Many-body ^{4}He boson with Restricted Boltzmann Machine

University of Oslo

Kari Eriksen

Desember 2019

Abstract

Abstract goes here

Contents

Part I

Theory

Chapter 1

Many-body quantum theory

Some introduction containing historical information and stuff, and motivation for this section.

1.1 Quantum Mechanics

In ordinary quantum mechanics an observable is a linear operator acting on a Hilbert space. Position operator and momentum operator, other observable quantities like angular momentum, energy, and so on, are linear operators constructed out of linear combinations of products of the position and momentum.[1]

1.2 Canonical quantization

Moving from a classical Hamiltonian description to a non-relativistic quantum mechanical description of the particle through a procedure called canonical quantization.(ref. Simen Kvaal)

Canonical communtation relations

$$\{q_k, p_l\} = \delta_{kl} \tag{1.1}$$

Any coordinate system $\epsilon = (q, p)$ we wish to use that satisfies 1.1 can be called canonical. (ref. Simen Kvaal)

Classical state: point in phase space

Quantum state: complex-valued wave function in an infinite dimensional Hilbert space, i.e. a complete vector space with an inner product. Standard choice of Hilbert space is the space of square integrable function, L^2, where the wave function ψ is a function depending on all coordinates:

$$\psi = \psi(x_1, x_2, \cdots, x_N) \tag{1.2}$$

where $(x_1, x_2, \cdots, x_N) \in X^N$ is a point in the configuration space of N particles.

The wave function is a map

$$\psi : X^N \longrightarrow \mathbb{C} \tag{1.3}$$

An observable is a *self-adjoint operator* (Hermitian operator) $\hat{\Omega}$, and its value in the state ψ is the expectation value.

Expectation value:

$$\langle \Omega \rangle = \langle \psi | \hat{\Omega} | \psi \rangle = \int \psi(x_1, \cdots, x_N) * [\hat{\Omega}\psi(x_1, \cdots, x_N)] dx_1 \cdots dx_N \tag{1.4}$$

$$i\hbar \frac{\partial}{\partial t} \left| \psi(t) \right\rangle = \hat{H} \left| \psi(t) \right\rangle \tag{1.5}$$

Replace classically canonical coordinates $\epsilon = (q, p)$ with operators $(\hat{q}, \hat{p})$ such that

$$\{q_j, p_k\} = \delta_{jk} \Rightarrow [\hat{q_j}, \hat{p_k}] = i\hbar\delta_{jk} \tag{1.6}$$

1.3 Canonical transformation

A one-to-one algebra morphism of an observable, u, onto itself leaving the canonical commutation relations invariant. The canonical transformation is a map, $u \to u$

properties satisfying

-linearity -invariance -product conserving (s. 175 MBBS)

1.4 Second quantization

Now the particles themselves are discrete quanta created and destroyed with creation and annihilation operators

Position operator

$$[\hat{r}_i \psi](x_1, \cdots, x_N) = r_i \psi(x_1, \cdots, x_N) \tag{1.7}$$

Momentum operator

$$[\hat{p}_i \psi](x_1, \cdots, x_N) = -i\hbar \nabla_i \psi(x_1, \cdots, x_N) \tag{1.8}$$

Fock space

1.5 The Schrödinger equation

The many-body Hamiltonian (s. 12)

one-body operator

two-body operator

1.6 Wave function

Chapter 2

The Bose gas

Some introduction containing historical information and stuff, and motivation for this section.

2.1 Ideal Bose gas

Bosons are quantum particles with integer spin, meaning they follow Bose-Einstein statistics. Elementary: Higgs boson, photon, gluon, gravition

Composite: hydrogen, mesons, nucleus of deuterium

Consider a dilute Bose gas where the inter-particle distance is much greater than the scattering length. It can be approximated by a mean field theory. We describe it by a simple hard-sphere effective potential. (Master Joachim)

Bogoliubov -

Ground state energy

2.2 The Bose-Einstein Condensation

The ability of bosons at low temperature to collectively inhabit the same energy state, compared to fermions who are restricted to the Pauli exclusion principle.

Free Bose gas -

$$H_V^{free} = \int_V dx \frac{1}{2m} \nabla a*(x).\nabla a(x) \tag{2.1}$$

BEC in interaction -

BEC in trap -

The Bose-Einstein condensation ground state can be described by the Gross-Pitaevskii equation, which uses the Hartree-Fock approximation and the pseduopotential intereaction model. (wiki) A a many-body boson system, the Gross-Pitaevskii approach is a mean-field approach. (MBBS) If the spacing between the particles are greater than the scattering

length (in the so-called dilute limit), then one can approximate the true interaction potential that features in this equation by a psedopotential. READ more about why this is.

The wave function in the Hartree-Fock approximation of a system of N bosons is represented by the product of single-particle wave functions ψ

$$\psi(r_1, r_2, \cdots, r_N) = \psi(r_1)\psi(r_2)\cdots\psi(r_N) \tag{2.2}$$

The pseudopotential model Hamiltonian of the system is given as

$$H = \sum_{i=1}^{N}\left(-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial r_i^2} + V(r_i)\right) + \sum_{i<j}\frac{4\pi\hbar^2 a_s}{m}\delta(r_i - r_j) \tag{2.3}$$

m is the mass of the boson, V is the external potential, a_s is the boson-boson scattering length and $\delta(r)$ the Dirac delta-function. This is an effective potential, the original with modifications

The Gross-Pitaevskii equation reads,

$$\left(-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial r^2} + V(r) + \frac{4\pi\hbar^2 a_s}{m}|\psi(r)|^2\right)\psi(r) = \mu\psi(r) \tag{2.4}$$

Se 104 MBBS

Solutions: free particle V(r) = 0

Soliton?

Thomas-Fermi approximation

Bogoliubov approximation

Interacting Bose gas - Landau's phenomenological theory of superfluidity. Based on the idea that a quantum liquid remains a classical fluid even at zero temperature and that the classical hydrodynamical laws remain valid. (Many body boson system.) Many models have been suggested to create a microscopical theory of the superfluidity. Vortex ring model, hard-sphere model, guassian cluster approach. (wiki)

The properties of these liquids are described in terms of the spectrum of the collective excitations.

2.3 The wave function

We are studying a system of two electrons confined in a harmonic oscillator trap described by the Hamiltonian

$$\hat{H} = \sum_{i=1}^{N}\left(-\frac{1}{2}\nabla_i^2 + \frac{1}{2}\omega^2 r_i^2\right) + \sum_{i<j}\frac{1}{r_{ij}} \tag{2.5}$$

where the first sum is the standard harmonic oscillator part and the last is the interacting part between the electrons and N represent the number of particles. ω is the oscillator

frequency of the trap and r_i is the position of particle i, whereas r_{ij} is the distance between the particles and given as $r_{ij} = |\mathbf{r_i} - \mathbf{r_j}|$.
Our wave function is given from the energy of the restricted Boltzmann machine, see (4.1), which is the joint energy functional between the visible and hidden nodes. From the marginal probability of the joint probability distribution, see (4.4), we get our wave equation.

$$\Psi(X) = F_{rbm}(X) \tag{2.6}$$

$$= \frac{1}{Z} \exp\left(-\sum_i^M \frac{(X_i - a_i)^2}{2\sigma^2}\right) \prod_j^N \left(1 + \exp\left(b_j + \sum_i^M \frac{X_i \omega_{ij}}{\sigma^2}\right)\right) \tag{2.7}$$

Here Z is the partition function, X_i represents the visible nodes running up to M, and a_i and b_j are the biases described in the section below, (4.1), where number of hidden nodes j runs up to N. ω_{ij} is an $M \times N$ matrix holding the weights connecting the visible nodes with the hidden and σ is the standard deviation of the noise in our model.

2.4 Interaction

Chapter 3

VMC and DMC

Some introduction containing historical information and stuff, and motivation for this section.

3.1 Markov chains

3.2 Variational Monte Carlo

Monte Carlo simulations are widely used methods in numerical science that employs random walkers. In this project we are taking a closer look at trapped bosons. We are given a trail wave function we assume is as close to the real case as possible, $\Psi_T(\mathbf{R};\alpha)$ where $\mathbf{R} = (\mathbf{R}_1, ..., \mathbf{R}_N)$ is the position of the different particles. From quantum mechanics we know the probability distribution is given by the wave function.

$$P(\mathbf{R};\alpha) = \frac{|\Psi_T(\mathbf{R};\alpha)|^2}{\int |\Psi_T(\mathbf{R};\alpha)|^2 d\mathbf{R}}$$

Monte Carlo integration allow us to evaluate the integral at hand. The expectation value of the Hamiltonian is given as follows.

$$\langle \widehat{\mathbf{H}} \rangle = \frac{\int d\mathbf{R} \Psi^*(\mathbf{R}) H(\mathbf{R}) \Psi(\mathbf{R})}{\int d\mathbf{R} \Psi^*(\mathbf{R}) \Psi(\mathbf{R})}$$

The variational principle states that the expectation value of the Hamiltonian is an upper-bound for the ground state energy of the Hamiltonian.

$$E_0 \leq \langle H \rangle$$

This is what the Variational Monte Carlo method bases itself on. Given a probability distribution we can evaluate the wave function and look for a local minimum. We define the local energy by

$$\widehat{\mathbf{E}}_L(\mathbf{R};\alpha) = \frac{1}{\Psi_T(\mathbf{R};\alpha)}\widehat{\mathbf{H}}\Psi_T(\mathbf{R};\alpha).$$

Then the expectation value of the local energy is given by

$$\langle\widehat{\mathbf{E}}_L\rangle = \int P(\mathbf{R})\widehat{\mathbf{E}}_L d\mathbf{R} \approx \frac{1}{N}\sum_{i=1}^{N}\mathbf{E_L}(x_i)$$

where N is the number of Monte Carlo cycles. Now we can calculate the probability distribution and the local energy. And for each cycle we propose a new configuration $\mathbf{R}_p$ for the system and hopefully we come closer to the ground state.

$$\mathbf{R}_p = \mathbf{R} + r * \Delta\mathbf{R}$$

3.3 The Metropolis Algorithm

How we select the new configurations during the simulation is given by the Metropolis Algorithm. Ideally we would have a transition probability matrix which told us how likely it would be for each particle to move in the configuration space. Since we don't we try and model it. We define the following entities:

$P_i^{(n)} \rightarrow$ probability of finding the system in state i at the n'th step.
$j \rightarrow$ possible new step
$A_{i\rightarrow j} \rightarrow$ probability of acceptance.
$T_{j\rightarrow i} \rightarrow$ probability of making the transition

Now we can say that a transition probability matrix can be constructed by $T_{j\rightarrow i}A_{j\rightarrow i}$. The probability of finding the system in state i at step n is

$$P_i^{(n)} = \sum_j \left[P_j^{(n-1)}T_{j\rightarrow i}A_{j\rightarrow i} + P_j^{(n-1)}T_{i\rightarrow j}(1 - A_{i\rightarrow j})\right]. \tag{3.1}$$

We want to push the system towards high density regional space of $P_i^{(n)}$. /We want to select states according to the probability distribution. In that way to converge to the desired stationary distribution p_i.

$$P_i^{(n\rightarrow\infty)} \rightarrow p_i$$

Using this statement and the fact that the sum over all possible transitional probabilities is one we can rewrite the above equation 3.1.

$$\sum_j [p_j T_{j\rightarrow i}A_{j\rightarrow i} - p_i T_{i\rightarrow j}A_{i\rightarrow j}] = 0$$

In order to stop at the configuration where we have reached equilibrium we us the condition of detailed balance, and it gives us

Data: matrix $\mathbf{R}$
Result: float $\mathbf{E}_L$
number of MC-cycles N;
initialize $\mathbf{R}$;
calculate $|\Psi_T(\mathbf{R})|^2$;
for i *in* $[0, N]$ **do**
  $\mathbf{R}_p = \mathbf{R} + r * \Delta\mathbf{R}$;
  calculate $\omega = \frac{|\Psi_T(\mathbf{R}_p)|^2}{|\Psi_T(\mathbf{R})|^2}$;
  if $q \leq \omega$ **then**
    accept new move;
  else
    reject new move;
  end
  update energy, $\mathbf{E}_L$;
end

Algorithm 1: Monte Carlo with Metropolis-Hastings

r is a random number drawn from the distribution $r \in [0, \Lambda]$ with $\Lambda < \infty$

$$\frac{A_{j\to i}}{A_{i\to j}} = \frac{p_i T_{i\to j}}{p_j T_{j\to i}}.$$

We choose to accept when the acceptance is bigger than 1.

$$A_{j\to i} = \min\left(1, \frac{p_i T_{i\to j}}{p_j T_{j\to i}}\right)$$

What we must calculate is equation 3.2. And an example of the code is given further down, in Algorithm 1

$$\frac{p_i T_{i\to j}}{p_j T_{j\to i}} = \frac{|\Psi_T(\mathbf{R}_p)|^2}{|\Psi_T(\mathbf{R})|^2} \tag{3.2}$$

3.4 Importance Sampling

Now we move to an extension of the Metropolis algorithm, the Importance sampling. This method provides us with a better way of suggesting new moves. The expression for the new move, y, is given by the solution of the Langevin equation. It reads

$$y = x + DF(x)\Delta t + \xi\sqrt{\Delta t},$$

where ξ is a gaussian random variable, D is the diffusion coefficient and is $\frac{1}{2}$ and Δt is the time step which take values between $[0.001, 0.01]$. $F(x)$ is the drift force and is given by the gradient of the wave function.

$$F = 2\frac{1}{\Psi_T}\nabla\Psi_T$$

It is the drift force that ensures us we move particles towards regions of configuration space where the trail wave function is large. This method increases the efficiency of our program, since the standard Metropolis can suggest new moves in every direction with same probability. From the solution of the Fokker-Planck equation we get a new transition probability, the Green's function.

$$G(y, x, \Delta t) = \frac{1}{(4\pi D\Delta t)^{3N/2}} \exp\big(-(y - x - DF(x)\Delta t)^2/4D\Delta t\big)$$

Now equation 3.2 from Metropolis algorithm changes to

$$\frac{p_i T_{i\to j}}{p_j T_{j\to i}} = \frac{G(x, y, \Delta t)|\Psi_T(y)|^2}{G(y, x, \Delta t)|\Psi_T(x)|^2},$$

and we must calculate the drift force in order to find new moves and the Green's function to accept/reject these moves.

3.5 Gradient Descent

The variational quantum problem is highly dependent on a good optimization algorithm. One of the more famous and widely used is the gradient descent. It is a family of minimization methods where the idea is to iteratively adjust the parameters in the direction where the gradient of the cost function, usually an energy function, is large and negative [2]. If a multi-variable function $E(\theta)$ is defined and differentiable in a neighborhood of a point θ, then $E(\theta)$ decreases fastest if one goes from θ in the direction of the negative gradient of E at θ, $-\nabla E(\theta)$, (wiki). This way we move towards a local minimum in the configurational space. And the gradient descent can be represented as

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta E(\theta_t), \tag{3.3}$$

where η_t is called the step length/learning rate, it controls how large steps we take within the method. The easiest way to implement the learning rate is by giving it a constant value. One can however introduce a changing learning rate. This can help us avoid problems like overshooting or oscillating between two points. Such adaptive step size methods can be Newton's method, backtracking line search, Cauchy or Barzilai and Borwein. To be continued.

We begin by making a guess on the parameter θ, then we follow Eq. (3.8). The partial derivative of the energy with respect to the variational parameter can be found through the following equation.

$$\frac{dE_L}{d\alpha} = 2\left(\left\langle \frac{1}{\Psi_T}\frac{d\Psi_T}{d\alpha}E_L\right\rangle - \left\langle \frac{1}{\Psi_T}\frac{d\Psi_T}{d\alpha}\right\rangle \langle E_L\rangle\right) \tag{3.4}$$

Here we will need to calculate the expectation value of the local energy as well as the partial derivative of the wave function, and their product. The term $\frac{1}{\Psi_T}\frac{d\Psi_T}{d\alpha}$ is fairly easy given the wave function we are considering.

$$\frac{1}{\Psi_T}\frac{d\Psi_T}{d\alpha} = \frac{-\Psi_T\prod_i^N(x_i^2+y_i^2+\beta z_i^2)}{\Psi_T} \tag{3.5}$$

$$= -\prod_i^N(x_i^2+y_i^2+\beta z_i^2) \tag{3.6}$$

Since the correlation function $f(a, r_{ij})$ of the wave function is independent of the variational parameter α, Eq. (3.6) holds for the interactive case as well.

The last step of the algorithm is to check if the absolute value of the gradient squared is smaller or larger than some given small number ϵ. If larger then we continue the process, if smaller we consider our result good enough.

3.6 Stochastic gradient descent

In this project we are dealing with different types of cost functions. Depending on the method we use the minimization of the first derivative of the cost function is done differently. In the case of linear regression there is no use of numerical optimization methods as there exists an analytical solution to the derivative of the RSS. In the case of logistic regression and the neural network however, there are no closed form solutions to the cost functions being used. We therefore take use of two algorithms called gradient descent and stochastic gradient descent in order to update the parameters (weights).

We call the function we wish to minimize $E(\theta)$, as energy is the quantity we in most problems within physics are trying to minimize. In linear regression this is the MSE (RSS) and in logistic regression it is the cross entropy.

$$E(\theta) = \sum_{i=1}^{N} e_i(X_i, \theta) \tag{3.7}$$

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta E(\theta_t) \tag{3.8}$$

In regular gradient descent the algorithm simply updates the parameter θ_{t+1} according to Eq. (3.8). Since the cost function tells us something about how well the parameters work for our regression model we seek to minimize its gradient. Updateing the parameters with Eq. (3.8) causes the model to moves closer to a local minima. The η is called the learning rate and it decides how fast we want to move in the gradients direction. A to

large learning rate may cause the model to diverge, but given an optimal value or close to one the gradient descent will converge towards the minima.

Usually the first parameters are given random variables.

Gradient descent has some drawbacks, one being that it can get stuck in one local minima and never reaching the correct one. Another is that it is sensitvive to initial conditions, so what values we give the parameters in the begining matters. It is also somewhat computational expensive. A solution to this is stochastic gradient descent.

$$\nabla_\theta E(\theta) = \sum_i^n \nabla_\theta e_i(\mathbf{x}_i, \theta) \longrightarrow \sum_{i \in B_k} \nabla_\theta e_i(\mathbf{x}_i, \theta) \tag{3.9}$$

$$\nabla_\theta E^{MB}(\theta) = \sum_{i \in B_k}^M \nabla_\theta e_i(\mathbf{x}_i, \theta) \tag{3.10}$$

$$\theta_{t+1} = \theta_\mathbf{t} - \eta_t \nabla_\theta E^{MB}(\theta) \tag{3.11}$$

Here we devide the data set into smaller minibatches of size M creating n/M batches. We denote these minibatches B_k running from $k = 1...n/M$. Now we solve the gradient descent for the new minibatches and update the parameters according to 3.11. Not do we only speed up the computational process, but it also introduces stochasticity.

3.7 Blocking Method

For the statistical analysis and error estimates we have used the technique of blocking. We are looking for the expectation value of the ground state energy of our system. In order to say something about how accurate these results are we want to look at the standard deviation.

If our samples where uncorrelated we could calculate the standard deviation through the following equation

$$\sigma = \sqrt{\frac{1}{n-1}\left(\langle E_L^2\rangle - \langle E_L\rangle^2\right)}.$$

It can be showed that for correlated samples the equation for the standard deviation is

$$\sigma = \sqrt{\frac{1 + 2\tau/\Delta t}{n-1}\left(\langle E_L^2\rangle - \langle E_L\rangle^2\right)}$$

where τ is the time between one sample and the next uncorrelated sample and Δt is the time between each sample. If we knew what τ was we could simply find our σ, but since we don't we use blocking to find it. The method is simple enough, we divide our samples of data into blocks and calculating the mean of each block. So if we have an amount of samples of $\langle E_L\rangle$, we divide these into M blocks and calculate the mean. By plotting the standard deviation as function of the block size we can keep blocking and blocking until we see that the std. dev. stops increasing, and this is where the blocks are uncorrelated. Now we have an estimate for τ and thus the standard deviation.

3.8 Numerical differentiation

To evaluate the local energy of the system as defined in the project it necessary to compute the second derivative of the trial wave function ψ_T. We've chosen to implement the numerical differentiation as a finite difference approximation. Let $\mathbf{R}$ be the row major $N \times D$ matrix where N is the number of particles and D is their dimension. Then the second derivative can be found by the procedure listed as algorithm 2

Data: matrix $\mathbf{R}$
Result: float $\nabla^2\psi_T(R)$

$\Delta = 0$
$\mathbf{R_p} = \mathbf{R}$
$\mathbf{R_m} = \mathbf{R}$
for i *in* $[0, N-1]$ **do**
  for j *in* $[0, D-1]$ **do**
    $\mathbf{R_p}(i,j) += h$
    $\mathbf{R_m}(i,j) -= h$
    $\Delta = \psi_T(\mathbf{R_p}) + \psi_T(\mathbf{R_m}) - 2\psi_T(\mathbf{R})$
    $\mathbf{R_p}(i,j) -= h$
    $\mathbf{R_m}(i,j) += h$
  end
end
return $\frac{\Delta}{h^2}$

Algorithm 2: Numerical differentiation of the second order of the trial wave function on a system $\mathbf{R}$

Since the derivative involves three function calls for each particle the numerical derivative will obviously be quite computationally expensive. It is noted that the differentiation could be substantially optimized from the version included in the code, but is outside the scope of this project.

3.9 Diffusion Monte Carlo

Sources: Lecture notes, Morten [3]

Numerical method solves the many-body Schrödinger equation for the ground-state of a system of boson exactly. Since it does not depend on any a priori choice of wave function. In practice however, one needs a guess of the wave function that is within the range of the exact wave function, or the method would use an enormous amount of time to solve the problem. So in theory DMC can produce the exact ground state of any(?) system, but in practice this is not the case.

Diffusion Monte Carlo (DMC) is a method within a class of projector Monte Carlo methods. They are methods based on taking projections in Hilbert space.

$$\exp^{(-\hat{H}t)} \psi(x) = \sum_i c_i \exp^{(-\epsilon_i t)} \phi_i(x) \tag{3.12}$$

Notes from Jørgens master: Green's function, an ensemble of walkers can be iterated by transitioning between configurations r and r' with probability given the greens function. Rewriting a trail wave function to something looking like a diffusion equation.

Question: what does he mean by the diffusion eq. contributing to the variational method? Is this because of the time-dependent part? Or is this the case for all problems?

Chapter 4

Machine Learning: RBM/DBM

Some introduction containing historical information and stuff, and motivation for this section.

4.1 Restricted Boltzmann Machine

Restricted Boltzmann machine is a energy-based generative model which include hidden and visible variables (ref to Metha). It consists of a two-layer network with a two dimensional matrix W_{ij} telling how strong the connections between the hidden and visible nodes are. The energy related to a configuration of the nodes is what forms the basis of our model. It is given in (4.1).

$$E(\mathbf{v}, \mathbf{h}) = -\sum_i a_i(v_i) - \sum_\mu b_j(h_j) - \sum_{ij} W_{ij} v_i h_j \tag{4.1}$$

Here $a_i(v_i)$ and $b_j(h_j)$ are bias functions of the visible and hidden layers which we are allowed to choose our self. In our case we want the visible layer to take a continuous form and the hidden layer to be binary, (Gaussian-binary), meaning a_i and b_j takes the following from

$$a_i(v_i) = \frac{v_i^2}{2\sigma_i^2}, \quad b_j(h_j) = b_j h_j.$$

We are working with restricted Boltzmann machine meaning there is no connection between nodes within layers, only between layers. Also this network is a generative one so we want our network to learn a probability distribution. We begin with the joint probability distribution of the visible and hidden nodes is given in (4.2) where Z is the partition function, see (4.3).

$$F_{rbm}(\mathbf{X}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{X}, \mathbf{h})} \tag{4.2}$$

$$Z = \int \int \frac{1}{Z} e^{-E(\mathbf{X},\mathbf{h})} d\mathbf{x} d\mathbf{h} \tag{4.3}$$

From (4.2) we can marginalize over all the hidden units and get the distribution over the visible units. And as mentioned above this is what we use to represent our wave function.

$$F_{rbm}(\mathbf{X}) = \sum_{\mathbf{h}} F_{rbm}(\mathbf{X}, \mathbf{h}) \tag{4.4}$$

$$= \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{X},\mathbf{h})} \tag{4.5}$$

Since we have no data to train we use the technique of reinforcement learning. We feed the network with input using Monte Carlo method and based on the variational principle we find the ground state of the system by seeking the configuration that gives the lowest quantum mechanical energy. According to this principle we change the weights and biases by gradient descent method and hopefully the network will converge towards the correct state.

4.2 Gibbs Sampling

We represent the wave equation on a new form. This requires that the wave function is positive definite.

$$\psi(x) = \sqrt{(F_{rbm})} \tag{4.6}$$

Rather than on the form we have used before.

$$\psi(x) = (F_{rbm}) \tag{4.7}$$

Part II

Implementation and results

Chapter 5

Implementation

Chapter 6

Results

Chapter 7

Conclusion

Appendix A

Appendix Title

A.1 Derivations

A.1.1 Local energy

The local energy is given by

$$E_L(\mathbf{r}) = \frac{1}{\Psi_T(\mathbf{r})}\hat{H}\Psi_T(\mathbf{r})$$

with trial wave equation

$$\Psi_T(\mathbf{r}) = \prod_i g(\alpha, \beta, \mathbf{r_i}) \prod_{i<j} f(a, |\mathbf{r_i} - \mathbf{r_j}|)$$

The Hamiltonian becomes

$$\hat{H} = \sum_i^N \left(\frac{-\hbar^2}{2m}\nabla_i^2 + V_{ext}(\mathbf{r_i}) \right) + \sum_{i>j}^N V_{int}(\mathbf{r_i}, \mathbf{r_j})$$

where

$$V_{ext}(\mathbf{r_i}) = \begin{cases} \frac{1}{2}m\omega_{ho}^2 r^2 & (S) \\ \frac{1}{2}m[\omega_{ho}^2(x^2+y^2) + \omega_z^2 z^2] & (E) \end{cases}$$

and

$$V_{int}(|\mathbf{r_i} - \mathbf{r_j}|) = \begin{cases} \infty & |\mathbf{r_i} - \mathbf{r_j}| \leq a \\ 0 & |\mathbf{r_i} - \mathbf{r_j}| > a \end{cases}$$

First we solve only the harmonic oscillator (a=0) and we use $\beta = 1$ for one particle in 1D. And for one particle the trial wave equation, Hamiltonian and local energy becomes

$$\Psi_T(\mathbf{r}) = g(\alpha, x) = e^{-\alpha x_i^2}, \tag{A.1}$$

$$\hat{H} = \left(\frac{-\hbar^2}{2m}\nabla^2 + \frac{1}{2}m\omega_{ho}^2 x^2 \right) \tag{A.2}$$

$$E_L = \frac{1}{e^{-\alpha x^2}} \left(\frac{-\hbar^2}{2m}\nabla^2(e^{-\alpha x^2}) + \frac{1}{2}m\omega_{ho}^2 x^2 (e^{-\alpha x^2}) \right) \tag{A.3}$$

We then need to compute the laplacian of the trial wave-function.

$$\nabla^2(e^{-\alpha x^2}) = \frac{d^2 e^{-\alpha x^2}}{dx^2} \tag{A.4}$$

$$= \frac{d}{dx}\left(\frac{de^{-\alpha x^2}}{dx} \right) \tag{A.5}$$

$$= \frac{d}{dx}(-2\alpha x \cdot e^{-\alpha x^2}) \tag{A.6}$$

$$= (4\alpha^2 x^2 - 2\alpha)e^{-\alpha x^2} \tag{A.7}$$

Finally, the local energy becomes

$$E_L = \frac{e^{-\alpha x^2}}{e^{-\alpha x^2}} \left(\frac{-\hbar^2}{2m}(4\alpha^2 x^2 - 2\alpha) + \frac{1}{2}m\omega_{ho}^2 x^2 \right) \tag{A.8}$$

$$E_L = \frac{-\hbar^2}{2m}(4\alpha^2 x^2 - 2\alpha) + \frac{1}{2}m\omega_{ho}^2 x^2 \tag{A.9}$$

Solving the same problem for one particle in 2D and 3D we get something similar,

$$E_L = (\frac{-\hbar^2}{2m})4\alpha^2(x^2 + y^2) - 4\alpha + \frac{1}{2}m\omega_{ho}^2(x^2 + y^2), \tag{A.10}$$

$$E_L = (\frac{-\hbar^2}{2m})4\alpha^2(x^2 + y^2 + z^2) - 6\alpha + \frac{1}{2}m\omega_{ho}^2(x^2 + y^2 + z^2) \tag{A.11}$$

We can now easily see that the local energy energy for N number of particles in 1D, 2D and 3D becomes

$$\text{1D: } E_L = \sum_i^N (\frac{-\hbar^2}{2m})[4\alpha^2(x^2) - 2\alpha] + \frac{1}{2}m\omega_{ho}^2(x^2),$$

$$\text{2D: } E_L = \sum_i^N (\frac{-\hbar^2}{2m})[4\alpha^2(x^2 + y^2) - 4\alpha] + \frac{1}{2}m\omega_{ho}^2(x^2 + y^2) \tag{A.12}$$

$$\text{3D: } E_L = \sum_i^N (\frac{-\hbar^2}{2m})[4\alpha^2(x^2 + y^2 + \beta^2 z^2) - 4\alpha - 2\alpha\beta] + \frac{1}{2}m\omega_{ho}^2(x^2 + y^2 + \beta z^2)$$

since the energy is simply the sum of the derivatives of each particle.
Now we can try to solve the complete problem. The trial wave function is now

$$\Psi_T(\mathbf{r}) = \prod_i g(\alpha, \beta, \mathbf{r_i}) \prod_{i<j} f(a, |\mathbf{r_i} - \mathbf{r_j}|)$$

and first we rewrite it using

$$g(\alpha, \beta, \mathbf{r_i}) = \exp -\alpha(x_i^2 + y_i^2 + \beta z_i^2) = \phi(\mathbf{r_i})$$

and

$$f(r_{ij}) \exp\left(\sum_{i<j} u(r_{ij})\right)$$

getting

$$\Psi_T(\mathbf{r}) = \prod_i \phi(r_i) \exp\left(\sum_{i<j} u(r_{ij})\right).$$

The local energy for this problem becomes:

$$E_L = \frac{1}{\Psi_T(\boldsymbol{r})}\left(\sum_i^N \left(\frac{\hbar^2}{2m}\nabla_i^2 \Psi_T(\mathbf{r}) + \frac{1}{2} m\omega_{ho}^2 r^2 \Psi_T(\mathbf{r})\right) + \sum_{i<j}^N V_{int}(\mathbf{r_i}, \mathbf{r_j})\Psi_T(\mathbf{r})\right). \quad \text{(A.13)}$$

The difficulty in (A.13) is solving the derivatives of the wave equation given the complexity of the exponential. We begin with the first derivative.

$$\nabla_i^2 \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij})\right) = \nabla_i \cdot \nabla_i \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij})\right)$$

The first derivative of particle k:

$$\nabla_k \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij})\right) = \nabla_k \left(\prod_i \phi(\mathbf{r_i})\right) \exp\left(\sum_{i<j} u(r_{ij})\right) + \nabla_k \left(\exp\left(\sum_{i<j} u(r_{ij})\right)\right) \prod_i \phi(r_i)$$

$$\begin{aligned}
\nabla_k \left(\prod_i \phi(\mathbf{r_i}) \right) &= \nabla_k (\phi(r_1)(\phi(r_2)...(\phi(r_k)..(\phi(r_N)) \\
&= \nabla_k \left(e^{-\alpha(x_1^2+y_1^2+z_1^2)} e^{-\alpha(x_2^2+y_2^2+z_2^2)} ... e^{-\alpha(x_k^2+y_k^2+z_k^2)} ... e^{-\alpha(x_N^2+y_N^2+z_N^2)} \right) \\
&= \nabla_k \phi(r_k) \left[\prod_{i \neq k} \phi(\mathbf{r_i}) \right] \\
\nabla_k \exp \left(\sum_{i<j} u(r_{ij}) \right) &= \nabla_k \exp\big(u(r_{12}) + u(r_{13}) + ... + u(r_{23}) + ... + u(r_{kj}) + .. + u(r_{N-1,N}) \big) \\
&= \exp \left(\sum_{i<j} u(r_{ij}) \right) \sum_{i \neq k} \nabla_k u(r_{kj})
\end{aligned}$$

And the first derivative of the trial wave equation is

$$\nabla_k \Psi_T(\mathbf{r}) = \nabla_k \phi(r_k) \left[\prod_{i \neq k} \phi(\mathbf{r_i}) \right] \exp \left(\sum_{i<j} u(r_{ij}) \right) + \prod_i \phi(\mathbf{r_i}) \exp \left(\sum_{i<j} u(r_{ij}) \right) \sum_{j \neq k} \nabla_k u(r_{kj}).$$

Now we find the second derivative of the wave function.

$$\frac{1}{\Psi_T(\mathbf{r})}\nabla_k^2\Psi_T(\mathbf{r}) = \frac{1}{\nabla_k \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij})\right)} \left(\nabla_k \left(\nabla_k \phi(r_k) \left[\prod_{i\neq k} \phi(\mathbf{r_i}) \right] \right) \cdot \exp\left(\sum_{i<j} u(r_{ij}) \right) \right.$$

$$+ \nabla_k \phi(r_k) \left[\prod_{i\neq k} \phi(\mathbf{r_i}) \right] \cdot \nabla_k \left(\exp\left(\sum_{i<j} u(r_{ij}) \right) \right)$$

$$+ \nabla_k \left(\prod_i \phi(\mathbf{r_i}) \right) \cdot \exp\left(\sum_{i<j} u(r_{ij}) \right) \sum_{j\neq k} \nabla_k u(r_{kj})$$

$$+ \prod_i \phi(\mathbf{r_i}) \cdot \nabla_k \left(\exp\left(\sum_{i<j} u(r_{ij}) \right) \right) \sum_{j\neq k} \nabla_k u(r_{kj})$$

$$\left. + \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij}) \right) \cdot \nabla_k \left(\sum_{j\neq k} \nabla_k u(r_{kj}) \right) \right)$$

Solving these equations separately makes it easier.

$$\frac{\nabla_k^2 \phi(r_k) \left[\prod_{i\neq k} \phi(\mathbf{r_i}) \right] \cdot \exp\left(\sum_{i<j} u(r_{ij}) \right)}{\nabla_k \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij}) \right)} = \frac{\nabla_k^2 \phi(r_k)}{\phi(r_k)}$$

$$\frac{\nabla_k \phi(r_k) \left[\prod_{i\neq k} \phi(\mathbf{r_i}) \right] \cdot \nabla_k \left(\exp\left(\sum_{i<j} u(r_{ij}) \right) \right)}{\nabla_k \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij}) \right)} = \frac{\nabla_k \phi(r_k)}{\phi(r_k)} \sum_{j\neq k} \nabla_k u(r_{kj})$$

$$\frac{\nabla_k \left(\prod_i \phi(\mathbf{r_i}) \right) \cdot \exp\left(\sum_{i<j} u(r_{ij}) \right) \sum_{j\neq k} \nabla_k u(r_{kj})}{\nabla_k \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij}) \right)} = \frac{\nabla_k \phi(r_k)}{\phi(r_k)} \sum_{j\neq k} \nabla_k u(r_{kj})$$

$$\frac{\prod_i \phi(\mathbf{r_i}) \cdot \nabla_k \left(\exp\left(\sum_{i<j} u(r_{ij}) \right) \right) \sum_{j\neq k} \nabla_k u(r_{kj})}{\nabla_k \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij}) \right)} = \sum_{i\neq k} \nabla_k u(r_{ki}) \sum_{j\neq k} \nabla_k u(r_{kj})$$

$$\frac{\prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij})\right) \cdot \nabla_k \left(\sum_{j\neq k} \nabla_k u(r_{kj})\right)}{\nabla_k \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij})\right)} = \sum_{j\neq k} \nabla_k^2 u(r_{kj})$$

Putting them together again we get the following

$$\frac{1}{\Psi_T(\mathbf{r})}\nabla_k^2 \Psi_T(\mathbf{r}) = \frac{\nabla_k^2\phi(r_k)}{\phi(r_k)} + 2\frac{\nabla_k\phi(r_k)}{\phi(r_k)} \sum_{j\neq k} \nabla_k u(r_{kj}) + \sum_{i\neq k} \nabla_k u(r_{ki}) \sum_{j\neq k} \nabla_k u(r_{kj}) + \sum_{j\neq k} \nabla_k^2 u(r_{kj}) \tag{A.14}$$

We solve the first and second derivatives of $u(r_{kj})$.

$$\nabla_k u(r_{kj}) = \left(\vec{i}\frac{\partial}{\partial x_k} + \vec{j}\frac{\partial}{\partial y_k} + \vec{k}\frac{\partial}{\partial z_k}\right) u(r_{kj}) \tag{A.15}$$

From Rottmann p. 128 we have that

$$\begin{aligned}
\frac{\partial u(r_{kj})}{\partial x_k}\,\vec{i} &= \frac{\partial u(r_{kj})}{\partial r_{kj}}\frac{\partial r_{kj}}{\partial x_k}\,\vec{i} \\
&= u'(r_{kj})\frac{\partial\sqrt{|x_k - x_j|^2 + |y_k - y_j|^2 + |z_k - z_j|^2}}{\partial x_k}\vec{i} \\
&= u'(r_{kj})\frac{1}{2}2|x_k - x_j|\frac{1}{r_{kj}}\vec{i} \\
&= \frac{u'(r_{kj})|x_k - x_j|}{r_{kj}}\vec{i} \\
\frac{\partial u(r_{kj})}{\partial y_k}\,\vec{j} &= \frac{u'(r_{kj})|y_k - y_j|}{r_{kj}}\vec{j} \\
\frac{\partial u(r_{kj})}{\partial z_k}\,\vec{k} &= \frac{u'(r_{kj})|z_k - z_j|}{r_{kj}}\vec{k}
\end{aligned}$$

where we have used the fact that $r_{kj} = \sqrt{|x_k - x_j|^2 + |y_k - y_j|^2 + |z_k - z_j|^2}$. Now equation (A.15) becomes

$$\begin{aligned}
\nabla_k u(r_{kj}) &= u'(r_{kj})\frac{(|x_k - x_j|\vec{i} + |y_k - y_j|\vec{j} + |z_k - z_j|\vec{k})}{r_{kj}} \\
&= u'(r_{kj})\frac{(\boldsymbol{r_k} - \boldsymbol{r_j})}{r_{kj}}.
\end{aligned}$$

And for the second derivative have that

$$\nabla_k^2 u(r_{kj}) = \left(\frac{\partial^2}{\partial x_k^2} + \frac{\partial^2}{\partial y_k^2} + \frac{\partial^2}{\partial z_k^2}\right) u(r_{kj}) \tag{A.16}$$

and use Rottmann p. 128 again and see that

$$\begin{aligned}
\frac{\partial^2 u(r_{kj})}{\partial x_{kj}^2} &= \frac{\partial^2 u(r_{kj})}{\partial r_{kj}^2}\left(\frac{\partial r_{kj}}{\partial x_{kj}}\right)^2 + \frac{\partial u(r_{kj})}{\partial r_{kj}}\frac{\partial^2 r_{kj}}{\partial x_{kj}^2} \\
&= u''(r_{kj})\frac{(x_k - x_j)^2}{r_{kj}^2} + u'(r_{kj})\frac{\partial^2 r_{kj}}{\partial x_{kj}^2} \\
\frac{\partial^2 r_{kj}}{\partial x_{kj}^2} &= \frac{\partial^2 \sqrt{|x_k - x_j|^2 + |y_k - y_j|^2 + |z_k - z_j|^2}}{\partial x_{kj}^2} \\
&= \frac{\partial \frac{x_k - x_j}{\sqrt{|x_k - x_j|^2 + |y_k - y_j|^2 + |z_k - z_j|^2}}}{\partial x_{kj}} \\
&= \frac{r_{kj} - \frac{1}{2}(x_k - x_j)\cdot\frac{2(x_k - x_j)}{r_{kj}}}{r_{kj}^2} \\
&= \frac{1}{r_{kj}} - \frac{(x_k - x_j)^2}{r_{kj}^3}.
\end{aligned}$$

These equations put together and solving with respect to y and z we get

$$\begin{aligned}
\frac{\partial^2 u(r_{kj})}{\partial x_{kj}^2} &= u''(r_{kj})\frac{(x_k - x_j)^2}{r_{kj}^2} + u'(r_{kj})\left(\frac{1}{r_{kj}} - \frac{(x_k - x_j)^2}{r_{kj}^3}\right), \\
\frac{\partial^2 u(r_{kj})}{\partial y_{kj}^2} &= u''(r_{kj})\frac{(y_k - y_j)^2}{r_{kj}^2} + u'(r_{kj})\left(\frac{1}{r_{kj}} - \frac{(y_k - y_j)^2}{r_{kj}^3}\right), \\
\frac{\partial^2 u(r_{kj})}{\partial z_{kj}^2} &= u''(r_{kj})\frac{(z_k - z_j)^2}{r_{kj}^2} + u'(r_{kj})\left(\frac{1}{r_{kj}} - \frac{(z_k - z_j)^2}{r_{kj}^3}\right).
\end{aligned}$$

Now we can add them all together and equation (A.16) becomes

$$\begin{aligned}
\nabla_k^2 u(r_{kj}) &= \frac{u''(r_{kj})}{r_{kj}^2}((x_k - x_j)^2 + (y_k - y_j)^2 + (z_k - z_j)^2) \\
&\quad + u'(r_{kj})\left(\frac{3}{r_{kj}} - \frac{(x_k - x_j)^2 + (y_k - y_j)^2 + (z_k - z_j)^2}{r_{kj}^3}\right) \\
&= \frac{u''(r_{kj})r_{kj}^2}{r_{kj}^2} + u'(r_{kj})\left(\frac{3}{r_{kj}} - \frac{r_{kj}^2}{r_{kj}^3}\right) \\
&= u''(r_{kj}) + u'(r_{kj})\frac{2}{r_{kj}}
\end{aligned}$$

Now we can write out the complete second derivative, equation (A.14)

$$\frac{1}{\Psi_T(\mathbf{r})}\nabla_k^2\Psi_T(\mathbf{r}) = \frac{\nabla_k^2\phi(r_k)}{\phi(r_k)} + 2\frac{\nabla_k\phi(r_k)}{\phi(r_k)}\sum_{j\neq k}\frac{(\boldsymbol{r_k}-\boldsymbol{r_j})}{r_{kj}}u'(r_{kj}) + \sum_{ij\neq k}\frac{(\boldsymbol{r_k}-\boldsymbol{r_i})}{r_{ki}}\frac{(\boldsymbol{r_k}-\boldsymbol{r_j})}{r_{kj}}u'(r_{ki})u'(r_{kj}) + \sum_{j\neq k}\left(u''(r_{kj}) + \frac{2}{r_{kj}}u'(r_{kj})\right).$$

For the full analytical solution of the interacting problem we solve each term by it self.

$$\begin{aligned}\frac{1}{\phi(r_k)}\nabla_k^2\phi(r_k) &= \frac{\nabla_k^2\ exp(-\alpha(x_k^2+y_k^2+\beta z_k^2))}{exp(-\alpha(x_k^2+y_k^2+\beta z_k^2))}\\ &= ((2\alpha(2\alpha x_k^2-1)) + (2\alpha(2\alpha y_k^2-1)) + (2\alpha\beta(2\alpha\beta z_k^2-1)))\cdot\frac{\phi(r_k)}{\phi(r_k)}\\ &= -4\alpha^2 - 2\alpha\beta + 4\alpha^2(x_k^2+y_k^2+\beta z_k^2)\end{aligned}$$

$$\begin{aligned}\frac{1}{\phi(r_k)}\nabla_k\phi(r_k) &= \frac{\nabla_k\ exp(-\alpha(x_k^2+y_k^2+\beta z_k^2))}{exp(-\alpha(x_k^2+y_k^2+\beta z_k^2))}\\ &= (2\alpha x_k\vec{i} + 2\alpha y_k\vec{j} + 2\alpha\beta z_k\vec{k})\cdot\frac{\phi(r_k)}{\phi(r_k)}\\ &= (2\alpha x_k\vec{i} + 2\alpha y_k\vec{j} + 2\alpha\beta z_k\vec{k})\end{aligned}$$

Wolfram alpha gives to solution the first and second derivatives of function $u(r_{kj})$.

$$\begin{aligned}u'(r_{kj}) &= \frac{d(\ln f(r_{kj}))}{dr_{kj}}\\ &= \frac{d\ \ln\left(1-\frac{a}{(r_k-r_j)}\right)}{dr_{kj}}\\ &= -\frac{a}{ar_{kj}-r_{kj}^2}\end{aligned}$$

$$\begin{aligned}u''(r_{kj}) &= \frac{d^2(\ln f(r_{kj}))}{dr_{kj}^2}\\ &= \frac{a(a-2r_{kj})}{r_{kj}^2(a-r_{kj})^2}\end{aligned}$$

Putting all of this together we end up with the following expression for the second derivative of the wave equation divided by it self:

$$\frac{1}{\Psi_T(\mathbf{r})}\nabla_k^2\Psi_T(\mathbf{r}) = -4\alpha^2 - 2\alpha\beta + 4\alpha^2(x_k^2 + y_k^2 + \beta z_k^2)$$
$$+ 2((2\alpha x_k\vec{i} + 2\alpha y_k\vec{j} + 2\alpha\beta z_k\vec{k}))\sum_{j\neq k}\left(\frac{(x_k - x_j)\vec{i} + (y_k - y_j)\vec{j} + (z_k - z_j)\vec{k}}{r_{kj}}\left(\frac{-a}{ar_{kj} - r_{kj}^2}\right)\right.$$
$$+ \sum_{j\neq k}\left(\frac{(x_k - x_i)\vec{i} + (y_k - y_i)\vec{j} + (z_k - z_i)\vec{k}}{r_{ki}}\right)\left(\frac{(x_k - x_j)\vec{i} + (y_k - y_j)\vec{j} + (z_k - z_j)\vec{k}}{r_{kj}}\right)$$
$$* \left(\frac{-a}{ar_{ki} - r_{ki}^2}\right)\left(\frac{-a}{ar_{kj} - r_{kj}^2}\right)$$
$$+ \sum_{j\neq k}\left(\frac{a(a - 2r_{kj})}{r_{kj}^2(a - r_{kj})^2} + \frac{2}{r_{kj}} - \frac{a}{ar_{kj} - r_{kj}^2}\right)$$

A.1.2 Drift force

As for the drift force we simply write down the solutions for one particle in 1D, 2D and 3D as they are just the first derivative of the wave function,

$$F = \frac{2\nabla\Psi_T}{\Psi_T}.$$

And are easily solved;

$$F = -4\alpha x,$$

$$F = -4\alpha(x + y),$$

$$F = -4\alpha(x + y + \beta z).$$

For N-particles we must solve $\nabla_k\Psi_T$. In 1D this is

$$\nabla_k\Psi_T = -2\alpha x_k\prod_{k\neq i}^{N} e^{-\alpha x_i^2},$$

$$F_k = -4\alpha x_k\frac{\prod_{k\neq i}^{N} e^{-\alpha x_i^2}}{\prod_i^N e^{-\alpha x_i^2}} = -4\alpha x_k\frac{1}{e^{-\alpha x_k^2}},$$

$$F = \sum_i^N -4\alpha x_i\frac{1}{e^{-\alpha x_i^2}}.$$

And for 2D and 3D;

$$F = \sum_{i}^{N} -4\alpha(x_i + y_i)\frac{1}{e^{-\alpha(x_i^2+y_i^2)}},$$

$$F = \sum_{i}^{N} -4\alpha(x_i + y_i + \beta z_i)\frac{1}{e^{-\alpha(x_i^2+y_i^2+\beta z_i^2)}}.$$

Finally we can solve for N-particles with interaction.
We have already solved the first derivative of the wave equation when we dealt with the local energy problem.

$$\nabla_k \Psi_T(\mathbf{r}) = \nabla_k \phi(r_k) \left[\prod_{i\neq k} \phi(\mathbf{r_i})\right] \exp\left(\sum_{i<j} u(r_{ij})\right) + \prod_i \phi(\mathbf{r_i}) \exp\left(\sum_{i<j} u(r_{ij})\right) \sum_{j\neq k} \nabla_k u(r_{kj})$$

$$\begin{aligned}
\nabla_k \phi(r_k) &= \frac{\partial}{\partial x_k} e^{(-\alpha(x_k^2+y_k^2+\beta z_k^2))\vec{i}} \\
&+ \frac{\partial}{\partial y_k} e^{(-\alpha(x_k^2+y_k^2+\beta z_k^2))\vec{j}} \\
&+ \frac{\partial}{\partial z_k} e^{(-\alpha(x_k^2+y_k^2+\beta z_k^2))\vec{k}} \\
&= (-2\alpha x_k \vec{i} - 2\alpha y_k \vec{j} - 2\alpha\beta z_k \vec{k}) e^{(-\alpha(x_k^2+y_k^2+\beta z_k^2))} \\
\nabla_k u(r_{kj}) &= u'(r_{kj}) \frac{(\mathbf{r_k} - \mathbf{r_j})}{r_{kj}} \\
&= -\frac{a(\vec{r_k} - \vec{r_j})}{a r_{kj}^2 - r_{kj})^3}
\end{aligned}$$

$$\nabla_k \Psi_T(\mathbf{r}) = (-2\alpha x_k \vec{i} - 2\alpha y_k \vec{j} - 2\alpha\beta z_k \vec{k}) \Psi_T(\mathbf{r}) + \sum_{j\neq k} \left(-\frac{a(\vec{r_k} - \vec{r_j})}{a r_{kj}^2 - r_{kj})^3}\right) \Psi_T(\mathbf{r})$$

$$F = (-4\alpha x_k \vec{i} - 4\alpha y_k \vec{j} - 4\alpha\beta z_k \vec{k}) + \sum_{j\neq k} \left(-\frac{a(\vec{r_k} - \vec{r_j})}{a r_{kj}^2 - r_{kj})^3}\right)$$

For the elliptical trap we are given an external potential that turns the Hamiltonian to the following

$$\hat{H} = \sum_i^N \left(\frac{-\hbar^2}{2m}\nabla_i^2 + \frac{1}{2}m[\omega_{ho}^2(x^2+y^2) + \omega_z^2 z^2] \right) + \sum_{i<j}^N V_{int}(\mathbf{r_i}, \mathbf{r_j}).$$

Using that $\omega_{ho} = \omega_\perp$ and $\omega_z = \lambda\omega_{ho}$ we see easily that we can rewrite this to

$$\hat{H} = \sum_i^N \left(\frac{-\hbar^2}{2m}\nabla_i^2 + \frac{1}{2}m\omega_{ho}^2(x^2+y^2+\lambda^2 z^2) \right) + \sum_{i<j}^N V_{int}(\mathbf{r_i}, \mathbf{r_j}), \tag{A.17}$$

which is simply the Hamiltonian for the spherical trap with interaction and $\beta = \gamma = \lambda^2$.

A.2 Green's function ratio

$$\frac{G(x,y,\Delta t)}{G(y,x,\Delta t)} = \frac{\exp\big(-(x-y-DF(y)\Delta t)\big)^2/4D\Delta t}{\exp\big(-(y-x-DF(x)\Delta t)\big)^2/4D\Delta t} \tag{A.18}$$

$$\begin{aligned}(x-y-DF(y)\Delta t)^2 &= x^2 - xy - xDF(y)\Delta t - xy - y^2 \\ &\quad + yDF(y)\Delta t - xDF(y)\Delta t + yDF(y)\Delta t + (DF(y)\Delta t)^2 \\ &= x^2 + y^2 - 2xy - 2yDF(y)\Delta t + 2xDF(y)\Delta t + (DF(y)\Delta t)^2 \quad (i)\end{aligned}$$

$$(y-x-DF(x)\Delta t)^2 = y^2 + x^2 - 2xy - 2yDF(x)\Delta t + 2xDF(x)\Delta t + (DF(x)\Delta t)^2 \quad (ii)$$

$$\begin{aligned}\frac{G(x,y,\Delta t)}{G(y,x,\Delta t)} &= \exp\left(\frac{-(i)+(ii)}{4D\Delta t}\right) \\ &= \exp\left(\frac{-(DF(y)\Delta t)^2 + 2DF(y)\Delta t(x-y) + (DF(x)\Delta t)^2 + 2DF(x)\Delta t(x-y)}{4D\Delta t}\right) \\ &= \exp\left(\frac{2D\Delta t(x-y)(F(x)+F(y)) + (D\Delta t)^2(F(x)^2 - F(y)^2)}{4D\Delta t}\right) \\ &= \exp\left(\frac{(x-y)(F(x)+F(y)) + 2D\Delta t(F(x)^2 - F(y)^2)}{2}\right)\end{aligned}$$

We then make use of the following equality.

$$(F(x)^2 - F(y)^2) = (F(x)+F(y))(F(x)-F(y)) \tag{A.19}$$

The final result for the Green's function ratio becomes

$$\begin{aligned}\frac{G(x,y,\Delta t)}{G(y,x,\Delta t)} &= \exp\left(\frac{(x-y)(F(x)+F(y))+2D\Delta t(F(x)+F(y))(F(x)-F(y))}{2}\right)\\ &= \exp\left(\frac{(F(x)+F(y))((x-y)+2D\Delta t(F(x)-F(y))}{2}\right)\end{aligned}$$

Bibliography

[1] S. Kvaal. *Lecture notes for FYS-KJM4480*. Lecture notes. Nov. 2017. URL: http://theory.rutgers.edu/~giese/notes/DFT.pdf.

[2] Pankaj Mehta et al. "A high-bias, low-variance introduction to machine learning for physicists". In: *Physics Reports* (2019).

[3] M. Hjorth-Jensen. *Computational Physics*. Lecture notes. Aug. 2015. URL: https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Lectures/lectures2015.pdf.