Project 2 - FYS 4411 The restricted Boltzmannn machine applied to the quantum many body problem

Davide Saccardo
Mathias Storhaug Reistad

June 10, 2018

Abstract

The ground state energy of Coulomb interacting and non-interacting electrons in a harmonic oscillator potential is studied using Variational Monte Carlo methods. The wavefunction is approximated using a Restricted Boltzmann Machine which is trained using the Stochastic Gradient Descent method. The study is done for 1 and 2 electrons in 1 and 2 dimensions for different number of hidden nodes. We test three sampling methods: Metropolis sampling, Importance sampling and Gibbs sampling. We find exact agreement with theoretical results for the non-interacting electrons using all 3 samplers. The Gibbs sampler fails to produce any meaningful ground state for the interacting electrons. The best result for 2 interacting electrons in 2 dimensions is found using 2 hidden nodes with Metropolis sampling: we report $(3.068 \pm 9 \times 10^{-3})$ $\hbar\omega$ as an upper bound for the ground state energy. This is in good agreement with the theoretical result of 3 $\hbar\omega$. Using more than 2 hidden nodes does not improve results. All programs used can be found on the GIT pagee: `https://github.com/mathisre/FYS4411---Project-2-ML`.

1 Introduction

The purpose of this project is to study the energy of a system of 2 interacting fermions in a 2d spherical harmonic oscillator. Fermions are particles that exhibit the Pauli exclusion principle stating that two fermions cannot occupy the same quantum state simultaneously. As a consequence the ground state can be filled by just two particles (in the case of spin$-1/2$ fermions). Therefore by studying two electrons, we can ignore dealing with the Pauli principle, which would have involved the computation of the Slater's determinant.

The electron interaction is modeled using the Coulomb potential. The result is that the electrons repel each another and have to find balance between particles repulsion and the curvature of the harmonic oscillator potential. The produced results will be compared with Taut's analytical solutions from [1]. The ground state energy is computed through Variational Monte Carlo methods, the sampling methods used are standard Metropolis, Importance Sampling and Gibbs sampling.

In recent times Machine Learning (ML) methods using neural networks (NN) has become increasingly better at simulating quantum systems. In [2] Carleo and Troyer discuss the impressive potential of ML and NN methods over traditional methods. The idea is to use a NN of visible and hidden nodes to represent the system wavefunction Ψ. The weights of these nodes act as variational parameters leading to the convenient use of the variational method to approximate the ground state. The energy of Ψ is calculated and the (Stochastic) Gradient Descent (SGD) method is applied to find the set of parameters that minimize the energy function. This project uses a Gaussan-Binary Restricted Boltzmann Machine (RBM) as choice of NN to represent Ψ. We use the SGD method with a fixed learning rate, η.

2 System description

We consider a system of N_p interacting fermions in a spherical harmonic oscillator potential. The interaction is given by the Coulomb potential. The Hamiltonian of the system is

$$H = \sum_{i=1}^{N_p} \left[-\frac{1}{2}\nabla_i^2 + \frac{1}{2}w^2 r_i^2 \right] + \sum_{i<j} \frac{1}{r_{ij}},$$

where $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$ is the distance between particle i and j.

When we set the interaction to be zero, we are left with a harmonic oscillator potential which is known to exhibit exact solutions. The energy of one particle in a 2d potential is $\epsilon_{n_x,n_y} = \hbar\omega(n_x + n_y + 1)$, where n_x and n_y are the quantum numbers for the x and y oscillator. The ground state energy is therefore $\epsilon_{0,0} = \hbar\omega$. Because electrons can have spin up or down, there exists two distinct degenerate ground states. Hence if the electrons have opposite spin then they can both be in the ground state. For 2 non-interacting electrons in 2d the energy is then $\epsilon = 2\hbar\omega$. Setting d as the dimensionality of the system, and N_p as the number of particles, the general energy of the non-interacting fermions in the ground state is

$$\epsilon(N_p, d) = \frac{1}{2} d N_p \hbar\omega. \tag{1}$$

This expression only holds for systems of 1 or 2 fermions, any more and the Pauli exclusion principle will need to be accounted for.

When we take the electron repulsion into account, the system can be solved analytically. We benchmark our results with the one found in [1].

2.1 Restricted Boltzmann Machine (RBM)

Our choice of NN is the Restricted Boltzmann Machine. It consists of one layer of visible nodes and one layer of hidden nodes. There is a connection from every node in the visible layer to every node in the hidden layer. The term restricted means that there is no direct link between nodes in the same layer. The goal of the RBM is to learn the probability distribution of the simulated system. In quantum systems the probability distribution is the wavefunction Ψ. The output we wish to produce is the positions of the electrons.

In some cases one might already have data that can be used to train the RBM about the system. This is not the case for this project. This project is an example of unsupervised or reinforced learning. The reinforcement in this case is the quantum variational principle. The variational principle states that the wavefunction that gives the lowest energy will be the best approximation to the ground state wavefunction. The variational principle can only give upper bounds on ground state energies.

The join probability distribution of the RBM is defined as

$$F_{RBM}(\mathbf{X}, \mathbf{H}) = \frac{1}{Z} \exp(-E(\mathbf{X}, \mathbf{H})),$$

where $\mathbf{X}$ is the visible nodes and $\mathbf{H}$ is the hidden nodes. The visible nodes in this case is the output of the system (the positions of the particles). Z is the partition function or normalization constant of the system.

$E(\mathbf{X}, \mathbf{H})$ is the function that specifies the relation between the visible and hidden nodes. It is called the energy of the node configuration. This is different from the energy of the quantum mechanical system. The choice of $E(\mathbf{X}, \mathbf{H})$ is the heart of what sort of RBM we have. In the fermionic case, the visible nodes need to take continuous values to properly represent the particle positions. We then look at the Gaussian-Binary RBM. In other cases it might be more desirable to have binary outputs. For these cases the Binary-Binary RBM can be used. The Gaussian-Binary RBM is given as

$$E(\mathbf{X}, \mathbf{H}) = \sum_{i}^{M} \frac{(X_i - a_i)^2}{2\sigma} - \sum_{j}^{N} b_j H_j + \sum_{ij}^{M,N} \frac{X_i w_{ij} H_j}{\sigma^2}, \tag{2}$$

where $\mathbf{a}$ is the visible bias, $\mathbf{b}$ is the hidden bias and $\mathbf{w}_{ij}$ is the weight of the connection between visible node i and hidden node j. We have M visible nodes and N hidden nodes. To represent the particle positions M has to be the number of particles $\times$ the number of dimensions. The RBM should have fewer hidden than visible nodes.

It is not the joint probability distribution that represents the wavefunction. Rather it is the marginal pdf. It is found by summing over all the hidden nodes. We find

$$\Psi(\mathbf{X}) = \sum_H F_{RBM}(\mathbf{X}, \mathbf{H}) = \frac{1}{Z}\sum_H \exp(-E(\mathbf{X}, \mathbf{H})).$$

Setting in the Gaussian-Binary RBM gives the final result

$$\Psi(\mathbf{X}) = \frac{1}{Z}\exp\left[-\sum_i^M \frac{(X_i - a_i)^2}{2\sigma^2}\right]\prod_j^N \left(1 + \exp\left[b_j + \sum_i^M \frac{X_i w_{ij}}{\sigma^2}\right]\right). \quad (3)$$

3 Methods

To compute the ground state energy of fermions, we use the variational method from quantum mechanics which gives an upper bound for the ground state energy E_{gs} given a trial wavefunction (in our case the Gaussian-Binary RBM marginal probability density). The variational parameters are the biases of the NN, $\boldsymbol{\theta} = a_1, ..., a_M, b_1, ..., b_N, w_{11}, ...w_{12}$.

$$E_{gs} \leq \frac{\langle \Psi_T | H | \Psi_T \rangle}{\langle \Psi_T | \Psi_T \rangle} = \frac{\int \Psi_T^*(\mathbf{X}, \boldsymbol{\theta}) H \Psi_T(\mathbf{X}, \boldsymbol{\theta}) d\mathbf{X}}{\int \Psi_T^*(\mathbf{X}, \boldsymbol{\theta}) \Psi_T(\mathbf{X}, \boldsymbol{\theta}) d\mathbf{X}}.$$

this integral is impossible to evaluate with methods such as Gaussian quadrature in a resonable amount of time. In such a situation, Monte Carlo methods are needed.

Let us consider the trial wavefunction (eq. (3)) and define the probability density function (PDF)

$$P(\mathbf{X}, \boldsymbol{\theta}) = \frac{|\Psi_T|^2}{\int d\mathbf{X} |\Psi_T|^2} \quad (4)$$

and define the local energy as

$$E_L = \frac{1}{\Psi_T} H \Psi_T. \quad (5)$$

By using eq. (4) and eq. (5), we have

$$E_{gs} \leq \frac{\int d\mathbf{X} \Psi_T^* H \Psi_T}{\int d\mathbf{X} |\Psi_T|^2} = \int d\mathbf{X}\; P(\mathbf{X}, \boldsymbol{\theta}) E_L(\mathbf{X}, \boldsymbol{\theta}) \simeq \frac{1}{N_{MC}} \sum_{i=1}^N P(\mathbf{X}_i, \boldsymbol{\theta}) E_L(\mathbf{X}_i, \boldsymbol{\theta})$$

where N_{MC} is the number of Monte Carlo steps.

In our case we will use MC methods to calculate the local energy for a set of randomly initiated $\boldsymbol{\theta}$, then apply the Stochastic Gradient Method (SGD) (more in section 3.2) to find a new $\boldsymbol{\theta}$ that will give a lower energy. This is an iterative method to find a function minimum. The final result should be a good estimate for the ground state, if it works. Since the SGD method is to be applied many times, the number of MC cycles for each SGD iteration should not be very large (to save time). Rather once the SGD cycles have finished we do a run of the final set of parameters with a large number of MC cycles to give a final answer. This is called the Variational Monte Carlo [3] (VMC) method.

However this method will not always be applicable. If the RBM is not able to converge to a precise energy but notably fluctuates, then it is not meaningful to use a final set of variational parameters for a final MC run. Instead the SGD local energy is used instead. A linear fit can be applied to the SGD data after equilibration. In this case it is actually a constant fit. This will produce a mean local energy that can be used as a final result.
During the whole project, we will use natural units $\hbar = \omega = 1$. So the energy is unitless.

3.1 Analytical solutions

The local energy is defined as

$$E_L = \frac{1}{\Psi(\mathbf{X})}\mathbf{H}\Psi(\mathbf{X}) = \frac{1}{\Psi(\mathbf{X})}\left(\sum_{i=1}^{N_p}\left[-\frac{1}{2}\nabla_i^2 + \frac{1}{2}w^2 r_i^2\right] + \sum_{i<j}\frac{1}{r_{ij}}\right)\Psi(\mathbf{X}). \tag{6}$$

The tricky to compute part about this expression is $\nabla^2\Psi$. It can be solved analytically. We start by taking the gradient, the derivative is with respect to the visible nodes. We look at the gradient with respect to the coordinates of the j'th particle. We begin with using the product rule:

$$\begin{aligned}\nabla_j\Psi(\mathbf{X}) = &-\left(\sum_{k=j}^{j+d}\frac{X_k - a_k}{\sigma^2}\mathbf{X}'_k\right) e^{\sum_i^M \frac{(X_i-a_i)^2}{2\sigma^2}} \prod_l^N \left(1 + e^{b_L} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}\right) \\ &+ e^{\sum_i^M \frac{(X_i-a_i)^2}{2\sigma^2}} \sum_{i=j}^{j+d}\prod_{l\neq j}^{M}\left(1 + e^{b_L} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}\right) e^{b_j} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}} \sum_{k=j}^{j+d}\frac{W_{kl}}{\sigma^2}\mathbf{X}'_k,\end{aligned}$$

where $\mathbf{X'}_k$ is the unit vector in the direction of $\mathbf{X}_k$ and d is the number of dimensions. We can collect terms and recognize $\Psi(\mathbf{X})$ in the expression above. We then have

$$\nabla_j\Psi(\mathbf{X}) = \Psi(\mathbf{X})\left(-\sum_{k=j}^{j+d}\frac{X_k - a_k}{\sigma^2}\mathbf{X'}_k + \sum_l^N \frac{e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}{1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}} \sum_{k=j}^{j+d}\frac{W_{kl}}{\sigma^2}\mathbf{X'}_k\right), \tag{7}$$

When finding the laplacian it will be convenient to treat the gradient as one expression. Using the product rule we then get

$$\begin{aligned}\nabla_j^2\Psi(\mathbf{X}) &= \frac{(\nabla_j\Psi(\mathbf{X}))^2}{\Psi(\mathbf{X})} + \Psi(\mathbf{X})\left[\sum_{k=j}^{j+d}\frac{1}{\sigma^2} + \sum_{l=1}^{N}\frac{e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}} \sum_{k=j}^{j+d}\frac{W_{kl}}{\sigma^2}\mathbf{X'}_k (1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}})}{(1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}})^2}\right. \\ &\qquad \left. - \frac{e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}} \sum_{k=j}^{j+d}\frac{W_{kl}}{\sigma^2}\mathbf{X'}_k e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}{(1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}})^2} \sum_{k=j}^{j+d}\frac{W_{kl}}{\sigma^2}\mathbf{X'}_k\right] \\ &= \frac{(\nabla_j\Psi(\mathbf{X}))^2}{\Psi(\mathbf{X})} + \Psi(\mathbf{X})\left[\frac{d}{\sigma^2} + \sum_{l=1}^{N} e^{b_l}\frac{e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}(1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}) - e^{b_l} e^{2\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}{(1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}})^2}\sum_{k=j}^{j+d}\frac{W_{kl}^2}{\sigma^4}\right] \\ &= \frac{(\nabla_j\Psi(\mathbf{X}))^2}{\Psi(\mathbf{X})} + \Psi(\mathbf{X})\left[-\frac{d}{\sigma^2} + \sum_{l=1}^{N}\frac{e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}{(1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}})^2}\sum_{k=j}^{j+d}\frac{W_{kl}^2}{\sigma^4}\right]\end{aligned}$$

Finally we wish look at the Laplacian normalized by the wavefunction. The result is

$$\frac{\nabla_j^2\Psi(\mathbf{X})}{\Psi(\mathbf{X})} = \left(\frac{\nabla_j\Psi(\mathbf{X})}{\Psi(\mathbf{X})}\right)^2 - \frac{d}{\sigma^2} + \sum_{l=1}^{N}\frac{e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}{(1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}})^2}\sum_{k=j}^{j+d}\frac{W_{kl}^2}{\sigma^4} \tag{8}$$

In the first term we can recognize that $\nabla_j\Psi/\Psi$ is the quantum force term $\mathbf{F}_j$ (to be further discussed in section 3.3.2).

3.2 Stochastic gradient descent (SGD)

The method of choice used to optimize our Boltzmann machine is the Stochastic Gradient Descent. The basic idea is that we want to find the minimum of the local energy as a function of the variational parameters $\boldsymbol{\theta}$.

This is done by following the direction of the negative gradient towards the minimum. The set of parameters is $\boldsymbol{\theta} = (a_1...a_M, b_1, ...b_N, w_{11}, w_{MN})$. The method updates the parameters by

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta \nabla E_L(\boldsymbol{\theta}),$$

where η is the learning rate of our RGB. We use a fixed learning rate. There exists methods that use adaptive η to speed up or optimize minimum searches. One potential downside to this method is that it is impossible to know whether the method has found a local minimum or the global minimum. In this project it is the global minimum we are interested in.

We can find expressions for the parameter gradient of the local energy. The general expression is

$$\frac{\partial \langle E_{loc} \rangle}{\partial \boldsymbol{\theta}_k} = 2 \left[\langle \frac{E_{loc}}{\Psi} \frac{\partial \Psi}{\partial \boldsymbol{\theta}_k} \rangle - \langle E_{loc} \rangle \langle \frac{1}{\Psi} \frac{\partial \Psi}{\partial \boldsymbol{\theta}_k} \rangle \right]. \tag{9}$$

To compute this value, the parameter derivatives of the wavefunction are needed. These can be solved analytically with straight forward methods. For simplicity we define $s_k = \exp\left(-b_k - \sum_i^M \frac{X_i W_{ik}}{\sigma^2}\right)$. The derivatives are then found as

$$\frac{1}{\Psi} \frac{\partial \Psi}{\partial a_k} = \frac{X_k - a_k}{\sigma^2}$$
$$\frac{1}{\Psi} \frac{\partial \Psi}{\partial b_k} = \frac{1}{1 + s_k}$$
$$\frac{1}{\Psi} \frac{\partial \Psi}{\partial w_{kl}} = \frac{X_k}{\sigma^2} \frac{1}{1 + s_l}$$

The VMC program then has to sample these quantities during the MC steps to create the means required in Eq. 9.

3.3 Sampling methods

We use MC methods to measure system quantities such as energy. The basic recipe is to choose one particle at random, and propose a step for that particle. Then according to some rule we wish to either accept or decline the move. This project will use two methods to propose and accept moves: standard Metropolis and Metropolis-Hastings.

3.3.1 Metropolis sampling

The standard Metropolis algorithm (also referred to in this text as Brute Force Metropolis) follows the basic recipe for choosing new positions given by

$$x_{k+1} = x_k + Lr,$$

where L is a step length and r is a uniformly distributed random variable $\in [0, 1]$. We accept moves that are deemed likely by the ratio of the absolute square wavefunctions before and after the move. We choose a new random number r, and if

$$r \leq \frac{|\Psi(\mathbf{r}_{k+1})|^2}{|\Psi(\mathbf{r}_k)|^2},$$

then we accept the move. If not we decline it.

3.3.2 Importance sampling

The issue with brute force Metropolis sampling is that the algorithm proposes a lot of moves that get rejected. These rejected moves are wasted computation time and as such, it becomes desirable to use an algorithm that proposes moves that are more likely to be accepted. For this we will use the Importance Sampling method. The Fokker-Planck equation is

$$\frac{\partial P}{\partial t} = D\frac{\partial}{\partial x}\left(\frac{\partial}{\partial x} - F\right)P,$$

where P is the probability density function of the system, D is the diffusion coefficient and F is the drift force. This equation governs the time evolution of the probability density function for a diffusing system under random forces. In the fermionic system the probability density function is the wavefunction itself and we call F the quantum force. $D = 1/2$ comes from the half factor in the kinetic energy.

The solution of the Fokker-Planck equation is

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

The gradient is already calculated in Eq. 9. Dividing by $\Psi(\mathbf{X})$ we get the quantum force

$$\mathbf{F}_j = \frac{2\nabla\Psi}{\Psi} = 2\left(\sum_{k=j}^{j+2}\frac{X_k - a_k}{\sigma^2}\mathbf{X'}_k + \sum_{l=1}^{N}\frac{e^{b_l}e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}{1+e^{b_l}e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}\sum_{k=j}^{j+2}\frac{W_{kl}}{\sigma^2}\mathbf{X'}_k\right).$$

The new move is given by

$$x_{k+1} = x_k + DF\Delta t + r\sqrt{\Delta t},$$

where Δt is some time step parameter that can be tuned.

Now that we have changed the proposal of moves we need to alter the method of accepting such moves. We move then to the Metropolis-Hastings algorithm. The probability of diffusion can be solved from the Fokker-Planck equation through a Green function

$$G(x_{k+1}, x_k, \Delta t) = \frac{1}{(2\pi\Delta t)^{3N/2}}\exp\left[-\frac{(x_{k+1} - x_k - \Delta t F(x_k))^2}{2\Delta t}\right].$$

The new moves are now accepted following the rule

$$r \le \frac{G(x_k, x_{k+1}, \Delta t)|\Psi(x_{k+1})|^2}{G(x_{k+1}, x_k, \Delta t)|\Psi(x_k)|^2},$$

where r is a Gaussian distributed random number

3.3.3 Gibbs sampling

Another sampling method is the so called Gibbs sampling: it uses alternatively the conditional probabilities $P(x_i|\mathbf{h})$ and $P(h_j|\mathbf{x})$ to generate new samples. These are accepted with probability one. For the Gaussian-Binary RBM, the conditional probabilities are

$$P(h_j = 1|\mathbf{x}) = \frac{1}{1+e^{-b_j - \frac{1}{\sigma^2}\sum_i x_i w_{ij}}} \quad (10)$$

$$P(x_i|\mathbf{h}) = \mathcal{N}(x_i, a_i + \sum_j h_j w_{ij}, \sigma^2) \quad (11)$$

Therefore we use the old position to generate the new hidden nodes (eq. (10)) which are used to compute new positions according to eq. (11).

For the Gibbs sampler we have to use the square root of the RBM wavefunction, $\Psi_G(\mathbf{X}) = \sqrt{F_{RBM}(\mathbf{X}}$. This means we have to calculate new expressions for the local energy, as well as the expressions for the parameter gradients. A very useful trick can be used in this situation, as we have already calculated all the derivatives for $\Psi(\mathbf{X}) = F_{FBM}(\mathbf{X})$. We look at the logarithm of the wavefuntion and its derivative, $\ln(\Psi_G) = \frac{1}{2}\ln(\Psi)$.

$$\frac{\partial \ln(\Psi_G)}{\partial x} = \frac{1}{\Psi_G}\frac{\partial \Psi_G}{\partial x} = \frac{1}{2\Psi}\frac{\partial \Psi}{\partial x}.$$

Thus, when we look at local values (normalized by wavefunction), we only need multiply in a 1/2. This can be applied to the SGD gradients (Eqs. 3.2-3.2).

For the local energy we need to consider the second derivative, this complicates matters a bit. Starting by taking the general gradient of the new wavefunction we get

$$\nabla_j \Psi^{1/2} = \frac{1}{2\Psi^{1/2}}\nabla_j \psi$$

$$\nabla_j^2 \Psi^{1/2} = -\frac{1}{4\Psi^{3/2}}(\nabla_j \Psi)^2 + \frac{1}{2\Psi^{1/2}}\nabla_j^2 \Psi$$

Dividing by the wavefunction to get the local energy returns

$$\frac{\nabla_j^2 \Psi^{1/2}}{\Psi^{1/2}} = -\left(\frac{\nabla_j \Psi}{2\Psi}\right)^2 + \frac{1}{2\Psi}\nabla_j^2 \Psi$$

Inserting eq. (8) as the solution to the second derivative yields

$$\frac{\nabla_j^2 \Psi^{1/2}}{\Psi^{1/2}} = \left(\frac{\nabla_j \Psi}{2\Psi}\right)^2 + \frac{1}{2}\frac{d}{\sigma^2} + \frac{1}{2}\sum_{l=1}^{N} \frac{e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}}}{(1 + e^{b_l} e^{\sum_i^M \frac{X_i W_{il}}{\sigma^2}})^2} \sum_{k=j}^{j+d} \frac{W_{kl}^2}{\sigma^4}.$$

A method of implementing this to the code is to define a parameter that is 1 for the Metropolis sampling methods and 1/2 for Gibbs sampling. Then all that is required is to multiply this parameter in the appropriate places.

3.4 Statistical analysis - blocking method

It is important to have an estimate for the error of our results, without it they are useless. The simple way of getting an error estimate is to take the standard deviation of our data. The issue with that method is that the standard deviation does not take the correlation of the data into account and, in this project, in fact, data are correlated. It is simple to see that because the energy is a time series: the energy does change discontinuously, in Monte Carlo steps, but the change is typically small compared to the total energy and thus it can be classified as a time series.

An excellent tool for analyzing time series correlation is the blocking method [4]. The basic idea is to start with a correlated dataset and turn it into an uncorrelated dataset which we take the standard deviation of. The method works in this way: suppose that our initial dataset is

$$\mathbf{X}_0 = (X_1, X_2, ..., X_N) \quad \text{where } X_1 = (\mathbf{X_0})_1,$$

we average $X_1$and X_2 and place it into $(\mathbf{X}_1)_1$. Then we do the same for X_3 and X_4 and place it in $(\mathbf{X}_1)_2$ and so on. This is one blocking iteration. Our new array is less correlated because the elements are further from each other than in the original time series.

By following the calculations of [4], the variance of the data set after blocking k times is

$$V(\bar{\mathbf{X}}_k) = \frac{\sigma_k^2}{n_k} + \frac{2}{n_k}\sum_{h=1}^{n_k-1}\left(1-\frac{h}{n_k}\right)\gamma_k(h) = \frac{\sigma_k^2}{n_k} + \epsilon_k,$$

where $\bar{\mathbf{X}}_k$ is the mean of $\mathbf{X}_k$, σ_k is the standard deviation in $\mathbf{X}_k$, n_k is the number of elements in $\mathbf{X}_k$ and γ_k is the autocovariance of $\mathbf{X}_k$. We call ϵ_k the truncation error. It can be shown [4] that ϵ_k tends to zero as we keep on blocking. The result is that the error of the mean is just the standard deviation of the data set. This error should better reflect the error of our results than the simple standard deviation and it should also be larger since it takes into account correlations.

The blocking method is specifically useful for large data sets since the computation cost is of the order of $O(n)$ whereas other methods are $O(n^2)$ or worse. Indeed, when we are dealing with data sets sizes of the order of $10^6 - 10^7$, there is a large speed difference between $O(n)$ and $O(n^2)$.

4 Results

Before presenting the results from the different sampling methods we need to tune parameters such as Metropolis step length, Importance Sampling time step, and the learning rate. This is discussed in these subsections. At the end, the energies from the different sampling methods are presented together.

4.1 Equilibrating sampling methods to non-interacting electrons

In the non-interacting case, the best solution is when the $\boldsymbol{b}$ and $\boldsymbol{w}$ are close to zero. Therefore, if we choose a small standard deviation when generating initial parameters, we will already start with a guess that is close to the final solution. This is not very interesting, because it won't allow us to see how the RBM develops. Therefore we set the parameter $\sigma = 0.5$ to give a spread and an initial energy that is not correct.

4.1.1 Metropolis sampling

First of all, we study how the system reacts when the steplength of the metropolis sampling is changed. In Fig. (1) we show how the SGD energy optimizes towards equilibrium for various Metropolis step lengths with a learning rate $\eta = 0.01$ applied to 2 particles in 2 dimensions. In this case, from eq. (1), we expect the correct energy to be $\epsilon = 2$. As we can see from the figure, $L = 0.01$ and $L = 0.10$ use a long time to equilibrate whereas the others are quite quick. In particular $L \geq 0.5$ reaches equilibrium after only a few cycles, which is desirable. Therefore it appears that a larger step length produces a faster equilibrium. This is reasonable: with a large step length we move more the particles at each step and thus even though we start with a bad configuration, we will reach faster a good one through the SGD. Nevertheless we note that all the step lengths converge (in time) to the correct energy. Moreover we study also the behaviour of the acceptance ration as a function of the step length. The results are shown in Tab. (1). We note that using a very large L results in a small acceptance ratio and by consequence we sample the same positions many times, effectively producing worse statistics. Therefore it is desirable to use a step length that results in a higher acceptance ratio.

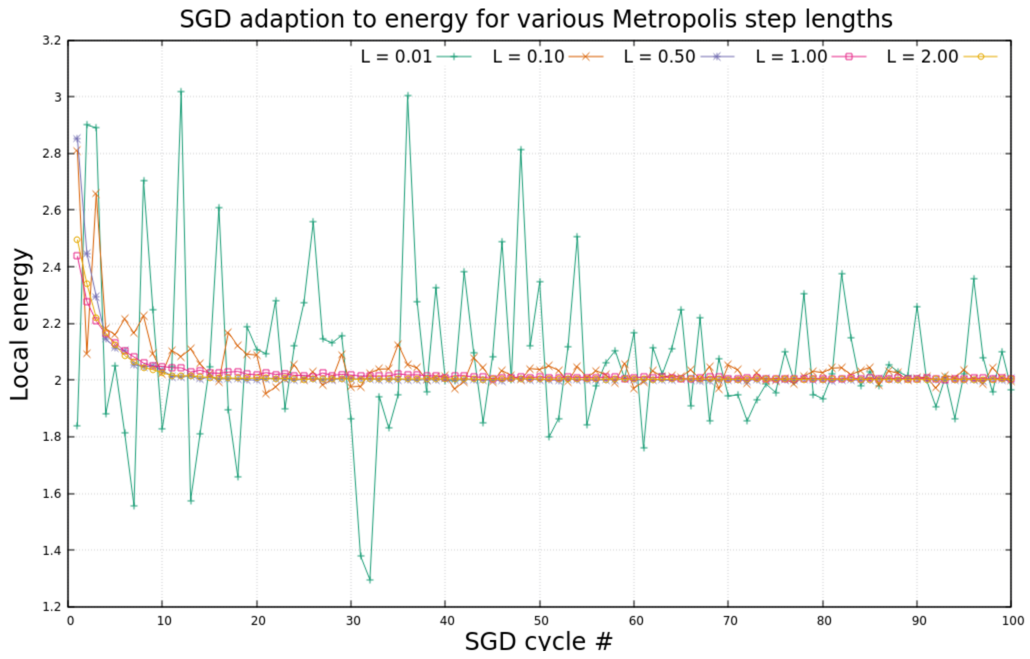

Figure 1: Local energy evolution calculated as the program adapts using stochastic gradient descent for various Metropolis step lengths. The figures are made using $\eta = 0.10$. Notice how for $L \geq 0.5$ are quite stable at $E_L = 2$.

Table 1: Acceptance ratio as function of Metropolis step length. The data are made using 2 particles in 2 dimensions.

L	**Acceptance ratio**
2	0.73
1	0.85
0.5	0.93
0.1	0.985
0.01	0.999

In Fig. (2) we study the evolution of the standard deviation of the local energy as the the SGD cycles advance. From the figure we can extract that by using small step length ($L = 0.01, 0.1$) the error is larger than the other cases and that the std. dev. undergoes large fluctuations. On the other hand for larger step length, the fluctuations decreases, but the error is still larger than the medium step length case. Therefore the best option is to use a medium step length. In the figure, $L = 0.50$ produced the best results: the system converges in about 20 cycles and than it fluctuates around the value reached ($\sigma_{E_L} \simeq 10^{-3.9}$).

By combining the data from Figs. (1), Fig. (2) and Tab. 1, we conclude that $L = 0.50$ is an appropriate choice of step length for the Metropolis sampler.

To understand what is the best learning rate to use, we plot in Fig. (3) the SGD energy optimization of the RBM applied to 1 electron in 2d for multiple learning rates. The figure shows that small learning rates use a long time to converge. $\eta = 0.01$ does converge to the same value after approximately 100 SGD cycles. $\eta = 0.05$ converges after 50 cycles and $\eta = 0.1, 0.2, 0.5$ converges after 20, 10 and 5 cycles respectively. Increasing η beyond 0.6 results in an energy diverging towards $\pm\infty$.

It is useless to choose a learning rate that takes a long time to converge. Hence we choose to use $\eta = 0.20$. Changing the number of hidden nodes does not produce interesting results for the Metropolis sampler.

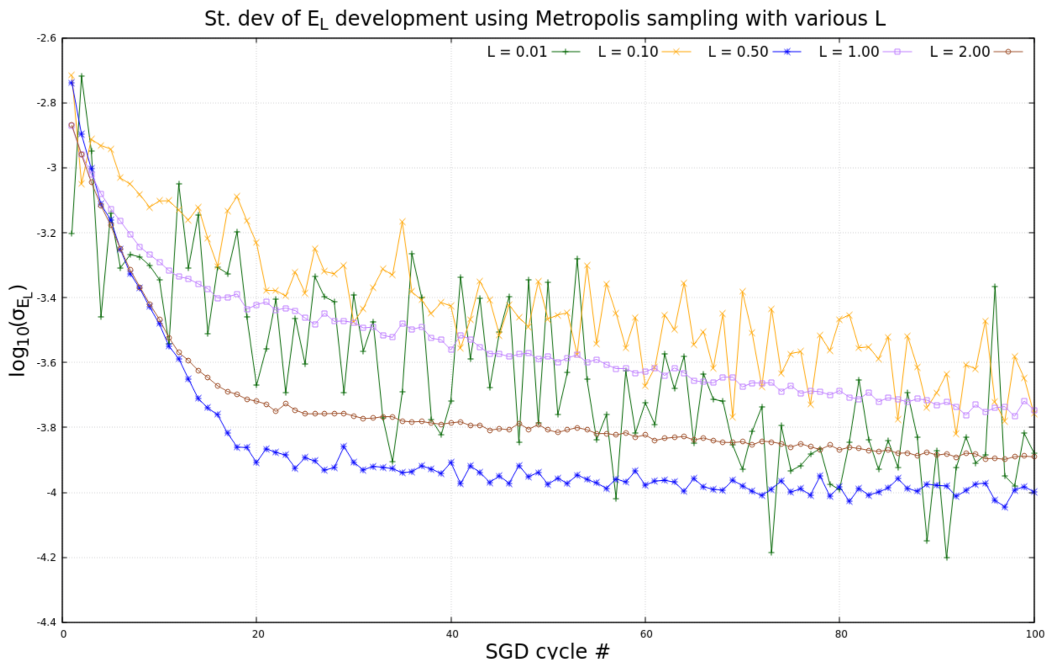

Figure 2: St. dev of local energy development changes with L. All the data are made using 2 particles in 2 dimensions and $\eta = 0.1$. The data suggests $L = 0.5$ is an appropriate step size. The log of the st. dev is used for visual clarity.

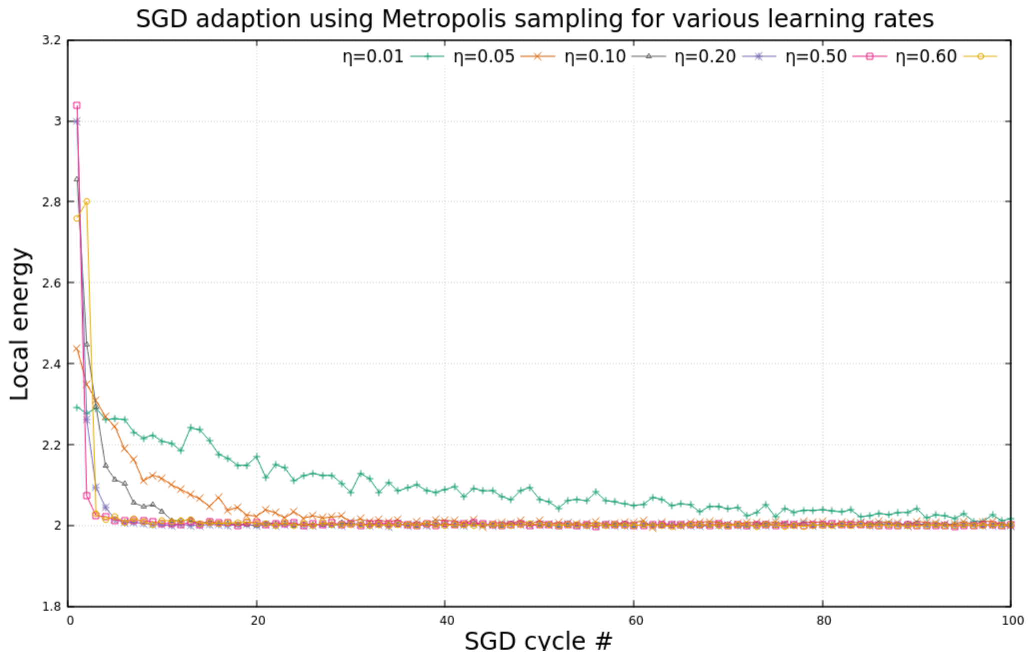

Figure 3: Local energy evolution calculated using standard Metropolis sampling as the program adapts using SGD for one particle in 2d. Metropolis step length 0.5 is used for all the runs. Each SGD cycle uses 300000 MC steps. $\eta > 0.60$ diverges and leaves the plot after few cycles. The rest all converge and the smaller learning rates give slower convergence.

4.1.2 Importance Sampling

As we have done above with the Metropolis Sampling, we study the system with the Importance Sampling by taking into account its behaviour as we change the time step Δt.

In Fig. (4) is shows the SGD cycle energy adaption for multiple Importance Sampling Δt. The data are made using $\eta = 0.10$. We can see that the velocity of convergence for Δ t $\in [0.001, 2.00]$ is almost the same. However, in the case of $\Delta t = 0.001$, the system does not converge well to the energy minimum: this is the only data series that visibly fluctuates. From this we can conclude that we should use $\Delta t > 0.001$.

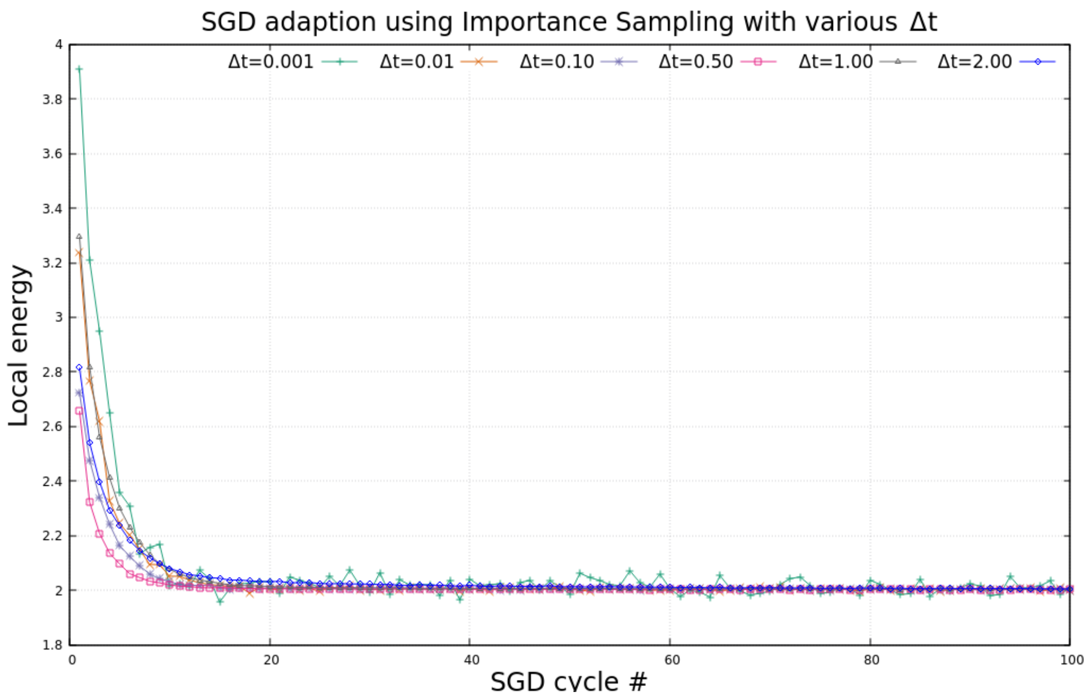

Figure 4: SGD adaption of Importance Sampling to local energy of 2 electron in 2d. The data are made using $\eta = 0.10$ and 2 hidden nodes. The convergence time is very fast, and is almost independent of Δt. Using $\Delta t = 0.50$ is faster than the other time steps. All the time steps converge to the same energy. Using $\Delta t = 0.001$, the local energy fluctuates visibly for 100 cycles. It will therefore be favorable to use $\Delta t > 0.001$.

To narrow the time step internal, we plot (Fig. (5)) the logarithm of the std. dev. of the local energy of the data from Fig. 4 as function of SGD cycles. The data show that approximately the same std. dev. is produced by using $\Delta t \in [0.01, 0.50]$. The std. dev is much more smooth for $\Delta t = 0.10, 0.50$. Using smaller or larger Δt results in larger std. dev.

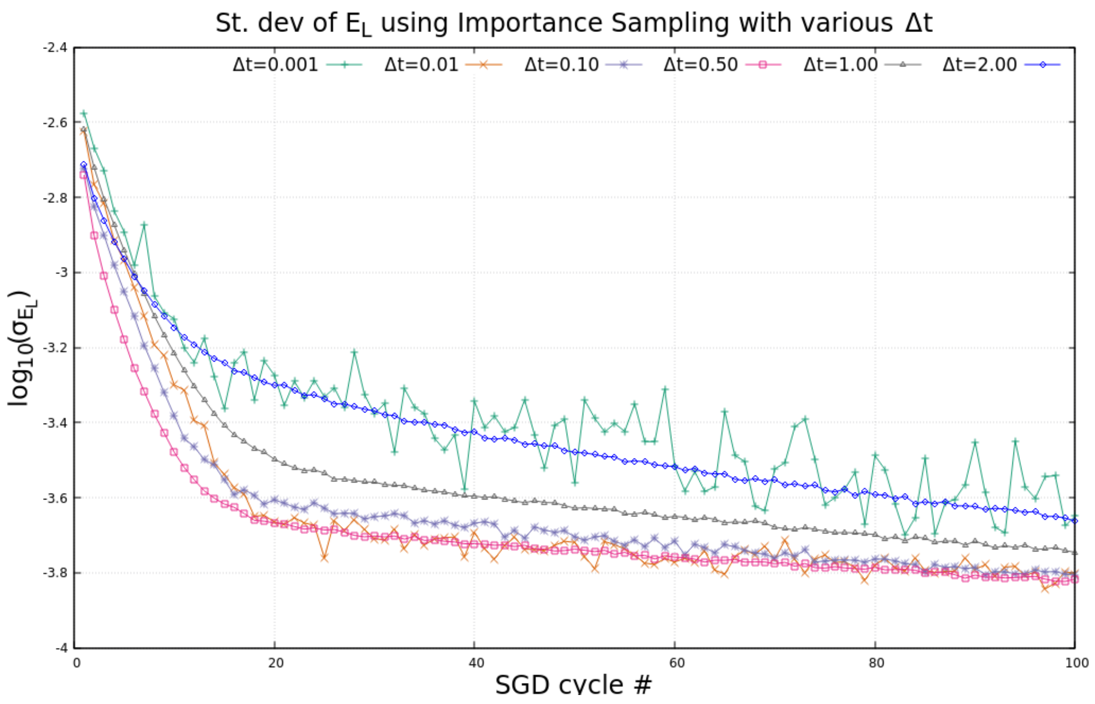

Figure 5: SGD adaption of Importance Sampling to local energy of 2 electron in 2d. The data are made using $\eta = 0.1$. $\Delta t \leq 0.05$ use a long time to converge to the energy. It will therefore be favorable to use $\Delta t > 0.05$.

In Tab. 2 we show the acceptance rate of Metropolis steps using Importance Sampling as a function of Δt. Again we wish to have a large acceptance ratio. This ratio should not be close to one, though. In fact that would mean that we are accepting all the moves which is a signal that we are not moving enough the particles at each Metropolis step. From the table, the best option is obviously $\Delta t = 0.50$.

Table 2: Acceptance ratio as function of Metropolis step length. The data are made using 2 particles in 2 dimensions and 2 hidden nodes.

Δt	**Acceptance ratio**
2	0.49
1	0.78
0.5	0.92
0.1	0.993
0.05	0.997
0.01	0.9998
0.001	0.99999

By combining this with the considerations expressed above, we conclude $\Delta t = 0.50$ is an acceptable time step. It will be used in the further computations.

As regards the learning rate, we note that varying η produces results very similar to the ones of Fig. (3). Therefore we take the same η used also for brute force Metropolis which was $\eta = 0.2$.

4.1.3 Gibbs Sampling

After having implemented the Gibbs sampler, we not that using $\sigma = 1.00$ does not give good results. Therefore we decide to vary sigma and to check how the system reacts.

In Fig. (6) we show the SGD adaption to the local energy of 2 electrons in 2d using the Gibbs sampler with various σ. The system converges to a value of local energy which depends on σ used. The exact results is 2, therefore as we can see from the figure, $\sigma = 0.7$ gives the closest energy to the exact ground state. By increasing it to 0.8 or by decreasing it to 0.6, we obtain worse results. It is difficult to fine tune this precision to more decimals because of how little is the change of the local energy with small changes of σ around 0.7. However we note that this sampler is much more dependent on the parameter σ than the others. We consider σ to be constant, but in principle it should be a parameter which is varied through SGD as all the others (a_i, b_i, etc). We do not have any prove but we can suppose that by implementing the variation of σ trhrough SGD we could overcome the problems connected with Gibbs sampling. Anyway, we choose $\sigma = 0.7$ for the non-interacting electrons.

Then, we study the behaviour of the system with the Gibbs sampling when the learning rate is varied. Fig. (7) shows the Gibbs sampler adapting to the ground state of 2 electrons in 2 dimensions for multiple η as function of the SGD cycles with $\sigma = 0.7$. We note that by using an $\eta \in [0.01, 0.6]$, the energy converges to a result. Increasing the value of η past this interval results in divergence. All the learning rates chosen converge to the right result (2) in about 25 cycles except for $\eta = 0.01$ which is consequently removed from our considerations. Learning rates greater than 0.2 converge basically at the same velocity. We choose $\eta = 0.2$.

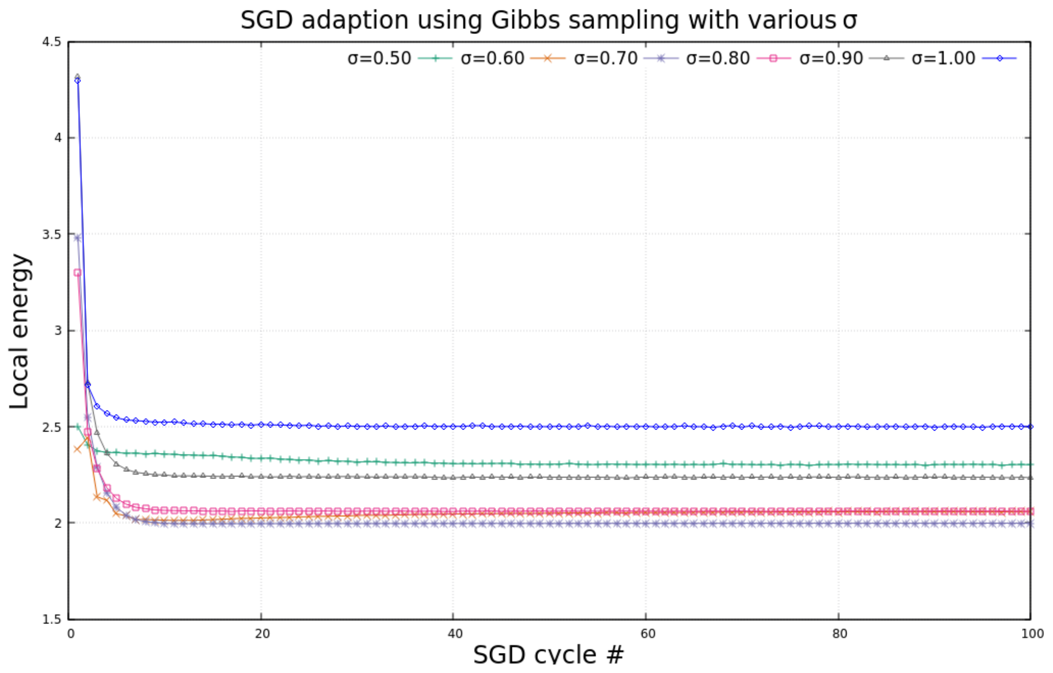

Figure 6: SGD adaption of Gibbs sampler to local energy of 2 electrons in 2d using 2 hidden nodes for varying σ. The data are made using $\eta = 0.2$. Setting $\sigma = 0.7$ gives the closest result to the exact local energy.

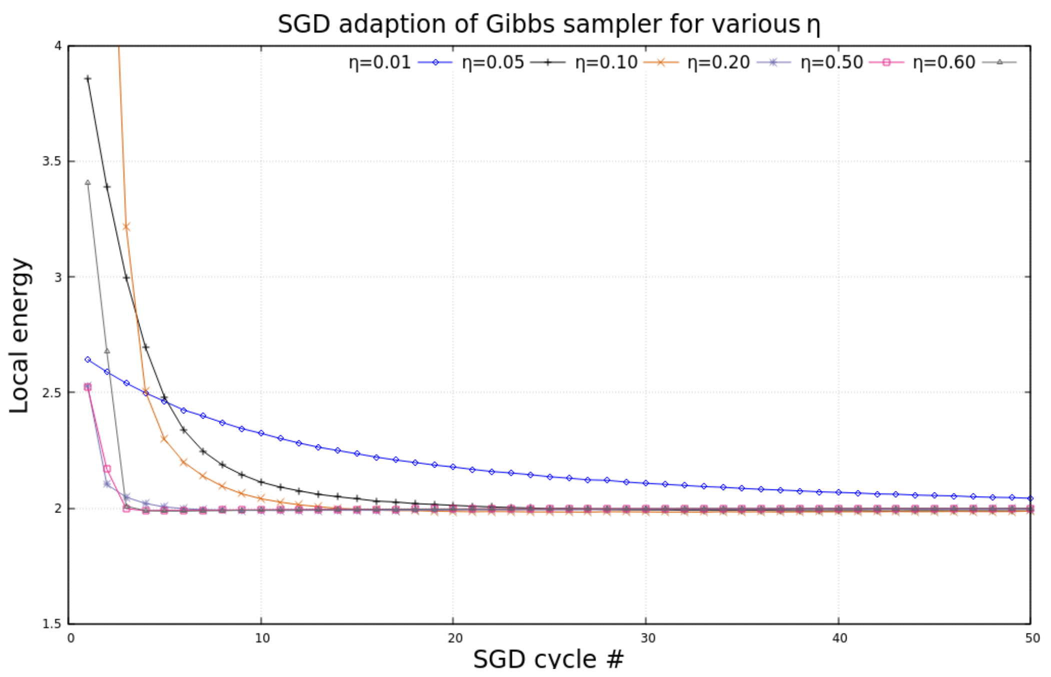

Figure 7: SGD adaption of Gibbs sampler to local energy of one 2 electron in 2 dimensions for various η with 2 hidden nodes. All the learning rates in $[0.05, 0.6]$ converge well to the same energy in about 25 cycles.

4.2 Energy of non-interacting electrons

Having found all the "right" parameters for each sampler, we proceed to give some values of energy for different configurations of particles, dimensions and number of hidden nodes.

In Tab. 3 we show the ground state local energy of non-interacting electrons computed with the different samplers. Metropolis, with and without importance sampling, and Gibbs sampling all produce exactly correct results with an error of the order of $10^{-3} - 10^{-6}$. In general it appears importance sampling produces smaller errors than standard Metropolis. For the Metropolis algorithms the error increases as more hidden nodes are added. For Gibbs sampling, the error is approximately constant with changes in number of hidden nodes.

Generally Gibbs sampling returns the smallest error. This could be improved by a further analysis of the Gibbs optimal σ.

Table 3: Local energy of non-interacting electrons. N_p is the number of electrons, d is the dimensionality and N_H is the number of hidden nodes. All energies are in units of $\hbar\omega$. All the calculations are computed using 300 SGD cycles of 300 000 MC steps. The final run to compute the energy is done with 10^7 MC steps. $\eta = 0.2$ and $\omega = 1$. The Metropolis step length is $L = 0.5$, Importance Sampling timestep is $\Delta t = 0.5$. $\sigma = 1.00$ is used for Metropolis and Importance Sampling, $\sigma = 0.70$ is used for Gibbs sampling. The error is calculated using the blocking method.

N_p	d	N_H	Exact	Metropolis	Importance	Gibbs
1	1	1	0.5	$0.5 \pm 3 \times 10^{-5}$	$0.5 \pm 2 \times 10^{-5}$	$0.5 \pm 1 \times 10^{-6}$
1	2	1	1.0	$1.0 \pm 2 \times 10^{-4}$	$1.0 \pm 4 \times 10^{-6}$	$1.0 \pm 2 \times 10^{-5}$
1	2	2	1.0	$1.0 \pm 1 \times 10^{-4}$	$1.0 \pm 5 \times 10^{-5}$	$1.0 \pm 1 \times 10^{-5}$
2	1	1	1.0	$1.0 \pm 2 \times 10^{-4}$	$1.0 \pm 5 \times 10^{-5}$	$1.0 \pm 2 \times 10^{-5}$
2	1	2	1.0	$1.0 \pm 2 \times 10^{-4}$	$1.0 \pm 3 \times 10^{-5}$	$1.0 \pm 2 \times 10^{-5}$
2	2	1	2.0	$2.0 \pm 3 \times 10^{-4}$	$2.0 \pm 5 \times 10^{-5}$	$2.0 \pm 3 \times 10^{-5}$
2	2	2	2.0	$2.0 \pm 5 \times 10^{-4}$	$2.0 \pm 1 \times 10^{-4}$	$2.0 \pm 2 \times 10^{-5}$
2	2	3	2.0	$2.0 \pm 1 \times 10^{-3}$	$2.0 \pm 1 \times 10^{-4}$	$2.0 \pm 2 \times 10^{-5}$
2	2	4	2.0	$2.0 \pm 1 \times 10^{-3}$	$2.0 \pm 1 \times 10^{-4}$	$2.0 \pm 2 \times 10^{-5}$

4.3 Equilibrating sampling methods to interacting electrons

Our aim is to find the local energy of 2 interacting electrons in 2 dimensions because we can benchmark those results with the literature [1].

We set the spread of initial variational parameters to be $\sigma = 0.5$. We will use the same Metropolis step length and Importance sampling time step as we did for the non-interacting case. They were $L = 0.5$ and $\Delta t = 0.5$.

4.3.1 Metropolis sampling

First of all we study our system with the Metropolis sampling by checking how it reacts when the learning rate is varied. In Fig. (8) we show how the Metropolis sampler adapts to 2 interacting electrons in 2d using 2 hidden nodes for different η. All the learning rates make the RBM converge to the same local energy. Using $\eta = 1.0, 0.5$, the system converges at about the same rate. An oddity occurs between $\eta = 0.1$ and $\eta = 0.2$. Using $\eta = 0.2$, the local energy is almost constant for 200 cycles before descending at a fast rate. Using $\eta = 0.1$, the local energy decreases in a linear fashion. Despite this, both learning rate converge to the same local energy using approximately the same number of SGD cycles.

Generally it can also be seen that the RBM uses more cycles to adapt to the interacting electrons wavefunction than the non-interacting one. Plotting the std. dev development shows no notable difference between the learning rates. From this we determine to use $\eta = 0.1$. Then we need to do 400 cycles to let the system to reach equilibrium and then we do 200 cycles to gather data.

We can also see that the standard VMC method of doing a final run with a final set of variational parameters is not applicable to the results from the interacting electrons. The local energy is fluctuating too much for this to give meaningful results. Instead the SGD cycle data will be used directly to give final energies using a linear fit to the local energy after equilibration.

The energy of interacting electrons is much more difficult to estimate than the non-interacting one for the Metropolis sampler. The local energy produced varies much more with the number of hidden nodes. Fig. (9) shows the SGD adaption for different number of hidden nodes. The data are made using $\eta = 0.1$ and $\sigma = 1.0$. The results of using 2, 3 or 4 hidden nodes are really similar. It appears using more hidden nodes allows the system to converge using slightly fewer SGD cycles. Using only 1 hidden node results in a notably larger and worse local energy than for the other numbers of hidden nodes.

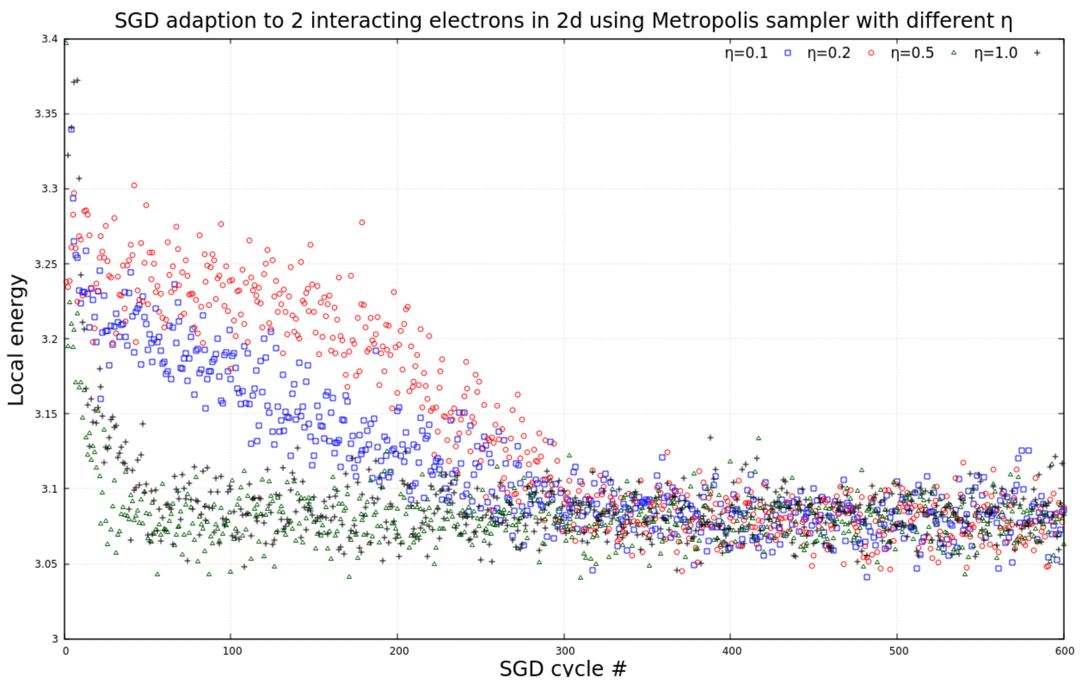

Figure 8: Metropolis sampler adapting to 2 interacting electrons in 2d with 2 hidden nodes, using different learning rates. The data are made using $\sigma = 1$ and $L = 0.5$. Increasing η from 0.1 to 0.2 changes the behavior but does not change the convergence time. Increasing past this makes the RBM converge with less cycles.

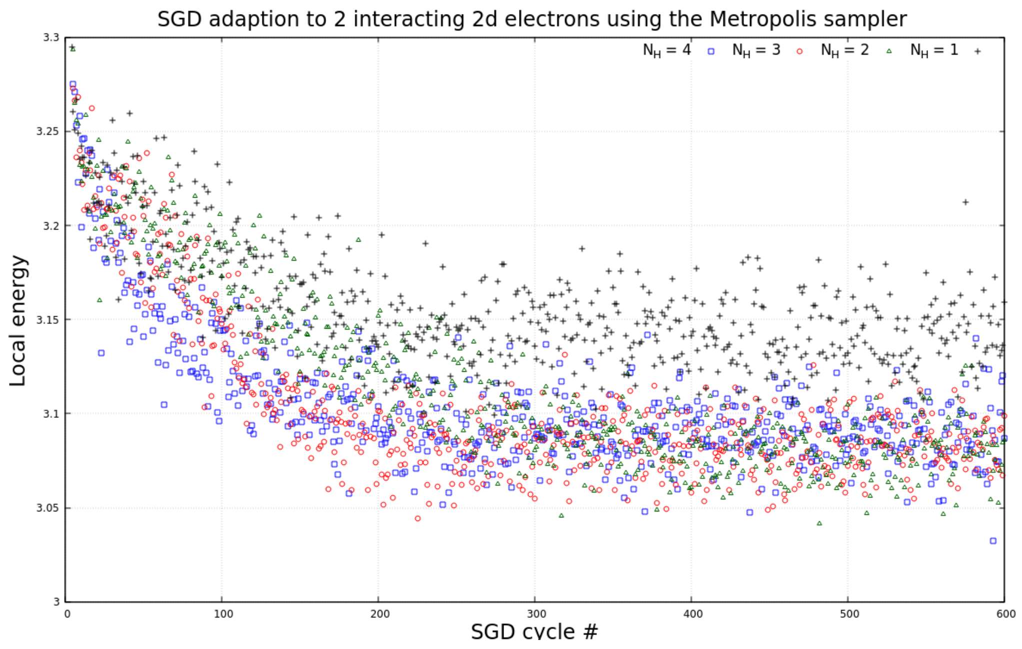

Figure 9: SGD adaption of Metropolis sampler to local energy of 2 interacting electrons in 2d for different number of hidden nodes. For this, $\sigma = 1$, $\eta = 0.2$ and $L = 0.5$. The exact local energy is 3. Using 1 hidden node produces bad results, using 2, 3 or 4 gives about the same result (see Tab. 4 for more detailed analysis).

4.3.2 Importance Sampling

For the non-interacting electrons, the SGD adaption variation between the Metropolis and Importance samplers was identical with regards to variations in the learning rate. In the interacting case there are large visual differences between the two samplers. Fig. (10) shows the Importance sampler adapting to interacting electrons in 2d while varying the learning rate. In the case of $\eta = 0.1$, the local energy drops quickly for a few cycles and then it descends steadily towards convergence. As for $\eta = 0.2$, the local energy shifts between being approximately constant and developing in sharp descents. Despite taking very different paths, $\eta = 0.1$ and $\eta = 0.2$ converge after approximately 375 SGD cycles. $\eta = 0.5, 1.0$ quickly descend and converge in less than 100 SGD cycles. By looking at the local energy after convergence for the different η, it is possible to see that $\eta = 0.1$ gives a larger energy than the other η. Therefore this learning rate should not be used with Importance sampling. We choose to use $\eta = 0.2$ in this case.

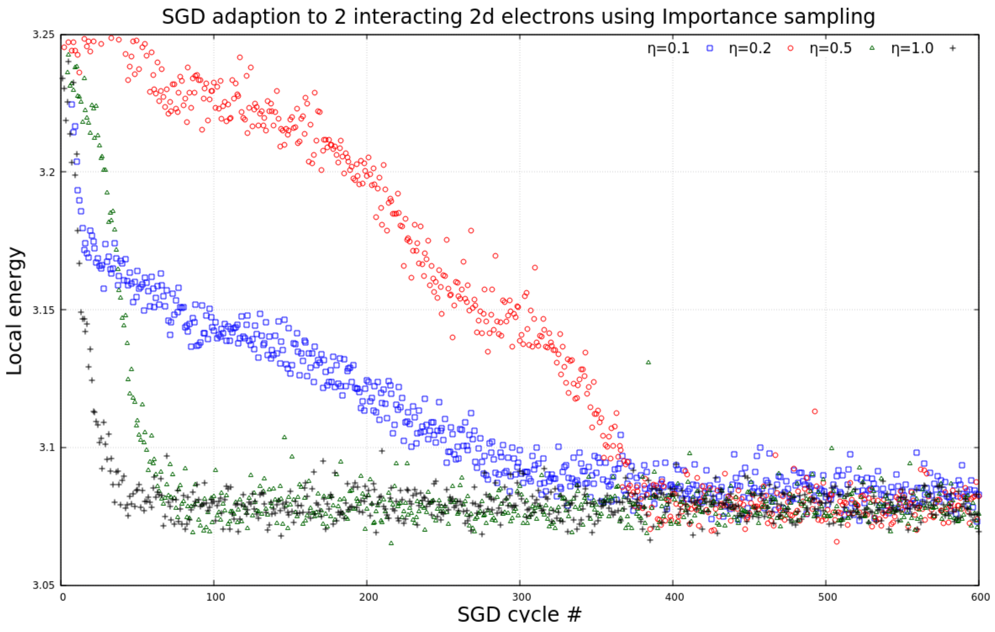

Figure 10: SGD adaption of Importance sampler to local energy of 2 interacting electrons in 2d using 2 hidden nodes for different learning rates. For this, $\sigma = 1$ and $\Delta t = 0.5$. The exact local energy is 3. As in the Metropolis case (Fig. (8)), the convergence is slower using $\eta = 0.2$ than $\eta = 0.1$.

As we have done above for the brute force Metropolis, in Fig. (11) we check the local energy SGD adaption to 2 electrons in 2 dimensions as function of SGD cycles using the Importance sampler for different number of hidden nodes. As with the Metropolis sampler, using only one hidden node does not give a good final result. Using 2, 3, or 4 hidden nodes gives approximately the same result. Using 4 nodes gives the fastest convergence and using fewer makes it slower.

An interesting property appears when using only 2 hidden nodes. After approximately 250 SGD cycles the local energy stabilizes for 100 or so cycles before converging to the final energy minimum. It seems that the energy stabilizes at the same energy reached with 1 hidden node. This means that when the RBM only has one hidden node it is not able to escape the minimum, whereas with 2 hidden nodes the RBM can. By using 3 or 4 hidden nodes, the RBM does not become trapped in the minimum at all.

By comparing Fig. (11) with Fig. (9) we can see that both the Metropolis and Importance samplers converge to an energy minimum with less SGD cycles when the number of hidden nodes is increased. And we can see that for the same learning rate, all the Metropolis runs have converged after 300 cycles whereas the fastest Importance sampling run converges at 300 and the slowest after 400 cycles.

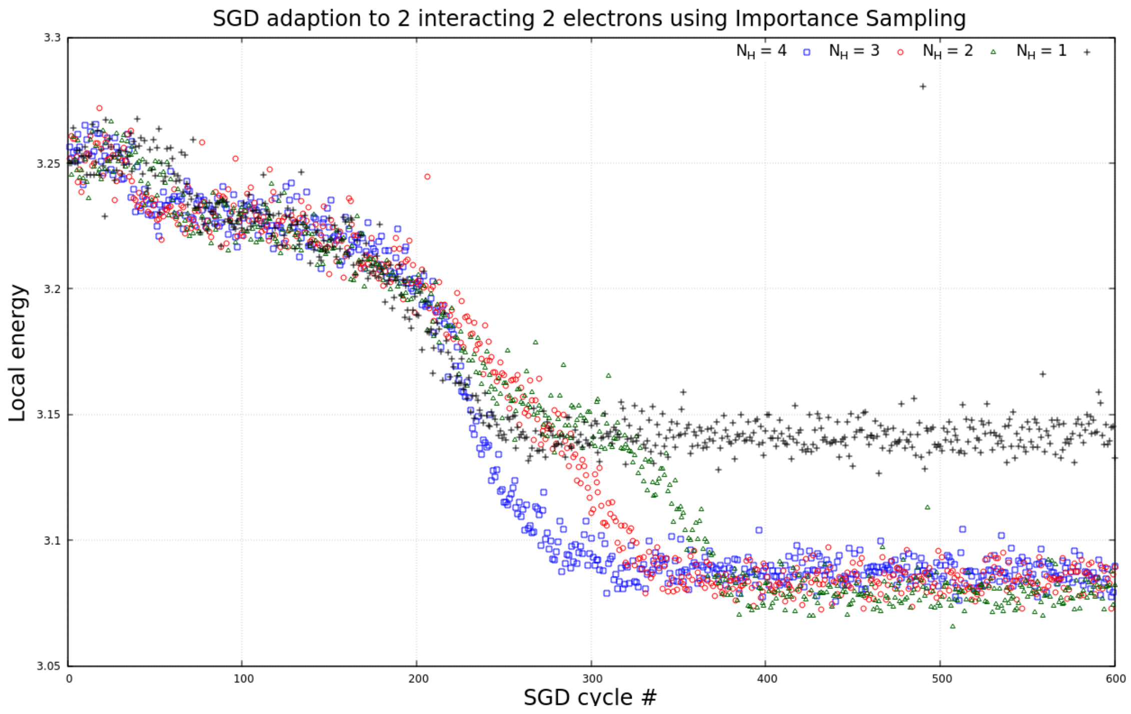

Figure 11: SGD adaption of Importance sampler to local energy of 2 interacting electrons in 2d for different number of hidden nodes. The data are made using $\sigma = 1$,$\eta = 0.2$ and $\Delta t = 0.5$. The exact local energy is 3.0. Using 1 hidden node produces a value completely different from what we expect, whereas by using 2,3 or 4 we get about the same result (see Tab. 4 for more detailed analysis). Increasing the number of hidden nodes makes the sampler converge to an energy minimum using fewer SGD cycles.

4.3.3 Gibbs sampler

The Gibbs sampler does not do a good job at representing the interacting electron wavefunction. An example of this is seen in Fig. (12) where the local energy adaption using the Gibbs sampler is plotted for two runs using the same setting except for the initial parameters which are drawn from a gaussian distribution. In both cases, the sampler converges after only a few cycles. The issue is that the system converges to two completely different values of energy. Hence the local energy the sampler converges to is meaningless for our discussion.

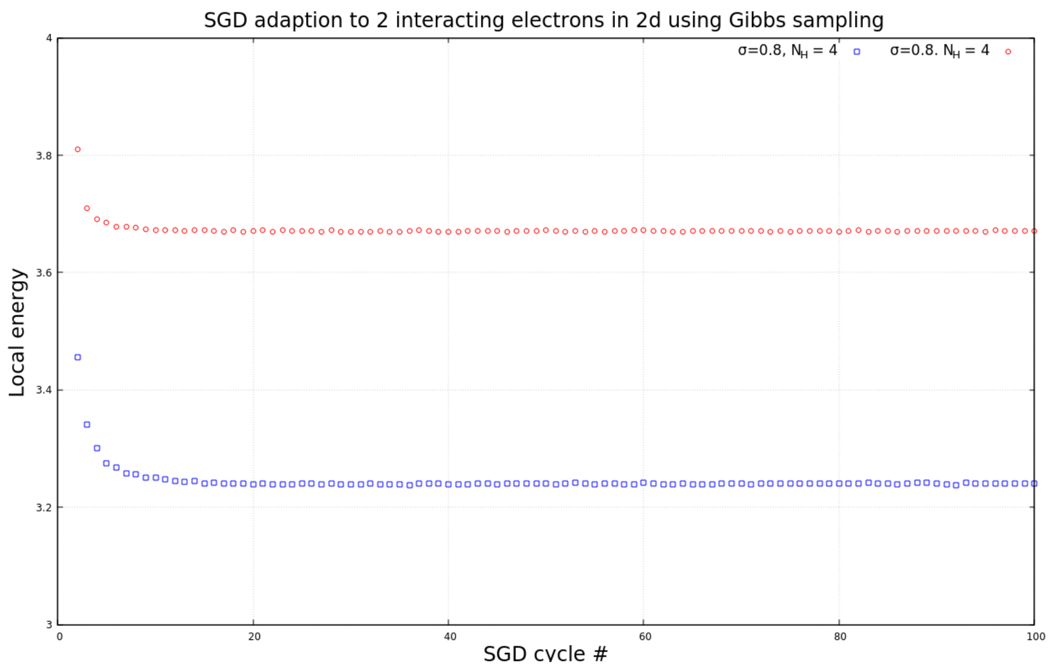

Figure 12: SGD adaption of Gibbs sampler to local energy of 2 interacting electrons in 2d using 4 hidden nodes. The figure shows two runs using the same controlled parameters, the only difference being the initial configuration of variational parameters. The issue is that the sampler does not converge to a defined energy minimum but instead is dependent on the initial configuration.

4.4 Energy of interacting electrons

In Tab. 4 we present the local energy calculated by the different samplers for the interacting electrons. All the runs with all variations of hidden nodes appear to produce quite good results for an upper limit to the ground state energy. Using 2, 3 or 4 hidden nodes is much better than using only 1. There appears to be a slight trend in which using 2 hidden nodes is better than using 3 or 4. This happens for both the standard Metropolis and the Importance sampler.

The results are quite similar for both the Metropolis and Importance samplers. The Importance sampler tends to produce errors that are 3 times better than the Metropolis errors. Using 2 or 3 hidden nodes, the Metropolis sampler produces better results than the Importance sampler. The best result is found using 2 hidden nodes with the Metropolis sampler, it is $\epsilon = 3.068 \pm 9 \times 10^{-3}$.

Table 4: Local energy of interacting electrons. N_p is the number of electrons, d is the dimensionality and N_H is the number of hidden nodes. All energies are in units of $\hbar\omega$. All the calculations are computed using 600 SGD cycles of 300 000 MC steps. The local energy is calculated using linear fits to SGD data after equilibration (400 cycles, see above sections). The Metropolis sampler uses $\eta = 0.1$ and step length $\Delta L = 0.5$. The Importance sampler uses $\eta = 0.1$ and time step $\Delta t = 0.5$. Both samplers use $\omega = 1$ and $\sigma = 1.00$. The Gibbs sampler did not produce usable results for the system interacting electrons.

N_p	d	N_H	Exact	Metropolis	Importance	Gibbs
2	2	1	3.0	$3.13 \pm 1 \times 10^{-2}$	$3.14 \pm 1 \times 10^{-2}$	-
2	2	2	3.0	$3.068 \pm 9 \times 10^{-3}$	$3.080 \pm 3 \times 10^{-3}$	-
2	2	3	3.0	$3.072 \pm 9 \times 10^{-3}$	$3.085 \pm 3 \times 10^{-3}$	-
2	2	4	3.0	$3.096 \pm 1 \times 10^{-2}$	$3.088 \pm 3 \times 10^{-3}$	-

5 Conclusions

The goal of this report was to use a RBM with VMC methods to calculate the ground state energy of interacting and non-interacting electrons in 2-dimensions. Using standard Metropolis, Importance and Gibbs sampling we precisely replicate the wavefunction of 1 and 2 non-interacting electrons. We were able to replicate the theoretical results to numerical precision. Applying the same methods to the interacting electrons we were able to produce good results using Metropolis and Importance sampling. The Gibbs sampler did not work for the interacting electrons. For 2 interacting electrons in 2-dimensions, the best result was produced by the Metropolis sampler using 2 hidden nodes. It is $\epsilon_{gs} \leq 3.068 \pm 9 \times 10^{-3}\ \hbar\omega$. The exact result is $\epsilon_{gs} = 3\ \hbar\omega$. The variational method only allows us to find an upper bound for the ground state. With this in mind our results are quite good.

A tendency was found in the results when varying the number of hidden nodes. Both the Metropolis and Importance samplers produced the best results when using 2 hidden nodes. Adding more nodes gave larger ground state energies. In the non-interacting case, adding more nodes produced larger errors. This behavior might be due to the increased complexity of the stochastic gradient. Adding more variational parameters might allow for more errors as the RBM is attempting to converge to an energy minimum.

We did not get the Gibbs sampler to work when applied to the system of interacting electrons. It is strange that it produced good results for the non-interacting electrons, and completely random results for the interacting ones. This leads us to think that perhaps the sampler wasn't working properly for the non-interacting electrons either. Perhaps it was just by coincidence that it was able to reproduce the non-interacting ground state. A possible way to fix this issue could be to vary σ with the SGD method as we did for the parameters a_i, b_i, W_{ij}.

The best result was found using the Metropolis sampler. Importance sampling is thought of as a correction to Metropolis sampling. Therefore we would have expected to get better results with Importance sampling. The only improvement we got was a lower error. From Fig. (10) we know that the convergence of the local energy changes with η. It might be that we found an η that gave a good result for Metropolis but not for Importance sampling. There might then be some other η that when applied to the Importance sampler would give a better ground state upper bound.

References

[1] M. Taut, "Two electrons in an external oscillator potential: Particular analytic solutions of a Coulomb correlation problem," *Physical Review A*, vol. 48, no. 5, pp. 3561–3566, Nov. 1993.

[2] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks," *Science*, vol. 355, no. 6325, pp. 602–606, Feb. 2017.

[3] M. H. Jensen, *Computational Physics*, 2015.

[4] M. Jonsson, "Lecture notes for blocking method," March 2018.