Variational Monte Carlo studies of bosonic systems

Roar Emaus

March 21, 2016

Abstract

The purpose of this project was to use Variational Monte Carlo (VMC) calculations to find the ground state energy of a trapped bosonic system in a harmonic oscillator. Using Metropolis methods with and without interaction, looking for effiecency and accuracy using blocking as an error analysis. For the brute force and Importance Sampling methods without interaction the results were exact with our trial wavefunction. Using the analytical solutions of the Laplacians reduced the CPU-time usage significantlty and the solutions were without variance. The interacting method became problematic where only a small number of particles with the numerical solutions gave results that were similiar to the benchmarks recieved by email.

Contents

1 Introduction **3**

2 Method **3**
2.1 System 3
2.1.1 Interacting local energy 4
2.1.2 Brute force Metropolis 5
2.1.3 Importance sampling 6
2.1.4 Blocking 7
2.1.5 One body density 7

3 Results **8**
3.0.6 Brute force Metropolis 8
3.0.7 Importance sampling 8
3.0.8 Time comparison 10
3.0.9 Interacting Hamiltonian 11
3.0.10 Blocking 12
3.0.11 Particle distribution 13

4 Conclusion **13**

A Tables **14**
A.1 Brute force Metropolis algorithm 14
A.1.1 Analytical 2D 14
A.1.2 Analytical 3D 14
A.1.3 Numerical 2D 14
A.1.4 Numerical 3D 14
A.2 Metropolis algorithm with Importance sampling 15
A.2.1 Analytical 2D 15
A.2.2 Analytical 3D 15
A.2.3 Numerical 2D 15
A.2.4 Numerical 3D 15

B Links **16**

1 Introduction

The Bose-Einstein condensate in gases of trapped alkali atoms have low density and can therefore be studied considering two body collisions. This making it a good test for the accuracies of our methods. Here we are going to simulate 2 different systems with different number of particles in harmonic oscillator traps using a trial wavefunction [1], taking measurments for the ground state energy. The systems being with and without interaction, where both will be tested with brute force Metropolis and Importance Sampling. The report starts with a Method section to describe the theory behind, then introducing the results and a conclusion.

2 Method

2.1 System

Using the oscillators and wavefunctions found in DuBois and Glydes [1] article we have the following terms. A two-body Hamiltonian of this system of the form

$$H = \sum_{i}^{N} \left(\frac{-\hbar^2}{2m} \nabla_i^2 + V_{ext}(\boldsymbol{r}_i) \right) + \sum_{i<j}^{N} V_{int}(\boldsymbol{r}_i, \boldsymbol{r}_j) \tag{1}$$

where the external potential for the trap is for the spherical and elliptical part given by

$$V_{ext}(\boldsymbol{r}) = \begin{cases} \frac{1}{2} m\omega^2 r^2 & \text{Spherical} \\ \frac{1}{2} m[\omega^2(x^2+y^2) + \gamma^2 z^2] & \text{Elliptical} \end{cases} \tag{2}$$

where ω is the trap potential strength, and in the elliptical trap γ is the strength in z-direction. The internal potential which represents the repulsion when two boson gets close is

$$V_{int}(|\boldsymbol{r}_i - \boldsymbol{r}_j|) = \begin{cases} \infty & |\boldsymbol{r}_i - \boldsymbol{r}_j| \leq a \\ 0 & |\boldsymbol{r}_i - \boldsymbol{r}_j| > a \end{cases} \tag{3}$$

with a as the hard-core diamater of the bosons. The trial wave function we are using have the form

$$\Psi_T(\mathbf{R}) = \Psi_T(\mathbf{r}_1, \mathbf{r}_2, \ldots \mathbf{r}_N, \alpha, \beta) = \prod_i g(\alpha, \beta, \mathbf{r}_i) \prod_{i<j} f(a, |\mathbf{r}_i - \mathbf{r}_j|), \tag{4}$$

with α and β as variational parameters, and

$$g(\alpha, \beta, \boldsymbol{r}_i) = e^{-\alpha(x_i^2 + y_i^2 + \beta z_i^2)}$$

and

$$f(a, |\mathbf{r}_i - \mathbf{r}_j|) = \begin{cases} 0 & |\mathbf{r}_i - \mathbf{r}_j| \leq a \\ (1 - \frac{a}{|\mathbf{r}_i - \mathbf{r}_j|}) & |\mathbf{r}_i - \mathbf{r}_j| > a. \end{cases} \tag{5}$$

For the simplest cases where we set the boson size $a = 0$ and $\beta = 1$ the trial wavefunction becomes

$$\Psi_T(\boldsymbol{R}) = e^{-\alpha r^2} \tag{6}$$

With $r^2 = x^2+y^2+z^2$, $\omega = \gamma = 1$. Calculating the double derivative of the wavefunction returns

$$\nabla^2\Psi_T = \nabla^2 e^{-\alpha r^2} = \nabla - \alpha 2re^{-\alpha r^2} = 2\alpha e^{-\alpha r^2}(2\alpha r^2 - 1) \tag{7}$$

then inserting this result into the expression for the local energy yields

$$E_L(\boldsymbol{R}) = \frac{1}{\Psi_T(\boldsymbol{R})} H\Psi_T(\boldsymbol{R}) = 2\alpha(2\alpha r^2 - 1) \tag{8}$$

which is the analytical experssion for the local energy in a spherical trap without interaction.

2.1.1 Interacting local energy

To get the analytical solution for the interacting case we use the full $\Psi_T(\boldsymbol{R})4$, rewriting $g(\alpha, \beta, \boldsymbol{r}_i)$ and $f(a, |\boldsymbol{r}_i - \boldsymbol{r}_j|)$ with $r_{ij} = |\boldsymbol{r}_i - \boldsymbol{r}_j|$ to $\phi(\boldsymbol{r}_i)$ and $u(r_{ij}) = \ln f(r_{ij})$ respectivly.

$$\begin{aligned}
\nabla_k\Psi_T(\boldsymbol{R}) &= \nabla_k \prod_i \phi(\boldsymbol{r}_i) \exp\left[\sum_{i<j} u(r_{ij})\right] \\
&= \nabla_k\phi(\boldsymbol{r}_k) \prod_{i\neq k} \phi(\boldsymbol{r}_i) \exp\left[\sum_{i<j} u(r_{ij})\right] + \prod_i \phi(\boldsymbol{r}_i) \exp\left[\sum_{i<j} u(r_{ij})\right] \sum_{j\neq k} \nabla_k u(r_{ij})
\end{aligned}$$

And dertivating a second time gives us

$$\begin{aligned}
\nabla_k^2\Psi_T(\mathbf{R}) = {} & \nabla_k^2\phi(\boldsymbol{r}_k) \prod_{i\neq k} \phi(\boldsymbol{r}_i) \exp\left(\sum_{i<j} u(r_{ij})\right) \\
& + \nabla_k\phi(\boldsymbol{r}_k) \prod_{i\neq k} \phi(\boldsymbol{r}_i) \sum_{j\neq k} \nabla_k u(r_{ij}) \exp\left[\sum_{i<j} u(r_{ij})\right] \\
& + \nabla_k\phi(\boldsymbol{r}_k) \prod_{i\neq k} \phi(\boldsymbol{r}_i) \sum_{j\neq k} \nabla_k u(r_{ij}) \exp\left[\sum_{i<j} u(r_{ij})\right] \\
& + \left(\sum_{j\neq k} \nabla_k u(r_{ij}) \sum_{i\neq k} \nabla_k u(r_{ij})\right) \prod_i \phi(\boldsymbol{r}_i) \exp\left[\sum_{i<j} u(r_{ij})\right] \\
& + \left(\nabla_k \cdot \sum_{j\neq k} \nabla_k u(r_{ij})\right) \prod_i \phi(\boldsymbol{r}_i) \exp\left[\sum_{i<j} u(r_{ij})\right]
\end{aligned}$$

dividing by the wavefunction gives us

$$\frac{1}{\Psi_T(\mathbf{R})}\nabla_k^2\Psi_T(\mathbf{R}) = \frac{\nabla_k^2\phi(\boldsymbol{r}_k)}{\phi(\boldsymbol{r}_k)} + 2\frac{\nabla_k\phi(\boldsymbol{r}_k)}{\phi(\boldsymbol{r}_k)}\sum_{j\neq k}\nabla_k u(r_{ij}) + \left(\sum_{j\neq k}\nabla_k u(r_{ij})\sum_{i\neq k}\nabla_k u(r_{ij})\right) + \left(\nabla_k\cdot\sum_{j\neq k}\nabla_k u(r_{ij})\right)$$

Taking the gradient of $u(r_{ij})$ gives us

$$\nabla_k u(r_{ij}) = \frac{(\boldsymbol{r}_k - \boldsymbol{r}_j)}{r_{kj}}u'(r_{ij})$$

and substituting that into the equation gives

$$\frac{1}{\Psi_T(\mathbf{R})}\nabla_k^2\Psi_T(\mathbf{R}) = \frac{\nabla_k^2\phi(\boldsymbol{r}_k)}{\phi(\boldsymbol{r}_k)} + 2\frac{\nabla_k\phi(\boldsymbol{r}_k)}{\phi(\boldsymbol{r}_k)}\sum_{j\neq k}\frac{(\boldsymbol{r}_k - \boldsymbol{r}_j)}{r_{kj}}u'(r_{kj}) + \sum_{j\neq k}\frac{(\boldsymbol{r}_k - \boldsymbol{r}_j)}{r_{kj}}u'(r_{kj})\frac{(\boldsymbol{r}_k - \boldsymbol{r}_i)}{r_{ki}}u'(r_{ki}) + \sum_{j\neq k}\left(u''(r_{kj}) + \frac{2}{r_{kj}}u(r_{kj})\right)$$

I did not finalize the last term on my own, but this is the analytic local energy which will be used for the interacting bosons.

2.1.2 Brute force Metropolis

For the first part we use a Variational Monte Carlo program with a brute force Metropolis sampling to calculate the ground state energy. The Metropolis algorithm uses the probability density functions of the wave function, comparing the function in the old position $A_{j\to i}$ against the function after moving the system one iteration into a new, uniform random position chosen $A_{i\to j}$.

$$\frac{A_{j\to i}}{A_{i\to j}} = \frac{p_i T_{i\to j}}{p_j T_{j\to i}} \tag{9}$$

where T is the transition probability and p is the probability distribution. Given that the transition probability is independent of direction we are left with the

$$\frac{A_{j\to i}}{A_{i\to j}} = \frac{p_i}{p_j} \tag{10}$$

The Metropolis choice is to maximize the A values

$$A_{j\to i} = \min\left(1, \frac{p_i}{p_j}\right) \tag{11}$$

This is done by first placing the system in a random, Gaussian distributed position around 0 with a $\sigma = 1/\sqrt{2}$. Evaluating the wavefunction according to equation 6. Then choosing a random particle and dimension, and moving it with a Gaussian distribution to a new position. Do the same evaluation for the new position and compare

$$\frac{|\Psi_{New}|^2}{|\Psi_{Old}|^2}$$

Then by taking a random, uniformly distributed number between 0 and 1 and compare it to the ratio between the new and old wavefunction. If the ratio is larger we accept the new step, if not we revert back to the old position. Then lastly we sample the energy for the resulting system.

Listing 1: Brute force Metropolis

```
randomMove        = normalDistribution;
waveFunctionOld = my_waveFunction->evaluate();

my_particles[chosenParticle]->changePosition(chosenDimension, randomMove);

waveFunctionNew = my_waveFunction->evaluate();

waveFunctionsCompared = (waveFunctionNew*waveFunctionNew)/
                (waveFunctionOld*waveFunctionOld);

if (waveFunctionsCompared < uniformDistribution{
  my_particles[chosenParticle]->changePosition(chosenDimension, -randomMove);
  return false;
}
return true;
```

When using this method we get the result which is shown in table 1. This result is in exact correspondence to the analytical solution.

2.1.3 Importance sampling

To increase the relevance in our choice of movement we use importance sampling. Here there is a biased direction in the new step which is dependent on the trial wavefunction. The new position is given as the solutions to Langevin equation,

$$y = x + DF(x)\Delta t + \xi\sqrt{\Delta t} \tag{12}$$

Here x is the old position, Δt the step unit, D is the diffusion coefficient, $F(x)$ quantum force and ξ is a normal distributed random variable. The quantum force is given by

$$\mathbf{F} = 2\frac{1}{\Psi_T}\nabla\Psi_T \tag{13}$$

which gives the walker an incentive to go towards areas where the wavefunction is large. The new comparison is now

$$\frac{G(x, y, \Delta t)|\Psi_T(y)|^2}{G(y, x, \Delta t)|\Psi_T(x)|^2} \tag{14}$$

where

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

is the Greens function of the Fokker-Planck equation

$$\frac{\partial^2 P}{\partial t} = \sum_i D\frac{\partial}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i - \boldsymbol{F}_i)P(\boldsymbol{x},t)$$

2.1.4 Blocking

The stochastic nature of a Monte Carlo simulation opens up for the same analysis as we would do on experimental data. For uncorrelated measurements we could use

$$\sigma = \sqrt{\frac{1}{N}(\langle E^2\rangle - \langle E\rangle^2)}$$

as the standard deviation σ, but in the case of an interacting system we need to take into account the correlation between samples. Therefore we need to add a autocorrelation function with our variance [2] [1]

$$\sigma = \sqrt{\frac{1+2\tau\Delta t}{N}(\langle E^2\rangle - \langle E\rangle^2)}$$

where τ is the autocorrelation time, N number of samples and Δt the step length. To caluclate this correlation factor we use the method of blocking. By dividing our samples up in blocks and calculate the variance and mean for each block, then increase the block size until we find blocks which are uncorrelated, we can extract the $\tau = n\Delta t$, n being the block size.

2.1.5 One body density

The one body density is the probability of finding a particle a distance $\boldsymbol{r}_i$ from origo. Where then the probability density function can be calculated by

$$\rho(\boldsymbol{r}_k) = \int d\boldsymbol{r}_i |\Psi_T(\boldsymbol{r}_k, \boldsymbol{r}_i)|^2, \quad i \neq k$$

[1] Computational Physics Lecture notes, 2015, page 412, equation (12.18)

3 Results

3.0.6 Brute force Metropolis

N particles	$< E >$	Variance	Accepted	Time [s]
1	5.000000e-01	0.000000e+00	0.553895	0.018
10	5.000000e+00	0.000000e+00	0.548784	0.024
100	5.000000e+01	0.000000e+00	0.550417	0.075
500	2.500000e+02	0.000000e+00	0.550962	0.312

Table 1: Benchmark of the brute force Metropolis method with analytical calculation of the Laplacian. Using 10^5 cycles and 1.7 as step length

With the brute force Metropolis method and analytical calculation of the Laplacian the results are exactly right, as seen in table for 1 dimension, 10^5 cycles and step length of 1.7. The choice of step length is to get an accpetance rate of about 50%. With lower step length the acceptance rate would go up, but then the energies would be sampled at a more narrow range and we could not be sure if it is the acutal ground state. In table A.1.1 and A.1.2 the runs have been done with the same parameters but with 2 and 3 dimensions. The result is the same with only a slight increase in time for the highest number of particles.

N particles	$< E >$	Variance	Accepted	Time [s]
1	5.000000e-01	-9.436896e-16	0.550639	0.023
10	5.000000e+00	-5.115908e-13	0.548139	0.099
100	5.000000e+01	3.092282e-11	0.551551	2.868
500	2.500000e+02	3.419700e-10	0.550273	61.191

Table 2: Benchmark of the brute force Metropolis method with numerical calculation of the Laplacian. Using 10^5 cycles and a step length of 1.7.

Using numerical derivation to find the Laplacian slows the speed of the program by about 200 times for 500 particles. The precision is the same as with the analytical calculation and so is the acceptance rate. This tells us that using the numerical calculation is a bad choice when the analytical solution is available for the brute force Metropolis. The same results continuos when increasing to 2 and 3 dimensions, as shown in A.1.3 and A.1.4.

3.0.7 Importance sampling

Given the difference in how the step length is implemented in the two different methods it is hard to make any direct comparison of them. The energy and variance have the same values as is to be expected and the time difference is negligeble in the range we have tested here.

N particles	$< E >$	Variance	Accepted	Time [s]
1	5.000000e-01	0.000000e+00	0.998522	0.03
10	5.000000e+00	0.000000e+00	0.998700	0.038
100	5.000000e+01	0.000000e+00	0.998644	0.111
500	2.500000e+02	0.000000e+00	0.998922	0.451

Table 3: Benchmark of importance sampling Metropolis algorithm with analytical calculated Laplacian. Using 10^5 cycles and a time step of 0.05.

N particles	$< E >$	Variance	Accepted	Time [s]
1	4.999999e-01	1.337819e-14	0.998844	0.035
10	4.999999e+00	2.948752e-13	0.998800	0.111
100	4.999999e+01	-2.273737e-11	0.998711	2.907
500	2.500000e+02	1.455192e-10	0.998767	61.364

Table 4: Benchmark of importance sampling Metropolis algorithm with numerical calculation of the Laplacian. Using 10^5 cycles and a time step of 0.05.

The results for the Metropolis importance sampling algorithm has shown to be much the same as for the brute force method, with the obvious difference in acceptance ratio which is not comparable due to the way we choose the time/step length. The CPU-time is almost identical to that of the brute force method. This means that the calculation of the quantum force is negligeble compared to the rest of the calculations.

3.0.8 Time comparison

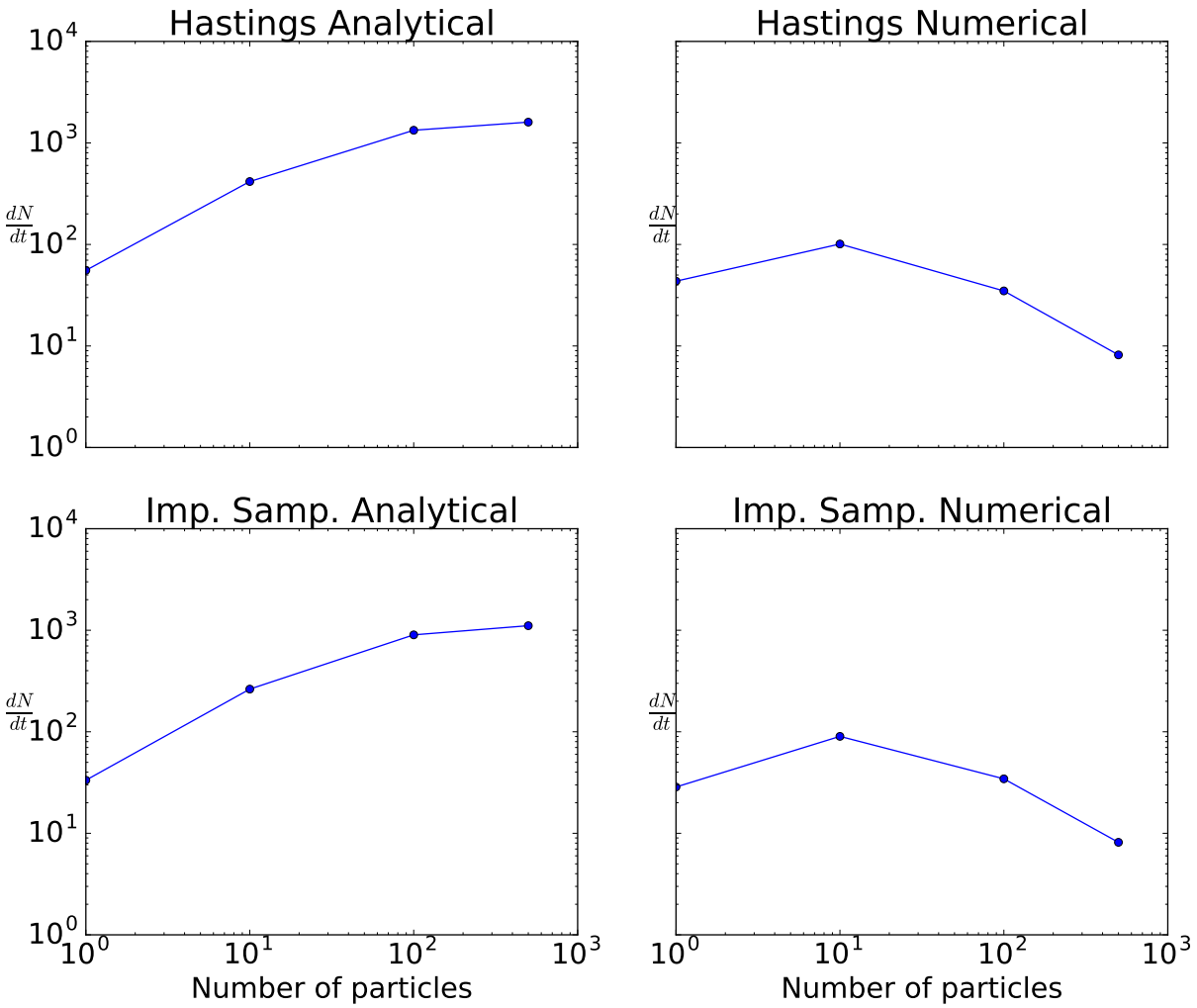

Figure 1: The x-axis is number of particles and y-axis are number of particles per time/step unit. The lower the graph goes, the more time per particle the program spends. This shows that the numerical Laplacians loses effectiveness as the number of particles grows, as for the analytical solutions the methods seem to converge toward a top.

For lower number of particles the time spent on operations other than calculating the energies is high. By increasing the number of particles the code spend more time calculating the laplacian and the graph seem to converge to a number around 1000 particles per unit step. With the numerical calculation of the Laplacian, the effectiveness of the computation decreases steeply as number of particles increases.

3.0.9 Interacting Hamiltonian

I could not find the error that was causing the interacting model to give bad results. There is an error increasing with number of particles, and it seems to have a linear correlation with the number of particles. It is also unstable, where it suddenly becomes NAN or similar. I have even compared my program with others and I can not find the error unfortunatly. I am hoping to get it fixed, but not in time for this report.

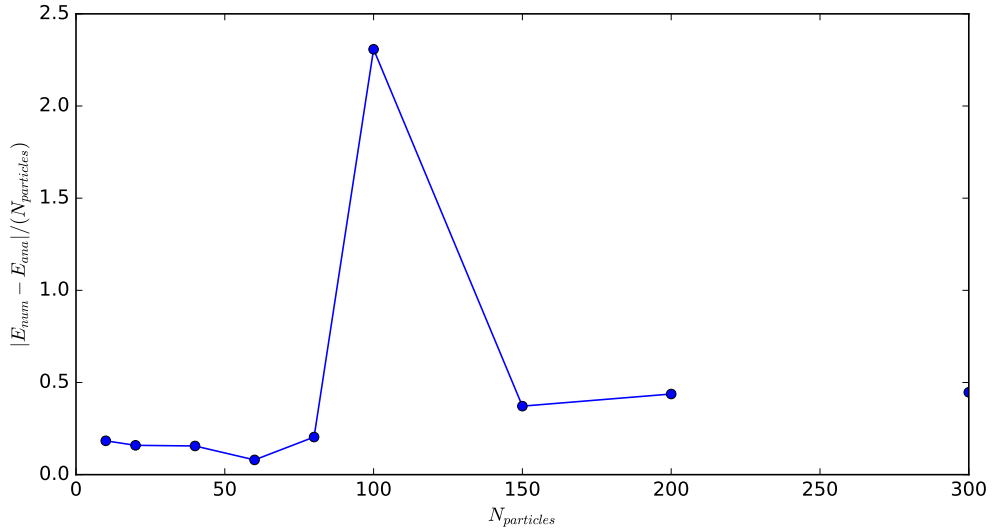

Figure 2: Plotted on the y-axis is the mean energy difference between analytical and numerical calculation per total number of particles. There seem to be a linear corerlation between error with the the exceptions of some unstable parts. This might suggest that the error might come from the movement of particles or in one of the single loops.

3.0.10 Blocking

These are the results from a run of 500 particles with Importance Sampling since I do not have any interacting data to use.

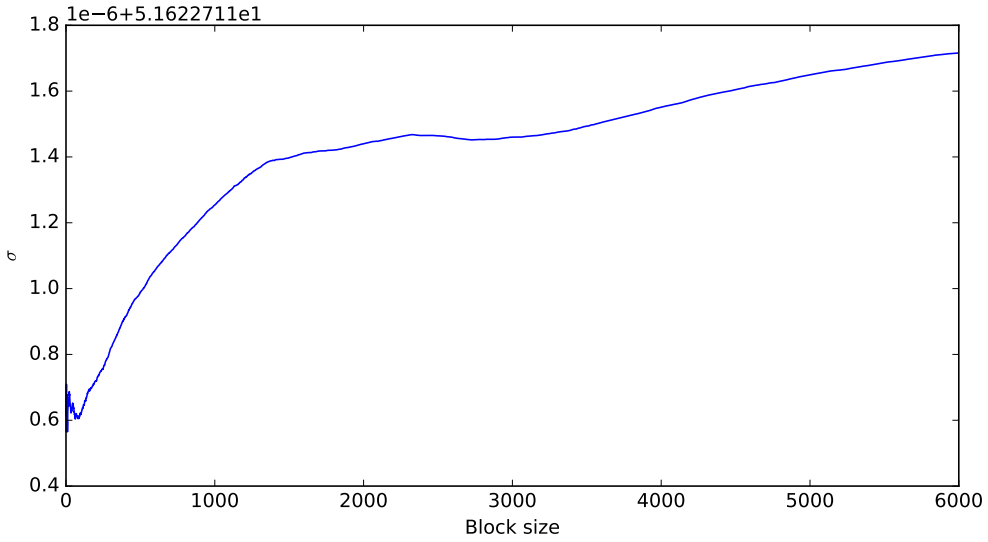

Figure 3: 500 particles and 3 dimensions, 10^5 Monte Carlo cycles, $\alpha = 0.5$There is a very low variance due to high precision when calculating this system which makes it harder to see the effect of blocking even though we can see the right shape. the STD varies with $1E-6$.

3.0.11 Particle distribution

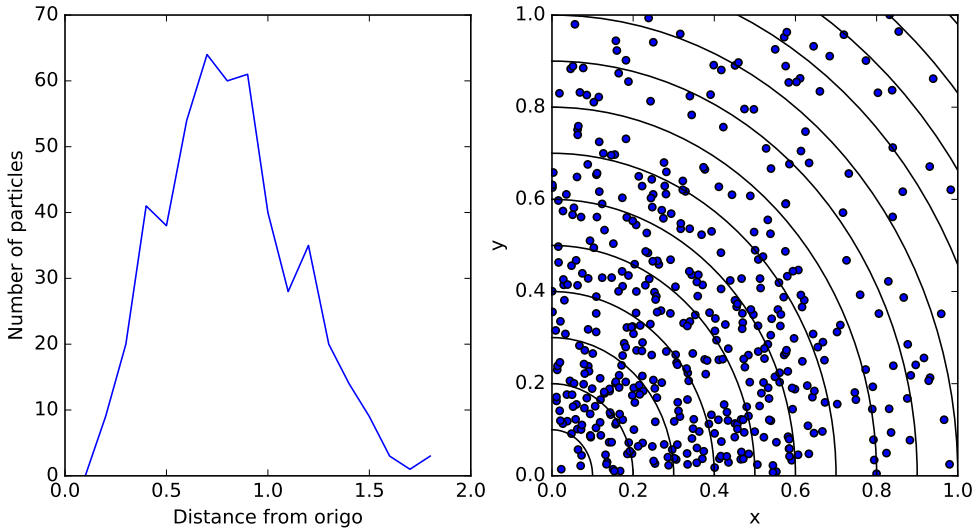

Figure 4: To the left is the distribution of particles after 8000 cycles where we see a tendency to move towards 0.75. On the right side is a cross section of the particles shown in the xy-plane, where the lines are the shells chosen when counting the distribution.

4 Conclusion

For the non-interacting system the VMC gave the exact results we would expect, and comparing the analytical solutions to the numerical showed little difference in result, but a significant increase in efficiency. Giving such low variance making it unnessecary to utilize blocking for this system. The interacting system gave a high level of resistance when trying to compute the energies, only reaching a small amount of accuracy for 10 particles with the numerical solution. Furterhmore I hope to find the missing puzzle piece to get it up and running as fast as possible.

A Tables

A.1 Brute force Metropolis algorithm

All runs are with with 10^5 cycles and step length of 1.7.

A.1.1 Analytical 2D

N particles	$< E >$	Variance	Accepted	Time [s]
1	1.000000e+00	0.000000e+00	0.548395	0.019
10	1.000000e+01	0.000000e+00	0.549895	0.029
100	1.000000e+02	0.000000e+00	0.551184	0.121
500	5.000000e+02	0.000000e+00	0.551195	0.555

A.1.2 Analytical 3D

N particles	$< E >$	Variance	Accepted	Time [s]
1	1.500000e+00	0.000000e+00	0.550362	0.019
10	1.500000e+01	0.000000e+00	0.553095	0.034
100	1.500000e+02	0.000000e+00	0.549951	0.17
500	7.500000e+02	0.000000e+00	0.543150	0.8

A.1.3 Numerical 2D

N particles	$< E >$	Variance	Accepted	Time [s]
1	9.999999e-01	-1.576517e-14	0.549651	0.029
10	9.999999e+00	1.961098e-12	0.551617	0.208
100	9.999999e+01	1.509761e-10	0.552251	10.063
500	5.000000e+02	-5.675247e-09	0.551295	232.441

A.1.4 Numerical 3D

N particles	$< E >$	Variance	Accepted	Time [s]
1	1.500000e+00	6.394885e-14	0.553662	0.035
10	1.500000e+01	3.097966e-12	0.552217	0.364
100	1.500000e+02	-6.184564e-11	0.550728	21.808
500	7.499999e+02	-2.328306e-10	0.546217	517.259

A.2 Metropolis algorithm with Importance sampling

All runs are with with 10^5 cycles and step length of 0.05.

A.2.1 Analytical 2D

N particles	$< E >$	Variance	Accepted	Time [s]
1	1.000000e+00	0.000000e+00	0.998744	0.03
10	1.000000e+01	0.000000e+00	0.998644	0.044
100	1.000000e+02	0.000000e+00	0.998689	0.181
500	5.000000e+02	0.000000e+00	0.998800	0.817

A.2.2 Analytical 3D

N particles	$< E >$	Variance	Accepted	Time [s]
1	1.500000e+00	0.000000e+00	0.998756	0.031
10	1.500000e+01	0.000000e+00	0.998767	0.051
100	1.500000e+02	0.000000e+00	0.998967	0.254
500	7.500000e+02	0.000000e+00	0.998833	1.176

A.2.3 Numerical 2D

N particles	$< E >$	Variance	Accepted	Time [s]
1	9.999999e-01	1.099121e-14	0.998756	0.041
10	9.999999e+00	6.963319e-13	0.998767	0.223
100	9.999999e+01	2.546585e-11	0.998711	10.117
500	4.999999e+02	-5.587935e-09	0.998789	232.673

A.2.4 Numerical 3D

N particles	$< E >$	Variance	Accepted	Time [s]
1	1.500000e+00	1.332268e-15	0.998767	0.046
10	1.500000e+01	2.131628e-12	0.998833	0.38
100	1.500000e+02	-5.456968e-11	0.998611	21.95
500	7.499999e+02	1.012813e-08	0.998667	517.499

B Links

Source code [3]

References

[1] J. L. DuBois and H. R. Glyde. Bose-einstein condensation in trapped bosons: A variational monte carlo analysis. *Phys. Rev. A*, 63, 023602, 2001.

[2] M. Hjorth-Jensen. *Computational Physics: Lecture Notes Fall 2015*. 2015.

[3] Roar Emaus. Github repository. `https://github.com/roarem/compPhys2_p1`.