VMC part 1

Stian Roti Svendby

3. mars 2015

Abstract:

We have estimated the groud state of helium and beryllium using different wavefunctions. This has been done with Monte Carlo simulations using brute force Metropolis algorithm and Metropolis-Hasting algorithm.

Introduction:

Usually, in classical mechanics, we analyze forces acting on and between objects in our system, before we develop and solve systems of differential equations to estimate positions at a later time. This is doable for an easy system like the hydrogen atom, but when the system consist of more particles in three dimensions, the system of equations grows fastly, and the CPU expences become too large to solve the system at all with classical mechanics. The solution is quantum mechanics, combined with Monte Carlo (MC) simulations using Metropolis algorithm, using a statistical approach. In this project we have used Variational Monte Carlo to estimate the ground state energy, i.e find the system configuration that minimizes the energy, for both the Helium and Beryllium atom. We have found closed form expressions for the local energies used in the calculations to save CPU time for Helium, and estimated energy minumum, onebody density and charge desity. First we are going to look at the Hamiltonian for Helium, introduce two different wavefunctions and define and show how the local energy is used in the computation of the energy. Thereafter we will explain how the simulations are performed, both with and without importance sampling, and extend all the theory to Beryllium. In the result section we will compare how different wavefunctions and importance sampling affect our computations both for Helium and Beryllium.

Methods:

The dimensionless Hamiltonian for Helium in three dimensions is given by

$$\hat{\boldsymbol{H}} = -\frac{\nabla_1^2}{2} - \frac{\nabla_2^2}{2} - \frac{2}{r_1} - \frac{2}{r_2} + \frac{1}{r_{12}}$$

, consisting of the kinetic and potential energy, and containing all information about the system (the helium atom). ∇ is the derivative in three spatial dimensions, $r_1$and r_2 the radial distances between the nucleus and the two electrons, and r_{12} the distance between the two electrons with. $-\frac{\nabla_1^2}{2} - \frac{\nabla_2^2}{2}$ is the kinetic energy, $-\frac{2}{r_1} - \frac{2}{r_2}$ potential energy due to the electron attraction from the nucleus, and $\frac{1}{r_{12}}$ is the potential energy due to the interaction between the electrons.

The time-independent Schrödinger equation is given by:

$$\hat{\boldsymbol{H}}\psi = E\psi$$

, which states that

$$E\left[H\right] = \langle H \rangle = \frac{\int d\boldsymbol{r_1} d\boldsymbol{r_2} \psi_{Ti}^*(\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{r_{12}}) \hat{\boldsymbol{H}}(\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{r_{12}}) \psi_{Ti}(\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{r_{12}})}{\int dr \psi_T^*(R)\psi_T(R)}$$

, which is an upper bound to the ground state energy E_0:

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

To find the energy, we need to find a wavefunction that describes the system well. The easiest approach is to use two hydrogenic wavefunctions and multiply them, which yield:

$$\psi_{T1}(\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{r_{12}}, \boldsymbol{\alpha}) = e^{-\alpha(r_1+r_2)}$$

, where α is a variational parameter we have to find.

A better guess of the wavefunction is however to include a *Jastrow factor*:

$$\psi_{T2}(\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{r_{12}}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = e^{-\alpha(r_1+r_2)} e^{\frac{r_{12}}{2(1+\beta r_{12}}}$$

, where β also is a variational parameter. The choosen wavefunction is our probability distribution ($R = (\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{r_{12}})$):

$$P(R, \alpha, \beta) = \frac{|\psi_T(R, \alpha, \beta)|^2}{\int |\psi_T(R, \alpha, \beta)|^2 dR}$$

Our goal is to compute the ground state of atom, and from the time-independent Schrödinger equation we introduce *local energy*, defined as:

$$E_L(R_i, \alpha, \beta) = \frac{1}{\psi_T(R, \alpha, \beta)} \hat{\boldsymbol{H}} \psi_T(R, \alpha, \beta)$$

$$= -\frac{1}{2\psi_T(R, \alpha, \beta)} \nabla_1^2 \psi_T(R, \alpha, \beta) - \frac{1}{2\psi_T(R, \alpha, \beta)} \nabla_2^2 \psi_T(R, \alpha, \beta) - \frac{2}{r_1} - \frac{2}{r_2} + \frac{1}{r_{12}}$$

With this in hand, we are able to estimate the ground state of a given atom:

$$E[H(\alpha, \beta)] = \int P(R) E_L(R) dR \approx \frac{1}{N} \sum_{i=1}^{N} P(R_i, \alpha, \beta) E_L(R_i, \alpha, \beta)$$

The estimate is actually a MC simulation to solve the above integral. We try a random move in space for an electron, use Metropolis algorithm to accept or reject the move, before we have to calculate the new local energy (using the wavefunction) every time we try a new configuration of the atom. The new local energy is added to a sum for each MC cycle. To get our estimate for the ground state, we just divide the sum of local energies by the number of MC cycles, or MC cycle times electrons depending on how the sum of local energies is constructed.

Since we have to compute the local energy every time we move a particle, this is also the most CPU critical part of the computations. We se that we have to carry out two double derivatives for the helium with two electrons. Those derivatives can be solved bruce force numerically, but that is an ineffective way of solving it. If we instead carry out the analytical solution of the local energy, we should be able to save a lot of CPU time. If we are able to compute the same energy numerical as with the analytical expression, it will also be a nice beckmark to verify that the program work. The derivatives are easiest carried out by polar-coordinates by hand or by using *Symbolic Python* (sympy). The local energies of helium for the to different wavefunctions can then be written:

$$E_{L1} = (\alpha - Z)\left(\frac{1}{r_1} + \frac{1}{r_2}\right) + \frac{1}{r_{12}} - \alpha^2$$

$$E_{L2} = E_{L1} + \frac{1}{2(1+\beta r_{12})^2}\left(\frac{\alpha(r_1+r_2)}{r_{12}}(1 - \frac{\boldsymbol{r_1}\boldsymbol{r_2}}{r_1 r_2}) - \frac{1}{2(1+\beta r_{12})^2} - \frac{2}{r_{12}} + \frac{2\beta}{1+\beta r_{12}}\right)$$

A simple code to find E_{L1} with sympy can be:

```
1  import sympy as sp
2
3  def LocalEnergy(phi):
4    return  ((-sp.diff(phi, x1, 2) - sp.diff(phi, y1, 2) - sp.diff(phi, z1,2) +        (- sp.diff(phi
         , x2, 2) - sp.diff(phi, y2, 2) - sp.diff(phi, z2, 2)))/(2*phi) -       Z/r1 - Z/r2 + 1 /r12)
5
6  # make variables symbolic
7  x1, x2, y1, y2, z1, z2, alpha, Z = sp.symbols('x1, x2, y1, y2, z1, z2, alpha, Z')
8
9  # creates a symbolic equivalent of r1 and r2
```

```
R1, R2, R12 = sp.symbols('r1 r2 r12')
r1 = sp.sqrt(x1*x1 + y1*y1 + z1*z1) r2 = sp.sqrt(x2*x2 + y2*y2 + z2*z2) r12 = sp.sqrt((x1 - x2)**2
     + (y1 - y2)**2 + (z1 - z2)**2)

# our wavefunction
phi1 = sp.exp(-alpha*(r1 + r2))

# compute the local energy
local_energy = LocalEnergy(phi1)

#Simplify the expression local_energy = local_energy.subs(r12,R12).factor().subs(r1, R1).subs(r1
    **2,R1**2).subs(r2,R2).subs(r2**2,R2**2).simplify().collect(Z).collect(alpha)

# Print out analytical expression for the local energy as latex and c code
print sp.printing.latex(local_energy)
print sp.printing.ccode(local_energy)
```

As mentioned we use Metropolis algorithm to accept or reject random suggested moves of electrons. A suggested move are given by:

$$y = x + r\delta$$

, where x is the old position, r is is a random number between -0.5 and 0.5, and δ is the steplength. In our program suggested moves are done by the code:

```
rNew(i,j) = rOld(i,j) + stepLength*(ran2(&idum) - 0.5);
```

, where *steplength* is a predefined value choosen to give an acceptance ratio about 0.5 in Metropolis algorithm. $ran2(\&idum)$ generates random numbers between 0 and 1. With the new configuration of the atom we use Metropolis algorithm. The Metropolis algorithm is given by:

$$A(y,x) = min(1, q(y,x))$$

$$q(y,x) = \frac{|\psi_T(y)|^2}{|\psi_T(x)|^2}$$

A code for doing this is:

```
if(ran2(&idum)<=(waveFunctionNew*waveFunctionNew)/(waveFunctionOld*waveFunctionOld)){
  for(int j = 0; j < nDimensions; j++){
    rOld(i,j) = rNew(i,j);
    waveFunctionOld = waveFunctionNew;
  }
}
else{
  for(int j = 0; j < nDimensions; j++){
    rNew(i,j) = rOld(i,j);
}
```

We see that if the new configuration gives lower energy, there will be a probability of 0.5 for the right steplength for accepted move. If the energy is higher in the new configuration than in the previous one, we reject the move.

We can also introduce *importance sampling*, in this particular case apply a *quantum force*. That means that we apply a force that make the electrons tend to its minimum, so a new suggested move will almost always give a lower energy. A new step is suggested as:

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

, where $D = 0.5$ is a factor from the kinetic energy, $F(x)$ is the quantum force, $\Delta t \in [0.001, 0.01]$ (for stabel values) is the timestep, and ξ is a is a gaussian probability distribution. The quantum force are given by:

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

and pushes the electrons towards regions where the wavefunction is large, so this case we will accept almost every move. Suggested moves are in this case done by the code:

```
rNew(i,j) = rOld(i,j) + GaussianDeviate(&idum)*sqrt(timestep)+QForceOld(i,j)*timestep*D;
```

GaussianDeviate is the gaussian probability distribution, *timestep* control the steplength and the acceptance ratio of suggested steps. A low value of $timestep = \Delta t$ will give high acceptance ratio, but the steps will also become shorter, so we must have a balance (values in [0.001,0.01]) for stabel result.

Instead of the brute force Metropolis algorithm, we switch to Metropolis-Hasting algorithm given by:

$$A(y,x) = min(1, q(y,x))$$

$$q(y,x) = \frac{G(x,y,\Delta t)|\psi_T(y)|^2}{G(y,x,\Delta t)|\psi_T(x)|^2}$$

The Metropolis-Hasting algorithm (with importance sampling) are implemented by:

```
if(ran2(&idum)<=GreensFunction*(waveFunctionNew*waveFunctionNew)/(waveFunctionOld*waveFunctionOld)
    ){
  for(int j = 0; j < nDimensions; j++){
    rOld(i,j) = rNew(i,j);
    QForceOld(i,j) = QForceNew(i,j);
    waveFunctionOld = waveFunctionNew;
  }
}
else{
  for(int j = 0; j < nDimensions; j++){
    rNew(i,j) = rOld(i,j);
    QForceNew(i,j) = QForceOld(i,j);
  }
}
```

This has the same properties as without quantum force, but we also multiply with the *Greensfunction* computed by:

```
GreensFunction = 0.0;
for(int j=0; j < nDimensions; j++){
  GreensFunction += 0.5*(QForceOld(i,j)+QForceNew(i,j))*                    (D*timestep
      *0.5*(QForceOld(i,j)-QForceNew(i,j))-rNew(i,j)+rOld(i,j));          }
GreensFunction = exp(GreensFunction);
```

We now have all the theory we need to compute the ground state for Helium with and without Jastrow factor and with and without importance sampling. We will in the section with results see how this affect the computations.

If we study larger atoms such as Beryllium, we have to introduce a *Slater determinant*, since we cannot have more than two eletrons in the lowest orbital. In our project we use wavefunctions for Beryllium at the form:

$$\psi_T(\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{r_3}, \boldsymbol{r_4}) = Det(\phi_1(\boldsymbol{r}_1)\phi_2(\boldsymbol{r}_2)\phi_3(\boldsymbol{r}_3)\phi_4(\boldsymbol{r}_4)) \prod_{i<j}^{4} e^{\frac{r_{ij}}{2(1+\beta r_{ij})}}$$

where the first part is the Slater determinant, while the last part is the Jastrow factor. The Slater determinant is given by:

$$Det(\phi_1(\boldsymbol{r}_1)\phi_2(\boldsymbol{r}_2)\phi_3(\boldsymbol{r}_3)\phi_4(\boldsymbol{r}_4)) = \frac{1}{\sqrt{4}} \begin{vmatrix} \phi_1(\boldsymbol{r}_1) & \phi_2(\boldsymbol{r}_1) & \phi_3(\boldsymbol{r}_1) & \phi_4(\boldsymbol{r}_1) \\ \phi_1(\boldsymbol{r}_2) & \phi_2(\boldsymbol{r}_2) & \phi_3(\boldsymbol{r}_2) & \phi_4(\boldsymbol{r}_2) \\ \phi_1(\boldsymbol{r}_3) & \phi_2(\boldsymbol{r}_3) & \phi_3(\boldsymbol{r}_2) & \phi_4(\boldsymbol{r}_3) \\ \phi_1(\boldsymbol{r}_4) & \phi_2(\boldsymbol{r}_4) & \phi_3(\boldsymbol{r}_4) & \phi_4(\boldsymbol{r}_4) \end{vmatrix}$$

For ϕ we use the hydrogen orbitals:

$$\phi_{1s}(\boldsymbol{r}_i) = e^{-\alpha r_i}$$

$$\phi_{2s}(\boldsymbol{r}_i) = (1 - \frac{\alpha}{2} r_i) e^{-\frac{\alpha}{2} r_i}$$

Then we can rewrite the Slater determinant as:

$$Det(\phi_1(\boldsymbol{r}_1)\phi_2(\boldsymbol{r}_2)\phi_3(\boldsymbol{r}_3)\phi_4(\boldsymbol{r}_4)) = \frac{1}{\sqrt{4}} \begin{vmatrix} \phi_{1s}(\boldsymbol{r}_1) & \phi_{1s}(\boldsymbol{r}_2) \\ \phi_{2s}(\boldsymbol{r}_1) & \phi_{2s}(\boldsymbol{r}_2) \end{vmatrix} \begin{vmatrix} \phi_{1s}(\boldsymbol{r}_3) & \phi_{1s}(\boldsymbol{r}_4) \\ \phi_{2s}(\boldsymbol{r}_3) & \phi_{2s}(\boldsymbol{r}_4) \end{vmatrix}$$

This is implemented with the folliwing code, with *argument* as a vector with the four instantaneous radial distances between the nucleus and the electrons:

```
double VMCSolver::SlaterDeterminant(){
  vec argument = zeros(nParticles);
  argument = r_centre;
  double wf = (psi1s(argument(0))*psi2s(argument(1))-psi1s(argument(1))*psi2s(argument(0)))*(
      psi1s(argument(2))*psi2s(argument(3))-psi1s(argument(3))*psi2s(argument(2)));
  return wf;
}
```

We note that we have skipped the factor $\frac{1}{\sqrt{4}}$ since it vanish in Metropolis ratio in our algorithm.

The second part of our wavefunction for Beryllium, the Jastrow factor, is computed by:

$$\Psi_C = e^{\sum_{i<j} \frac{a_{ij} r_{ij}}{1-\beta r_{ij}}}$$

Since two fermions cannot be in the same state unless they have different spinn, we chose to give 2 electrons spin *up* and 2 electrons spinn *down*. a_{ij} is a matrix containing information about spins of the electrons, since $a_{ij} = \frac{1}{4}$ if equal spinn and $a_{ij} = \frac{1}{2}$ if opposite spinn. Assuming a_{ij} is computed and the distance between the elektrons are known, code to compute the Jastrow factro is given by::

```
double VMCSolver::JastrowFactor(){
  double Psi = 1.0;
  for(int j=0; j < nParticles; j++){
    for(int i=0; i < j; i++){
      Psi *= exp((a_matrix(i,j)*r_distance(i,j))/(1.0 + beta*r_distance(i,j)));          }
    }
  return Psi;
}
```

Now we got all the theory to make etimates of the ground state energy for both Helium and Beryllium, but we have to find a proper way to present the estimate. This is done by include the standard deviation or σ and present the estimated energy as:

$$E_{VMC} \pm \sigma$$

If our samples are uncorrelated, the standard deviation of our etimate of the mean would have been:

$$\sigma = \sqrt{\frac{1}{N}\left(\langle E^2_{mean}\rangle - \langle E_{mean}\rangle^2\right)}$$

But since our samples actually are correlated, the computed σ will be too low, and the correct standard deviation is given by:

$$\sigma = \sqrt{\frac{1+2\tau/\Delta t}{N}\left(\langle E^2_{mean}\rangle - \langle E_{mean}\rangle^2\right)}$$

, where τ is *correlation time* (time between uncorrelated samples) and Δt is time between samples. Unfortunately τ is unknown and cost a lot of CPU, fortunately this can be solved with *blocking*. Blocking means that we run a MC simulation, store every computed energy, divide the stored energy into different block sizes, and compute and plot the standard deviation (SD) as a function of the blocksize. The SD for each blocksize is found by the formula:

$$\sigma_{trial} = \sqrt{\frac{1}{N_{block}}\sum_{N_{block}} \langle E^2_{block}\rangle - \left(\frac{1}{N_{block}}\sum_{N_{block}} \langle E_{block}\rangle\right)^2}$$

A matlab code for doing the blocking can be:

```
for i=1:block_trials
  blocks = data_size/(i);
  energy_trial_sum = 0;
  energySquared_trial_sum = 0;
  for j=1:blocks
    block_size = data_size/blocks;
    index_start = floor(1 + (j-1)*block_size);
    index_stop = floor(j*block_size);
    n = (index_stop - index_start) + 1;

    energy_mean_block = sum(energy(index_start:index_stop))/n;
    energy_trial_sum = energy_trial_sum + energy_mean_block;
    energySquared_trial_sum = energySquared_trial_sum+ energy_mean_block*energy_mean_block;
```

```
14      end
15
16      energy_mean_blockSize = energy_trial_sum/blocks;
17      energySquared_mean_blockSize = energySquared_trial_sum/blocks;
18      variance_mean_blockSize = (energySquared_mean_blockSize - (energy_mean_blockSize*
            energy_mean_blockSize))/blocks;
19      block_size_values(i) = floor(block_size);
20      block_size_variance(i) = variance_mean_blockSize;
21  end
22
23  SD = sqrt(block_size_variance);
24  plot(block_size_values, SD)
```

After plotting the SD as a function of block size, we should see a "plateu" where the SD is independent of the block size, and the SD on this plateu we name σ_{block}, with the corresponding $N_{blocksize}$ where the plateu starts. Then the actual σ we want to find is given by:

$$\sigma = \frac{\sigma_{block}}{\sqrt{N/N_{blocksize}}}$$

, where N is the size of the dataset we have done blocking on.

Now we are able to get result and present then properly, so its time to move on with some results.

Results:

First of all we need to find optimal parameters for α and β. We have developed an bruth force algorithm that find *steplength* = 1.3 which gives an accteptance ratio about 0.5. Doing a MC simulation running over different values of α and β, gives a mesh of energies. The result is given in Figure 1.

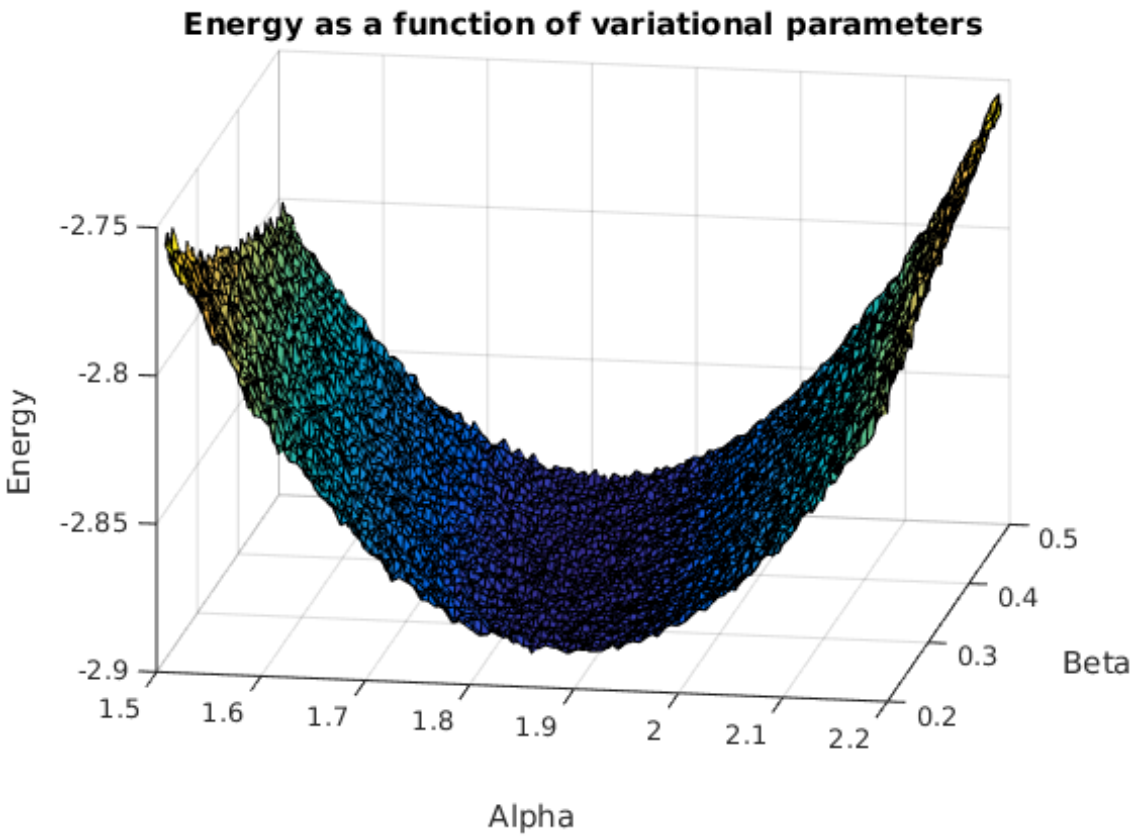

Figure 1: Out of this mesh, we can se which pair of α and β that minimizes the energy for the Helium atom with Jatrow factor.

From the figure we see that the energy is minimized for values $\alpha = 1.85$ and $\beta = 0.35$. These values will be used in all MC simulations of Helium in this project. This is not exact values because of uncertainty in the computed energies, but very many samples in each MC simulation will gain precision. An alternative approach it to look at the variance instead of the energy, an see how the variance is minimized. The physical interpretation of α can be explained from the Hydrogen atom. For hydrogen, the exact wavefunction is given by:

$$\psi_T(\boldsymbol{r}, \boldsymbol{\alpha}) = e^{-\alpha r}$$

For $\alpha = 1$, we find the exact result with zero variance in the computations down to machine precision Ref[1]. We see that α correspond to the charge of the Hydrogen atom. For helium we could then think that $\alpha = 2$ would be the best value, but thats not the case. Our value $\alpha = 1.85$ is a bit lower than we could expect at a first glaze, but the interaction between the electrons are disturbing a "clean" charge field around the nucleus as for the hydrogen atom, so it looks like a reasonalbe value for α after a second thought.

We have earlier found the analytical expressions for the local energies. How the CPU time is with the analytical expressions is compared to the numerical solver is given in Table 1.

$N = 10^8$	*numerical*	*analytical*	*speedup analytical*
without Jastrow factor	189.48	56.50	3.35
with Jastrow factor	430.58	78.34	5.50

Table 1: We see that we get a huge speedup on the MC simulation with analytical expressions for the local energy of Helium.

After a look at Table 1, it should be clear that we prefer using the analytical expressions with huge speedup, and so is done in the rest of the project.

If we perform some simple MC simulations with our parameters for the Helium atom, we will now be able to estimate the ground state of Helium and look at the mean distance between the electrons. The result are given in Table 2 and Table 3. We note that all computations in Table 2 and Table 3 are done by using the analytical expressions for the local energies for Helium, but there are actually no visible difference on the results when using the numerical integration.

$N = 10^8$	*importance sampling off*	*importance sampling on*
without Jastrow factor	-2.8214	-2.8214
with Jastrow factor	-2.8902	-2.8905

Table 2: Estimate of ground state energy of Helium.

$N = 10^8$	*importance sampling off*	*importance sampling on*
without Jastrow factor	1.1824	1.1841
with Jastrow factor	1.3564	1.3569

Table 3: Estimate of the mean distance between the electrons in the Helium atom.

From Table 2 we see that the Jastrow factor minimized the energy much better than when not using it, so the wavefunction including the Jastrow factor is the better guess for the wavefunction. In fact we are very close the referance value $E_0 = -2.9037\, a.u.$. From Table 3 we see that the mean distance between the electrons is bigger using the Jastrow factor.

It can be interesting to see how the energy is minimized as a funtion of MC cycles. Doing a MC simulation with Jastrow factor gived the result shown in Figure 2.

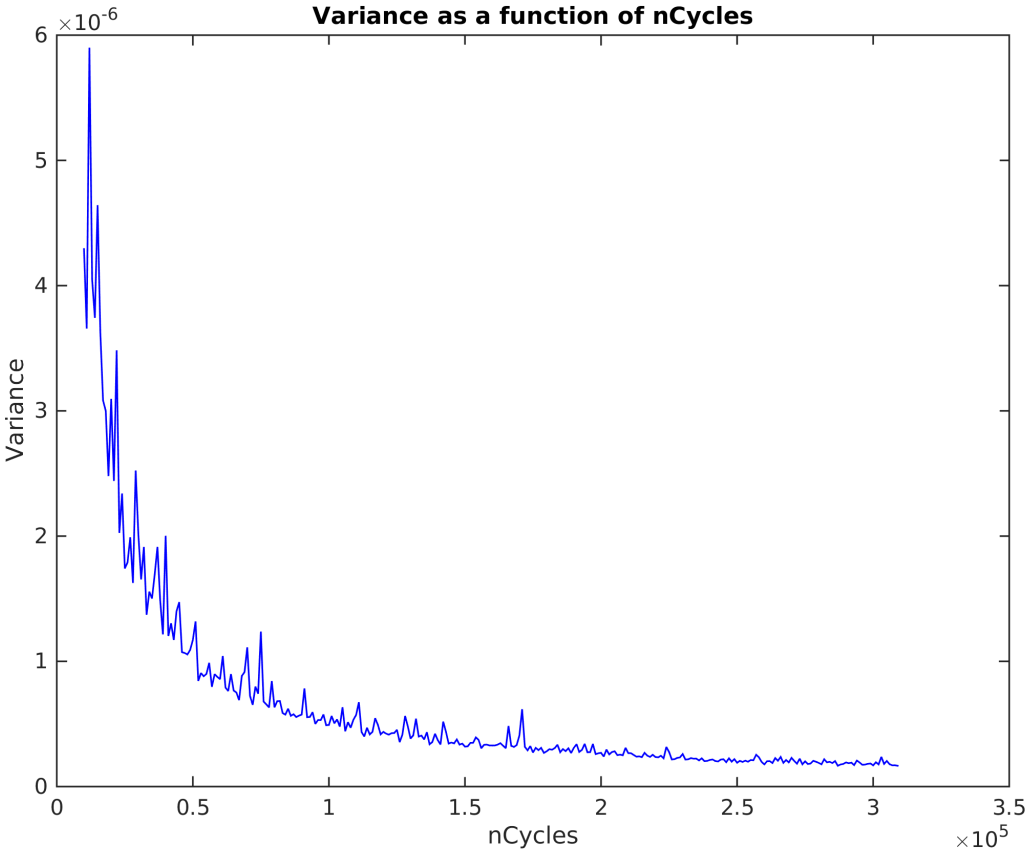

Figure 2: We see how the variance become smaller when MC cycles are increased in simulation of the Helium atom.

As expected, the variance of the energy is decreasing for increasing MC cycles. We always want to use as many cycles as possible within the range of CPU-time.

If we now turn on importance sampling, we have to check how the energy and mean electron distance depends on the choosen value of *timestep*. The result from a simulation with $N = 10^7$ are given by Figure 3:

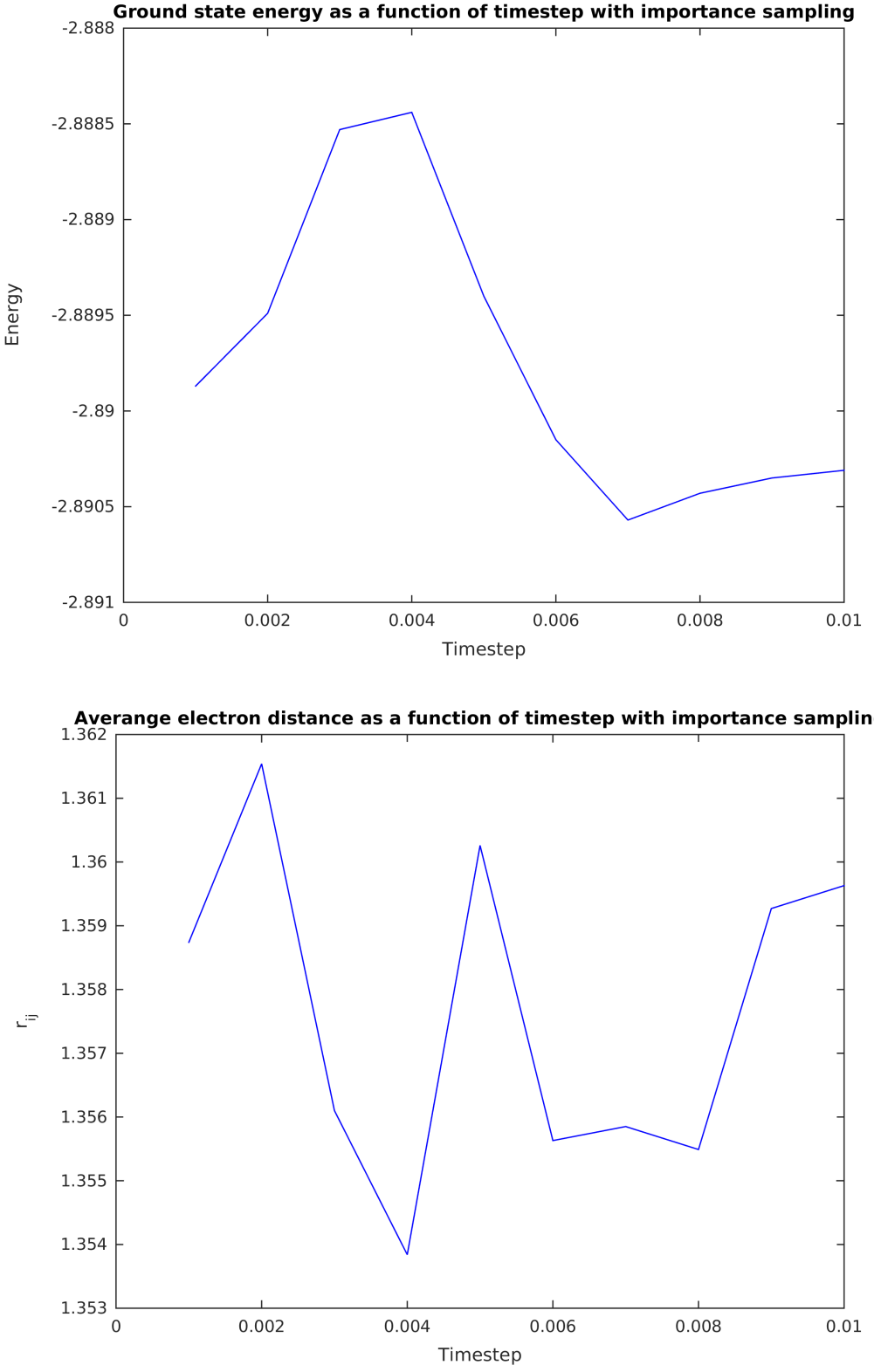

Figure 3: Energy and mean distance between electrons as a function of Δt.

We cannot see any clear dependence between timestep and energy from Figure 3, so the value of Δt is not important within the range for Δt values. The same conclusion can also be drawn for averange electron distance r_{ij}. So a proper value for Δt could be 0.002, with an acceptance ratio about 0.94.

It is also interesting to see how the CPU-time change when importance sampling is activated. Running a simulation for helium with Jastrow factor gives the result shown in Figure 4:

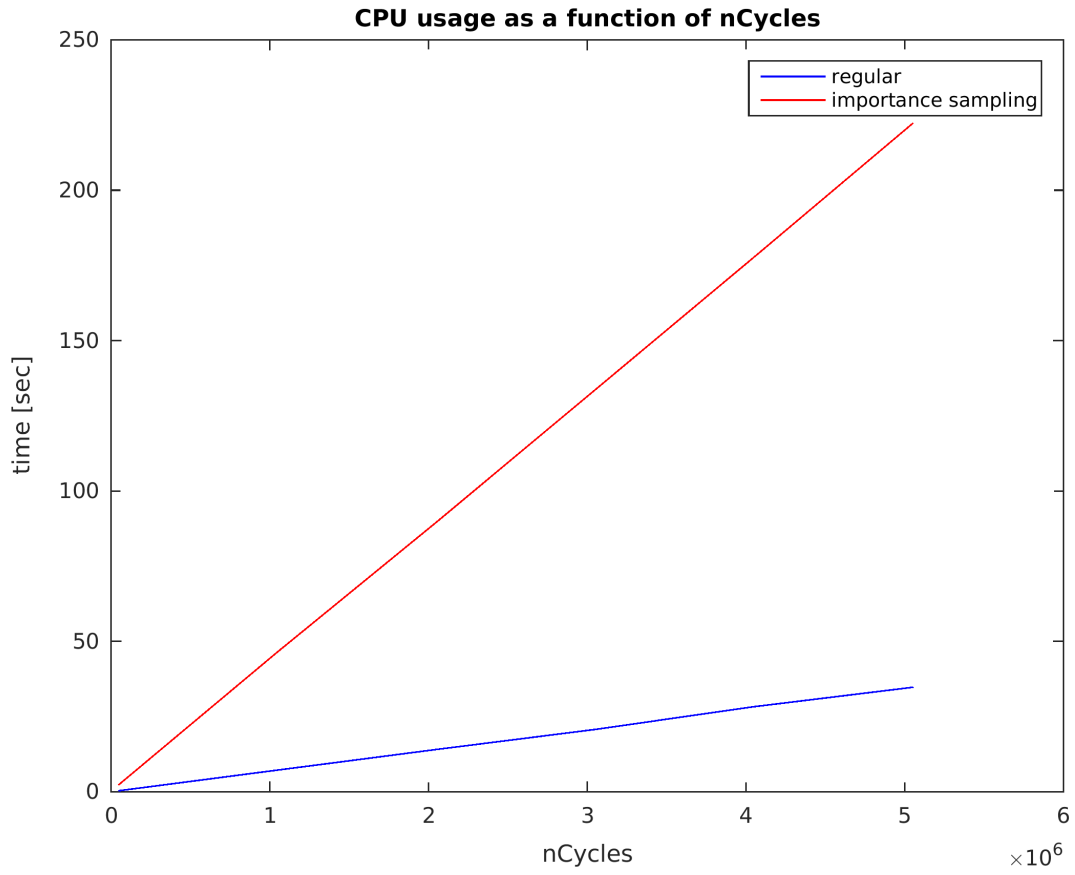

Figure 4: CPU-time as a function of MC cycles for helium with and without importance sampling.

From Figure 4 we see that including the importance sampling increase the CPU-time a lot, so the quantum force should make the electrons reach the ground state in pretty few cycles, else the computations are way to heavy compared with not including the quantum force.

Since the quantum force push the electrons in the right direction (where the wavefunction is large) , it's reasonable to think that we need fewer MC cycles to reach an energy close to the ground state energy. A plot show what it looks like:

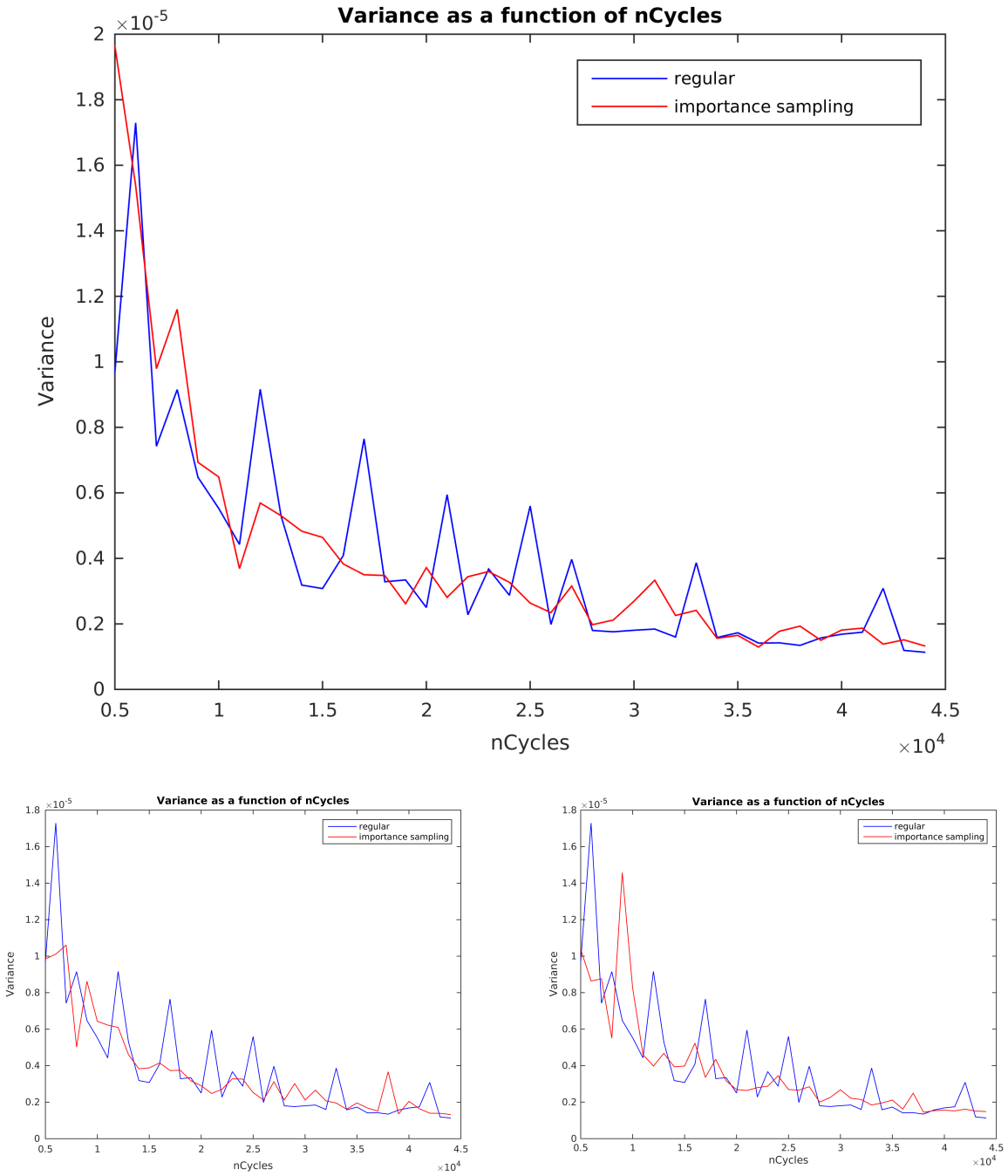

Figure 5: We see that the variance actually creases equally with and without importance sampling, but with importance sampling there are less variation in the variance, so there should be less variation in the computed energy for few cycles. $\Delta t = 0.002$ in the first figure, $\Delta t = 0.005$ in the figure down left, and $\Delta t = 0.01$ in the figure down right.

Now when we know optimal values for α and β, showed that the Jastrow factor give improved results, we can estimate the ground state of Helium with standard deviation using blocking. If we run a MC simulation with $N = 10^6$ of helium with Jastrow factor without importance samling and do blocking, we get Figure 6:

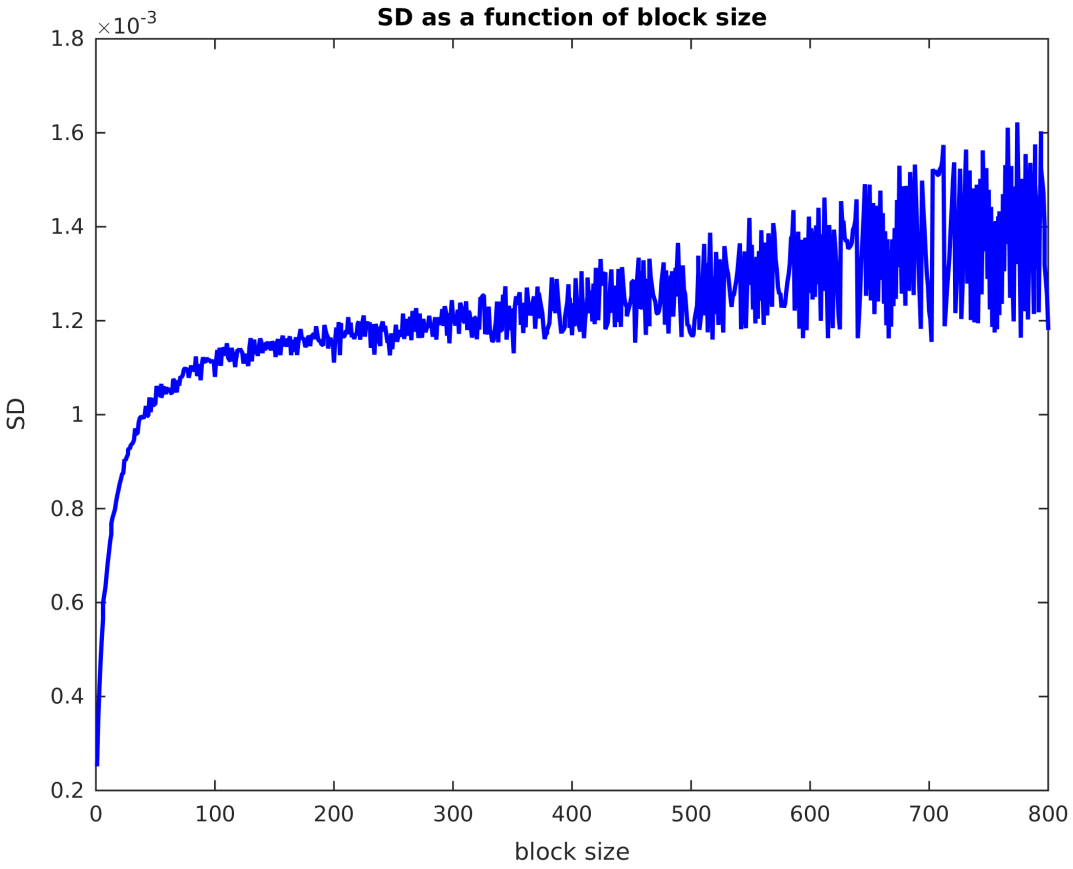

Figure 6: Blocking performed on the Helium atom with $N = 10^6$. We see a "plateu" where $\sigma_{block} \approx 1.2 \times 10^{-3}$ and $N_{blocksize} \approx 200$.

Using value $\sigma_{block} \approx 1.2 \times 10^{-3}$ and $N_{blocksize} \approx 200$, we find the standard deviation:

$$\sigma = \frac{\sigma_{block}}{\sqrt{N/N_{blocksize}}} = \frac{1.2 \times 10^{-3}}{\sqrt{10^6/200}} = 1.697 \times 10^{-5}$$

If we now plot all positions after all moves, we can study the charge density and onebody density (positions of only one electron). For the Helium atom with and without Jastrow factor and with and without importance sampling are given by Figure 7-10:

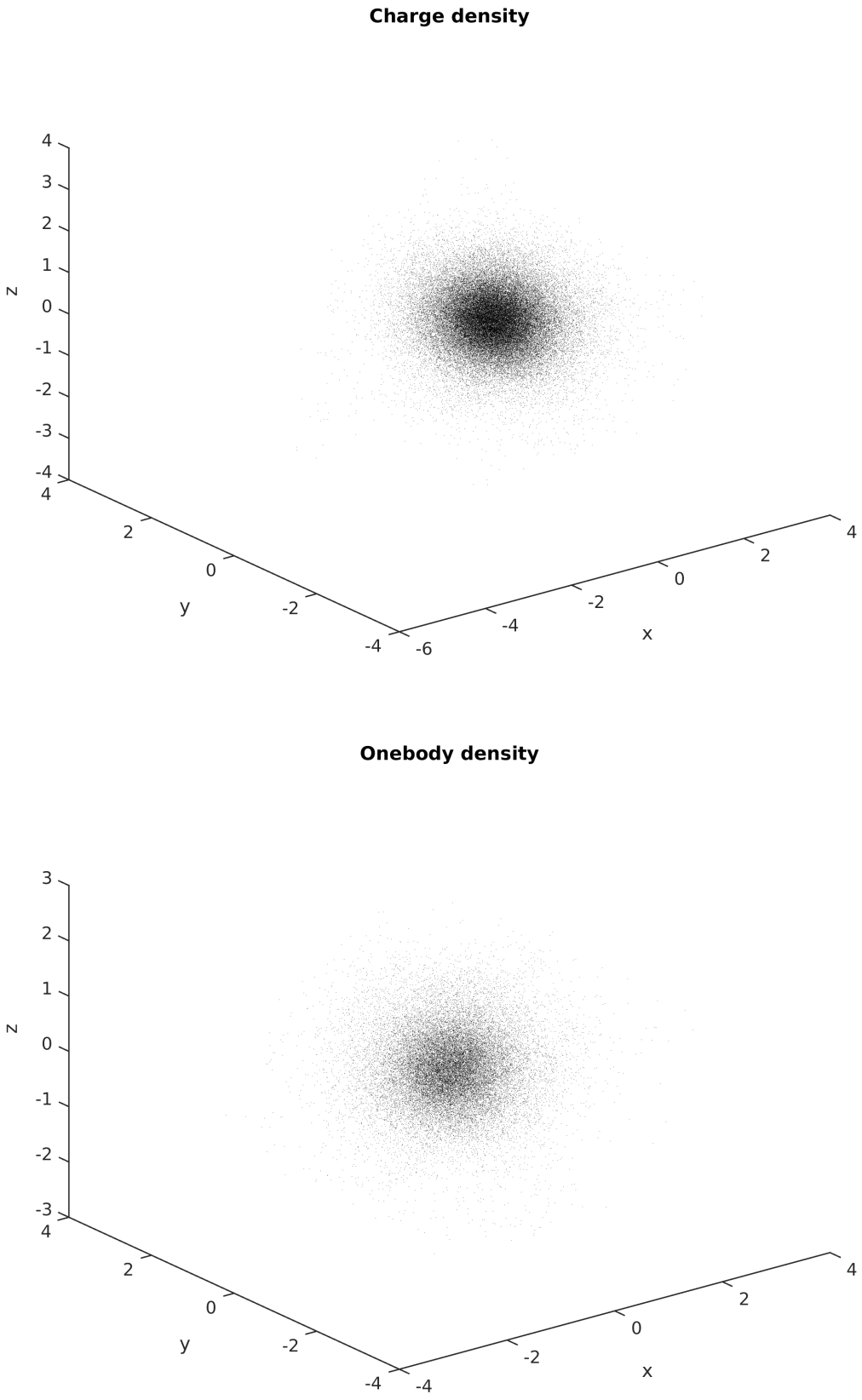

Figure 7: Charge density and onebody density for Helium with Jastrow factor, without importance sampling.

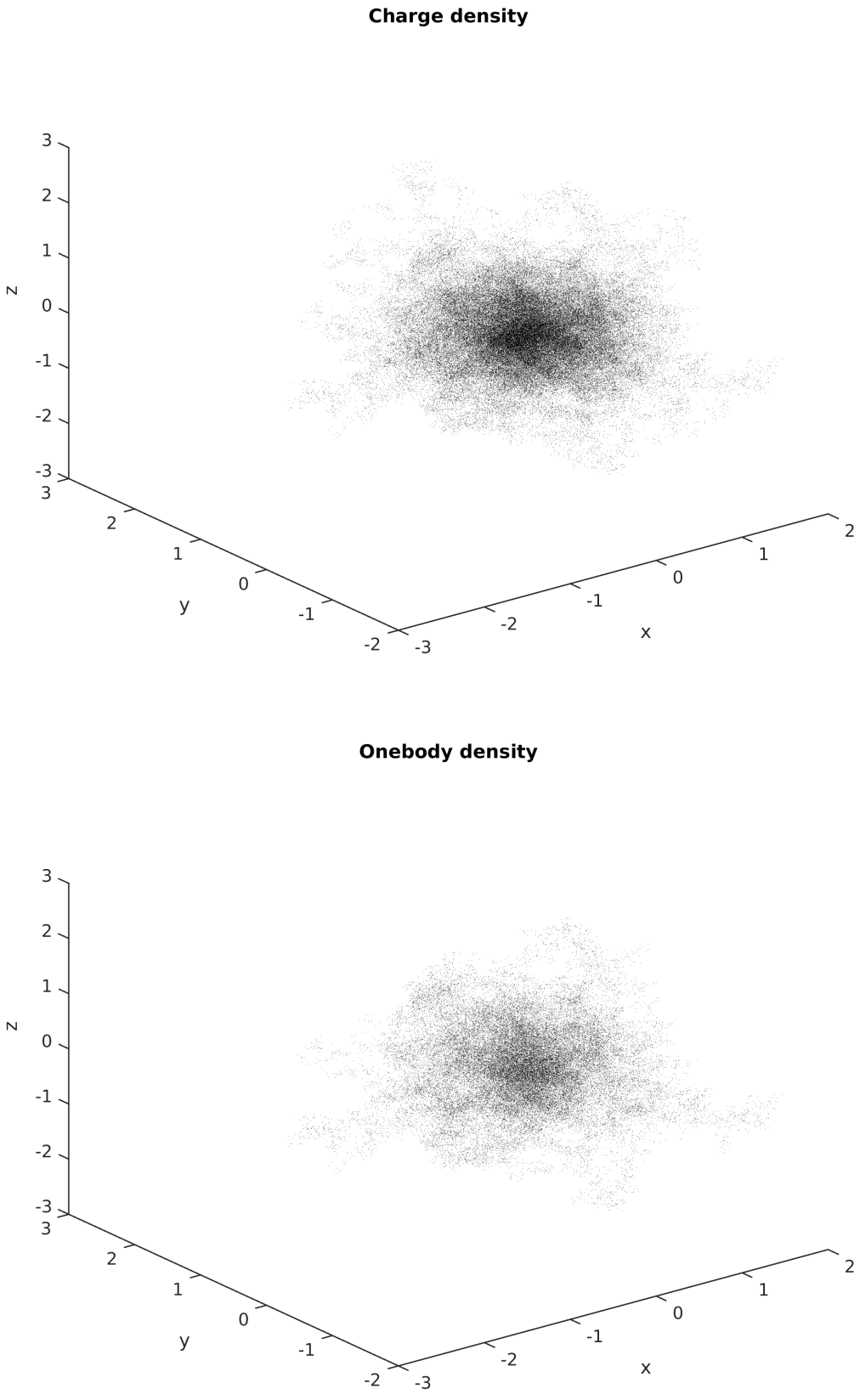

Figure 8: Charge density and onebody density for Helium with Jastrow factor, with importance sampling.

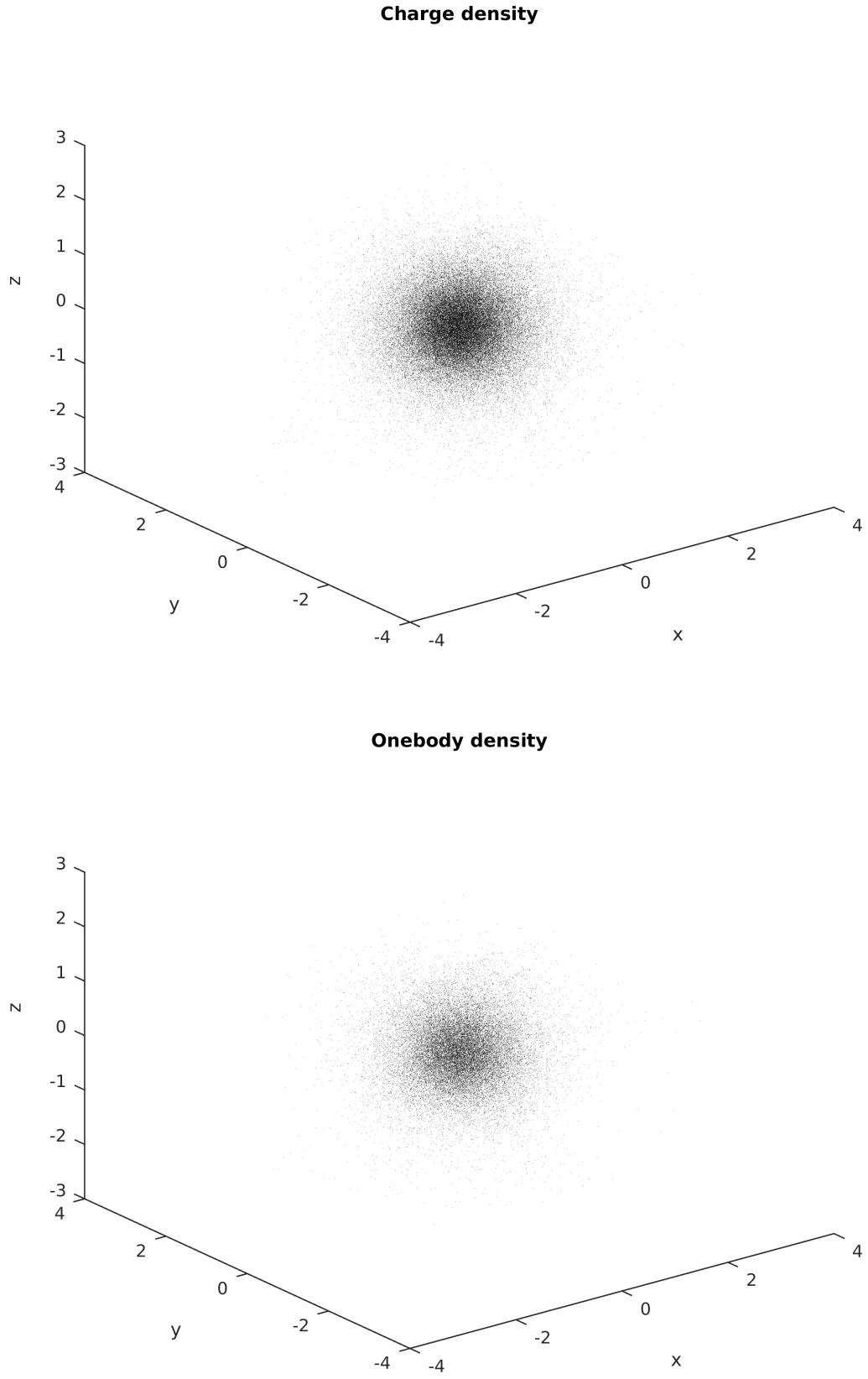

Figure 9: Charge density and onebody density for Helium without Jastrow factor, without importance sampling.

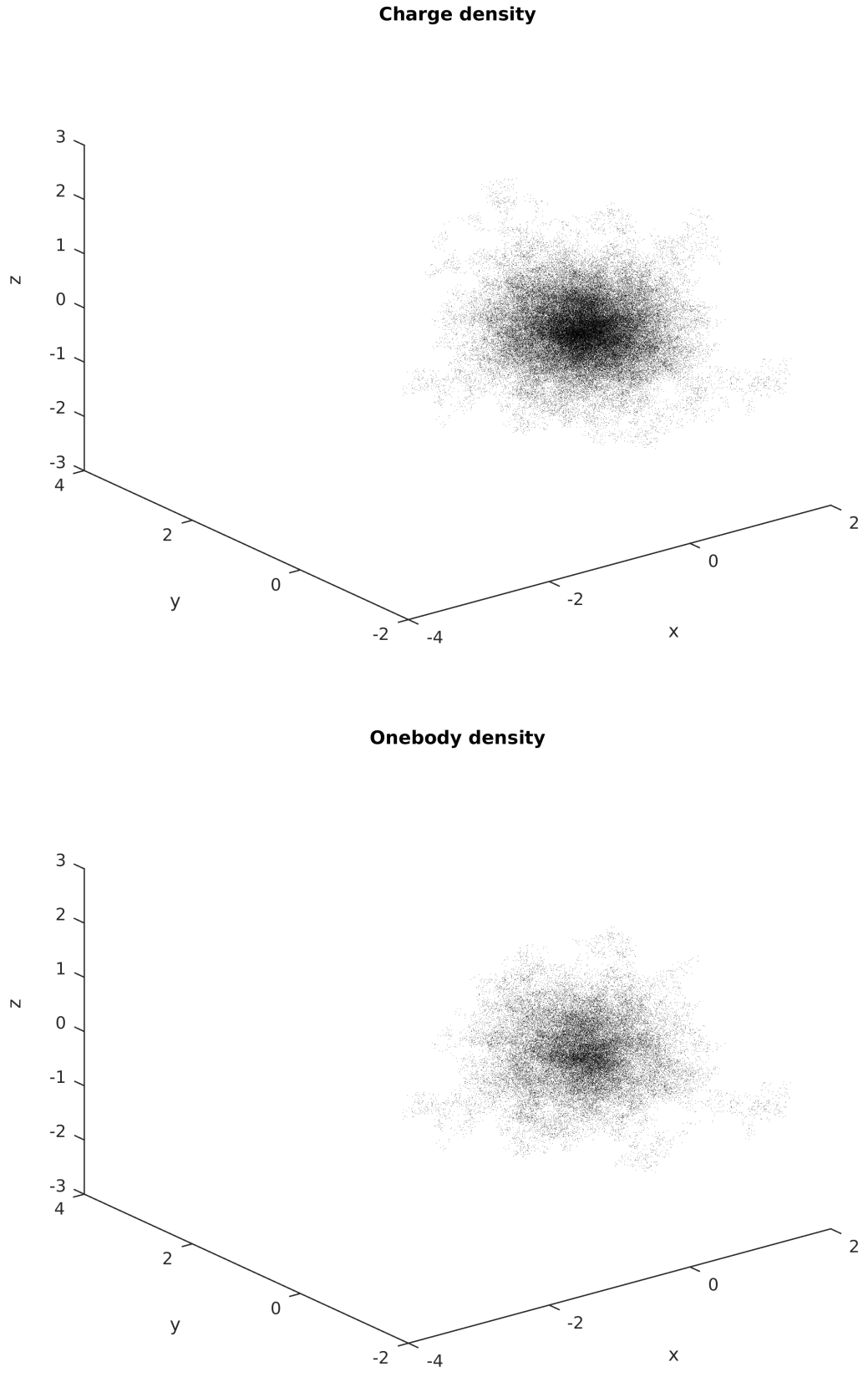

Figure 10: Charge density and onebody density for Helium without Jastrow factor, with importance sampling.

$atom$	α	β	E_{VMC}	E_0	σ	N
He	1.85	0.35	-2.8893	-2.9037	1.697×10^{-5}	10^6
Be	3.9	0.1	-14.4877	-14.667	3.8×10^{-4}	10^6

Table 4: Summary of the estimates of both Helium and Beryllium.

From the figures it is hard to see any difference between the pure hydrogenic wavefunction and the wavefunction including the Jastrow factor. On the other hand its easy to see a difference in the figures where we turn on and off importance sampling. All the plots have been made with only 50000 MC cycles, and we can easily see a special pattern when using importance sampling. So it is clearly an effect of the applied quantum force.

We now go over to Beryllium and do all the same as for helium. We run over different values for α and β to find the combination that minimizes the energy as for helium, and find approximate $\alpha = 3.9$ and $\beta = 0.1$. We set $\Delta t = 0.002$, active importance sampling, and run a MC simulation with $N = 10^6$ to estimate the ground state energy. We perform blocking to find the corresponding σ. We find $E_{VMC} = -14.4877$. The corresponding blocking is shown in Figure 11:

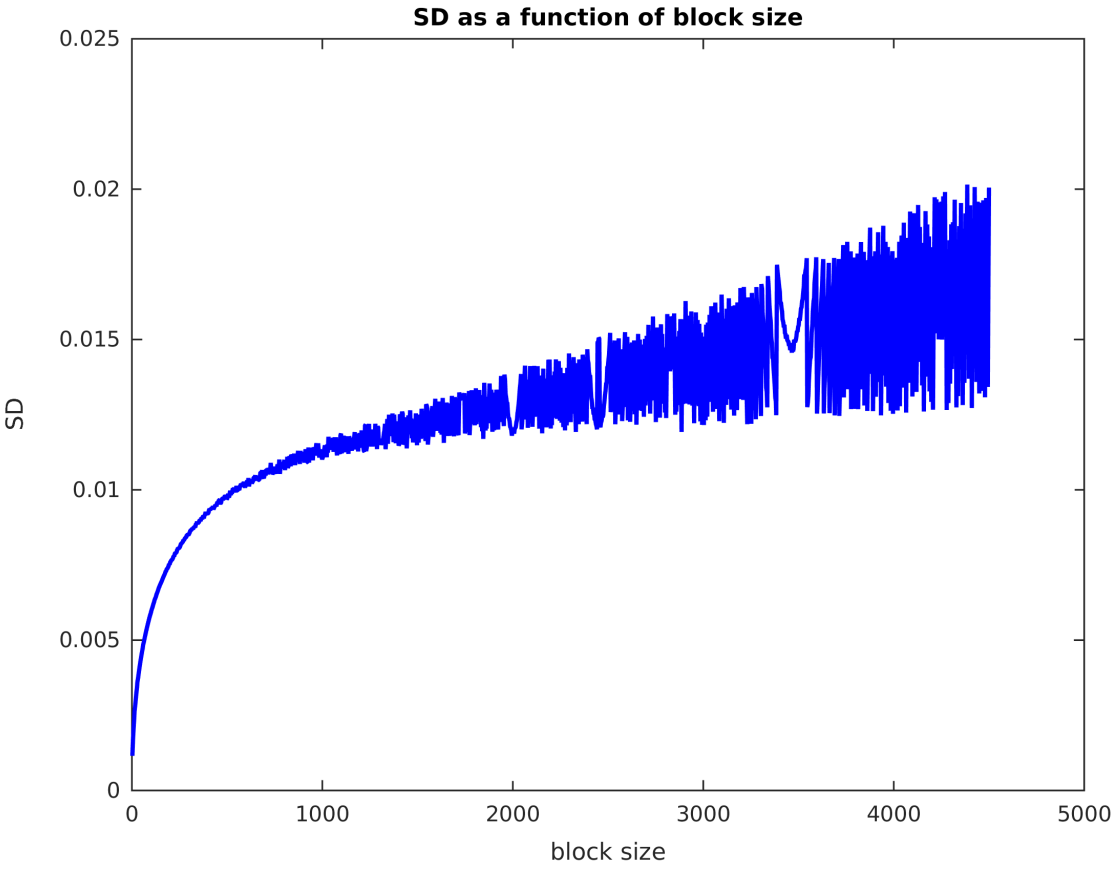

Figure 11: Blocking performed on the Beryllium atom with $N = 10^6$. We see a "plateu" where $\sigma_{block} \approx 0.012$ and $N_{blocksize} \approx 1000$.

Using value $\sigma_{block} \approx 0.012$ and $N_{blocksize} \approx 1000$, we find the standard deviation:

$$\sigma = \frac{\sigma_{block}}{\sqrt{N/N_{blocksize}}} = \frac{0.012}{\sqrt{10^6/1000}} = 3.8 \times 10^{-4}$$

The blocking analysis of both helium and Beryllium is collected in Table 4.

Again we plot the charge density and onebody desity, but this time for Belyllium. We are again turning on and off Jastrow factor and and on and off importance sampling. Results are given by Figure 12-15:

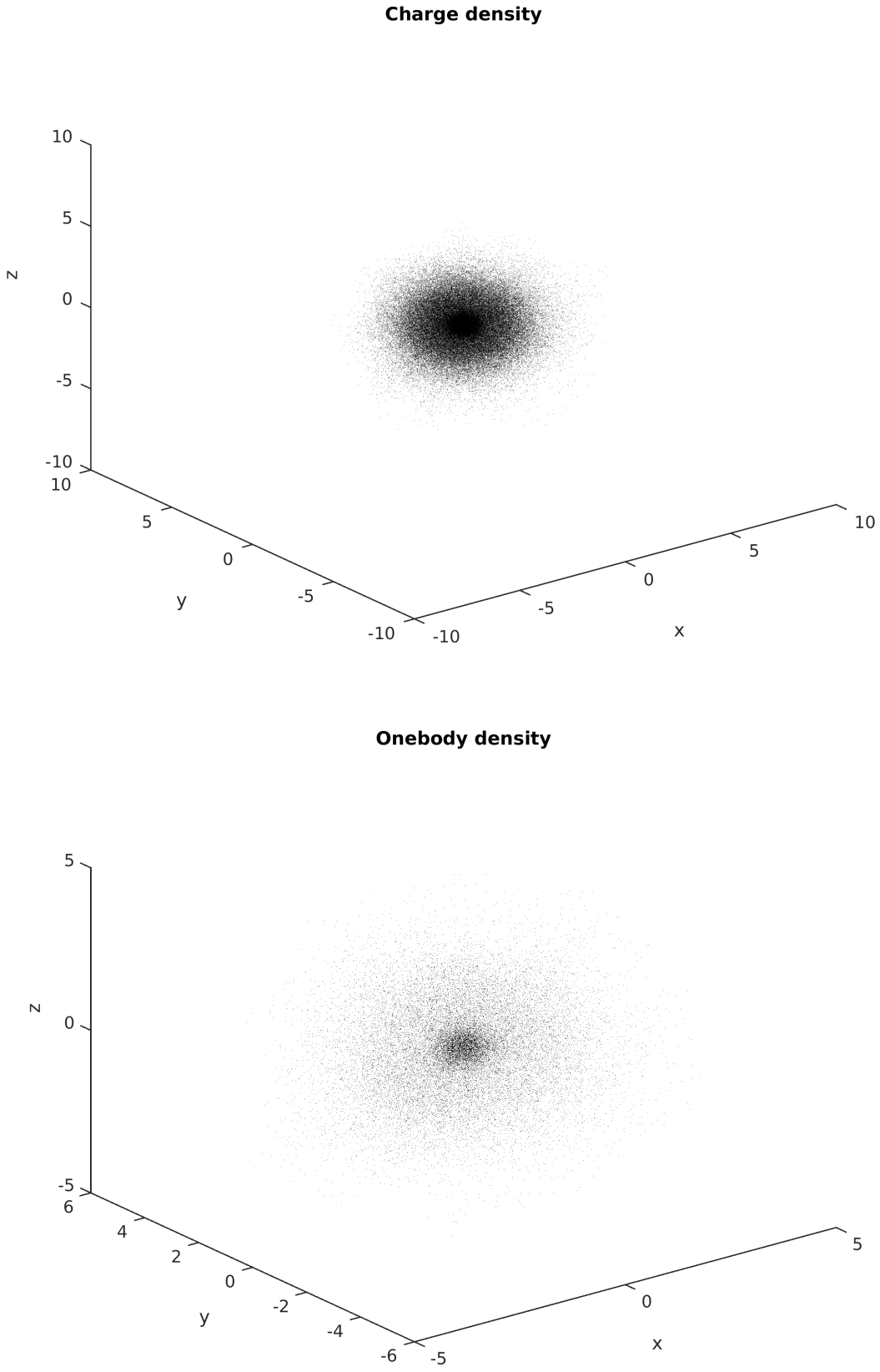

Figure 12: Charge density and onebody density for Beryllium with Jastrow factor, without importance sampling.

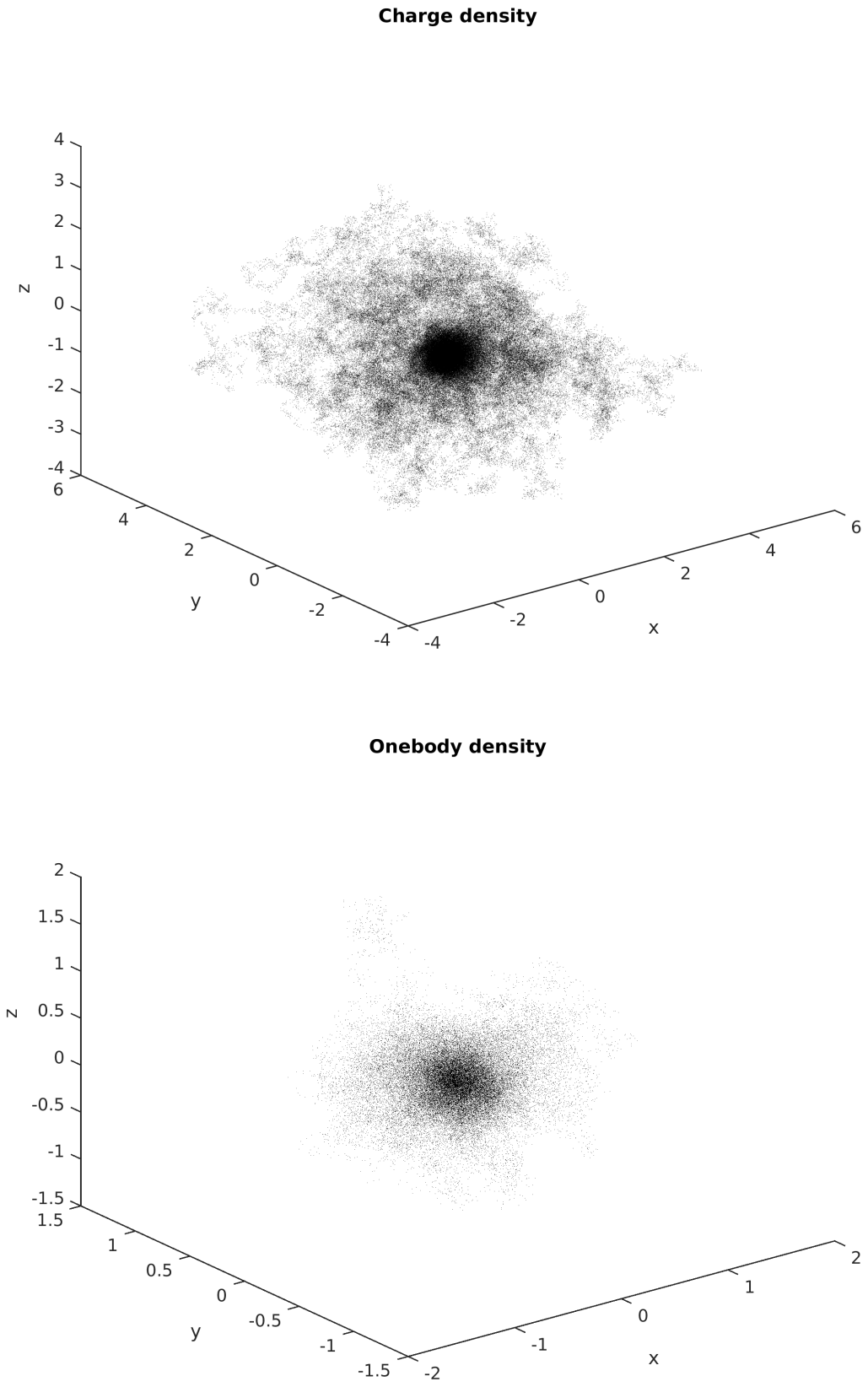

Figure 13: Charge density and onebody density for Beryllium with Jastrow factor, with importance sampling.

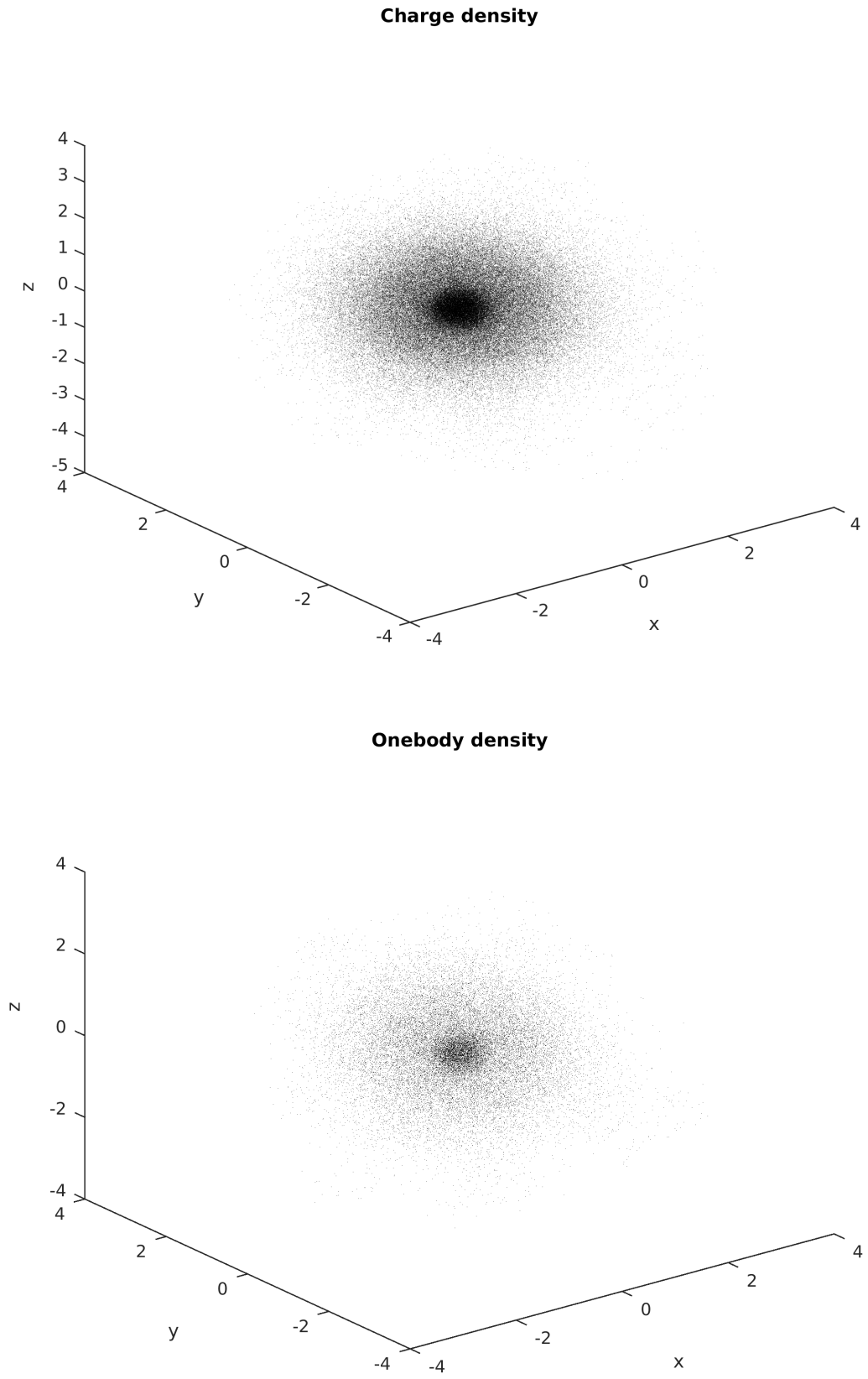

Figure 14: Charge density and onebody density for Beryllium without Jastrow factor, without importance sampling.

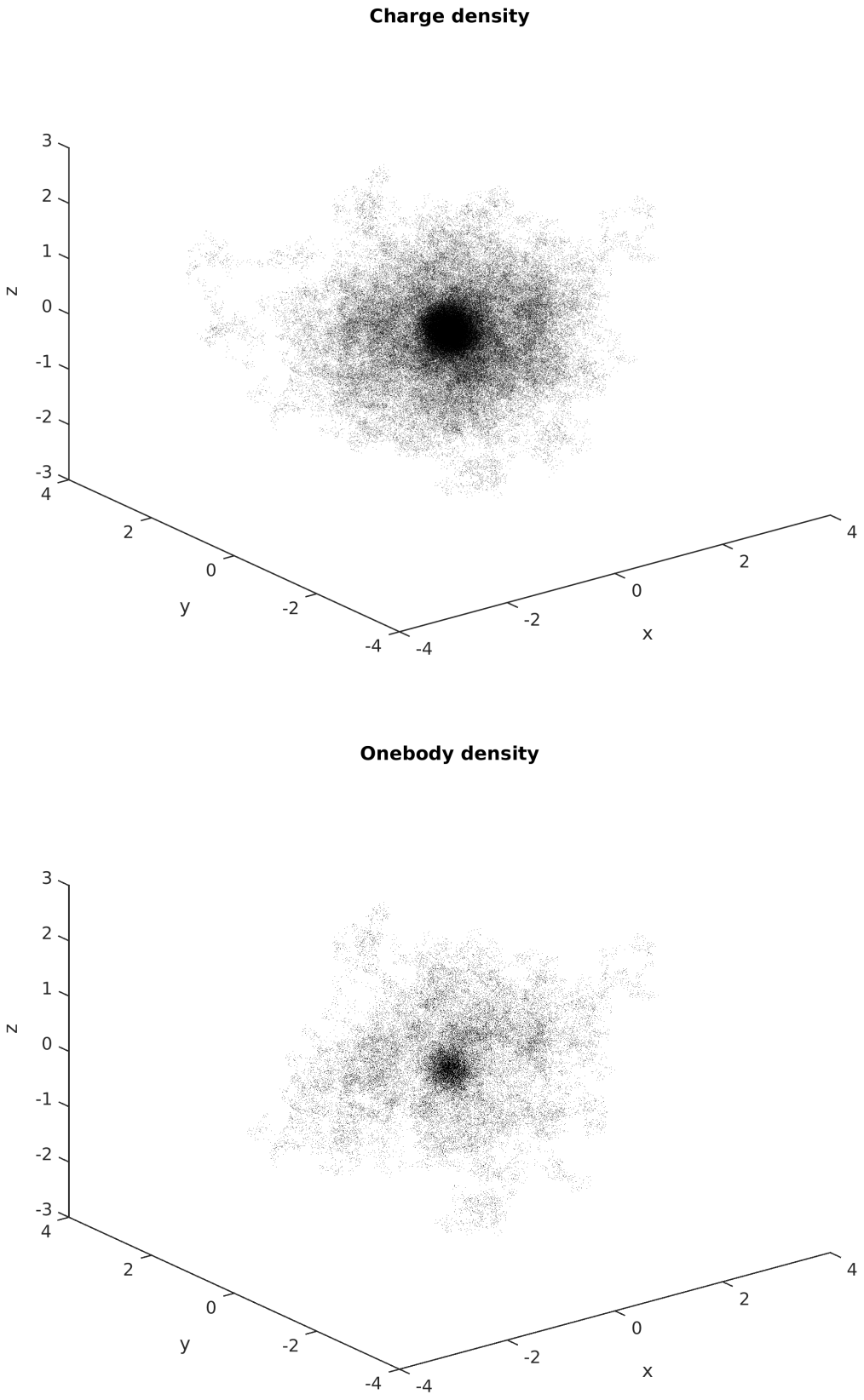

Figure 15: Charge density and onebody density for Beryllium without Jastrow factor, with importance sampling.

In the same way as for helium, we see that without the quantum force, the electrons are equally distributed with higher density close to a centre. We also see clear paterns when the importance sampling is activated, so it clearly has some effect.

Conclusion:

In our project we have found out that the estimate of the energy become more precise for increasing MC cycles, and the Jastrow factor gives a better estimate of the ground state of both helium and beryllium than a pure hydrogenic wavefunction. We have also seen that an applied quantum force gives less variance in the energy computed than without it, but it didn't look like the energy converges faster with importance sampling for few MC cycles, and we must have in mind that importance sampling uses a lot more CPU. Blocking has been uses to find the real standard deviation of our ground state estimate. Blocking makes us able to find the standard deviation in an CPU-effisient way.

In the following we have to do major improvments to the structure of the code to make it easier to change atom and corresponding parameters. It could also be used a better algorithm to find variational parameters that minimizes the energy since this his very CPU-heavy to do in the way done in this project.

Further improvements like parallelization and a more general way to compute the slater determinant are topics for the next project.

References:

Ref[1]: http://compphysics.github.io/ComputationalPhysics2/doc/pub/vmc/pdf/vmc-print.pdf (2. March 2015)