

Project 1, Fys 4411

Karl R. Leikanger

May 9, 2012

Contents

1	Quantum dots	
2	The Monte Carlo method	
2.1	Introduction	1
2.2	Markov Chain Monte Carlo	2
2.3	Metropolis-Hasting algorithm	2
2.4	Choice of transition kernel	2
3	Statistical analysis of the data	
4	Quantum monte Carlo	
4.1	Introduction	3
4.2	Transition kernels	3
4.3	Variational Monte Carlo	4
4.4	The trial wave functions	5
5	Minimization of the wave functions	
6	Implementation and optimization	
6.1	Random number generators	6
6.2	Optimizing the Slater	6
6.3	Optimizing the Jastrow	8
6.4	Implementing $(\partial_\alpha, \partial_\beta)\langle E_L \rangle$	9
7	Results	
7.1	Scaling and time consume	9
7.2	Convergence rate: DMC vs VMC	9
7.3	Systematic errors	9
7.4	Numerical results	9
7.5	Verification of the results (literature, articles)	10
A	Pseudocode: The SGA algorithm.	11
B	Derivation of the analytical expressions for $(\partial_\alpha, \partial_\beta)\langle \hat{H} \rangle$	12

Abstract

1 blablabla

1 Quantum dots

2 Our system of choice

2 The Monte Carlo method

2.1 Introduction

3 Monte Carlo (MC) methods includes a wide range
3 of computational methods for solving integrals.
4 The main idea is to sample the integrand $f(\mathbf{r})$ in
4 random points $\{\mathbf{r}_i\}$, and finding an approximate
5 solution to the integral by doing statistical analysis
5 on the sampling data. The statistical error will give
6 us an immediate idea of the precision of the result.

In general, any integral can be written on the form

$$\int_{\mathcal{D}} f(\mathbf{r}) d\mathbf{r} = \int_{\mathcal{D}} P(\mathbf{r}) g(\mathbf{r}) d\mathbf{r}, \quad (1)$$

where $P(\mathbf{r})$ is a probability distribution. The integral can now be approximated by the sum

$$\int_{\mathcal{D}} f(\mathbf{r}) d\mathbf{r} \approx \frac{1}{N} \sum_{i=1}^N g(\mathbf{r}_i), \quad (2)$$

where $\{\mathbf{r}_i\}$ is a set of random numbers drawn from the statistical distribution $P(\mathbf{r})$.

The simplest example, possible whenever the region of the of the integral is finite, is when P is a uniform distribution bounded by the integration limits. Then $P(\mathbf{r})$ is a constant and $g(\mathbf{r}) = f(\mathbf{r})$. However, for many integrals this method will converge slowly and will require a very large number of samples to give accurate results. By choosing a

smart $P(\mathbf{r})$ that is large in the regions of interest, more samples will be drawn from the important regions, often leading to a more rapid convergence.

2.2 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) provides us with a method for drawing the correct samples from P . Instead of drawing uncorrelated random samples we use a markov chain to generate a sequence $\{\mathbf{r}_n\}$ of vectors. A Markov chain is characterized by a transition kernel $q(\mathbf{x}|\mathbf{y})$ which is a probability density distribution of the variable \mathbf{y} , and which tells us the probability of a transition from \mathbf{x} to \mathbf{y} . For each \mathbf{r}_n , we draw a new \mathbf{r}_{n+1} from the distribution $q(\mathbf{r}_n, \mathbf{y})$. The distribution of the vectors in the set $\{\mathbf{r}_n\}$ will converge towards the distribution P if the following conditions are fulfilled (reference):

- P is an invariant of the kernel $q(\mathbf{x}|\mathbf{y})$ in the sense that

$$\int_{\mathcal{D}} d\mathbf{r}_A P(\mathbf{r}_A) q(\mathbf{r}_A|\mathbf{r}_B) = P(\mathbf{r}_B), \quad (3)$$

- The chain is ergodic, which means that any point \mathbf{r}_n is accessible in a finite number transitions from any starting point \mathbf{r}_0 .

It is customary to make the analogy to a set of λ biased random walkers, where the density of walkers P_w is (close to) stationary and proportional to P . This analogy is reasonable as long as the sequence $\{\mathbf{r}_n\}$ can be split into λ subsequences $\{\mathbf{r}_n^\lambda\}$ which are (practically speaking) uncorrelated. Then one can equally well interpret the samples as N samples from one markov chain, as N/λ samples from λ markov chains.

For P_w to be stationary (), the population of walkers in any point \mathbf{r}_A must be () constant,

$$\sum_n P_w(\mathbf{r}_A) q(\mathbf{r}_A|\mathbf{r}_n) \approx \sum_n P_w(\mathbf{r}_n) q(\mathbf{r}_n|\mathbf{r}_A) \quad (4)$$

This equation resembles (3) since both equations states that the density is stationary. The condition of detailed balance

$$P_w(\mathbf{r}_A) q(\mathbf{r}_A|\mathbf{r}_B) = P_w(\mathbf{r}_B) q(\mathbf{r}_B|\mathbf{r}_A). \quad (5)$$

assures that (4) is fulfilled. As long as the markov chain is ergodic the walkers will have access to all regions of the integration domain.

2.3 Metropolis-Hasting algorithm

We will now outline a general algorithm for the movement of the random walkers that fulfills (5). We introduce new scalar field to our equation $\gamma(\mathbf{r}_A, \mathbf{r}_B)$, which is a measure of the probability of accepting a move from \mathbf{r}_A to \mathbf{r}_B . We split our transition kernel in two parts

$$q(\mathbf{r}_A|\mathbf{r}_B) \rightarrow \gamma(\mathbf{r}_A|\mathbf{r}_B) g(\mathbf{r}_A|\mathbf{r}_B). \quad (6)$$

I will refer to $g(\mathbf{r}_A|\mathbf{r}_B)$ as the suggestion density, and $\gamma(\mathbf{r}_A|\mathbf{r}_B)$ as the acceptance probability. We set

$$\begin{aligned} \gamma(\mathbf{r}_B|\mathbf{r}_A) &= 1 \text{ when} \\ P_w(\mathbf{r}_A) g(\mathbf{r}_A|\mathbf{r}_B) &> P_w(\mathbf{r}_B) g(\mathbf{r}_B|\mathbf{r}_A). \end{aligned} \quad (7)$$

Then, by inserting this equations into (5), we get

$$\gamma(\mathbf{r}_A|\mathbf{r}_B) = \min \left(\frac{P_w(\mathbf{r}_B) g(\mathbf{r}_B|\mathbf{r}_A)}{P_w(\mathbf{r}_A) g(\mathbf{r}_A|\mathbf{r}_B)}, 1 \right) \quad (8)$$

By making this choice for γ , (5) will be fulfilled and the sequence $\{\mathbf{r}_i\}$ will sample P . Note that only the relative ratio between the left and the right hand side of (eq) are of interest to us. Therefore, P does not have to be normalized. These equations are implemented in the well known Metropolis-Hastings algorithm as follows:

loop

$\mathbf{r}_B \leftarrow$ random from distribution $g(\mathbf{r}_T|\mathbf{r}_B)$.

$\mathcal{X} \leftarrow$ random uniform. ($\mathcal{X} \in (0, 1)$)

Calculate $\gamma(\mathbf{r}_n|\mathbf{r}_T)$.

if $\mathcal{X} \leq \gamma$ **then**

$\mathbf{r}_{n+1} \leftarrow \mathbf{r}_T$

else

$\mathbf{r}_{n+1} \leftarrow \mathbf{r}_n$

end if

$n \leftarrow n + 1$

end loop

2.4 Choice of transition kernel

One possibility us to use a suggestion density $g(\mathbf{r}_n|\mathbf{r}_{n+1})$ that is symmetric around \mathbf{r}_n along all axis. This method is referred to as brute force sampling. In this case $\gamma(\mathbf{r}_{n+1}|\mathbf{r}_n)$ can be expressed as

$$\gamma(\mathbf{r}_{n+1}|\mathbf{r}_n) = \min \left(\frac{P_w(\mathbf{r}_{n+1})}{P_w(\mathbf{r}_n)}, 1 \right) \quad (9)$$

A common choice is to set

$$g(\mathbf{r}_n|\mathbf{r}_{n+1}) = l \left(\mathcal{U}(\mathbf{r}_{n+1} - \mathbf{r}_n) - \frac{1}{2} \right) \quad (10)$$

where l is a constant which we will refer to as the step length, and $\mathcal{U}(\mathbf{r})$ is the uniform distribution

$$\mathcal{U}(\mathbf{r}) = \begin{cases} 1, & \text{if all } \mathbf{r} \cdot \mathbf{e}_i \in [0, 1] \\ 0, & \text{otherwise} \end{cases}. \quad (11)$$

Here \mathbf{e}_i is the unit vectors of \mathcal{D} . The step length l will alter the number of accepted steps during a simulation, and will have an effect on the rate of convergence.

An other class of transition kernels makes use of our knowledge of the mathematical form of P by using a suggestion density that is dependent on it. By solving the Fokker-Planck equation for the stationary distribution P one can construct a transition kernel that in principle will sample P perfectly without the Metropolis-Hastings accept/reject procedure [6, 9]. However, it is necessary to discretize the dynamical equations in time $t \rightarrow t_n = \Delta t n$, and an time step error arises leading to an imperfect sampling of P . The Metropolis-Hastings algorithm corrects this error by restoring detailed balance.

The suggestion density becomes [6]

$$g(\mathbf{r}_{n+1}|\mathbf{r}_n) = K \exp(-D\Delta t \frac{\nabla P(\mathbf{r}_n)}{P(\mathbf{r}_n)} + \mathbf{r}_{n+1} - \mathbf{r}_n)^2, \quad (12)$$

where D is the diffusion constant originating from the Fokker-Planck equation and K is a normalization constant. This is simply a normal distribution with a shift $\mathbf{r}_n + D\Delta t \frac{\nabla P(\mathbf{r}_n)}{P(\mathbf{r}_n)}$. Samples from this density are generated using the equation

$$\mathbf{r}_{n+1} = \mathbf{r}_n + D\Delta t \frac{\nabla P(\mathbf{r}_n)}{P(\mathbf{r}_n)} + \sqrt{\Delta t 2D} \mathcal{N}, \quad (13)$$

where \mathcal{N} is a random normal distributed vector. From this equation it is evident that a small Δt will lead to a small average step length $|\mathbf{r}_{n+1} - \mathbf{r}_n|$. This might slow down the convergence of the system even though the acceptance rate will increase. In practice one has to experiment to find the optimal values for Δt that gives a ((small time step error)) and a rapid convergence.

3 Statistical analysis of the data

The statistical error is defined as the standard deviation of a sample mean, and can be estimated using the formula [5]

$$err \approx \sqrt{\frac{1}{N} Var(\mathcal{M})}, \quad (14)$$

where \mathcal{M} is a set of N uncorrelated sample means.

The samples obtained in the MCMC simulations is heavily correlated, mainly because of the high correlation between succeeding configurations \mathbf{R} . We get an estimate of err by dividing our data into $N_\tau = N/\tau$ bins of τ samples, and calculating the standard deviation of the N_τ mean values.

$$err(\tau) \approx \sqrt{\frac{\tau}{N_\tau} Var(\mathcal{M}_\tau)} = \sqrt{\frac{\tau}{N_\tau} \langle \mathcal{M}_\tau^2 \rangle - \langle \mathcal{M}_\tau \rangle^2}. \quad (15)$$

This is a good estimate if 1) τ is large enough so that the means \mathcal{M}_τ are practically uncorrelated and 2) N_τ is large enough to give us a good statistical value. Starting with a small τ , $err(\tau)$ will converge towards the correct statistical error as τ increases. This method is referred to as blocking analysis. Fig. (??) shows a plot of $err(\tau)$ for a simulation of a 6 particle quantum dot with an error close to $\pm 10^{-5}$ Hartrees. The time between two practically uncorrelated samples, when the curve $err(\tau)$ flattens, will be referred to as the correlation length.

4 Quantum monte Carlo

4.1 Introduction

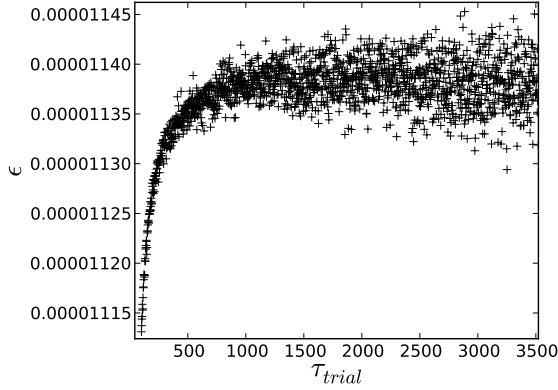
The expectation values of an observable \hat{O} can be expressed on the general form

$$\langle O \rangle = \int d\mathbf{r} \Psi(\mathbf{r}) \hat{O} \Psi(\mathbf{r})^*, \quad (16)$$

where $\Psi(\mathbf{r})$ is a solution to the schrödinger equation. The integral can also be written

$$\langle O \rangle = \int |\Psi(\mathbf{r})|^2 \frac{\hat{O} \Psi(\mathbf{r})^\dagger}{\Psi(\mathbf{r})^\dagger}, \quad (17)$$

Figure 1: *Blocking results for a 6 particle quantum dot with $\omega = 1$ and $\Delta t = 0.05$. A total of 10^8 samples has been used in the simulation. From the plot we can see that the statistical error is close to $\pm 10^{-5}$ Hartrees, and that the correlation length is of the order $\mathcal{O}(10^3)$.*



Since $|\Psi(\mathbf{r})|^2$ is a probability density distribution we see that

$$\langle O \rangle \approx \sum_{i=1}^N \frac{\hat{O}\Psi(\mathbf{r}_i)^\dagger}{\Psi(\mathbf{r}_i)^\dagger}, \quad (18)$$

where the set of numbers $\{\mathbf{r}_i\}$ are drawn from $|\Psi|^2$. This sum can be sampled using the Metropolis-Hastings algorithm with the acceptance probability

$$\gamma(r_i|r_{i+1}) = \frac{g(r_i|r_{i+1})|\Psi(\mathbf{r}_i)|^2}{g(r_{i+1}|r_i)|\Psi(\mathbf{r}_{i+1})|^2}. \quad (19)$$

$g(r_i|r_{i+1})$ is the suggestion density as described earlier. Note that Ψ does not need to be normalized in the above equations. This saves us a great deal of work since the normalization of most wave functions would have to be done numerically for every new trial function.

4.2 Transition kernels

Brute force sampling is implemented using the suggestion density eq. (10) and a acceptance probability

$$\gamma(\mathbf{R}_{n+1}|\mathbf{R}_n) = \max \left(\frac{|\Psi(\mathbf{R}_{n+1})|^2}{|\Psi(\mathbf{R}_n)|^2}, 1 \right) \quad (20)$$

Importance sampling based on Langevin dynamics is implemented using the following equations. The gradient of $P \rightarrow |\Psi|^2$ is

$$\mathbf{F}_n = \frac{\nabla |\Psi(\mathbf{R}_n)|^2}{|\Psi(\mathbf{R}_n)|^2} = 2 \frac{\nabla \Psi(\mathbf{R}_n)}{\Psi(\mathbf{R}_n)}, \quad (21)$$

and the diffusion constant D can be shown to be $1/2$ in our dimensionless units(ref). Combining these two quantities with eq. (13) gives us an equation to sample the suggestion density eq. (12).

$$\begin{aligned} \mathbf{R}_{n+1} = \mathbf{R}_n + \Delta t \frac{\nabla \Psi(\mathbf{R}_n)}{\Psi(\mathbf{R}_n)} \\ + \sqrt{\Delta t} \mathcal{N}(\mathbf{R}_{n+1} - \mathbf{R}_n). \end{aligned} \quad (22)$$

The acceptance probability is found combining eq. (12) and eq. (8)

$$\gamma(\mathbf{R}_{n+1}|\mathbf{R}_n) = \min \left(G(\mathbf{R}_{n+1}, \mathbf{R}_n) \frac{|\Psi(\mathbf{R}_{n+1})|^2}{|\Psi(\mathbf{R}_n)|^2}, 1 \right) \quad (23)$$

where $G(\mathbf{R}_{n+1}, \mathbf{R})$ is the greens function

$$\begin{aligned} G(\mathbf{R}_{n+1}, \mathbf{R}_n) = e^{-\frac{1}{2}(\mathbf{F}_{n+1} + \mathbf{F}_n)} \times \\ e^{\left(\frac{1}{4}\Delta t(\mathbf{F}_n - \mathbf{F}_{n+1}) + \mathbf{R}_n - \mathbf{R}_{n+1}\right)}. \end{aligned} \quad (24)$$

4.3 Variational Monte Carlo

According to the ritz principle, the expectation value for the energy

$$\epsilon_0 \geq \langle \Psi_T | \hat{H} | \Psi_T \rangle \forall |\Psi_T\rangle \in \mathcal{H}. \quad (25)$$

This means that any expectation value $\langle \hat{H} \rangle$ sets an upper limit to the energy of the system. This principle allows us to perform a systematic search for the lowest $\langle \hat{H} \rangle$ using a set of trial wave functions $|\Psi_T(\alpha, \beta, \dots)\rangle$ where α, β, \dots are variational parameters.

The search in the parameter space (α, β, \dots) can be done by performing a number of MCMC calculations on a grid. More sophisticated methods are also in use, like the stochastic gradient approximation which will be described later in this text.

Since the variance $\langle \hat{H}^2 \rangle - \langle \hat{H} \rangle^2 = 0$ for the eigenfunctions of \hat{H} , the variance is a measure of how close $|\Psi_T, \alpha, \beta, \dots\rangle$ is to the 'true' ground state

wave function of the system. Even if the lowest energy obtained in our simulations will be closest to the ground state energy, the trial function with the lowest variance might be a better fit to the 'true' ground state wave function.

4.4 The trial wave functions

Our system is a closed shell model with the hamiltonian

$$\hat{H} = -\frac{\nabla^2}{2} + \omega|\mathbf{R}|^2 - \sum_{i<j} \frac{1}{r_{ij}}. \quad (26)$$

Here $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_N)$ is a vector containing the position of all electrons in the system. , and $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$ is the distance between electron i and electron j . The equations

$$\begin{aligned} \phi_{n_x n_y}^\alpha(\mathbf{r}_i) &= H_{n_y}(\sqrt{\omega\alpha}y)H_{n_x}(\sqrt{\omega\alpha}x) \\ &\times \exp(-\frac{1}{2}\omega\alpha|\mathbf{r}_i|^2) \end{aligned} \quad (27)$$

are the eigenstates for the hamiltonian eq. (26) when $\alpha = 1$. $H_n(x)$ is the Hermite polynomials and α is a variational parameter $\alpha \in [0, 1)$ which represents the "shielding" effect that occurs when more than one electron is present in the system. Since the electrons carry a negative charge, the efficient external potential seen by the individual particles will be lower than the actual external potential.

Our trial wave function can be written on the form

$$\Psi(\mathbf{R}) = \mathcal{D}_\downarrow^\alpha(\mathbf{R})\mathcal{D}_\uparrow^\alpha(\mathbf{R})\mathcal{J}^\beta(\mathbf{R}) \quad (28)$$

Here $\mathcal{D}_\uparrow(\mathbf{R})$ is a slater determinant consisting of the one particle solutions eq. (27). Table 1 shows which orbitals that enters the slater determinant up to the 6'th full shell. The slater determinant automatically fulfills the Pauli principle stating that any wave function consisting of identical fermions must be antisymmetric w.r.t. the exchange of two particles. $\mathcal{D}_\downarrow^\alpha(\mathbf{R})$ is the corresponding determinant for the spin down states. Note that in the case of a spin dependent hamiltonian, $\mathcal{D}_\downarrow^\alpha\mathcal{D}_\uparrow^\alpha$ would have to be written as one single slater determinant to preserve the antisymmetry of the wavefunction. This is also the case for the open shell quantum dots since they can have a non zero net spin. $\mathcal{J}^\beta(\mathbf{R})$ is the Jastrow factor which removes the singularities

Shell	Electrons	One particle orbitals
1	2	ψ_{00}
2	6	ψ_{10}, ψ_{01}
3	12	$\psi_{20}, \psi_{11}, \psi_{02}$
4	20	$\psi_{30}, \psi_{21}, \psi_{12}, \psi_{03}$
5	30	$\psi_{40}, \psi_{31}, \psi_{22}, \psi_{13}, \psi_{04}$
6	42	$\psi_{50}, \psi_{41}, \psi_{32}, \psi_{23}, \psi_{14}, \psi_{05}$

Table 1: *This table shows the one particle orbitals that is part of the slater determinants. Only the first orbital state ψ_{00} is needed to fill the first shell. This corresponds to a two electron quantum dot, one spin up and one spin down electron. The three first orbitals (six electrons) $\psi_{20}, \psi_{11}, \psi_{02}$ are needed to fill the second shell. Six orbitals (twelve electrons) are needed to fill the third shell and so on.*

in $\hat{H}\Psi(\mathbf{R})$ at $\mathbf{r}_i = \mathbf{r}_j$ (ref). We use a Jastrow factor with one variational parameter β ,

$$\begin{aligned} \mathcal{J}^\beta(\mathbf{R}) &= \prod_{i<j} \mathcal{J}_{ij} \\ \mathcal{J}_{ij} &= \exp\left(\frac{a_{ij}r_{ij}}{1 + \beta r_{ij}}\right) \end{aligned} \quad (29)$$

$a_{ij} = 1$ if the spins of particle i and j are parallel and $a_{ij} = \frac{1}{3}$ if the spins are perpendicular. See ref. [6] for a detailed derivation of the Jastrow factor.

The wave functions $\psi^{\alpha n_x, n_y}(\mathbf{R})$ is $n + 1$ times degenerated in the n 'th energy level (starting at 0). In the s 'th full shell we then have $2 \sum_{a=1}^s a$ particles. The basis states used in the different energy levels are listed in table (1).

According to an article written by Tauts (ref tauts), the energy of the 2 particle $\omega = 1$ quantum dot should be exactly 3 Hartrees. The kinetic energy *** (e without the) ($\hat{H} \rightarrow -\nabla^2/2$) of the slater matrices can be shown to be (ref,appendix): 2ω Hartrees fo two particles, 10ω Hartrees for 6 particles, 28ω Hartrees for 12 particles and 60 Hartrees for 20 particles. (formula)?. We have used this values to validate the part of the code involving the slater matrices.

5 Minimization of the wave functions

The minimization was performed using the stochastic gradient approach (SGA) [3, 4, 8]. This is a stochastic procedure which minimizes the gradient of the energy using a large number of sampling points. The variational parameters are updated iteratively using the formula

$$(\alpha_{n+1}, \beta_{n+1}) = (\alpha_n, \beta_n) - \gamma_n (\partial_\alpha, \partial_\beta) \langle E_L \rangle \quad (30)$$

The variable steplength γ_n is a system dependent parameter that must fulfill

$$\sum_{n=0}^{\infty} \gamma_n^2 < \sum_{n=0}^{\infty} \gamma_n = \infty, \quad \gamma_n > 0, \quad \lim_{n \rightarrow \infty} \gamma_n \rightarrow 0. \quad (31)$$

The SGA is based on the Robbins-Monroe theorem which states that under certain mathematical conditions $\lim_{n \rightarrow \infty} (\alpha_n, \beta_n)$ (30) are guaranteed to converge towards the minima $(\alpha_{min}, \beta_{min})$. The mathematical conditions include (31) and some constraints on the objective function which is typically met by QMC wavefunctions. The basis for the theory is discussed in [8] with references to the original articles.

I have used a γ_n on the form

$$\gamma_n = (k * n + l)^{-0.8}, \quad (32)$$

where k and l are parameters that is calculated as a function of the average values of $|\partial_\alpha \langle E_L \rangle|$ and $|\partial_\beta \langle E_L \rangle|$ during the first iterations. l is initially chosen to be 0, and k to be 1. There is one counter n for each of the parameters α, β which is updated every time the gradient changes its sign. The reason for this is that it is assumed that whenever the gradient changes its sign, the system is close to it's minima[4].

The maximum step length is set to 0.1. This stabilizes the simulation during the initial phase when γ_n is small and the step length might be large. After the thermalization phase γ_n is chosen so that the average step length becomes much smaller than 0.1.

The approximate value of the gradient $(\partial_\alpha, \partial_\beta) \langle E_L \rangle$ is obtained during a short MC run where $m = \mathcal{O}(10)$ uncorrelated samples are

collected. I have used m different walkers that collect one sample each every iteration. I also tried to use one walker collecting m samples every iteration, but the high correlation between the samples leads to bad results if the number of samples m is much smaller than the correlation length ($\mathcal{O}(10^3)$). After my experience, using many uncorrelated walkers instead of one speeds up the convergence of the algorithm considerably.

See algorithm ?? for a detailed pseudo code.

The algorithm proved efficient when applied on the full shell quantum dots. The optimization of the two particle quantum dots is done in a few seconds, the six particle quantum dot in less than thirty seconds and the twelve particle QD in about four minutes. All with an statistical error close to 10^{-3} .

Note that the optimization converges much faster than one can calculate the energy. For example, to optimize the two particle quantum dot with an error in the variational parameters $\sim 10^{-3}$, only a few hundred thousand samples is needed (see figure ?), with a minima that is accurate to $\mathcal{O}(10^{-5})$ Hartrees. To obtain the energy with an error $\sim 10^{-5}$ Hartrees about 10^8 MC-samples are needed.

6 Implementation and optimization

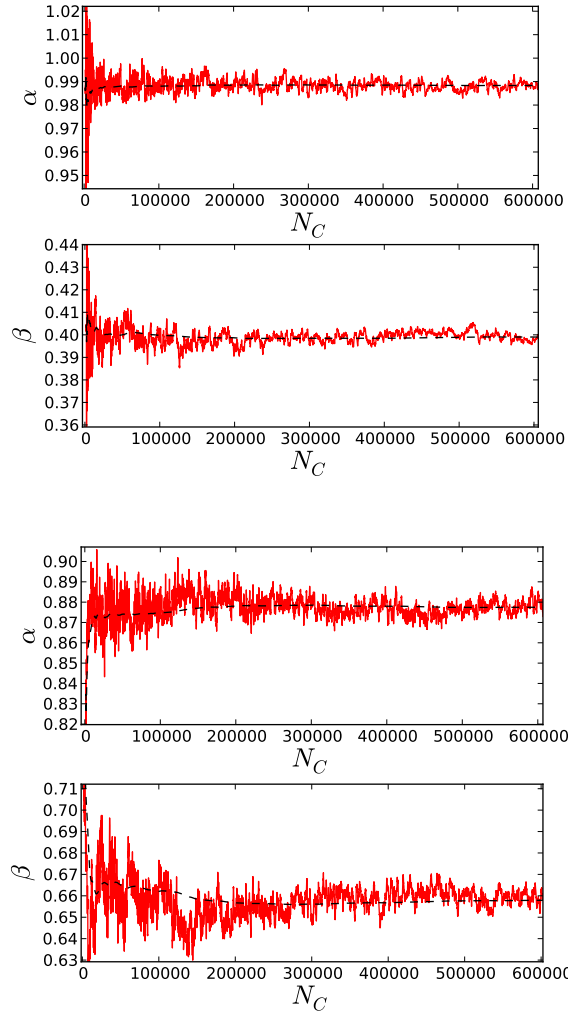
6.1 Random number generators

I have used the random number generator `DRanNormalZig32` for generating normal distributed random numbers and `DRan_MWC8222` for generating uniformly distributes random numbers. The random number generators are programmed by Jurgen A. Doornik [1].

6.2 Optimizing the Slater

The slater ratio $\mathcal{D}(\mathbf{R}_{n+1})/\mathcal{D}(\mathbf{R}_n)$ can be calculated as an $\mathcal{O}(n)$ operation if we know the inverse matrix $\mathcal{D}^{-1}(\mathbf{R}_n)$. If only one row of the determinant is updated, the inverse can be updated in $\mathcal{O}(n^2)$ op-

Figure 2: Minimization of a 6 particle (top) and a 12 particle (bottom) quantum dot using an $\omega = 1$ and a $\delta t = 0.05$. N_C is the total number of MC-samples. The red line shows the α and β values at N_C , while the black line shows the accumulated mean. The means are quickly converging towards the optimal results ($\beta = 0.399$, $\alpha = 0.988$) (top) and ($\beta = 0.877$, $\alpha = 0.657$) (bottom), only using a total of total of 6×10^5 samples.



erations using the Sherman-Morrison formula [2]

$$D_{kj}^{-1}(\mathbf{R}_{n+1}) = D_{kj}^{-1}(\mathbf{R}_n) - \frac{1}{R} D_{ki}^{-1}(\mathbf{R}_n) \times \sum_{l=1} D_{il}(\mathbf{R}_{n+1}) D_{lj}^{-1}(\mathbf{R}_n) \quad (33)$$

for $j \neq i$ and

$$D_{kj}^{-1}(\mathbf{R}_{n+1}) = \frac{1}{R} D_{ki}^{-1}(\mathbf{R}_n) \quad (34)$$

if $j = i$. R is the ratio between the old and the new determinant. Otherwise, calculating a determinant scales as $\mathcal{O}(n^3)$ ¹. As we will see, computing the gradient and the Laplacian of the determinant are also much faster when the inverse Slater matrix is known. In particular, evaluating the gradient becomes $\mathcal{O}(n^2)$ instead of $\mathcal{O}(n^4)$

The metropolis-hastings test is performed each cycle, while the energy and other observables are updated when we have cycled through all electrons. Evaluating the energy every time one electron has been moved would lead to a much higher correlation between the samples. Since the evaluation of the local energy is an expensive operation this strategy leads to a slower convergence in terms of cpu time.

The inverse D^{-1} of a matrix D can be written as the cofactor matrix $C/|D|$. C has the elements $C_{ij} = |S_{ij}|$ where S_{ij} is a sub matrix of D with the i 'th row and the j 'th column removed. The determinant $|D|$ can be expressed as

$$1 = \sum_j D_{ij} D_{ji}^{-1} = \sum_j D_{ij} \frac{C_{ji}}{|D|} \Rightarrow \sum_j D_{ij} C_{ji} = |D| \quad (35)$$

From the definition of C_{ij} , we see that the i 'th column of $D^{-1} = C/|D|$ is unchanged if we change the i 'th row of D . Therefore,

$$\begin{aligned} \text{if } D'_{ij} &= D_{ij} \forall i \neq k \\ \sum_k D'_{kj} D_{jk}^{-1} &= \sum_k D'_{kj} D_{jk}^{-1} = \sum_k D'_{kj} \frac{C_{jk}^{-1}}{|D|} \\ \Rightarrow \sum_k D'_{kj} D_{jk}^{-1} &= \frac{|D'|}{|D|} \end{aligned} \quad (36)$$

¹The most commonly used methods: I.e. taking the determinant of the LUP-decomposed matrix.

Eq. (36) can be used to calculate the ratios

$$\frac{\mathcal{D}^\alpha(\mathbf{R}')}{D^\alpha(\mathbf{R})} = \sum_{ij} (\mathcal{D}_{ij}) \mathcal{D}_{ji}^{-1} \quad (37)$$

$$\frac{\nabla^2 \mathcal{D}^\alpha(\mathbf{R})}{D^\alpha(\mathbf{R})} = \sum_{ij} (\partial_{r_i}^2 \mathcal{D}_{ij}) \mathcal{D}_{ji}^{-1} \quad (38)$$

$$\frac{\nabla_i \mathcal{D}^\alpha(\mathbf{R})}{D^\alpha(\mathbf{R})} = \sum_j (\partial_{r_i} \mathcal{D}_{ij}) \mathcal{D}_{ji}^{-1} \quad (39)$$

since all expressions can be written as a sum of slater determinants where only one row is changed. \mathbf{R}' is equal to \mathbf{R} except for the coordinates of one electron. The expressions for $\partial_{r_i} \mathcal{D}_{ij}$ and $\partial_{r_i}^2 \mathcal{D}_{ij}$ is calculated by evaluating the analytical expressions

$$\begin{aligned} \partial_{r_i}^2 \mathcal{D}_{ij} &= \psi_{n_{xi}, n_{yi}}((x_j, y_j), \alpha) \omega \alpha \\ &\quad \left(\frac{\omega}{2} (x_j^2 + y_j^2) - 2 \right. \\ &\quad + 4n_{xi}(n_{xi} - 1) \frac{H_{n_{xi}-2}(\sqrt{\omega\alpha}x_j)}{H_{n_{xi}}(\sqrt{\omega\alpha}x_j)} \\ &\quad + 4n_{yi}(n_{yi} - 1) \frac{H_{n_{yi}-2}(\sqrt{\omega\alpha}y_j)}{H_{n_{yi}}(\sqrt{\omega\alpha}y_j)} \\ &\quad - 4x\sqrt{\omega\alpha} n_{xi} \frac{H_{n_{xi}-1}(\sqrt{\omega\alpha}x_j)}{H_{n_{xi}}(\sqrt{\omega\alpha}x_j)} \\ &\quad \left. - 4y\sqrt{\omega\alpha} n_{yi} \frac{H_{n_{yi}-1}(\sqrt{\omega\alpha}y_j)}{H_{n_{yi}}(\sqrt{\omega\alpha}y_j)} \right) \quad (40) \end{aligned}$$

$$\begin{aligned} \partial_{r_i} \mathcal{D}_{ij} &= \psi_{n_{xi}, n_{yi}}((x_j, y_j), \alpha) \\ &\quad \left(\mathbf{e}_x (2n_{xi}\sqrt{\omega\alpha} \frac{H_{n_{xi}-1}(\sqrt{\omega\alpha}x_j)}{H_{n_{xi}}(\sqrt{\omega\alpha}x_j)} - x\omega\alpha) \right. \\ &\quad \left. \mathbf{e}_y (2n_{yi}\sqrt{\omega\alpha} \frac{H_{n_{yi}-1}(\sqrt{\omega\alpha}y_j)}{H_{n_{yi}}(\sqrt{\omega\alpha}y_j)} - y\omega\alpha) \right) \quad (41) \end{aligned}$$

The expressions is derived using the same strategy as we have used when $(\partial_\alpha)\psi_i(\mathbf{r}_j)$ is derived in appendix A. The full derivation is found in [6].

Note only one of the determinants \mathcal{D}_\downarrow , \mathcal{D}_\uparrow needs to be evaluated when a single particle is moved.

6.3 Optimizing the Jastrow

There are only $n(n-1)/2$ distinct values of r_{ij} . Therefore, all values can be stored in a lower tri-di-

agonal matrix with zeros on the diagonal.

$$P = \begin{pmatrix} 0 & 0 & 0 & \dots \\ r_{21} & 0 & 0 & \\ r_{31} & r_{32} & 0 & \\ r_{41} & r_{42} & r_{43} & \\ \vdots & & & \end{pmatrix} \quad (42)$$

If we move one electron, only $(n-1)$ elements has to be updated.

The electron-electron potential can be directly calculated by summing the $(n-1)$ inverse elements of P . The Jastrow ratio can be calculated

$$\frac{\mathcal{J}^\beta(\mathbf{R}')}{\mathcal{J}^\beta(\mathbf{R})} = \prod_{i=k \text{ or } j=k} \frac{\mathcal{J}_{ij}^\beta(\mathbf{R}')}{\mathcal{J}_{ij}^\beta(\mathbf{R})} \quad (43)$$

when only the electron with coordinates \mathbf{r}_k has been moved. This expression can be efficiently evaluated as the exponential of the sum of $2(n-1)$ elements

$$\frac{\mathcal{J}^\beta(\mathbf{R}')}{\mathcal{J}^\beta(\mathbf{R})} = \exp \left(\sum_{i=k \text{ or } j=k} j(r_{ij}) - j(r'_{ij}) \right) \quad (44)$$

$$j(r_{ij}) = \frac{a_{ij}r_{ij}}{1 + \beta r_{ij}}. \quad (45)$$

the gradient: subtract $n-1$ old values, add $n-1$ new values. Large optimization. Each element in the vector can be calculated using the formula

$$\frac{\partial_{x_i} \mathcal{J}^\beta}{\mathcal{J}^\beta} = \sum_{k=1}^{i-1} \frac{a_{ki}(x_i - x_k)}{r_{ki}(1 + \beta r_{ik})^3} + \sum_{k=i+1}^{n_p} \frac{a_{ik}(x_i - x_k)}{r_{ik}(1 + \beta r_{ki})^3} \quad (46)$$

where $r_i = (x_i, y_i)$. The formula looks identical if $x \rightarrow y$. If one particle is moved, only $(n_p - 1)$ distances r_{ik} is changed. Therefore the gradient can be calculated by adding the $n-1$ new elements and subtracting the corresponding old ones.

The Laplacian are given by the formula

$$\begin{aligned} \frac{\nabla^2 \mathcal{J}^\beta}{\mathcal{J}^\beta} &= \sum_{k=1}^{i-1} \frac{a_{ki}(1 + \beta r_{ik})}{r_{ki}(1 + \beta r_{ik})^3} \\ &\quad + \sum_{i=1}^{n_p} \frac{a_{ik}(1 + \beta r_{ik})}{r_{ik}(1 + \beta r_{ki})^3} + \left(\frac{\partial_{x_i} \mathcal{J}^\beta}{\mathcal{J}^\beta} \right)^2 \quad (47) \end{aligned}$$

Note that only half of the elements needs to be calculated since $r_{kj} = r_{jk}$. A detailed derivation of these expressions is found in [6]

6.4 Implementing $(\partial_\alpha, \partial_\beta)\langle E_L \rangle$

The gradient are calculated using an analytical expression for the energy. [7]

$$\begin{aligned} & (\partial_\alpha, \partial_\beta) \langle E_L \rangle = \\ & 2 \left\langle \frac{(\partial_\alpha, \partial_\beta) \Psi}{\Psi} E_L \right\rangle - 2 \left\langle \frac{(\partial_\alpha, \partial_\beta) \Psi}{\Psi} \right\rangle \langle E_L \rangle, \end{aligned} \quad (48)$$

where

$$\begin{aligned} & \frac{(\partial_\alpha, \partial_\beta) \Psi(\mathbf{R}, \alpha, \beta)}{\Psi(\mathbf{R}, \alpha, \beta)} \\ & = \left(\frac{\partial_\alpha \mathcal{D}(\mathbf{R}, \alpha)}{\mathcal{D}(\mathbf{R}, \alpha)}, \frac{\partial_\beta \mathcal{J}(\mathbf{R}, \beta)}{\mathcal{J}(\mathbf{R}, \beta)} \right) \end{aligned} \quad (49)$$

From the definition of the Jastrow eq. (..) we can derive

$$\frac{\partial_\beta \mathcal{J}(\mathbf{R}, \beta)}{\mathcal{J}(\mathbf{R}, \beta)} = \sum_{k < l} -a \left(\frac{r_{kl}}{1 + \beta r_{kl}} \right)^2 \quad (50)$$

The expression is calculated as a sum over the relevant distances $r_{kl} = |\mathbf{r}_k - \mathbf{r}_l|$ which is already stored in the matrix P (42). The gradient of alpha is calculated using the expression:

$$\frac{\partial_\alpha \mathcal{D}(\mathbf{R}, \alpha)}{\mathcal{D}(\mathbf{R}, \alpha)} = \sum_i \sum_j (\partial_\alpha \psi_i(\mathbf{r}_j, \alpha)) \mathcal{D}_{ji}^{-1} \quad (51)$$

$$\partial_\alpha \psi_i(\mathbf{r}_j, \alpha) =$$

$$\begin{aligned} & \psi_{n_{xi}, n_{yi}}((x_j, y_j), \alpha) \left(n_{xi} \sqrt{\frac{\omega}{\alpha}} \frac{H_{n_{xi}-1}(\sqrt{\omega\alpha}x_j)}{H_{n_{xi}}(\sqrt{\omega\alpha}x_j)} \right. \\ & \left. + n_{yi} \sqrt{\frac{\omega}{\alpha}} \frac{H_{n_{yi}-1}(\sqrt{\omega\alpha}y_j)}{H_{n_{yi}}(\sqrt{\omega\alpha}y_j)} - \frac{\omega}{2}(x_j^2 + y_j^2) \right) \end{aligned} \quad (52)$$

Here $r_j \rightarrow (x_j, y_j)$ and $i \rightarrow n_{xi}, n_{yi}$. The values $\partial_\alpha \psi_i(\mathbf{r}_j, \alpha)$ are already stored in the matrix \mathcal{D} , and does not have to be calculated. All of the analytical expressions above is derivated in appendix A.

7 Results

7.1 Scaling and time consume

I have used the package GNU gprof to profile my program.

The cpu time spent on running the code for $n_p = \{2, 6, 12, 20\}$ particles is listed in table 2. A

curve fit to this data indicates that the algorithm scales as $\mathcal{O}(n_p^{2.55})$. The theoretical limit is $\mathcal{O}(n_p^2)$ if one ignores the evaluation of the orbitals and their derivatives.

In table 2 I have also listed the percentage of the total floating point operation spent on the three most expensive functions. The evaluation of the first derivatives of the single particle orbitals soon becomes the most time consuming part of the code, since the mathematical expressions for the orbitals eq. (27) gets more complex as the orbital energies increases. The problem is that half of these derivatives are calculated each time one particle is moved. It is possible to store the values, reducing the number of analytical expressions that has to be calculated $n/2$ times (?). This did not pay off when I tried to implement it since the number of variables that has to be stored is too high $(1/2(n_p)^2)$ for our system). The copying and accessing of the elements when accepting or rejecting a move is simply too expensive.

This shows us how important it is to use the analytical derivatives to optimize the code. For example, for a two particle quantum dot, the evaluation of the derivatives took about 23 times longer when using numerical instead of analytical derivatives.

7.2 Convergence rate: DMC vs VMC

7.3 Systematic errors

Systematic time step error see HAMMOND [2] (monte carlo methods in ab initio quantum chemistry) This is not the error imposed by the discretization of the Langevin equation, since this is corrected by the Metropolis-Hastings algorithm. !!

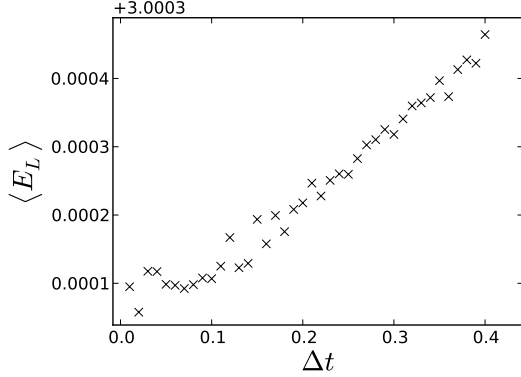
7.4 Numerical results

The results are summarized in tables 4 and 5. I have used the values $\omega = 0.28, 0.5, 1.0$ and $\Delta t = 0.01, 0.25, 0.05$ to be able to compare my results with ref(...). I have also calculated the values using the same number of cycles. The results are well consistent with the results of (refLE),..

Speed tests.

The timestep error: We have calculated all energy with three different values for Δt . The timestep error seems to be of the same order as

Figure 3: Energy as a function of Δt for a 2 particle quantum dot with $\alpha = 0.988, \beta = 0.399, \omega = 1$. All energies are calculated using 10^7 samples. The energy are plotted for 40 values of Δt between 0.01 and 0.4.



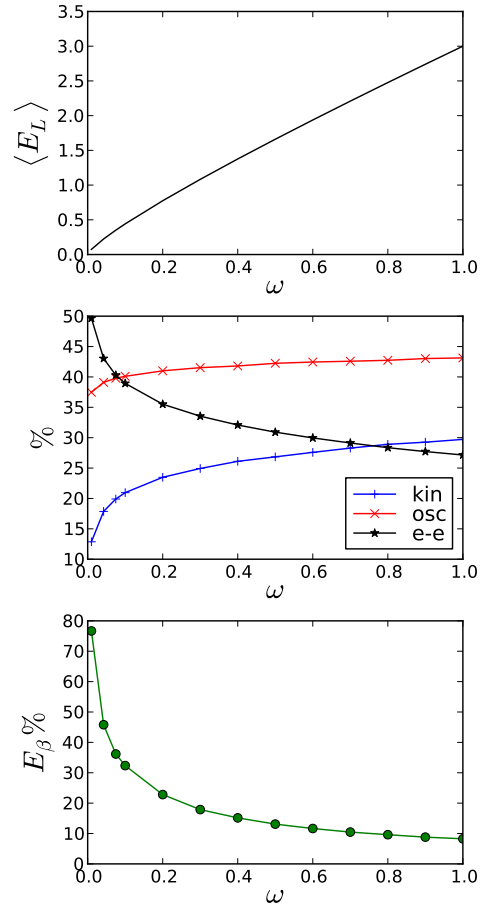
the statistical error for our values of Δt . There is also a general trend in the variance and the statistical error as a function of Δt . The variance and the statistical error increases as Δt gets smaller. The correlation length((plots?)) are generally increasing when Δt gets smaller, probably explaining the increase in the variance and statistical error since the system will use a longer time to equilibriate after random fluctuations away from equilibrium (eg. if a sequence of mc-steps with a very low metropolis probability are accepted).

7.5 Verification of the results (literature,articles)

can see that the timestep error a

The SGA algorithm works well. First of all, it seems as the algorithm is working for different systems. Some parameter tuning seems inevitable for any new system. This can be solved by implementing some sort of line search along the gradient $\partial \dots$. One possibility is to implement a Newtonian approach to estimate the minima for each iteration. The hessian matrix can be evaluated analytically for each new configuration of the system [ref]. This approach has many advantages. In particular for trial functions with asymmetrical minima, where the gradient is larger on one side of the minima

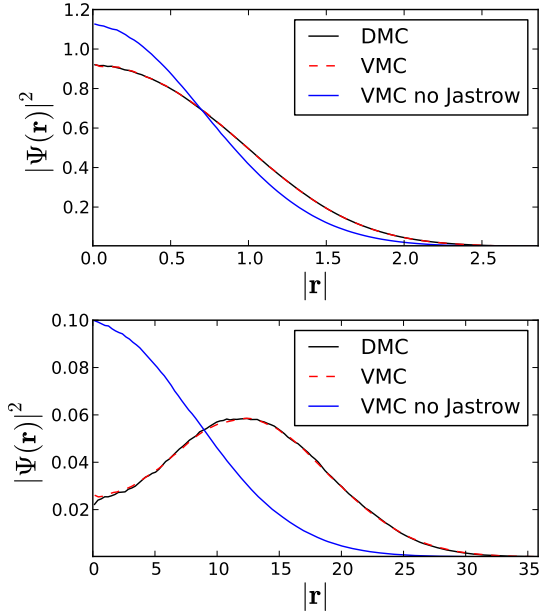
Figure 4: Different quantities are plotted as a function of ω for a 2 particle quantum dot with the optimal variational parameters. All energies are calculated using 10^7 samples and $\Delta t = 0.05$. Top plot: The total energy. Middle plot: Blue line (—+): kinetic energy, red line (—×) oscillator potential energy, black line (—*) electron-electron potential energy. Percentage of total energy for each ω plotted. Bottom plot: The Jastrow energy as a percentage of the total energy.



then on the other. For an algorithm without an adaptive step size, the step length will have to be very small to give accurate results.

Two reasons: 1) mean value of variational parameters taken gives higher stability. The algorithm as it is implemented by (refs) Does not take the mean, but lets γ increase faster to obtain a

Figure 5: The single particle densities for a two particle dot with $\omega = 1$ (top plot) and $\omega = 0.01$ (bottom plot). The single particle densities are calculated with and without the Jastrow function (VMC). The DMC density is very almost identical to the VMC density, showing that most correlations are accounted for by the Jastrow. These plots also demonstrates that the correlation effects included in the Jastrow function are more important when ω gets small.



A Pseudocode: The SGA algorithm.

My implementation of the SGA. At $n = N_A/N_B$ start/stop collecting mean value of gradient. At $n = N_B$, k, l is calculated. At N_C the system should be thermalized and we start collecting data. v_{min} is the lower limit of the variational parameters α, β . d_{max} is the upper limit for the length of a step in each variational parameter. k, l are dependent on D, E, F . E is the (negative) exponent limited by eq. (31). I have used $m = 30$, $N_A = 50$, $N_B = 120$, $N_C = 200$, $v_{min} = 0.01$, $d_{max} = 0.1$, $D = 300$, $E = 0.8$, $F = 20$ in all calculations.

N_p	cpu time	d1	jasgrad	slagrad
2	5.85s	4%	10%	6%
6	54.7s	10%	16%	14%
12	252s	20%	14%	13%
20	920s	37%	11%	16%

Table 2: Running time for $\omega = 1$ quantum dots with N_p particles and the optimal values for the variational parameters. All runs done on one 2.4MHz processor with 10^7 iterations. The percentage of time consume of the most expensive functions are listed in the table. (d1) :evaluation of the first derivatives of the orbitals, (jasgrad) :evaluation of the gradient of the Jastrow factor, (slagrad) :evaluation of the gradient of the Slater matrix (d1 not included).

N_p	ω	α_{min}	β_{min}
2	1.0	0.9886	0.3987
2	0.28	0.9714	0.2531
2	0.01	0.8965	0.07248
6	1.0	0.9245	0.5561
6	0.28	0.8736	0.3259
6	0.01	0.6141	0.09327
12	1.0	0.8775	0.6574
12	0.28	0.8090	0.3788
12	0.01	0.4935	0.1128
20	1.0	0.8384	0.7392
20	0.28	0.7500	0.4274
20	0.01	0.3958	0.1418

Table 3: Minimization results. N_p is the number of particles and N_s is the number of cycles used to obtain the energies and the gradient during the minimization. α_{min} and β_{min} is the values of the variational parameters at the minima. We have used a $\Delta t = 0.05$ and 10^5 iterations in the SGA loop.

****Initialize m walkers****

$k_\alpha \leftarrow 0, l_\alpha \leftarrow 1$

$k_\beta \leftarrow 0, l_\beta \leftarrow 1$

loop

$n = n + 1$

for all walkers do

propose new position for walker

accept/reject move (metropolis test)

sample $(\partial_\alpha, \partial_\beta) E_L, E_L, (\partial_\alpha, \partial_\beta) E_L * E_L$

N_p	ω	Δ_t	$\langle E_L \rangle$
2	1.0	0.007	3.00001(2)
2	1	0.005	3.00000(2)
2	1	0.001	3.00001(3)
2	0.28	0.007	1.02167()
2	0.28	0.005	1.02166()
2	0.28	0.001	1.20167()
2	0.01	0.007	0.07385()
2	0.01	0.005	0.07386()
2	0.01	0.001	0.07386()
6	1.0	0.9245	0.5561
6	0.28	0.8736	0.3259
6	0.01	0.6141	0.09327
12	1.0	0.8775	0.6574
12	0.28	0.8090	0.3788
12	0.01	0.4935	0.1128
20	1.0	0.8384	0.7392
20	0.28	0.7500	0.4274
20	0.01	0.3958	0.1418

Table 4: *Minimization results.* N_p is the number of particles and N_s is the number of cycles used to obtain the energies and the gradient during the minimization. α_{min} and β_{min} is the values of the variational parameters at the minima. We have used a $\Delta t = 0.05$ and 10^5 iterations in the SGA loop.

```

end for
 $(\partial_\alpha, \partial_\beta) \langle E_L \rangle \leftarrow$  use eq. (48)
for  $\nu = \{\alpha, \beta\}$  do
  if  $n \in [N_A, N_B]$  then
     $\Delta_\nu \leftarrow \Delta_\nu + |\partial_\nu \langle E_L \rangle|$ 
  end if
  if  $n = N_B$  then
     $l_\nu \leftarrow D^E(\Delta_\nu / (N_B - N_A))^{-E}$ 
     $k_\nu \leftarrow l_\nu / F$ 
     $n \leftarrow 0$ 
  end if
  if  $|\partial_\nu \langle E_L \rangle| > d_{max}$  then
     $\nu \leftarrow \nu - d_{max} * \partial_\nu \langle E_L \rangle / |\partial_\nu \langle E_L \rangle|$ 
  else
     $\nu \leftarrow \nu - \partial_\nu \langle E_L \rangle (k_\nu * n + l_\nu)^{-E}$ 
  end if
  if  $\nu < v_{min}$  then
     $\nu \leftarrow v_{min}$ 
  end if
  if  $\partial_\nu \langle E_L \rangle$  has changed its sign then
     $n_\nu \leftarrow n_\nu + 1$ 

```

```

    end if
  end for
  if  $n > N_C$  then
    accumulate mean values of  $\alpha, \beta$ 
  end if
end loop

```

B Derivation of the analytical expressions for $(\partial_\alpha, \partial_\beta) \langle \hat{H} \rangle$

The derivative of the jastrow factor can be written

$$\begin{aligned} \partial_\beta \mathcal{J}(\mathbf{R}, \beta) &= \partial_\beta \prod_{i < j} \mathcal{J}_{ij}(\mathbf{R}, \beta) \\ &= \sum_{k < l} \partial_\beta \mathcal{J}_{kl}(\mathbf{R}, \beta) \prod_{i < j, (i,j) \neq (k,l)} \mathcal{J}_{ij}(\mathbf{R}, \beta). \end{aligned} \quad (53)$$

We are interested in the ratio

$$\frac{\partial_\beta \mathcal{J}(\mathbf{R}, \beta)}{\mathcal{J}(\mathbf{R}, \beta)} = \frac{\partial_\beta \prod_{i < j} \mathcal{J}_{ij}(\mathbf{R}, \beta)}{\prod_{i < j} \mathcal{J}_{ij}(\mathbf{R}, \beta)} = \sum_{k < l} \frac{\partial_\beta \mathcal{J}_{kl}(\mathbf{R}, \beta)}{\mathcal{J}_{kl}(\mathbf{R}, \beta)}. \quad (54)$$

Evaluation of the terms $\partial_\beta \mathcal{J}_{kl}(\mathbf{R}, \beta)$ is straight forward, and the full expressions becomes

$$\frac{\partial_\beta \mathcal{J}(\mathbf{R}, \beta)}{\mathcal{J}(\mathbf{R}, \beta)} = \sum_{k < l} -a_{kl} \left(\frac{r_{kl}}{1 + \beta r_{kl}} \right)^2. \quad (55)$$

The differentiation $\partial_\alpha \mathcal{D}(\mathbf{R}, \alpha)$ can be written on the following form: (using partial derivation(??))

$$\begin{aligned} \partial_\alpha \mathcal{D}(\mathbf{R}, \alpha) &= \begin{vmatrix} \partial_\alpha \psi_1(\mathbf{r}_1, \alpha) & \partial_\alpha \psi_1(\mathbf{r}_2, \alpha) & \partial_\alpha \psi_1(\mathbf{r}_3, \alpha) & \dots \\ \psi_2(\mathbf{r}_1, \alpha) & \psi_2(\mathbf{r}_2, \alpha) & \psi_2(\mathbf{r}_3, \alpha) & \\ \psi_3(\mathbf{r}_1, \alpha) & \psi_3(\mathbf{r}_2, \alpha) & \psi_3(\mathbf{r}_3, \alpha) & \\ \vdots & & & \end{vmatrix} \\ &+ \begin{vmatrix} \psi_1(\mathbf{r}_1, \alpha) & \psi_1(\mathbf{r}_2, \alpha) & \psi_1(\mathbf{r}_3, \alpha) & \dots \\ \partial_\alpha \psi_2(\mathbf{r}_1, \alpha) & \partial_\alpha \psi_2(\mathbf{r}_2, \alpha) & \partial_\alpha \psi_2(\mathbf{r}_3, \alpha) & \\ \psi_3(\mathbf{r}_1, \alpha) & \psi_3(\mathbf{r}_2, \alpha) & \psi_3(\mathbf{r}_3, \alpha) & \\ \vdots & & & \end{vmatrix} \\ &+ \dots \end{aligned} \quad (56)$$

$$= \mathcal{D}^{(1)}(\mathbf{r}, \alpha) + \mathcal{D}^{(2)}(\mathbf{r}, \alpha) + \dots \quad (57)$$

Since only one row in the determinant $\mathcal{D}^{(i)}(\mathbf{R}, \alpha)$ is changed, following the same reasoning as in section

?.?, we can write

$$\frac{\mathcal{D}^{(i)}(\mathbf{R}, \alpha)}{\mathcal{D}(\mathbf{R}, \alpha)} = \sum_j (\partial_\alpha \psi_i(\mathbf{r}_j, \alpha)) \mathcal{D}_{ji}^{-1} \quad (58)$$

giving the expression

$$\frac{\partial_\alpha \mathcal{D}(\mathbf{R}, \alpha)}{\mathcal{D}(\mathbf{R}, \alpha)} = \sum_i \sum_j (\partial_\alpha \psi_i(\mathbf{r}_j, \alpha)) \mathcal{D}_{ji}^{-1}. \quad (59)$$

The terms $\partial_\alpha \psi_i(\mathbf{r}_j, \alpha)$ are derived below. We write $\psi_i(\mathbf{r}_j, \alpha) \rightarrow \psi_{n_{xi}, n_{yi}}(\mathbf{r}_j, \alpha)$. where n_{xi} and n_{yi} is quantum numbers equal to or larger than zero.

$$\begin{aligned} \psi_{n_{xi}, n_{yi}}((x_i, y_i), \alpha) \\ = H_{n_{xi}}(\sqrt{\omega\alpha}x_j) H_{n_{yi}}(\sqrt{\omega\alpha}y_j) \psi_{00}((x_i, y_i), \alpha) \end{aligned} \quad (60)$$

$H_n(x)$ is the hermite polynomials. Differentiation gives

$$\begin{aligned} \partial_\alpha \psi_{n_{xi}, n_{yi}}((x_i, y_i), \alpha) = \\ \left(\partial_\alpha H_{n_{xi}}(\sqrt{\omega\alpha}x_j) \right) H_{n_{yi}}(\sqrt{\omega\alpha}y_j) \psi_{00}((x_i, y_i), \alpha) \\ + H_{n_{xi}}(\sqrt{\omega\alpha}x_j) \left(\partial_\alpha H_{n_{yi}}(\sqrt{\omega\alpha}y_j) \right) \psi_{00}((x_i, y_i), \alpha) \\ + H_{n_{xi}}(\sqrt{\omega\alpha}x_j) H_{n_{yi}}(\sqrt{\omega\alpha}y_j) \left(\partial_\alpha \psi_{00}((x_i, y_i), \alpha) \right) \end{aligned} \quad (61)$$

We make use of the identity $\delta_x H_n(x) = 2nH_{n-1}(x)$,

$$\begin{aligned} \partial_\alpha H_n(\sqrt{\alpha\omega}x) &= \frac{\partial H_n(\sqrt{\alpha\omega}x)}{\partial(\sqrt{\alpha\omega}x)} \frac{\partial(\sqrt{\alpha\omega}x)}{\partial\alpha} \\ &= nx \sqrt{\frac{\omega}{\alpha}} H_{n-1}(\sqrt{\omega\alpha}x) \end{aligned} \quad (62)$$

Derivation of $\psi_{00}((x, y), \alpha)$ yields

$$\partial_\alpha \psi_{00}((x, y), \alpha) = -\frac{\omega}{2}(x^2 + y^2) \psi_{00}((x, y), \alpha). \quad (63)$$

We combine eq. (61), eq. (62) and eq. (63)

$$\begin{aligned} \partial_\alpha \psi_i(\mathbf{r}_j, \alpha) = \\ \psi_{n_{xi}, n_{yi}}((x_j, y_j), \alpha) \left(x_j n_{xi} \sqrt{\frac{\omega}{\alpha}} \frac{H_{n_{xi}-1}(\sqrt{\omega\alpha}x_j)}{H_{n_{xi}}(\sqrt{\omega\alpha}x_j)} \right. \\ \left. + y_j n_{yi} \sqrt{\frac{\omega}{\alpha}} \frac{H_{n_{yi}-1}(\sqrt{\omega\alpha}y_j)}{H_{n_{yi}}(\sqrt{\omega\alpha}y_j)} - \frac{\omega}{2}(x_j^2 + y_j^2) \right) \end{aligned} \quad (64)$$

References

- [1] Jurgen A Doornik. An improved ziggurat method to generate normal random samples, 2005.
- [2] B. L. Hammond, W. A. Lester, and P. J. Reynolds. *Monte Carlo Methods in Ab Initio Quantum Chemistry*. World Scientific, 1994.
- [3] A. Harju, B. Barbiellini, S. Siljamäki, R. M. Nieminen, and G. Ortiz. Stochastic gradient approximation: An efficient method to optimize many-body wave functions. *Phys. Rev. Lett.*, 79:1173–1177, Aug 1997.
- [4] Ari Harju. Variational monte carlo for interacting electrons in quantum dots. *Journal of Low Temperature Physics*, 140:181–210, 2005. 10.1007/s10909-005-6308-7.
- [5] Morten Hjorth-Jensen. Computational physics. Lecture Notes, Fys4411, University of Oslo, 2011.
- [6] Lars Eivind Lervåg. Vmc calculations of two-dimensional quantum dots. Master’s thesis, University of Oslo, 2010.
- [7] Xi Lin, Hongkai Zhang, and Andrew M. Rappe. Optimization of quantum monte carlo wave functions using analytical energy derivatives. *The Journal of Chemical Physics*, 112(6):2650–2654, 2000.
- [8] Daniel Andrew Nissenbaum. *The stochastic gradient approximation: an application to li nanoclusters*. PhD thesis, Northeastern University, 2008.
- [9] Anthony Scemama, Tony Lelièvre, Gabriel Stoltz, Eric Cancès, and Michel Caffarel. An Efficient Sampling Algorithm for Variational Monte Carlo. *The Journal of Chemical Physics*, 125(11):114105, 2006.

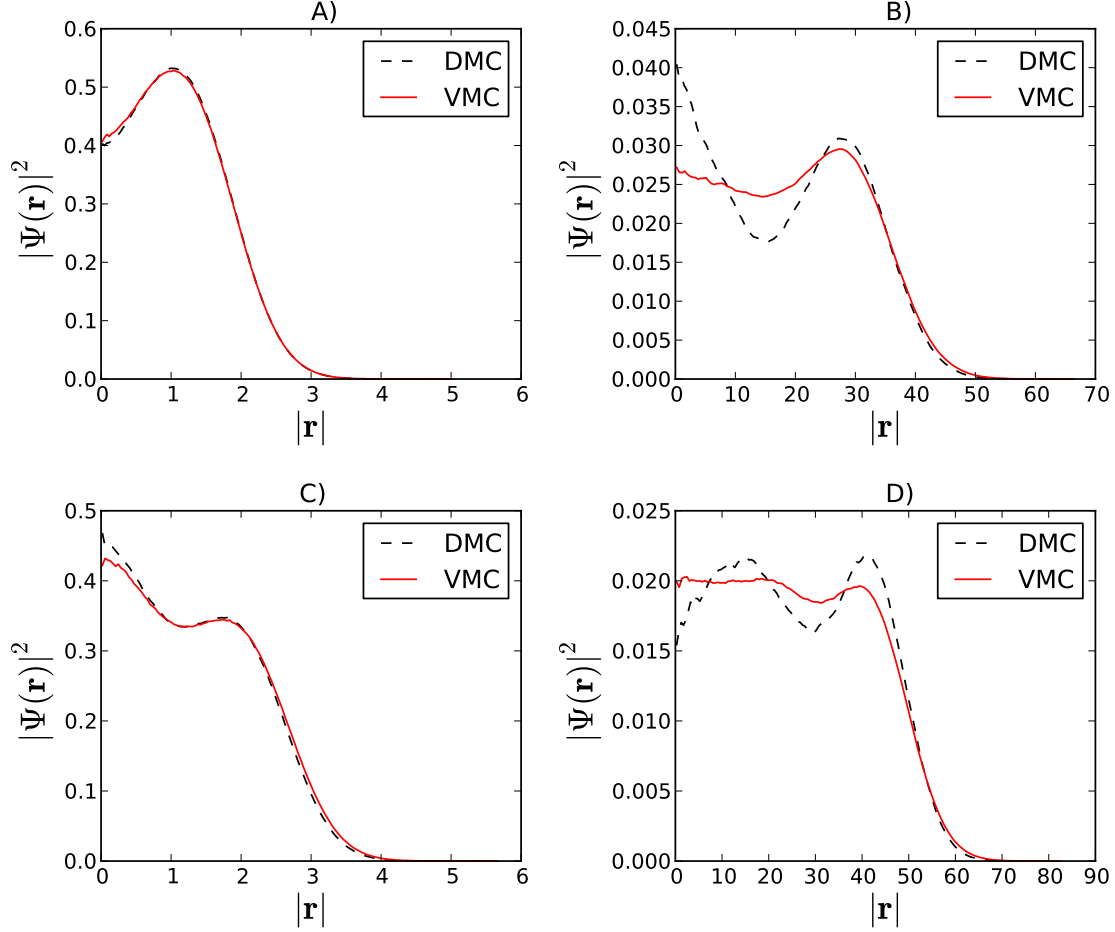


Figure 6: The single particle densities A) 6 particle QD with $\omega = 1$ B) 6 particle QD with $\omega = 0.01$ C) 12 particle QD with $\omega = 1$ D) 12 particle QD with $\omega = 0.01$. The correlations gets more important for smaller ω or a higher number of particles. The DMC density is very almost identical to the VMC density for the $\omega = 1$ plots (left), while the $\omega = 0.01$ plots (right) showing strong correlations effects that can not be accounted for by the 2 particle Jastrow.

N_p	ω	Δt	$\langle E_L \rangle$	σ^2	E_{kin}	E_{ocs}	E_{e-e}	R_A
2	1.0	0.05	3.00040(1)	0.0018334	29.7%	43.3%	27.1%	99.48%
		0.025	3.00036(2)	0.0018346				99.80%
		0.01	3.00035(3)	0.0018411				99.95%
2	0.28	0.05	1.02217(2)	7.4875×10^{-4}	24.7%	41.4%	33.9%	99.91%
		0.025	1.02218(2)	7.4943×10^{-4}				99.96
		0.01	1.02213(3)	7.5364×10^{-4}				99.99%
2	0.01	0.05	0.07407(1)	9.9381×10^{-6}	13.0%	37.6%	49.4%	> 99.99%
		0.025	0.07407(1)	9.9323×10^{-6}				> 99.99%
		0.01	0.07406(2)	9.8967×10^{-6}				> 99.99%
6	1.0	0.05	20.1905(1)	0.115568	17.7%	39.3%	43.0%	98.81%
		0.025	20.1904(2)	0.119353				99.52%
		0.01	20.1908(3)	0.123143				99.86%
6	0.28	0.05	7.6216(1)	0.017358	12.3%	37.4%	50.4%	99.81%
		0.025	7.6216(1)	0.017655				99.92%
		0.01	7.6215(2)	0.017838				99.98%
6	0.01	0.05	0.6984(1)	2.5323×10^{-4}	3.9%	34.6%	61.4%	> 99.99%
		0.025	0.6984(1)	2.5522×10^{-4}				> 99.99%
		0.01	0.6983(2)	2.4233×10^{-4}				> 99.99%
12	1.0	0.05	65.7906(2)	0.40948	13.4%	37.8%	48.8%	98.13%
		0.025	65.7903(3)	0.42128				99.22%
		0.01	65.7912(5)	0.44123				99.76%
12	0.28	0.05	25.6991(2)	0.060032	8.5%	36.2%	55.3%	99.70%
		0.025	25.6991(3)	0.061213				99.88%
		0.01	25.6996(4)	0.062310				99.96%
12	0.01	0.05	2.4980(2)	9.0817×10^{-4}	2.3%	33.3%	64.3%	> 99.99%
		0.025	2.4981(3)	9.1378×10^{-4}				> 99.99%
		0.01	2.4980(4)	8.9618×10^{-4}				> 99.99%
20	1.0	0.05	155.9949(3)	1.08344	11.0%	37.0%	52.0%	95.83%
		0.025	155.9952(4)	1.08273				97.41%
		0.01	155.9946(6)	1.13446				98.48%
20	0.28	0.05	62.0391(3)	0.14986	6.6%	35.6%	57.8%	98.66%
		0.025	62.0395(3)	0.15339				99.01%
		0.01	62.0398(6)	0.15632				99.44
20	0.01	0.05	6.1996(3)	0.0023159	1.6%	32.1%	66.3%	93.49%
		0.025	6.2002(4)	0.0023496				99.90%
		0.01	6.2002()	0.0029175				99.93%

Table 5: VMC results using the variational parameters in table (4). Minimization results. N_p is the number of particles and N_s is the number of cycles used to obtain the energies and the gradient during the minimization. All results are obtained using 10^8 MC - cycles. The error is estimated using blocking analysis of the data. The oscillator energies are quite stable, but the ratio between E_{kin} and E_{e-e} varies with ω . * * * Generally, the lower this ratio is, the more important are the correlations not included in the Jastrow. See plots.