mathrm{DW}} \,\middle|\, \psi_q^{\mathrm{DW}} \right\rangle &= \delta_{pq} \varepsilon_p^{\mathrm{DW}} \delta_{pq} \\
\left\langle \psi_p^{\mathrm{DW}} \,\middle|\, h^{\mathrm{DW}} \,\middle|\, \psi_q^{\mathrm{DW}} \right\rangle &= \varepsilon_p^{\mathrm{DW}} \delta_{pq} \\
\left\langle \psi_p^{\mathrm{DW}} \psi_q^{\mathrm{DW}} \,\middle|\, \frac{1}{r_{12}} \,\middle|\, \psi_r^{\mathrm{DW}} \psi_s^{\mathrm{DW}} \right\rangle &= \sum_{tuvw}^{ijkl} C_{tp} C_{uq} C_{vr} C_{ws} \left\langle \psi_t^{\mathrm{HO}} \psi_u^{\mathrm{HO}} \,\middle|\, \frac{1}{r_{12}} \,\middle|\, \psi_v^{\mathrm{HO}} \psi_w^{\mathrm{HO}} \right\rangle
\end{aligned}
\tag{4.58}
$$

Where the two-body elements over the harmonic oscillator functions can be calculated by expansion in s-type Gaussian constituents and then using equation 4.48.

4.4.2   Choosing the Basis Functions

In order to actually solve the eigenvalue equation we use Python and the NumPy package, however we still need to choose the ψ^{HO}'s first. This will be experimented with and we will choose enough to reach to *Hartree-Fock limit*. The choice will also follow the Harmonicoscillator levels in terms of degeneracy(see figures 6.3b and 6.3a). This means for instance that if we choose to only use 1 spacial function we only need the ground-state function, but there is no reason to believe that the electron-configuration would prefer any two(or 3 in 3D) of the functions in the second level over one another. This means that we choose the basisfunctions according to the *magic numbers* of the Harmonicoscillator system basis.

4.4.3   Degeneracy

When we tackled the single-well problem the energy-levels were degenerate and followed the magic numbers. For the double-well only the degeneracy due to spin is present. This means that we can(still with two-spin fermions) run the simulations with $N = 2, 4, 6, \ldots$ instead of the original $N = 2, 6, 12, 20, \ldots$ in two dimensions and $N = 2, 8, 20, 30 \ldots$ in three.

4.5   Summary

This chapter tackled the one- and two-body integrals over s-type Gaussian functions in two dimensions by using the results and approach of Trygve Helgaker. We will here rewrite the expressions found and write out the full expression for integrals over harmonic oscillator functions.

Firstly the integrals over Hermite-Gaussians (monomials multiplied by exponential). The Hermite-Gaussian was expressed as

$$g_n(\boldsymbol{\alpha}; \boldsymbol{r}, \boldsymbol{A}) = \prod_d \left(\frac{\partial}{\partial A_d} \right)^{n_d} \mathrm{e}^{-\alpha_d (x_d - A_d)^2} \tag{4.59}$$

An expansion of these in terms of Hermite-polynomials was then made to arrive at an overlap distribution, with the Gaussian product rule for the product of two Hermite-Gaussians, with a recursive relation for the

expansion coefficients

$$\Omega_{ij}(\alpha, \beta, \boldsymbol{r}, \boldsymbol{A}, \boldsymbol{B}) = \sum_{t=0}^{i+j} E_t^{ij} g_t(\alpha, \beta, \boldsymbol{r}, \boldsymbol{P})$$

$$E_t^{i+1,j} = \frac{1}{2(\alpha+\beta)} E_{t-1}^{ij} - \frac{\beta}{\alpha+\beta}(A_x - B_x) E_t^{ij} + (t+1) E_{t+1}^{ij} \qquad (4.60)$$

$$E_t^{i,j+1} = \frac{1}{2(\alpha+\beta)} E_{t-1}^{ij} - \frac{\alpha}{\alpha+\beta}(A_x - B_x) E_t^{ij} + (t+1) E_{t+1}^{ij}$$

$$E_0^{00} = K_{AB}$$

The overlap integral was then found to be

$$\langle g_i(\boldsymbol{\alpha}; \boldsymbol{r}, \boldsymbol{A}) \,|\, g_j(\boldsymbol{\beta}; \boldsymbol{r}, \boldsymbol{B}) \rangle = \sum_p^{i+j} E_p^{ij} \left(\frac{((-1)^p - 1)\,\Gamma\left(\frac{p+1}{2}\right)}{2(\alpha_p + \beta_p)^{\frac{p+1}{2}}} \right)^d \qquad (4.61)$$

The integral over a potential x_d^k was found with the binomial expansion to be

$$\langle g_i(\boldsymbol{\alpha}; \boldsymbol{r}, \boldsymbol{A}) \,|\, x_d^k \,|\, g_j(\boldsymbol{\beta}; \boldsymbol{r}, \boldsymbol{B}) \rangle = \sum_{p=\text{even}}^{i+j} E_p^{i+j} \left(\frac{\Gamma\left(\frac{p+1}{2}\right)}{2(\alpha_p + \beta_p)^{\frac{p+1}{2}}} \right)^{D-1} \sum_{\substack{l=0 \\ \text{even}}}^{k} \binom{k}{l} \frac{P_d^{k-l}}{2(\alpha_p + \beta_p)^{\frac{l}{2}}} \Gamma\left(\frac{l+1}{2}\right) \quad (4.62)$$

and the integral with the Laplacian was

$$\langle g_i(\boldsymbol{\alpha}; \boldsymbol{r}, \boldsymbol{A}) \,|\, \nabla^2 \,|\, g_j(\boldsymbol{\beta}; \boldsymbol{r}, \boldsymbol{B}) \rangle = \sum_d \prod_{d' \neq d} \langle g_{i_{d'}}(\alpha_{d'}; x_{d'}, A_{d'}) \,|\, \sigma_{d'}(S_d(\beta; x - B_d)) \,|\, g_{j_{d'}}(\beta d'; x_{d'}, B_{d'}) \rangle \quad (4.63)$$

with

$$S_d(\alpha; x_d - A_d) \equiv \left(4\alpha^2 (x_d - A_d)^{i_d + 2} - 2\alpha (2i_d + 1)(x_d - A_d)^{i_d} + i_d (i_d - 1)(x_d - A_d)^{i_d - 2} \right)$$

$$\sigma_d(S_d) \equiv \begin{cases} 1, & d' \neq d \\ S_d, & d' = d \end{cases} \qquad (4.64)$$

Finally the integral over the Coulomb interactions was calculated using a recursive relation for the one-dimensional integral it was defined by. The expressions are

$$\left\langle g_{iA}^{\boldsymbol{\alpha}} g_{jB}^{\boldsymbol{\beta}} \,\middle|\, \frac{1}{r_{12}} \,\middle|\, g_{kC}^{\boldsymbol{\delta}} g_{lD}^{\boldsymbol{\gamma}} \right\rangle = \frac{a}{\sqrt{(\alpha + \gamma + \beta + \delta)}} \sum_{pq}^{i+k,j+l} E_p^{ik} E_q^{jl} (-1)^q \xi_{p+q} \left(\frac{(\alpha+\gamma)(\beta+\delta)}{\alpha+\gamma+\beta+\delta}, \boldsymbol{R_{S_1 S_2}} \right) \quad (4.65)$$

with

$$a \equiv \begin{cases} \dfrac{\pi^{\frac{3}{2}}}{\sqrt{(\alpha+\gamma)(\beta+\delta)}}, & \text{2D} \\[4mm] \dfrac{\pi^{\frac{5}{2}}}{(\alpha+\gamma)(\beta+\delta)}, & \text{3D} \end{cases} \qquad (4.66)$$

And the recursive relation in two dimensions(left) and three dimensions (right)

$$\begin{aligned} \xi_{t+1,u}^n &= t\xi_{t-1,u}^{n+1} + X_{AB}\xi_{t,u}^{n+1} \\ \xi_{t,u+1}^n &= u\xi_{t,u-1}^{n+1} + Y_{AB}\xi_{t,u}^{n+1} \\ \xi_{00}^n &= \left(\frac{-2\alpha\beta}{\alpha+\beta} \right)^n \zeta_n\left(\frac{\alpha\beta}{\alpha+\beta} R_{AB}^2 \right) \\ \zeta_n(x) &= \int_{-1}^{1} \frac{u^{2n}}{\sqrt{1-u^2}} e^{-u^2 x} \mathrm{d}u \end{aligned} \qquad \begin{aligned} \xi_{t+1,u,v}^n &= t\xi_{t-1,u,v}^{n+1} + X_{AB}\xi_{t,u,v}^{n+1} \\ \xi_{t,u+1,v}^n &= u\xi_{t,u-1,v}^{n+1} + Y_{AB}\xi_{t,u,v}^{n+1} \\ \xi_{t,u,v+1}^n &= u\xi_{t,u,v-1}^{n+1} + Y_{AB}\xi_{t,u,v}^{n+1} \\ \xi_{000}^n &= (-2\alpha\beta)^n \zeta_n\left(\frac{\alpha\beta}{\alpha+\beta} R_{AB}^2 \right) \\ \zeta_n(x) &= \int_{-1}^{1} u^{2n} e^{-u^2 x} \mathrm{d}u \end{aligned} \qquad (4.67)$$

We also give a reminder again that these expressions are only valid for *isotropic* Gaussian functions, Gaussians whose scaling factor(i.e α) is the same in all dimensions. For the non-isotropic case the see [6].

And as promised, here are the full expressions for the integral elements with harmonic oscillator functions using equation 4.65 and the orthogonality of the harmonic oscillator functions

$$\left\langle \psi_i^{\mathrm{HO}} \,\middle|\, \psi_j^{\mathrm{HO}} \right\rangle = N_i \delta_{ij} \tag{4.68}$$

$$\left\langle \psi_i^{\mathrm{HO}} \,\middle|\, h^{\mathrm{HO}} \,\middle|\, \psi_j^{\mathrm{HO}} \right\rangle = N_i \varepsilon_i^{\mathrm{HO}} \delta_{ij} \tag{4.69}$$

$$\left\langle \psi_i^{\mathrm{HO}} \psi_j^{\mathrm{HO}} \,\middle|\, \frac{1}{r_{12}} \,\middle|\, \psi_k^{\mathrm{HO}} \psi_l^{\mathrm{HO}} \right\rangle = N_{ijkl} \frac{a}{\sqrt{2\omega}} \sum_{tuvw}^{ijkl} H_{tuvw}^{ijkl} \sum_{pq}^{t+v,u+w} E_p^{tv} E_q^{uw} (-1)^q \xi_{p+q} \left(\frac{\omega}{2}, \mathbf{0} \right) \tag{4.70}$$

$$a \equiv \begin{cases} \dfrac{\pi^{\frac{3}{2}}}{\sqrt{\omega}}, & \text{2D} \\[2ex] \dfrac{\pi^{\frac{5}{2}}}{\omega}, & \text{3D} \end{cases} \tag{4.71}$$

with the notations

$$N_{ijkl} = X_i X_j X_k X_l$$
$$\sum_{tuvw}^{ijkl} = \sum_p^t \sum_q^u \sum_r^v \sum_s^w$$
$$\sum_{pq}^{t+v,u+w} = \sum_p^{t+v} \sum_q^{u+w} \tag{4.72}$$
$$H_{tuvw}^{ijkl} = H_{i,t} H_{j,u} H_{k,v} H_{l,w}$$
$$E_p^{tv} = E_{p,t+v}$$

And using the multi-index notation for the dimensions in p and q indices. The H are the Hermite coefficients.

4.6   Further Work

This concludes this chapter on basis functions. We have found an expression for the integral elements involving Hermite-Gaussians and used these to express the integral over harmonic oscillator functions which in turn gave an expression for the integrals over double-well functions as they were just expanded in harmonic oscillator functions. The next chapter will present optimizations methods used in the variational method and the implementations of these integrals are presented in chapter 6.

As a note for further work the expressions found here are for isotropic gaussians only. For certain systems, like the one presented in article[6], more flexibility in the Gaussian functions can be desired, such that finding expression for the integrals of Gaussian functions with higher order monomial factors and extending the existing code to calculate those can be of desire.

Chapter 5

Numerical Optimization

In the Variational Monte Carlo method in section 3.7 the essential point was to vary a set of *variational parameters* in order to reach an eigenbasis which gives the ground-state energy of the Hamiltonian in question. There are many was one could approach this. One way could be to wildly guess random parameters and hope for the best, obviously this is a poor approach. The more sound approach would be to optimize(minimization int the VMC case) the wavefunction using methods from a popular field in mathematics called *numerical optimization*. The two methods used in this thesis was the *Conjugate Gradient method* and a version of the *Adaptive Stochastic Gradient Descent*. The explaining of the approaches for numerical optimization is only explained briefly. For a better mathematical explanation see » REF THESE «

5.1   The Optimization Problem

We will explain the general approach for minimizing a multi-variate function and set the terminology in this section.

The problem in question is the following. Given a continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, for what set of parameters $\{\alpha\}_{k=1}^n$ is

$$\nabla_\alpha f = \mathbf{0} \tag{5.1}$$

fulfilled[1]. This means we seek a point $\boldsymbol{\alpha}_m$ in the real where variation of the value of f is zero. In reality the condition in equation 5.1 is only approximate, that is we terminate the search for a minimum if we reached a point where the absolute value of f is within a threshold ϵ

$$|\nabla_\alpha f| \leq \epsilon \tag{5.2}$$

One might at this point have the question, wouldn't the condition presented in equation 5.2(and equation 5.1 be valid for a maximum as well? The answer is yes, it would. The simple fix to this is to define the *search direction*, more precisely the sign of the search direction. The next section explains this in better detail.

5.2   Gradient Descent

We defined the optimization problem and defined a simple condition for the extremal and mentioned a search direction in the previous section. A search direction in our context is a direction $\boldsymbol{p} \in \mathbb{R}^n$ which points towards $\boldsymbol{\alpha}_m$. To find $\boldsymbol{p}$ we use the well known *second derivative test* to determine the curvature of f. This, mentioned qualitatively, means that the gradient of f any point $\boldsymbol{\alpha}_i$ points towards an extremal and that the negative gradient(negative sign) points towards the minimum and the positive gradient points towards the maximum. This observation gives a simple rule for finding $\boldsymbol{\alpha}_m$. Start out with blindly guessing a point $\boldsymbol{\alpha}_0$ and keep updating the parameters according to the following recursive rule

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha_{n-1}} - \gamma \nabla_\alpha f \tag{5.3}$$

and terminate the search when equation 5.2 is fulfilled. » REF AND MAKE ALGORITHM «[2]

---

[1] $\nabla_\alpha = \sum_k \boldsymbol{e}_k \frac{\partial}{\partial \alpha_k}$ with $\boldsymbol{e}_k \in \mathbb{R}^n$ a unit vector along direction k.

[2] Change the negative sign in front of the gradient if a maximum is desired.

This method of finding the minimum is known as the method of *Gradient Descent* and is the simplest method for finding a minimum. The problem however is stability, the termination condition is firstly not optimal » REF THIS « and the step-size γ is a constant which can give allot of oscillations around minimum as the algorithm might get close to the minimum and then *over-shoot* and go past the minimum point, turn around(because the sign changes) and over-shoot again and then keep going. Many(seriously many) methods have been devised to account for these problems and other. We will contain ourselves with the methods mentioned in the introduction of this chapter.

5.3 Adaptive Stochastic Gradient Descent

Along with the limitations of the method of gradient descent, the *Adaptive Stochastic Gradient Descent* tries to account for those, but also takes into account the variance introduced by the stochastic nature of the probability distribution. As such many variations of the method have been proven to be popular among problems in which the function to be minimized is an expectation value. The method used in this thesis is the one described in [39]. We will give a summary of the method here, for a more detailed outline and description see [39].

Like the gradient descent method the adaptive stochastic gradient descent method updates the parameters in the same manner as in equation 5.3, the difference however is that the step γ is changed for each iteration in accordance to the following

$$
\begin{aligned}
\gamma_{n+1} &= \frac{a}{t_{n+1} + A} \\
t_{n+1} &= \max(t_n + g(X_n), 0) \\
X_n &= -\nabla f_n \cdot \nabla f_{n+1} \\
g(x) &= g_{\min} + \frac{g_{\max} - g_{\min}}{1 - \frac{g_{\max}}{g_{\min}} \mathrm{e}^{-\frac{x}{\omega}}}
\end{aligned}
\tag{5.4}
$$

The whole idea of the method is that the form of g and the accumulative combination of gradient estimations for each step the total error would tend quickly to zero, meaning the central element(namely the gradient) in the minimization is well behaving.

The main concern with the method is the convergence, although the error in the gradient estimations tend towards zero, the step-sizes themselves will also be quite small after some iterations. For this reason we use the adaptive method with a quasi-Newton method. This means that we start with a random guess at the parameters and keep iterating with the quasi-Newton method until the norm of the gradient is below some threshold, at which the adaptive method is applied from that point and onward till convergence is reached.

5.4 Newtons-Method and Quasi-Newton Methods

We will here explain briefly *Newton's method* and *Quasi-Newton* methods as the ideas presented will be used in the next section.

Newton's method[25](or Newton-Raphson method) is originally a method for finding the zeros of a function. The rule states that given a real-valued function $f : \mathbb{R} \to \mathbb{R}$ and an initial guess $x \in \mathbb{R}$ for the zero-point, recursively find better approximations for the zero by

$$
x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}
\tag{5.5}
$$

This method would then within a number iterations find the zero that is closest to x_0.

For the optimization problem the condition for a point to be an extremal is equation 5.1 meaning, again, that one needs to find the zero of the derivative. Newton's method in this case would be

$$
x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}, \qquad n \geq 0
\tag{5.6}
$$

Of course in the real world one might work with multi-variate function, not to worry as Newton's method for optimization problems in the multi-variate case with $f : \mathbb{R}^n \to \mathbb{R}$ (still real-valued) is

$$
\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - |\boldsymbol{H}f(\boldsymbol{x}_n)|^{-1}\dot{\nabla} f(\boldsymbol{x}_n), \qquad n \geq 0
\tag{5.7}
$$

where $\boldsymbol{H}$ is the Hessian matrix. One might also introduce a step-length multiplied to the Hessian part in order to induce conditions[31] which ensure some stability of the method.

Newton's method, in most cases, converges faster(less iterations) towards the minimum than gradient descent making it favorable, however the full Hessian has to be known. This matrix(or its inverse) is in many cases too expensive to compute or difficult to express in closed-form. In these cases the class of methods knows as Quasi-Newton methods can be utilized.

Quasi-Newton methods give an estimate of the inverse Hessian by using the first derivatives. Introduce the Taylor approximation of f around an iteration point $\boldsymbol{x}_n$

$$f(\boldsymbol{x}_k + \boldsymbol{s}) \approx f(\boldsymbol{x}_k) + (\nabla f(\boldsymbol{x}_k))^T \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T \boldsymbol{H} \boldsymbol{s} \tag{5.8}$$

differentiate with respect to the change $\boldsymbol{s}$

$$\nabla_s f(\boldsymbol{x}_k + \boldsymbol{s}) \approx \nabla f(\boldsymbol{x}_k) + \boldsymbol{H}\boldsymbol{s} \tag{5.9}$$

and introduce the condition in equation 5.1 and set this gradient to zero to find the change $\boldsymbol{s}$

$$\boldsymbol{s} = -\boldsymbol{H}^{-1}\nabla f(\boldsymbol{x}_k) \tag{5.10}$$

Another way to determine this particular form for $\boldsymbol{s}$ is to say that the approximation to the Hessian must satisfy the *secant equation* which is equation 5.9. The updating rule for $\boldsymbol{x}_n$ is then given by

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \gamma_k \boldsymbol{H}_n^{-1}\nabla f(\boldsymbol{x}_n) \tag{5.11}$$

where γ_k is again introduced to give some stability conditions. The important part of this equation is the index on the inverse Hessian. This is essentially just a relabeling at the change $\boldsymbol{s}$ is technically applied for each iterate $\boldsymbol{x}_n$. Note also that $\boldsymbol{s}$ takes the role of the search direction in this case. The algorithm is then to make an initial guess on the Hessian(usually just the identity matrix) and then use a type of updating formula that finds a new approximation for the Hessian at each step n. There are a number of these updating formulas, just to mention some we have DFP, SR1, McCormick, Broyden, BFGS and more. The one we will mention in more detail is the BFGS method, but a main formula that shows up in all of the updating methods is the *Sherman-Morrison formula* for the inverse. This basically means that the need for calculation of the inverse matrix is completely removed.

With the mentioned expression we can devise an algorithm similar to Newton's method for finding the minimum $\boldsymbol{x}_m$. Starting with an initial guess for the inverse Hessian $\boldsymbol{H}_0^{-1}$ and minimum $\boldsymbol{x}_0$ with the condition that $\boldsymbol{H}_0^{-1}$ is positive-definite (identity matrix is a nice start if nothing else is known) proceed with

- $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \gamma_n \boldsymbol{H}_0^{-1}\nabla f(\boldsymbol{x}_n)$

- Calculate the gradient $\nabla f(\boldsymbol{x}_{n+1})$

- Use $\nabla f(\boldsymbol{x}_{n+1})$ and $f(\boldsymbol{x}_n)$ in a updating formula of choice to find $\boldsymbol{H}_{n+1}^{-1}$

5.5   BFGS Method

In the previous section we gave an outline for Newton's method and the class known as Quasi-Newton methods. The latter used an approximation for the inverse of the Hessian matrix, which was updated at each step in the algorithm. For the sake of brevity only conditions employed to arrive at the expression for the updating formula and the formula itself is given here, for more see [3, 11, 12, 31, 36]. The conditions enforced is

- Secant condition: $\boldsymbol{H}_{n+1}\boldsymbol{s}_n = \nabla f(\boldsymbol{x}_{n+1}) - \nabla f(\boldsymbol{x}_n)$

- Strong curvature: $\boldsymbol{s}_k^T \cdot (f(\boldsymbol{x}_{n+1}) - \nabla f(\boldsymbol{x}_n)) > 0$

and the resulting formula states with $\boldsymbol{y}_k = f(\boldsymbol{x}_{n+1}) - \nabla f(\boldsymbol{x}_n)$

$$\boldsymbol{H}_{n+1} = \boldsymbol{H}_n + \frac{\boldsymbol{y}_n \boldsymbol{y}_n^T}{\boldsymbol{y}_n^T \boldsymbol{s}_n} - \frac{\boldsymbol{H}_n \boldsymbol{s}_n \boldsymbol{s}_n^T \boldsymbol{H}_n}{\boldsymbol{s}_n^T \boldsymbol{H}_k \boldsymbol{s}_n} \tag{5.12}$$

With the Sherman-Morrison formula[34] the inverse is updated with

$$\boldsymbol{H}_{n+1}^{-1} = \boldsymbol{H}_n^{-1} + \frac{\left(\boldsymbol{s}_n^T \boldsymbol{y}_n + \boldsymbol{y}_n^T \boldsymbol{H}_n^{-1} \boldsymbol{y}_n\right)\left(\boldsymbol{s}_n \boldsymbol{s}_n^T\right)}{\left(\boldsymbol{s}_n^T \boldsymbol{y}_n\right)^2} - \frac{\boldsymbol{H}_n^{-1}\boldsymbol{y}_n \boldsymbol{s}_n^T + \boldsymbol{s}_n \boldsymbol{y}_n^T \boldsymbol{H}_n^{-1}}{\boldsymbol{s}_n^T \boldsymbol{y}_n} \tag{5.13}$$

5.6   Linesearch methods

In the optimization methods described in section 5.4 There was one important part neglected, namely how to find the step-length γ, introduced in the updating formula. As it is, one can choose it in any manner desired, however a class if one-dimensional minimization methods knows as *lineasearch methods* are often used to get an (usually rough) estimate for the step length at each iteration in the optimization. These methods all have some conditions for stability and convergence as an innate property, meaning the validity of the step length is better[3]. Some popular linesearch methods are backtracking linesearch, Hager-Zhang method, Strong Wolfe conditions and the More-Thuente linesearch method. The one used here is the latter. For an exact derivation and explanation of linesearch methods in general see [31]. See also the fantastic article by Jorge J. Moré and David J. Thuente[28].

The basic idea of linesearch methods is to solve a one-dimensional problem of minimizing

$$\phi(\alpha) = f(\alpha \boldsymbol{p}_k + \boldsymbol{x}_k) \tag{5.14}$$

with $f : \mathbb{R}^n \to \mathbb{R}$ and $\boldsymbol{p}_k$ is a search direction as described with the quasi-Newton methods and $\boldsymbol{x}_k$ is the current iterate(point) in the minimization. Notice also that

$$\frac{\partial \phi}{\partial \alpha} = \boldsymbol{p}_k \cdot \nabla f(\alpha \boldsymbol{p}_k + \boldsymbol{x}_k) \tag{5.15}$$

by the chain-rule and the gradient on the right hand side is over the parameters $\boldsymbol{x}_k$. One usually perform this linesearch loosely since the search direction is not necessarily directly pointing towards the minimum, meaning we only search for a step length that gives a *sufficient decrease* in the function value f.

5.7   Stochastic-Adaptive-BFGS

In section 5.5 we mentioned the popular BFGS method for updating the Hessian matrix and its inverse. A more resent method which uses that method with the stochastic nature of a functional expectation value is a method called SABFGS[13] described by Zhou C., Gao W. and Goldfarb D. This method uses the BFGS update for the Hessian, but uses an adaptive step instead of the deterministic linesearch for the step-size. There one problem however, the method itself is only valid for *self-concordant* functions. The energy-functional is by-far not within this criteria. This problem can be accounted for by using the Wolfe-conditions. That is to check that the step-size satisfies the Wolfe-conditions at each iteration before actually making an update. The algorithm presented takes this into account aswell.

Algorithm 1 SA-BFGS
_____

Input: $\boldsymbol{x}_0$, $\boldsymbol{H}_0$, $\boldsymbol{G}_0$, $\beta < 1$
for $k = 0$ to $M_{\max}$ **do**
  $\boldsymbol{g}_k = \nabla F_k(\boldsymbol{x}_k)$ ▷ Gradient in current step
  $\boldsymbol{d}_k = -\boldsymbol{H}_k \boldsymbol{g}_k$ ▷ Search direction
  $\delta_k = \sqrt{\boldsymbol{d}_k^T \boldsymbol{G}_k(\boldsymbol{x}_k) \boldsymbol{d}_k}$
  $\alpha_k = \frac{\boldsymbol{g}_k^T \boldsymbol{H}_k \boldsymbol{g}_k}{\delta_k^2}$
  $t_k = \frac{\alpha_k}{1 + \alpha_k \delta_k}$ ▷ Step size
  $\boldsymbol{g}_{k+1} = \nabla F_k(\boldsymbol{x}_k + t_k \boldsymbol{d}_k)$ ▷ Propose new set of parameters
  if $\boldsymbol{g}_{k+1}^T \boldsymbol{d}_k < \beta \boldsymbol{g}_k^T \boldsymbol{d}_k$ **then**   ▷ Wolfe-Conditions
    Set $\boldsymbol{d}_k = -\boldsymbol{g}_k$
    Recompute δ_k, α_k and t_k
    Set $\boldsymbol{H}_{k+1} = \boldsymbol{H}_k$ and $\boldsymbol{G}_{k+1} = \boldsymbol{G}_k$
  else
    Set $\boldsymbol{H}_{k+1}$ with BFGS inverse update   ▷ equation 5.13
    Set $\boldsymbol{G}_{k+1}$ with BFGS update   ▷ equation 5.12
  end if
  $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + t_k \boldsymbol{d}_k$ ▷ Update parameters
end for
_____

---

[3] It's actually present...

5.8   Simulated Annealing

A huge problem with the mentioned methods is the fact that they only converge towards a *local minimum* which is not necessarily the global minimum. Many methods already exists to account for this, the one used in this thesis and to be described in this section is the method known as *simulated annealing*. Simulated annealing follows a simple algorithm, namely

---

Algorithm 2 Simulated Annealing

    Initialize a solution $s = s_0$.   ▷ I.e a set of parameters $\{\alpha\}_{k=1}^{N}$
    for $j = 1$ to $M_{\max}$ **do**
        Set temperature T with specific function for $\frac{j}{M_{\max}}$
        Pick a new state s_{new} within some neighbour of s
        if $P(f(s), f(s_{\text{new}}), T) \geq \xi$ **then**   ▷ ξ uniformly distributed random in $[0, 1]$.
            $s = s_{\text{new}}$
        end if
    end for

    The idea is to start with searching a large part of the solution space, since a high temperature increases the search-range, and hope that as j reaches $M_{\max}$ the probability function P is such that the solution is trapped within the down-hill of the global minimum.

The specific form of P, T and how to choose a neighbour is specific from problem to problem however an effective and simple way to define these is by using the Metropolis-algorithm with

$$P = \exp\left(-\frac{f(\text{new}) - f(s)}{T}\right) \tag{5.16}$$

define the temperature as

$$T_j = \frac{j}{M_{\max}} \tag{5.17}$$

and choose new neighbours within some min/max range from the current s. » REF THIS «.

After the annealing is done a Quasi-Newton method is used and then the adaptive method is used to converge to the minimum.

Chapter 6

Implementation

This chapter will explain the code used in the thesis in detail. The code itself i given in » REF GITHUB REPO «. General information of the usage of packages are given in appendix » REF APPENDIX « while the structure and workflow of the code is given here. We will first present a workflow-chart of the two main code-bases used, namely the Hartree-Fock implementation and the Variational Monte-Carlo implmentation.
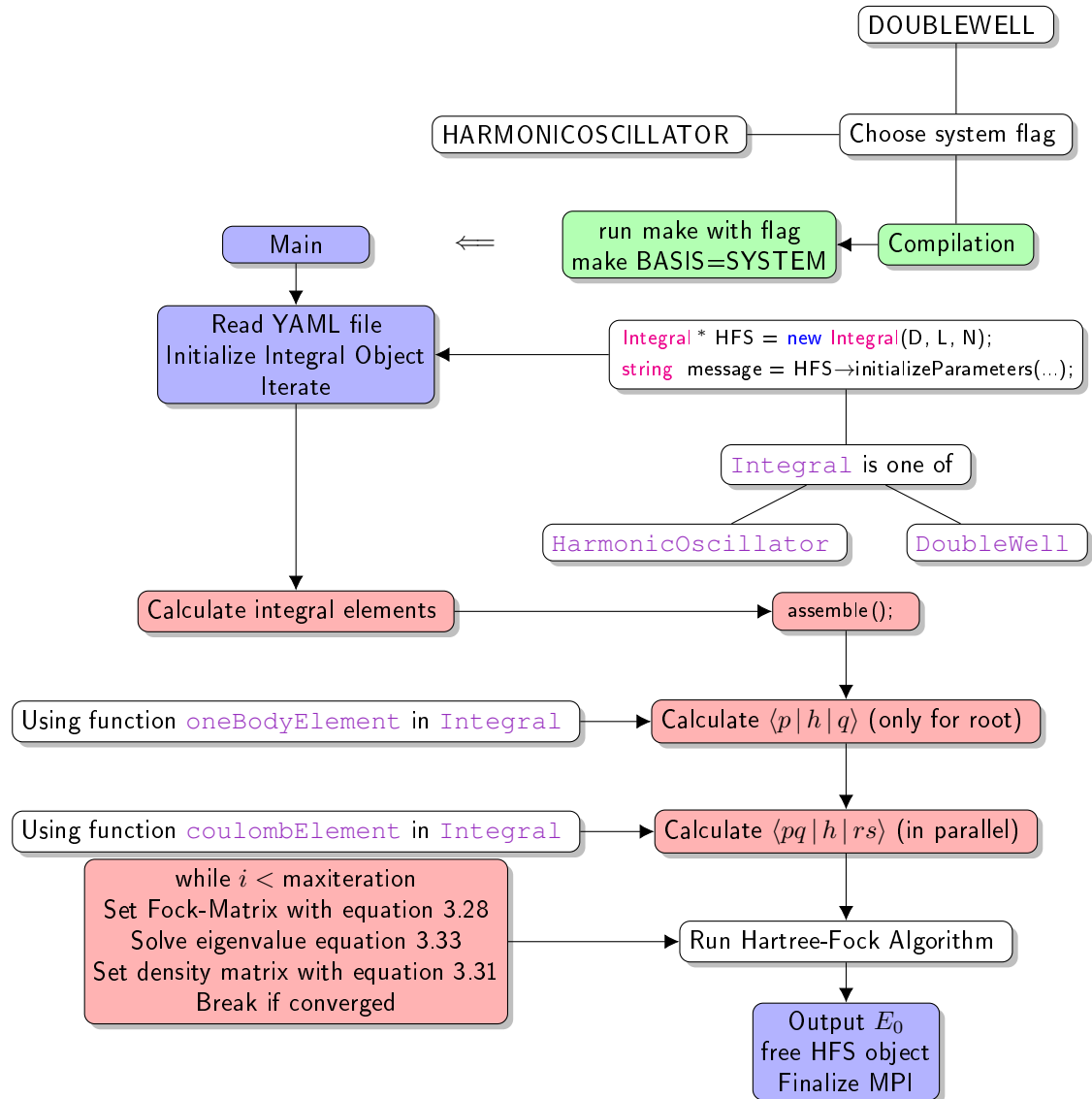
DOUBLEWELL

HARMONICOSCILLATOR — Choose system flag

run make with flag
make BASIS=SYSTEM ⟵ Compilation

Main ⟸

Read YAML file
Initialize Integral Object
Iterate

```
Integral * HFS = new Integral(D, L, N);
string   message = HFS→initializeParameters(...);
```

`Integral` is one of

`HarmonicOscillator`   `DoubleWell`

Calculate integral elements   assemble();

Using function `oneBodyElement` in `Integral`   Calculate $\langle p\,|\,h\,|\,q\rangle$ (only for root)

Using function `coulombElement` in `Integral`   Calculate $\langle pq\,|\,h\,|\,rs\rangle$ (in parallel)

while $i <$ maxiteration
Set Fock-Matrix with equation 3.28
Solve eigenvalue equation 3.33
Set density matrix with equation 3.31
Break if converged   Run Hartree-Fock Algorithm

Output E_0
free HFS object
Finalize MPI

Figure 6.1: Flow chart of Hartree-Fock implementation.

Choose JASTROW flag → Compilation ← Choose SYSTEM flag

PADEJASTROW
RBMJASTROW
EXPNQS

HARMONICOSCILLATOR
HARTREEFOCK
DOUBLEWELL

optional flag
TESTS=ON/OFF →
run make with flags
WAVEFUNCTION=SYSTEM
JASTROW=JASTROW
← optional flag
DEBUG=ON/OFF

⇓

SlaterJastrow
Slater

ImportanceSampling
BruteForce

T

Read YAML input file.

Choose one   Choose one   T

Main

T* wf = new T(D, N, P);
wf→initializeParameters (...);
wf→initializeMatrices ();

Create wavefunction object
Create sampler object

T* vmc = new T(wf, Δt, P, M);

Gaussian random if
NQS parameters.

Uniform random for
other parameters.

Minimizer<T>* m = new Minimizer<T>
(vmc, S, I, eps, M);
m→minimize();

Find initial parameters.

S: Method
- SA-BFGS
- BFGS (MTLS)
- ASGD
- SD
- CG (MTLS)

Find optimal set of parameters.

Run final sampling with optimal parameters.   vmc→sampler();

Output E_0
free Wavefunction object (wf)
free Sampler object (vmc)
free Minimizer object (m)

Figure 6.2: Flow chart of Variational Monte-Carlo implementation.

6.1 Cartesian Basis

In chapter 4 we mentioned the use of basis functions the different Many-Body methods. These can be pre-built using nifty intuition. One such observation is in the way harmonic oscillator functions station themselves on energy-levels(in the full-shell case). The following image[1] describes this for the first few levels

---

[1] As the old idiom goes; "A picture is worth a thousand words"

(a) Two-dimensional case with (n_x, n_y) notation.   (b) Three-dimensional case with (n_x, n_y, n_z) notation.
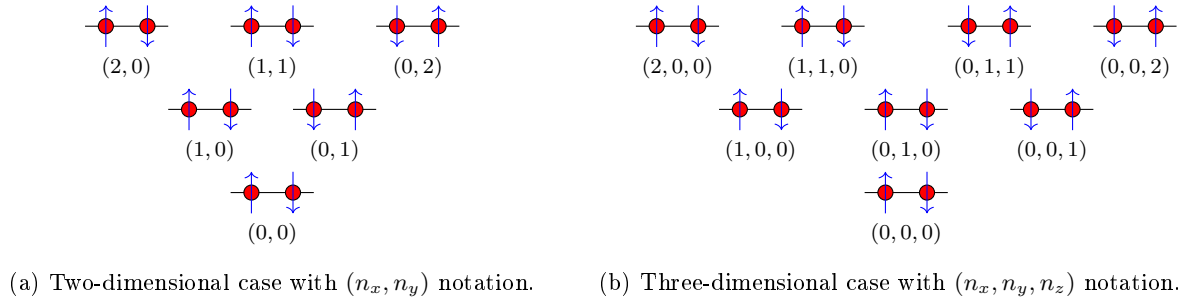
Figure 6.3: Harmonic Oscillator Levels

This specific arrangement of basis-functions is implemented in class `Cartesian` and is used in both the Hartree-Fock and VMC implementations. It essentially builds a matrix of states with the rows being the specific state and the columns containing the quantum numbers(in cartesian), the spin-value(as an integer), magic number and energy(in natural units proportional to the oscillator frequency). The essential form is

$$\begin{pmatrix} n_x & n_y & s & m_s & E & M \end{pmatrix} \quad \begin{pmatrix} n_x & n_y & n_z & s & m_s & E & M \end{pmatrix} \tag{6.1}$$

with the n's being the principal numbers, s the spin value m_s the spin projection(up or down in our case), E the energy and M the magic number. The `Cartesian` class builds the states with alternating spin (the spacial parts are doubled with spin), but also has a function for restructuring by setting the states with spin down in acending order first and the same states with spin up after.

6.2   Hartree-Fock

The Hartree-Fock method described in section 3.5 is implemented in » REF GITHUB «. Only the restricted case is implemented and is present as the class `HartreeFockSolver`. The matrix-elements(integrals) are implemented in `HermiteIntegrals` class. This class also uses an auto-generated header for the Hermite-coefficients, see section 6.5 below. The `HartreeFockSolver` is implemented in a general way such that an abstract class for the integral elements is all that is needed. The `HartreeFockSolver` can then be inherited and used. An example of how to create a solver object with the Double-Well system called HFS with number of dimensions D, number of basis functions L and number of particles N.

```
DoubleWell* HFS = new DoubleWell(D, L, N);
string message = HFS->initializeParameters(...);
```

With the ... meaning one initializes it with in however manner the function was made. The initialization function must also return a message determined by the success of the initialization. If it succeeds it return an empty message while if not it returns a pre-defined message.

Here is a simple example code-snippet which initializes and runs the Hartree-Fock algorithm

```
DoubleWell* HFS = new DoubleWell(D, L, N);

string message = HFS->initializeParameters(...);
if (message.compare("")) {
    if (myRank == 0) {
        std::cout << message << std::endl;
    } // end if
    delete HFS;
    finalize();
} // end if

double E = HFS->iterate(M, 1e-8, true);
```

The `iterate` function takes in M as the maximum number of iterations, the convergence tolerance (when to break the iteration) and a boolean for showing progress or not. It calculates the integral-elements and runs the Hartree-Fock algorithm and returns the estimation of the ground-state energy.

6.2.1   Parallelization of Two-Body Matrix

The most time-consuming part of the Hartree-Fock procedure is the calculation of the two-body matrix-elements giving the interaction terms. This is parallelized in the `assemble` function in `HartreeFockSolver`.

The basic premise is to represent the N^4 elements $\langle pq \,|\, r^{-1} \,|\, rs \rangle$ as a one-dimensional array with the mapping

$$(p, q, r, s) \rightarrow p + N(q + N(r + Ns)) \tag{6.2}$$

that is the element (p, q, r, s) is stored in position $(p + N(q + N(r + Ns)))$ in the one-dimensional array. The symmetry $(p, q, r, s) = (q, p, r, s)$ reduces the number of elements down to $N(N + 1)/2$. A $N(N + 1)/2 \times 2$ matrix of indices is then created with

---

Algorithm 3 Parallelization Index Mapping
___

Initialize matrix pqMap with $\frac{N}{2}(N + 1)$ rows and 2 columns.
j = 0
for p=0 : N-1 **do**
    for q=p : N-1 **do**
        pqMap[j,0] = p
        pqMap[j,1] = q
        j = j + 1
    end for
end for

The rows are then distributed evenly among P processes according to

$$\text{rows} = \begin{cases} \left\lceil \dfrac{\frac{N}{2}(N + 1)}{P} \right\rceil & \text{rank} < \left(\dfrac{N}{2}(N + 1) \mod P \right) \\[3ex] \left\lfloor \dfrac{\frac{N}{2}(N + 1)}{P} \right\rfloor & \text{else} \end{cases} \tag{6.3}$$

The problem now however is that processes of higher and higher ranks my end up with calculating more since larger indices involve computationally more heavy functions to be evaluated. We can account for this by weighting the number of rows each process gets by the sum of the indices. The algorithm is as follows

---

Algorithm 4 Even Weighting
___

Make array S of size P with each element being the sum of the (p, q) elements for the specific process.
Make an array sizes of size P with the number of elements for each process.
Make an array displ of size P with the displacement (index).
O = Mean of S.
for $i = 0$ to $P - 2$ **do**
    for j =displ[i] to displ[i]+sizes[$i + 1$] **do**   ▷ Iterate over elements in right adjacent process.
        $L = S[i]$+sum(pqMap.row(j))
        if $L < O$ **then**
            sizes[i] = sizes[i] + 1
            sizes[$i + 1$] = sizes[$i + 1$]−1;
            displ[$i + 1$] = displ[$i + 1$]+1;
        else if L equals $O − 1$ or L equals $O + 1$ or $L > O$ **then**
            Break inner-most loop.
        end if
    end for
    Set elements of S.
end for

Each process has access to the `pqMap` array such that the each process calculates its own chunk according to the size set and the `displ` tells the start-index. Each process then calculates the two-body elements according to

---

Algorithm 5 Two-Body Calculation

Initialize A for containing (p, q, r, s) elements.
Initialize B for containing (p, q, s, r) elements.
$i_{\text{start}} = \text{displ[rank]}$
$i_{\text{end}} = \text{displ[rank]} + \text{sizes[rank]}$
$k = 0$
for $l = i_{\text{start}}$ to $i_{\text{end}} - 1$ **do**
    for $r = 0$ to $N - 1$ **do**
        for $s = r$ to $N - 1$ **do**
            $p = \text{pqMap}[l, 0]$
            $q = \text{pqMap}[l, 1]$
            $A[l] = \left\langle pq \left| \frac{1}{r_{12}} \right| rs \right\rangle$
            $B[l] = \left\langle pq \left| \frac{1}{r_{12}} \right| sr \right\rangle$
        end for
    end for
end for

Each A and B is then sent to root process and concatenated to a large one-dimensional array and the actual two-body matrix of size N^4 is assembled and antisymmetrized. The Hartree-Fock algorithm is then run only on one process.

6.3   Variational Monte Carlo

The Variational Monte-Carlo implementation is mainly in three classes, namely VMC, BruteForce and ImportanceSampling. The structure is set such that both BruteForce and ImportanceSampling inherit VMC. This structure essentially gives room for splitting specific part of the Brute-Force algorithm from the Metropolis-Hastings algorithm, but still using the same code for minimization. We will explain the minimization parts in the next section.

The main input which VMC needs is a wavefunction. An abstract class-template is implemented and can be generated using a python script, also given in the same GitHub repository »REF GITHUB«. The template is built in such a way that one only needs to fill in specific analytic expressions for the gradient, Laplacian and gradient for the variational parameters. The latter part is optional.

The wavefunction itself is built using the Slater or SlaterJastrow class. However, in order to use the SlaterJastrow one has to specify which Jastrow function to use at compile time. A simple example illustrates this better

```
SlaterJastrow* wf = new SlaterJastrow(dim, numParticles, parameters);
wf->initializeParameters(omega);
wf->initializeMatrices();

ImportanceSampling<SlaterJastrow>* vmc = new
        ImportanceSampling<SlaterJastrow>(wf, stepmc, parameters,
                                          maxIterations, rank, numProcs);

double E = vmc->sampler();

delete vmc;
delete wf;
```

The first chunk initializes the SlaterJastrow class with the pre-specified system which needs to be given at compile time as a flag i.e WAVEFUNCTION=HarmonicOscillator. The second part sets the sampling to use Metropolis-Hastings algortithm. The parameters variable must be an Eigen[15] vector or array and contains the variational parameters present. The representation of each element in this vector(or array) is specific to each system. For a more detailed explanation see » REF GITHUB «.

This is just an example of how to build a simple run, however we have also built a run file which uses YAML[7]. The basic template is as follows

```
omega: 1.0
numparticles: 6
maxitermc: 100000
stepmc: 0.01
parameters: [0.99, 0.47]  #optional
numparameters: 2
```

```
jastrow: true #optional
importance: true #optional
```

This is a template used for the `HarmonicOscillator` with the Padé-function as Jastrow factor. If the NQS-function is used an additional parameter 'numhiddenbias' giving the number of hidden biases used, is needed. Again, for more detail see » REF GITHUB «

This form of input makes it fairly simple to actually run the code for different systems with ease. One needs to compile with the specific flag for the system(wavefunction and Jastrow) and then supply an YAML input file at runtime.

6.3.1 Slater Optimizations

In the Metropolis sampling we need access to the ratio of two determinants, namely the Slater wavefunction at the current state and the one at previous state. These are quite expensive to calculate[2]. This can be overcome by using the fact that moving only *one* particle at each iteration also constitutes to a state-transition. Following [21] and given row i as the index for the row that is changed, the following expressions are valid

$$\frac{\Psi}{\Psi'} = \sum_j \psi_{ij} \psi_{ji}'^{-1}$$

$$\frac{\nabla_i \Psi}{\Psi} = \sum_j \psi_{ij} \nabla_i \psi_{ji}^{-1} \qquad (6.4)$$

$$\frac{\nabla_i^2 \Psi}{\Psi} = \sum_j \psi_{ij} \nabla_i^2 \psi_{ji}^{-1}$$

with Ψ' being the wavefunction at previous state. One might ask now, but isn't this just worse? We have gone from needing two determinants to needing two inverses!? Fret not, the Sherman-Morrison[34] formula for updating an inverse matrix if only one row has changed in the original matrix comes to the rescue. The elements of the inverse can by this be expressed as(with i being the row that changed)

$$\psi_{kj}^{-1} = \begin{cases} \psi_{kj}'^{-1} - \dfrac{\Psi}{\Psi'} \psi_{ki}'^{-1} \displaystyle\sum_{l=1}^{N} \psi_{il} \psi_{lj}'^{-1}, & j \neq i \\[2em] \dfrac{\Psi}{\Psi'} \psi_{ki}'^{-1} \psi_{ki}'^{-1} \displaystyle\sum_{l=1}^{N} \psi_{il}' \psi_{lj}'^{-1}, & j = i \end{cases} \qquad (6.5)$$

This means that the inverses and the determinant ratios can be calculated fully once before the sampling and then updated using the above formulas. This procedure is implemented in the `Slater` class.

6.3.2 Jastrow Optimizations

The Jastrow factors presented in sections 3.7.7, 3.7.7 and 3.7.7 also give rise to optimizations in the case of moving only one particle at a time. In particular, the Padeé function and the simple exponential can both be represented by a matrix of size $N \times D$ as

$$\boldsymbol{J} = \big(\boldsymbol{J}(\boldsymbol{r}_1) \dots \boldsymbol{J}(\boldsymbol{r}_N)\big) \qquad (6.6)$$

Moving one particle at a time means only one column in $\boldsymbol{J}$ is changed meaning only that column needs to be updated. Additionally, the $\boldsymbol{J}$ matrix doesn't actually need to be present in the code since we are only interested in ratios J/J'. If only one index i changes we have

$$\frac{J}{J'} = \exp\left(\sum_{i<j} \big(f_{ij} - f_{ij}'\big)\right)$$

$$= \exp\left(\sum_{i \neq j} f_{ij}\right) \qquad (6.7)$$

---

[2]Actual complexity of a determinant is $n \times n$, with n being the size of the Slater matrix.

Notice that i is fixed in the last step.

The gradient of J can be represented as a $N \times N \times D$ matrix

$$\nabla \boldsymbol{J} = \begin{pmatrix} 0 & \nabla \boldsymbol{J}(\boldsymbol{r}_{1,2}) & \dots & \nabla \boldsymbol{J}(\boldsymbol{r}_{1,N-1}) \\ \nabla \boldsymbol{J}(\boldsymbol{r}_{2,1}) & 0 & \dots & \nabla \boldsymbol{J}(\boldsymbol{r}_{2,N-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \nabla \boldsymbol{J}(\boldsymbol{r}_{N,1}) & \dots & \dots & 0 \end{pmatrix} \tag{6.8}$$

When only one particle is moved at a time only one row and column changes in this matrix. In addition the matrix is symmetric with the expect a negative sign,

$$\nabla \boldsymbol{J}(\boldsymbol{r}_{ij}) = -\nabla \boldsymbol{J}(\boldsymbol{r}_{ji}) \tag{6.9}$$

Both of these optimizations are implemented in the `PadeJastrow` and `ExpNQS` classes.

For the NQS-Jastrow we notice that the sum in the exponential involving the weights can be represented as a matrix $\boldsymbol{W}$ of size $N \times H$ with H being the number of hidden biases with elements

$$W_{i,j} = \sum_d \frac{x_i^{(d)} w_{i+d,j}}{\sigma^2} \tag{6.10}$$

and the entire exponential with the hidden biases is represented as a vector $\boldsymbol{B}$ of size H with elements

$$B_j = \exp\left(b_j + \sum_i W_{i,j}\right) \tag{6.11}$$

So for each iteration only one row in $\boldsymbol{W}$ is updated and then the entire $\boldsymbol{B}$ vector is recalculated and reused. The part involving only the visible biases can be optimized in the same manner as with the Padeé function and simple exponential, meaning one only needs to calculate

$$\frac{J_a}{J_a'} = \exp\left(\frac{(r_i - a_i)^2}{\sigma^2}\right) \tag{6.12}$$

Again with i being the index of the moved particle.

6.3.3 Optimization For Tabulation

The `Slater` class checks on compile time the specific wavefunction class for the functions `set`, `reSetAll`, `initializeMatrices`, `update`, `reset`, `resetGradient`, `acceptState` and `acceptGradient`. If these functions are implemented in the wavefunction class they will be called in by the `VMC` class during the Metropolis sampling. The whole purpose for this is so that calculations can be tabulated and then only the parts which are changed be updated. Detailed explanation of what the functions can, and need to, do is given in the GitHub repository.

6.3.4 Tabulating Hermite Polynomials

In the `HarmonicOscillator` and `HartreeFock` classes used by `VMC` we calculate the Hermite polynomials

$$H_n(\sqrt{\omega} r) = \prod_d H_{n_d}\left(\sqrt{\omega} x_d\right)$$

$$H_n(\sqrt{\omega} r) = \prod_d H_{n_d - 1}\left(\sqrt{\omega} x_d\right) \tag{6.13}$$

$$H_n(\sqrt{\omega} r) = \prod_d H_{n_d - 2}\left(\sqrt{\omega} x_d\right)$$

for $n = 0, \dots L$ where L is the number of basis functions in the Hartree-Fock basis for `HartreeFock` and highest order of single-particle function for `HarmonicOscillator`. These are tabulated in a matrix of size $N \times D \times L$ with N being the number of particles and D. The mapping goes by

$$\boldsymbol{H}_{ijk} = H_{n_j^{(d)}}\left(\sqrt{\omega} x_i^{(d)}\right) \tag{6.14}$$

For each iteration in the Metropolis sampling, if we only move one particle at a time, only one row changes in $\boldsymbol{H}_{ijk}$. This update is reflected in the functions mentioned in section 6.3.3. `initializeMatrices` firstly allocates space for two matrices which represent the current and old versions of $\boldsymbol{H}_{ijk}$, the `set` function set calculates $\boldsymbol{H}_{ijk}$ using the auto-generated header explained in section 6.5, `update` then updates the row which has changed and then `acceptState` or `reset` is called depending on the Metropolis-Test.

6.4 Minimization

The central part of the Variational Monte-Carlo method is the actual *variation* of parameters. This is, as explained in chapter 3, a minimization problem. We will in this chapter explain the implementation of the methods presented in chapter 5 and also show some examples of how to use `Minimizer` class, in which the mentioned methods are implemented.

6.5 Auto-Generation

The analytic expressions involved in the quantum-dot systems are dependent on Hermite-Polynomials and their coefficients. These are calculated symbolically using the recurrence relation for Hermites with the SymPy package in python. These expressions are then written to C++ code and written to a C++ -header file. The script can be found in https://github.com/Oo1Insane1oO/Hermite. The generated header file is then included in the integral class in `HartreeFockSolver` and in the wavefunction classes(namely `HartreeFock` and `HarmonicOscillator`) in `VMC`.

6.6 Verification

» tables

Appendix A

A.1 Lagrange Multiplier Method

See [1, 4]. The optimization method of Lagrange multipliers maximizes(or minimizes) a function $f : \mathbb{R}^N \to \mathbb{R}$ with a constraint $g : \mathbb{R}^N \to \mathbb{R}$ We assume that f and g have continuous first derivatives in all variables(continuous first partial derivatives).

Given the above we can define a so-called Lagrangian $\mathcal{L}$

$$\mathcal{L}[x_1, \ldots, x_N, \lambda_1, \ldots, \lambda_M] = f(x_1, \ldots, x_N) - \lambda g(x_1, \ldots, x_N) \tag{A.1}$$

where the λ is called a Lagrange-multiplier. We now state that if $f(x_1^0, \ldots, x_N^0)$ is a maxima of $f(x_1, \ldots, x_N)$ then there exists a Lagrange-multiplier λ_0 such that $(x_1^0, \ldots, x_N^0, \lambda_0)$ is a stationary point for the Lagrangian. This then yields the $N + 1$ Lagrange-equations

$$\sum_{i=1}^{N} \frac{\partial \mathcal{L}}{\partial x_i} + \frac{\partial \mathcal{L}}{\partial \lambda} = 0 \tag{A.2}$$

to be solved for $x_1, \ldots, x_N$ and λ.

A.2 Interaction-Term in Fock-Operator

Introducing the so-called permutation operator P which interchanges the labels of particles meaning we can define

$$A \equiv \frac{1}{N!} \sum_p (-1)^p P \tag{A.3}$$

the so-called *antisymmetrization* operator. This operator has the following traits

- The Hamiltonian H and A commute since the Hamiltonian is invariant under permutation.

- A applied on itself (that is A^2) is equal to itself since permuting a permuted state reproduces the state.

We can now express our Slater Ψ_T in terms of A as

$$\Psi_T = \sqrt{N!} A \prod_{i,j} \psi_{ij} \tag{A.4}$$

where $\psi_{ij} = \psi_j(\boldsymbol{r}_i)$ is element i, j of the Slater matrix (the matrix associated with the Slater determinant Ψ_T).

The interaction part of H is then

$$\langle \Psi_T | H_I | \Psi_T \rangle = N! \prod_{i,j} \langle \psi_{ij} | A H_I A | \psi_{ij} \rangle \tag{A.5}$$

The interaction H_I and A commute since A commutes with H giving

$$A H_I A | \psi_{ij} \rangle = \frac{1}{N!^2} \sum_{i<j} \sum_p (-1)^{2p} f_{ij} P | \psi_{ij} \rangle \tag{A.6}$$

$$= \frac{1}{N!^2} \sum_{i<j} f_{ij} (1 - P_{ij}) | \psi_{ij} \rangle \tag{A.7}$$

The factor $1 - P_{ij}$ comes from the fact that contributions with $i \neq j$ vanishes due to orthogonality when P is applied. The final expression for the interaction term is thus

$$\langle \Psi_T | H_I | \Psi_T \rangle = \sum_{i<j} \prod_{k,l} \left[\langle \psi_{kl} | f_{ij} | \psi_{kl} \rangle - \langle \psi_{kl} | f_{ij} | \psi_{lk} \rangle \right] \tag{A.8}$$

Writing out the product and realizing the double summation over pairs of states we end up with

$$\langle \Psi_T | H_I | \Psi_T \rangle = \frac{1}{2} \sum_{i,j} \left[\langle \psi_{ij} \psi_{ji} | f_{ij} | \psi_{ij} \psi_{ji} \rangle - \langle \psi_{ij} \psi_{ji} | f_{ij} | \psi_{ji} \psi_{ij} \rangle \right] \tag{A.9}$$

More comprehensive details and derivations are given in [20, 38].

A.3   Multi-Index Notation

» REF THIS « This section will give a brief overlook of a notation which compresses indices running in similar fashion, the so-called *multi-index notation*. We will make use of this to reduce indices in each dimension down to one.

The rules are states as, given a n-tuple $(x_1, \ldots x_n)$ over any field $\mathbb{F}$ (real, complex, etc.), a multi index is defined to be

$$i = (n_1, \ldots i_n) \in \mathbb{Z}_+^n \tag{A.10}$$

with expansions

$$\begin{aligned} |i| &= i_1 + \cdots + i_n \\ i! &= i_1! \ldots i_n! \\ x^i &= x_1^{i_1} \ldots x_n^{i_n} \in \mathbb{F}[x] \\ i \pm j &= (i_1 \pm j_1, \ldots, i_n + j_n) \in \mathbb{Z} \end{aligned} \tag{A.11}$$

In essence the notation just wraps the notion of element-wise operations into one index variable.

Appendix B

B.1 Derivative of Energy and Variance

The derivative with respect to a general variational parameter α of the energy expectation value $\langle E \rangle$ is

$$\frac{\partial \langle E \rangle}{\partial \alpha} = 2 \left\langle \frac{E_L}{\psi} \frac{\partial \psi}{\partial \alpha} \right\rangle + \left\langle \frac{\partial E_L}{\partial \alpha} \right\rangle - 2 \left\langle \frac{\langle E \rangle}{\psi} \frac{\partial \psi}{\partial \alpha} \right\rangle$$
$$= 2 \left(\left\langle \frac{E_L}{\psi} \frac{\partial \psi}{\partial \alpha} \right\rangle - \langle E \rangle \left\langle \frac{1}{\psi} \frac{\partial \psi}{\partial \alpha} \right\rangle \right) \tag{B.1}$$

by using the hermiticity of the Hamiltonian (contained in the local energy E_L).
The derivative of the variance is » REF THIS STARTER «

$$\frac{\partial \sigma^2}{\partial \alpha} = 2 \left\langle \left(\frac{\partial E_L}{\partial \alpha} - \frac{\partial \langle E \rangle}{\partial \alpha} \right) (E_L - \langle E \rangle) \right\rangle$$
$$= 2 \left(\left\langle E_L \frac{\partial E_L}{\partial \alpha} \right\rangle - \langle E \rangle \left\langle \frac{\partial E_L}{\partial \alpha} \right\rangle + \left\langle \frac{\partial \langle E \rangle}{\partial \alpha} (\langle E \rangle - E_L) \right\rangle \right)$$
$$= 2 \langle E \rangle \left(\frac{\partial \langle E \rangle}{\partial \alpha} - 1 \right) \tag{B.2}$$

B.2 Derivatives of Hermite Functions

The gradient of Hermite functions on the form

$$\psi_n(r) = \prod_d \psi_{n_d}(x_d) = \prod_d H_{n_d}(\sqrt{\omega} x_d) e^{-\frac{\omega}{2} x_d^2} \tag{B.3}$$

is

$$\nabla \psi_n(r) = \sum_d \hat{e} \prod_{d' \neq d} \psi_{n_{d'}} \frac{\partial \psi_{n_d}}{\partial x_d}$$
$$= \sum_d \hat{e} \prod_{d' \neq d} \psi_{n_{d'}} \psi_{n_d} \sqrt{\omega} \left(\frac{\partial H_{n_d}}{\partial u} \frac{1}{H_{n_d}} - x_d \right)$$
$$= \psi_n \sqrt{\omega} \sum_d \hat{e} \left(2n_d \frac{H_{n_d-1}(\sqrt{\omega} x_d)}{H_{n_d}(\sqrt{\omega} x_d)} - x_d \right) \tag{B.4}$$

and the Laplacian follows

$$\nabla^2 \psi_n(r) = \sum_d \prod_{d' \neq d} \psi_{n_{d'}} \frac{\partial^2 \psi_{n_d}}{\partial x_d^2}$$
$$= \psi_n \omega \sum_d \left(\frac{(4n_d(n_d-1)H_{n_d-2}(\sqrt{\omega} x_d) - \sqrt{\omega} x_d H_{n_d-1}(\sqrt{\omega} x_d))}{H_{n_d}(\sqrt{\omega} x_d)} + \omega x_d^2 - 1 \right) \tag{B.5}$$

The derivative with respect to the variational parameter α is

$$
\begin{aligned}
\frac{\partial \psi_n}{\partial \alpha} &= \frac{\partial}{\partial \alpha} \prod_d H_{n_d}(\sqrt{\omega} x_d) \mathrm{e}^{-\frac{\omega}{2} x_d^2} \\
&= \sum_d \prod_{d' \neq d} \psi_{n_{d'}}(x_{d'}) \frac{\partial \psi_{n_d}(x_d)}{\partial \alpha} \\
&= \psi_n(r) \sum_d \omega x_d \left(\frac{n_d}{\sqrt{\alpha\omega}} \frac{H_{n_d-1}\left(\sqrt{\alpha\omega} x_d\right)}{H_{n_d}\left(\sqrt{\alpha\omega} x_d\right)} - \frac{\omega x_d^2}{2} \right) \\
&= \psi_n(r) \left(\sqrt{\frac{\omega}{\alpha}} \sum_d x_d n_d \frac{H_{n_d-1}\left(\sqrt{\alpha\omega} x_d\right)}{H_{n_d}\left(\sqrt{\alpha\omega} x_d\right)} - \frac{\omega}{2} r^2 \right)
\end{aligned}
\tag{B.6}
$$

B.3   Derivatives of Padé-Jastrow Function

Given the Padé-Jastrow function

$$
J = \exp\left(\sum_{i<j} f_{ij} \right), \qquad f_{ij} = \frac{a_{ij} r_{ij}}{1 + \beta r_{ij}}
\tag{B.7}
$$

The general expression for the gradient and Laplacian with respect to particle position k is

$$
\begin{aligned}
\nabla_k J &= J \sum_{j \neq k} \frac{\boldsymbol{r}_{kj}}{r_{kj}} \frac{\partial f_{kj}}{\partial r_{kj}} \\
\nabla_k^2 J &= \frac{(\nabla_k J)^2}{J} + J \sum_{j \neq k} \left(\frac{\partial f_{kj}}{\partial r_{kj}} \frac{D-1}{r_{kj}} + \frac{\partial^2 f_{kj}}{\partial r_{kj}{}^2} \right)
\end{aligned}
\tag{B.8}
$$

Notice that the sum with $j \neq k$ is only a sum over j with k fixed. The derivatives of f are

$$
\begin{aligned}
\frac{\partial f_{kj}}{\partial r_{kj}} &= \frac{a_{kj} r_{kj}^2}{(1 + \beta r_{kj})^2} \\
\frac{\partial^2 f_{kj}}{\partial r_{kj}{}^2} &= -\frac{2 a_{kj} \beta}{(1 + \beta r_{kj})^3}
\end{aligned}
\tag{B.9}
$$

And the derivative with respect to the variational parameter β

$$
\frac{\partial f_{kj}}{\partial \beta} = -\frac{a_{kj} r_{kj}^2}{(1 + \beta r_{kj})^2}
\tag{B.10}
$$

B.4   Derivatives of NQS-Wavefunction

Given the NQS-Wavefunction

$$
J = \exp\left(-\sum_{i=1}^N \frac{(\boldsymbol{r}_i - \boldsymbol{a}_i)^2}{2\sigma^2} \right) \prod_j^M (1 + E_j)
\tag{B.11}
$$

The gradient is

$$
\begin{aligned}
\frac{\nabla_k J}{J} &= \nabla_k \ln(J) \\
&= -\frac{\boldsymbol{r}_k - \boldsymbol{a}_k}{\sigma^2} + \sum_{j=0}^M \sum_{d=1}^D \hat{\boldsymbol{e}}_d \frac{w_{k+d,j}}{\sigma^2 \left(1 + \frac{1}{E_j}\right)}
\end{aligned}
\tag{B.12}
$$

and the Laplacian

$$
\begin{aligned}
\frac{\nabla_k^2 J}{J} &= \nabla_k^2 \ln(J) \\
&= \frac{(\boldsymbol{r}_k - \boldsymbol{a}_k)^2}{\sigma^4} - \frac{D}{\sigma^2} + \left(\sum_{j=1,d=1}^{M,D} \hat{\boldsymbol{e}}_d \frac{w_{k+d,j}}{\sigma^2 (1 + E_j)} \right)^2 + \sum_{j=1,d=1}^{M,D} \frac{w_{k+d,j}^2 E_j}{\sigma^4 (1 + E_j)^2}
\end{aligned}
\tag{B.13}
$$

with

$$W_j \equiv \sum_{i=1}^{N} \sum_{d=1}^{D} \frac{x_i^{(d)} w_{i+d,j}}{\sigma^2} \tag{B.14}$$

$$E_j \equiv \exp(b_j + W_j)$$

The derivative with respect to the visible bias is

$$\frac{1}{J} \frac{\partial J}{\partial a_l} = \frac{x_l - a_l}{\sigma^2} \tag{B.15}$$

and the hidden bias

$$\frac{1}{J} \frac{\partial J}{\partial b_l} = \frac{\partial}{\partial b_l} \ln(J)$$

$$= \frac{1}{1 + \frac{1}{E_j}} \tag{B.16}$$

and the weights

$$\frac{1}{J} \frac{\partial}{\partial w_{kl}} = \frac{\partial}{\partial w_{kl}} \ln(J)$$

$$= \frac{x_k}{\sigma^2 \left(1 + \frac{1}{E_j}\right)} \tag{B.17}$$

Notice that the indices k and l are here indiscriminate towards the dimensional component in relation to x. This means that x_l or x_k represents just a particle position in a dimension.

Bibliography

[1] Dacorogna B. *Introduction to the calculus of variations*. Imperial College Press; World Scientific, 2004. ISBN: 9781860945083.

[2] M. L. Boas. *Mathematical Methods in the Physical Sciences, 3rd Edition*. Wiley, 2005. ISBN: 978-0-471-19826-0.

[3] C. G. Broyden. "The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations". In: *IMA Journal of Applied Mathematics* (1970). DOI: 10.1093/imamat/6.1.76.

[4] van Brunt B. *The Calculus of Variations*. Springer, 2004. ISBN: 9780387402475.

[5] G. Carleo and M. Troyer. "Solving the Quantum Many-Body Problem with Artificial Neural Networks". In: *Science* 355.6325 (2017), pp. 602–606. ISSN: 0036-8075. DOI: 10.1126/science.aag2302.

[6] M. S. Carroll. "Implications Of Simultaneous Requirements For Low-Noise Exchange Gates In Double Quantum Dots". In: *Phys. Rev. B* (2010). DOI: 10.1103/PhysRevB.82.075319.

[7] YAML developers. *YAML*. http://yaml.org/.

[8] W. H. Dickhoff and D. Van Neck. *Many-body theory exposed!: Propagator Description of Quantum Mechanics in Many-Body Systems*. World Scientific, 2005. ISBN: 981256294X.

[9] N. D. Drummond, M. D. Towler, and R. J. Needs. "Jastrow Correlation Factor for Atoms, Molecules, and Solids". In: *Phys. Rev. B* 70 (2004), p. 235119. DOI: 10.1103/PhysRevB.70.235119.

[10] A. L. Fetter and J. D. Walecka. *Quantum Theory of Many-particle Systems*. Mcgraw-Hill College, 1971. ISBN: 9780070206533.

[11] R. Fletcher. "A New Approach To Variable Metric Algorithms". In: *The Computer Journal* (1970). DOI: 10.1093/comjnl/13.3.317.

[12] D. Goldfarb. "A Family of Variable-Metric Methods Derived by Variational Means". In: *Math. Comp.* 24 (1970). DOI: \url{https://doi.org/10.1090/S0025-5718-1970-0258249-6}.

[13] D. Goldfarb. "Stochastic Adaptive Quasi-Newton Methods for Minimizing Expected Values". In: *Proceedings of the 34th International Conference on Machine Learning*. Proceedings of Machine Learning Research. PMLR, 2017.

[14] D.J. Griffiths. *Introduction to quantum mechanics*. 2nd Edition. Pearson PH, 2005. ISBN: 0131911759.

[15] Gaël Guennebaud, Benoît Jacob, et al. *Eigen v3*. http://eigen.tuxfamily.org. 2010.

[16] W. A. Hammond B. L. Lester Jr. and P.J. Reynolds. *Monte Carlo Methods in Ab Initio Quantum Chemistry*. World Scientific, 1994. ISBN: 9810203225.

[17] T. Helgaker. "Molecular Integral Evaluation". In: (2006).

[18] T. Helgaker and P. R. Taylor. "Gaussian Basis Sets and Molecular Integrals". In: (1995).

[19] G. Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*. https://www.cs.toronto.edu/~hinton/absps/guideTR.pdf.

[20] M. Hjort-Jensen. *Computational Physics: Hartree-Fock methods and introduction to Many-Body Theory*. https://www.github.com/CompPhysics/ComputationalPhysics2/blob/gh-pages/doc/pub/basicMB/pdf. 2017.

[21] M. Hjort-Jensen. *Computational Physics: Variational Monte Carlo methods*. https://www.github.com/CompPhysics/ComputationalPhysics2/blob/gh-pages/doc/pub/vmc/pdf. 2017.

[22] Høgberget J. *Quantum Monte-Carlo Studies of Generalized Many-body Systems*. MA Thesis. University of Oslo, 2013.

[23] R. Jastrow. "Many-Body Problem with Strong Forces". In: *Phys. Rev.* (1955). DOI: 10.1103/PhysRev.98.1479.

[24] A. Jeffrey, ed. *Handbook of Mathematical Formulas and Integrals (Third Edition)*. Academic Press, 2004. ISBN: 978-0-12-382256-7. DOI: https://doi.org/10.1016/B978-012382256-7/50034-8.

[25] D. C. Lay. *Linear Algebra and Its Applications (4th Edition)*. Addison Wesley / Pearson, 2011. ISBN: 9780321385178.

[26] D. S. Lemons and A. Gythiel. "Paul Langevin's 1908 paper "On the Theory of Brownian Motion" ["Sur la théorie du mouvement brownien," C. R. Acad. Sci. (Paris) 146, 530–533 (1908)]". In: *American Journal of Physics* 65 (1997), pp. 1079–1081. DOI: 10.1119/1.18725.

[27] L. E. Lervåg. *VMC Calculations of Two-Dimensional Quantum Dots*. MA Thesis. University of Oslo, 2010.

[28] J. J. Moré and D. J. Thuente. "Line Search Algorithms with Guaranteed Sufficient Decrease". In: *ACM Trans. Math. Softw.* (1994). DOI: 10.1145/192115.192132.

[29] J. W. Moskowitz and M. H. Kalos. "A New Look At Correlations In Atomic And Molecular Systems. I. Application Of Fermion Monte Carlo Variational Method". In: *International Journal of Quantum Chemistry* (). DOI: 10.1002/qua.560200508.

[30] J. W. Negele and Orland H. *Quantum Many-Particle Systems*. Perseus BOOKs, 1998. ISBN: 9780738200521.

[31] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag New York, 2006. ISBN: 978-1-4939-3711-0. DOI: 10.1007/978-0-387-40065-5.

[32] F. Pederiva. "Variational Monte Carlo for Spin-orbit Interacting Systems". In: *Phys. Rev. B* (2012). DOI: 10.1103/PhysRevB.85.045115.

[33] E. Platen and N. Bruti-Liberati. *Numerical Solution of Stochastic Differential Equations with Jumps in Finance*. Springer-Verlag Berlin Heidelberg, 2010. ISBN: 9783642120572.

[34] Jack S. and M. Winifred. "Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix". In: *The Annals of Mathematical Statistics* 21 (1950). DOI: 10.1214/aoms/1177729893.

[35] J. P. Sethna. *Statistical Mechanics: Entropy, Order parameters and complexity*. Oxford University Press, USA, 2006. ISBN: 9780198566779.

[36] D. F. Shanno. "Conditioning of quasi-Newton methods for function minimization". In: *Math. Comp.* 0274029 (1970). DOI: 10.2307/2004840.

[37] J. Olsen T. Helgaker P. Jørgensen. *Molecular Electronic-Structure Theory*. Wiley, 2014. ISBN: 978-0-47-196755-2. DOI: 10.1002/9781119019572.

[38] J. M. Thijssen. *Computational Physics*. 2nd Edition. Cambridge University Press, 2013. ISBN: 9781107677135.

[39] M. A. Viergever. "Adaptive Stochastic Gradient Descent Optimisation for Image Registration". In: *Int J Comput. Vision* (2009). DOI: 10.1007/s11263-008-0168-y.

[40] Wolfram|Alpha. *Chebyshev-Gauss Quadrature*. http://mathworld.wolfram.com/Chebyshev-GaussQuadrature.html. 2018.

[41] Wolfram|Alpha. *Modified Bessel Function of the First Kind*. http://mathworld.wolfram.com/ModifiedBesselFunctionoftheFirstKind.html. 2018.