Application of restricted Boltzmann machines in variational Monte Carlo simulations.

Kaspara Skovli Gåsvær & Peder Lon Hauge*
University of Oslo - Department of Physics

(Dated: June 6, 2021)

At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident, similique sunt in culpa qui officia deserunt mollitia animi, id est laborum et dolorum fuga. Et harum quidem rerum facilis est et expedita distinctio. Nam libero tempore, cum soluta nobis est eligendi optio cumque nihil impedit quo minus id quod maxime placeat facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates repudiandae sint et molestiae non recusandae.

I. INTRODUCTION

The application of machine learning allows for new approaches in solving quantum mechanical problems. Calculating analytical ground-state energies for interacting multi-particle systems is usually not possible, so approximate methods such as Variational Monte Carlo (VMC) is needed.

When constructing a trial wave function $\Psi_T(\mathbf{r};\boldsymbol{\alpha})$, the challenge is to find a suitable set of variational parameters $\boldsymbol{\alpha}$. Using the framework given by machine learning, we are able to optimize trial wave functions with many more variational parameters. This increase in available degrees of freedom reduces the dependency of Ψ_T needing to resemble the true solution Ψ at the outset.

In this report, we will represent the trial wave function Ψ_T using a restricted Boltzmann machine (RBM). The resulting wave function is called a neural network quantum state (NQS). This method dates back to 2017, when Carleo and Grover used a RBM for studying lattice spin systems with the Ising and Heisenberg model [1]. We will apply the same method for finding the ground state energy of a fermionic system, consisting of one or two interacting electrons trapped in a harmonic oscillator potential.

In the pursuit of better numerical results, we will compare different sampling rules and optimization algorithms. We investigate both the symmetric random walk (Metropolis) and importance sampling (Metropolis-Hastings), and the methods known as gradient descent and ADAM.

Due to the anti-symmetry of fermionic wave functions, we are not able to generalize our results to higher numbers of particles. However, we can still benchmark our results to analytical values calculated by Taut [2, 3]. With further development, our approach can be used to study systems of more particles.

This report can be viewed as a continuation of earlier work on variational Monte Carlo methods [4]. Much of both the theory and implementation of methods is identical, so for the sake of brevity, we will refer to Ref. [4] for further details when possible.

II. THEORY

A. Sampling of local energy

Given a trial wave function $\Psi_T(\mathbf{r};\boldsymbol{\alpha})$ and a Hamiltonian H, the variational principle guarantees that the calculated expectance value $\langle H\rangle$ is higher than the real ground-state energy E_{gs} [4]. Thus, the best estimate of E_{gs} (for a given Ψ_T) is found by adjusting the variational parameters $\boldsymbol{\alpha}$ so that $\langle H\rangle$ is minimized.

In addition, the calculated energy variance σ^2_{std} can be used to determine the quality of the results, as $\sigma^2_{\text{std}} = 0$ if Ψ_T is equal to the true solution Ψ [4].

The expectation value $\langle H\rangle$ is given by

$$\langle H\rangle = \frac{\int d\mathbf{r}\Psi_T^* H\Psi_T}{\int d\mathbf{r}|\Psi_T|^2} = \int d\mathbf{r}\, P(\mathbf{r})E_L(\mathbf{r}), \tag{1}$$

where we in the last step define the local energy

$$E_L(\mathbf{r}) \equiv \frac{1}{\Psi_T(\mathbf{r})}H(\mathbf{r})\Psi_T(\mathbf{r}), \tag{2}$$

and the probability density function

$$P(\mathbf{r}) \equiv \frac{|\Psi_T(\mathbf{r})|^2}{\int d\mathbf{r}\,|\Psi_T(\mathbf{r})|^2}. \tag{3}$$

This re-write allows us to calculate $\langle H\rangle$ by sampling E_L during the Monte Carlo simulations. It can readily be shown (Ref. [4]) that $\langle E_L\rangle = \langle H\rangle$.

B. Hamiltonian of system

The system we consider consists of electrons confined in an isotropic harmonic oscillator potential. The Hamiltonian for P particles is given by

[*] Code repository: https://github.com/pederlh/FYS4411

$$H = \underbrace{\sum_{p=1}^{P}\left[-\frac{1}{2}\nabla_p^2 + \frac{1}{2}\omega^2 r_p^2\right]}_{H_0} + \underbrace{\sum_{p<q}\frac{1}{r_{pq}}}_{H_1}, \tag{4}$$

where H_0 denotes the standard harmonic oscillator part, and H_1 the repulsive electrostatic potential between two electrons. Since the Hamiltonian uses natural units ($\hbar = e = m_e = 1$), the energies are given in units of a.u. (atomic units). Furthermore, N is the number of electrons (either 1 or 2), ω the oscillator frequency and the distance between electrons is given by $r_{pq} = |\mathbf{r}_p - \mathbf{r}_q|$.

In this report we will compare results for the interactive case with the analytical ground state energies found by Taut for certain frequencies ω [2, 3]. For instance, when the Hamiltonian includes the term H_1, this energy is given by 3 a.u. (in two spatial dimensions), compared to 2 a.u. for non-interacting particles.

On the other hand, if only consider the term H_0 the particles can be viewed as non-interacting bosons. In this case, the ground state Ψ_0 has an analytical expression

$$\Psi_0 = A\prod_{p=1}^{P}\exp\left[-\frac{\omega r_p^2}{2}\right] \tag{5}$$

where A is a normalization factor. Finding analytic ground-state energy E_0 for Ψ_0 can be done in the same way as in Ref. [4]. For P particles in D spatial dimensions, we arrive at

$$E_0 = \frac{\omega P D}{2}\text{ a.u.} \tag{6}$$

C. Introducing the restricted Boltzmann machine

Before we can express Ψ_T in terms of a restricted Boltzmann machine, we need to cover the basics behind RBMs.

A restricted Boltzmann machines is a two-layer net with one layer of visible nodes (which results are read from) and another layer of hidden nodes. The M visible nodes are represented by a vector $\mathbf{x}$ (of length M), and similarly, a vector $\mathbf{h}$ stores the N hidden nodes. The nodes in $\mathbf{x}$ and $\mathbf{h}$ have biases (single-node weights) $\mathbf{a}$ and $\mathbf{b}$ respectively.

Furthermore, a matrix $\mathbf{W}$ contains the weights of the connections between the visible and hidden nodes. The term "restricted" stems from the fact that there are no connections between nodes within a single layer. An illustration showing the structure of the RBM is shown in Figure 1.

The RBM is a so-called generative network, whose goal is to learn a probability distribution (as opposed to the model producing an output directly). Ultimately, we wish to encode the particle positions $\mathbf{r}$ of the trial wave function $\Psi_T(\mathbf{r};\boldsymbol{\alpha})$ into the visible layer $\mathbf{x}$, and use Ψ_T in order to calculate the local energy E_L.

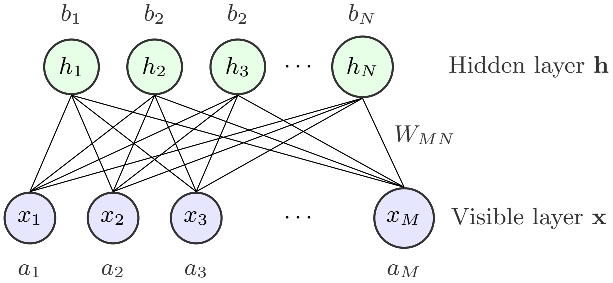

FIG. 1. Structure of a restricted Boltzmann machine. The two layers of nodes are shown with their weighted connections and biases.

Since the RBM is not working with training data, the network falls under the category of unsupervised/reinforcement learning. By the variational principle, the optimal result is found when the system's energy is minimized. This information guides the optimization ("learning") of the network's weights and biases.

D. Joint distribution and types of RBMs

The RBM model is described by a joint distribution of $\mathbf{x}$ and $\mathbf{h}$, the Boltzmann distribution defined as

$$F_{\text{RBM}}(\mathbf{x},\mathbf{h}) = \frac{1}{Z}e^{-\frac{1}{T_0}E(\mathbf{x},\mathbf{h})}. \tag{7}$$

The quantity T_0 is usually ignored by setting it to 1, and Z is the partition function (normalization constant) defined as

$$Z = \iint e^{-\frac{1}{T_0}E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}. \tag{8}$$

In the above equations, $E(\mathbf{x},\mathbf{h})$ is known as the energy of the configuration of the nodes (not the system itself), and it is a function that describes the relation between the hidden and visible nodes. A lower value of E translates to a higher configuration probability. RBMs can implement this energy function in different ways, and we chose to implement what's called a Gaussian-binary RBM.

In a Gaussian-binary RBM, the visible nodes take on continuous values (in order to model particle positions), while the hidden nodes only can have binary values. The energy function is then given by

$$E(\mathbf{x},\mathbf{h}) = \sum_{i=1}^{M}\frac{(x_i - a_i)^2}{2\sigma_i^2} - \sum_{j=1}^{N} b_j h_j - \sum_{i,j=1}^{M,N}\frac{x_i W_{ij} h_j}{\sigma_i^2}. \tag{9}$$

Letting $\sigma_i = \sigma$ for all visible nodes, this can also be stated as

$$E(\mathbf{x},\mathbf{h}) = \frac{|\mathbf{x}-\mathbf{a}|^2}{2\sigma^2} - \mathbf{b}^T\mathbf{h} - \frac{\mathbf{x}^T\mathbf{W}\mathbf{h}}{\sigma^2}.$$

We can now introduce the marginal distribution of $\mathbf{x}$

$$F_{\mathrm{RBM}}(\mathbf{x}) = \sum_{\{\mathbf{h}\}} F_{\mathrm{RBM}}(\mathbf{x},\mathbf{h}) = \frac{1}{Z}\sum_{\{\mathbf{h}\}} e^{-E(\mathbf{x},\mathbf{h})}, \tag{10}$$

which we will use to represent the trial wave function Ψ_T as a neural network quantum state (NQS). By inserting $E(\mathbf{x},\mathbf{h})$ from Eq. (9) into Eq. (10), we arrive at the following expression for the NQS:

$$\begin{aligned}\Psi_T(\mathbf{r};\boldsymbol{\alpha}) &= F_{\mathrm{RBM}}(\mathbf{x})\\ &= \frac{1}{Z} e^{-\sum_{i=1}^M \frac{(x_i-a_i)^2}{2\sigma^2}} \prod_{j=1}^N \left(1 + e^{b_j + \sum_{i=1}^M \frac{x_i W_{ij}}{\sigma^2}}\right).\end{aligned} \tag{11}$$

This above (spacial) wave function is symmetric under interchange of particles (by swapping corresponding sets of weights/biases), so for a two-electron system the spin state χ has to be anti-symmetric. This is the so-called singlet state

$$\chi = \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle)$$

with a total spin of 0.

More details about marginal and conditional probabilities of the Gaussian-binary RBM (i.e. how to find values of the hidden nodes $\mathbf{h}$) is described in Appendix A.

E. RBM: Cost function and training

The training of the RBM consists of tuning the weights such that a cost function is minimized. In our case it is natural to use the local energy E_L from Eq. (2) as a cost function, as we know that the best estimate of the ground state energy E_g is found by minimizing $\langle E_L\rangle$.

We now seek to optimize the RBM's weights θ. Since $\Psi_T(\mathbf{r};\boldsymbol{\alpha}) \in \mathbb{R}$, we can use the expression for the partial derivative $\frac{\partial\langle E_L\rangle}{\partial\theta}$ given in Appendix D of Ref. [4]:

$$\begin{aligned}\nabla_\theta E_L &\equiv \frac{\partial\langle E_L\rangle}{\partial\theta}\\ &= 2\left(\left\langle E_L \frac{1}{\Psi_T}\frac{\partial\Psi_T}{\partial\theta}\right\rangle - \langle E_L\rangle\left\langle \frac{1}{\Psi_T}\frac{\partial\Psi_T}{\partial\theta}\right\rangle\right)\end{aligned}$$

where $\theta \in \{a_1,\ldots,a_M,b_1,\ldots,b_N,W_{11},\ldots,W_{MN}\}$. This above equation can be further simplified to

$$\nabla_\theta E_L = 2\left(\left\langle E_L \frac{\partial\ln\Psi_T}{\partial\theta}\right\rangle - \langle E_L\rangle\left\langle \frac{\partial\ln\Psi_T}{\partial\theta}\right\rangle\right). \tag{12}$$

From Eq. (11) we find that

$$\begin{aligned}\ln\Psi_T(\mathbf{x}) = &-\frac{1}{2}\ln Z - \sum_{i=1}^M \frac{(x_i-a_i)^2}{4\sigma^2}\\ &+ \frac{1}{2}\sum_{j=1}^N \ln\left(1 + e^{b_j + \sum_{i=1}^M \frac{x_i W_{ij}}{\sigma^2}}\right),\end{aligned}$$

which leads to partial derivatives

$$\frac{\partial}{\partial a_i}\ln\Psi_T = \frac{1}{2\sigma^2}(x_i - a_i) \tag{13}$$

$$\frac{\partial}{\partial b_j}\ln\Psi_T = \frac{1}{2\left(e^{-b_j - \frac{1}{\sigma^2}\sum_{i=1}^M x_i W_{ij}} + 1\right)} \tag{14}$$

$$\frac{\partial}{\partial W_{ij}}\ln\Psi_T = \frac{x_i}{\sigma^2\left(e^{-b_j - \frac{1}{\sigma^2}\sum_{i'=1}^M x_i W_{i'j}} + 1\right)}. \tag{15}$$

As well as finding how $\langle E_L\rangle$ changes with the weights θ, we also want an efficient way of calculating E_L given a set of visible nodes $\mathbf{x}$. In Appendix B we show that if every visible node in the RBM represents one coordinate of a single particle, the local energy is

$$E_L = \frac{1}{2}\sum_{i=1}^M\left[-\left(\frac{\partial\ln\Psi_T}{\partial x_i}\right)^2 - \frac{\partial\ln\Psi_T}{\partial x_i^2} + \omega^2 x_i^2\right] + \sum_{p<q}\frac{1}{r_{pq}}, \tag{16}$$

with partial derivatives

$$\begin{aligned}\frac{\partial}{\partial x_i}\ln\Psi_T = &-\frac{1}{\sigma^2}(x_i - a_i)\\ &+ \frac{1}{\sigma^2}\sum_{j=1}^N \frac{W_{ij}}{e^{-b_j - \frac{1}{\sigma^2}\sum_{i'=1}^M x_{i'} W_{i'j}} + 1}\end{aligned} \tag{17}$$

$$\frac{\partial^2}{\partial x_i^2}\ln\Psi_T = -\frac{1}{\sigma^2} + \frac{1}{\sigma^4}\sum_{j=1}^N \frac{W_{ij}^2 \cdot e^{-b_j - \frac{1}{\sigma^2}\sum_{i'=1}^M x_{i'} W_{i'j}}}{\left(e^{-b_j - \frac{1}{\sigma^2}\sum_{i'=1}^M x_{i'} W_{i'j}} + 1\right)^2}. \tag{18}$$

III. METHOD

A. Monte Carlo sampling

Markov chain Monte Carlo methods provides a way of obtaining a sequence of samples from a probability distribution (like $P(\mathbf{r})$ from Eq. (3)). Concepts central to Markov Chain theory are ergodicity and detailed balance, which are explained in Ref. [4].

The acceptance rate $A(i \to j)$ of moving a random walker at position $\mathbf{r}_i$ to $\mathbf{r}_j$ can be decided by different sampling rules, and in this report we use two such algorithms. The first one is the plain (brute-force) Metropolis algorithm, which proposes moves using a uniform distribution, such that they are equally likely. This results in a rather simple acceptance rate.

Another way of performing the sampling process is by importance sampling, which uses the more general Metropolis-Hastings algorithm. In this case the random walkers are guided by the underlying probability distribution, which results in an overall higher acceptance rate. We will below present the equations being implemented for both cases, and we refer to Ref. [4] for more details.

For the brute-force Metropolis method, we define a step size h and a vector $\mathbf{g}$ with components drawn from a uniform distribution on the interval $[-1,1]$. A proposed move is then

$$\mathbf{r}_j = \mathbf{r}_i + h\mathbf{g}, \tag{19}$$

and the probability of it being accepted is

$$A(j \to i) = \min\left(1, \frac{|\Psi_T(\mathbf{r}_i)|^2}{|\Psi_T(\mathbf{r}_j)|^2}\right). \tag{20}$$

Importance sampling is slightly more complex to implement. We first define a quantity

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

called the quantum force (or drift term). The proposed moves are given by

$$\mathbf{r}_j = \mathbf{r}_i + D\mathbf{F}(\mathbf{r}_i)\Delta t + \boldsymbol{\chi}\sqrt{\Delta t}, \tag{21}$$

where we define the diffusion coefficient $D = \frac{1}{2}$ and time step Δt. In addition, $\boldsymbol{\chi}$ is a Gaussian random variable centered at zero, usually chosen to have standard deviation 1.

The acceptance probability for importance sampling is

$$A(j \to i) = \min\left(1, \frac{G(\mathbf{r}_j, \mathbf{r}_i, \Delta t)}{G(\mathbf{r}_i, \mathbf{r}_j, \Delta t)} \frac{|\Psi_T(\mathbf{r}_i)|^2}{|\Psi_T(\mathbf{r}_j)|^2}\right), \tag{22}$$

with the above equation including the Green's function

$$\begin{aligned} G(\mathbf{r}_j, \mathbf{r}_i, \Delta t) = &\frac{1}{(4\pi D\Delta t)^{3N/2}} \\ &\times \exp\left[\frac{-(\mathbf{r}_j - \mathbf{r}_i - D\Delta t\mathbf{F}(\mathbf{r}_i))^2}{4\pi D\Delta t}\right]. \end{aligned} \tag{23}$$

B. Gradient Descent

In order to update the weights and biases of the RBM such that they minimize the cost function $\langle E_L \rangle$, we need to implement a method for performing gradient descent. In this report we will focus on two of the many methods that exist: regular vanilla gradient descent and ADAM.

In gradient descent (GD), the parameters are initialized to some value θ_0, and then iteratively updated in the direction of the steepest descent through the updating scheme

$$\begin{aligned} v_t &= \eta_t \nabla_\theta E_L(\theta_t), \\ \theta_{t+1} &= \theta_t - v_t. \end{aligned} \tag{24}$$

Here $\nabla_\theta E_L(\theta_t)$ is the gradient of $\langle E_L \rangle$ with respect to the parameters θ at iteration step t, and η_t is the learning rate that's the step size in the direction of the gradient. Finding the perfect learning rate can be tricky. Choosing a small value for η_t means that the parameters "move" very slowly, that's to say that there is little change in each iteration. This means that the number of iterations spent reaching convergence grows large and the algorithm becomes computationally expensive. On the other hand, by choosing a large value of η_t we risk overshooting the minimum, thus never reaching convergence. Usually, a good starting point is a value between $0.01 - 0.001$ but the golden value for a given problem has to be found by either by a grid search or trial and error.

1. *Adding momentum*

A modification which deals with the constant learning rate η_t is to add a momentum term in the updating scheme, so that

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta_t \nabla_\theta E_L(\theta_t), \\ \theta_{t+1} &= \theta_t - v_t. \end{aligned} \tag{25}$$

The learning rate is still constant per se, but the momentum term serves as a memory of the previous time step. A fraction $\gamma \in (0,1)$ is added to the update vector v_{t-1} from the previous iteration step; this is then combined with the current gradient in the update vector v_t.

An interpretation of this is that we find a weighted mean of recent gradients, which means the GD algorithm will move in the direction of small and persistent gradients, while suppressing the effect of sudden oscillations. This will in many cases increase the convergence rate. However, the algorithm can in some cases overshoot the local minimum for some choices of γ and η_t, which makes the fitting algorithm oscillate about a minima, leading to a slower convergence rate.

2. *ADAM: Adding the second moment*

Another way to optimize GD is to additionally incorporate the second moment of the gradient. The resulting updating scheme is an algorithm known as ADAM:

$$\begin{aligned} g_t &= \nabla_\theta E_L(\theta_t), \\ m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t, \\ s_t &= \beta_2 s_{t-1} + (1-\beta_2) g_t^2, \\ \alpha_t &= \eta_t \frac{\sqrt{1-(\beta_2)^t}}{1-(\beta_1)^t}, \\ \epsilon_t &= \epsilon\sqrt{1-(\beta_2)^t}, \\ \theta_{t+1} &= \theta_t - \alpha_t \frac{m_t}{\sqrt{s_t} + \epsilon_t}. \end{aligned} \tag{26}$$

The quantity m_t is the average first moment of the gradient, s_t is the average second moment of the gradient, α_t the scaled learning rate at iteration step t, and ϵ_t is a parameter to avoid division by zero. The factors β_1 and β_2 controls the decay rates of the averages and are typically set to values just below 1.0 [5]. In few words one can think of ADAM as a combination of momentum and a adaptive learning rate, with more efficiently updated weights leading to faster convergence.

C. Other implementational details

Due to the samples being heavily correlated with each other, special care has to be taken when calculating the standard error in the measurements. In order to provide a better statistical analysis, we have implemented the blocking algorithm described in Ref. [4].

The harmonic oscillator system bears close resemblance to the system in described Ref. [4], so we chose to reuse the parameters for $h = 1.0$ and $\Delta t = 0.005$. These are the step sizes in Eqs. (19) and (21). While the parameter h was found by trial and error, a more detailed treatment of finding an optimal value of Δt is given in Ref. [4].

When performing gradient descent with ADAM (Eq.(26)), we adopted the decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and the smoothing term ϵ was set to be $\epsilon = 10^{-8}$. For both the regular gradient descent and ADAM we chose the convergence criteria that the sum of weight changes (absolute value for **a**, **b** and **W** separately) should be $\leq 9 \cdot 10^{-3}$.

The energy function $E(\mathbf{x}, \mathbf{h})$ (Eq. (9)) defining the RBM does not contain any information about the physical system, so we will investigate how beneficial it is to adjust parameters of the model so that it more closely resembles the ground state of a non-interacting system. Assuming that most weights and biases are small, we see that if they are all set to zero, the trial wave function in Eq. (11) goes as

$$\Psi_T(\mathbf{r}; \boldsymbol{\alpha}) \propto \prod_{i=1}^{M} \exp\left[-\frac{x_i^2}{2\sigma^2}\right].$$

Comparing the above expression with the analytical non-interacting solution in Eq. (5), we see that instead of simply setting $\sigma = 1$ for all frequencies ω, it may be advantageous to select $\sigma = 1/\sqrt{\omega}$ for better upper bounds for $\langle E_L \rangle$.

Unless otherwise noted, all results are found running the Monte Carlo simulation for 2^{20} cycles.

IV. RESULTS

A. Variable RBM parameter σ

We begin by comparing the choice of a constant $\sigma = 1$ to the more adaptive $\sigma = 1/\sqrt{\omega}$. As shown in Fig. 2, setting $\sigma = 1/\sqrt{\omega}$ leads to results much closer to the analytical ones, and we will use this choice of σ onward. This calculation was done with a RBM with two hidden nodes. A plot of the relative errors are shown in Fig. 3, and it shows that for all $\sigma \neq 1$, the better value of σ typically gives a smaller relative error on 2-4 orders of magnitude.

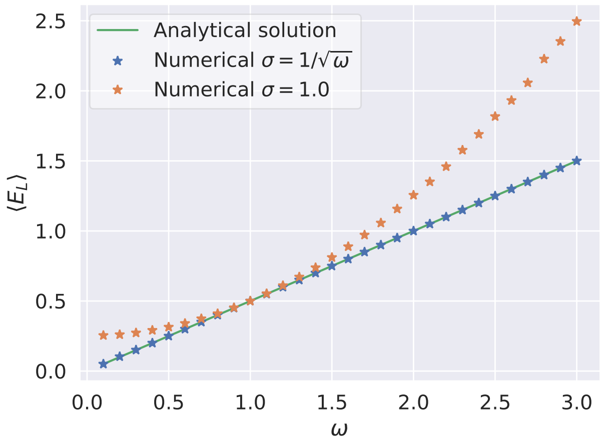

FIG. 2. $\langle E_L \rangle$ vs. ω for $\sigma = 1.0$ and $\sigma = 1/\sqrt{\omega}$. The system is a single particle in 1D, with the analytical solution is shown as a straight line. The RBM used had 2 hidden nodes and the simulation was.

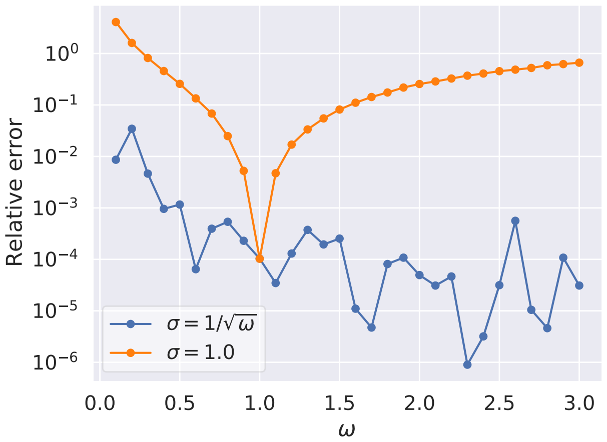

FIG. 3. The relative errors of the data in the $\langle E_L \rangle$ vs. ω plot (Fig. 2).

B. Number of hidden nodes

Next, we investigate how the number of hidden nodes N effects the result. In Fig. 4 we see how both $\langle E_L \rangle$ and the number of gradient descent iterations before convergence varies with N. In the gradient descent process, a learning rate of $\eta = 0.1$ is used. From the graphs we determined that having two hidden nodes was sufficient, and we kept $N = 2$ for the subsequent simulations.

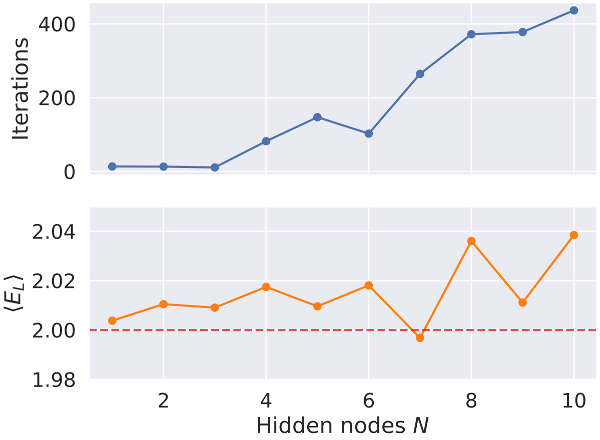

FIG. 4. $\langle E_L \rangle$ and the number of iterations before gradient descent convergence for different numbers of hidden nodes N. The dashed line marks the analytical ground state energy for two non-interacting particles in 2D.

C. Patching the interacting case program

When we started to study a system of interacting particles, we quickly ran into the problem of the RBM accepting particle positions extremely close to each other, leading to high values of the local energy E_L. This occurred frequently enough to significantly shift the final results of $\langle E_L \rangle$.

We sought out two possible ways to alleviate this problem. Since we are mostly interested in implementing importance sampling, one solution is to increase the variance in the distribution χ (Eq. (21)) so that the particles' proposed moves generally are more spread out. We tested a ten-fold increase of the variance from 1 to 10.

Another solution is to simply ignore the interaction term H_1 of the Hamiltonian (Eq. (4)) so that the Coulomb repulsion don't contribute to the local energy for small distances. The criteria we chose for this test was an inter-particle distance of $r \leq 5 \cdot 10^{-2}$.

Having defined the possible fixes, we tested both for 11 logarithmically spaced values of Monte Carlo cycles on the interval $[2^{10}, 2^{20}]$. From the results presented in Fig. 5, we chose to proceed the fix that ignores H_1 for close particles.

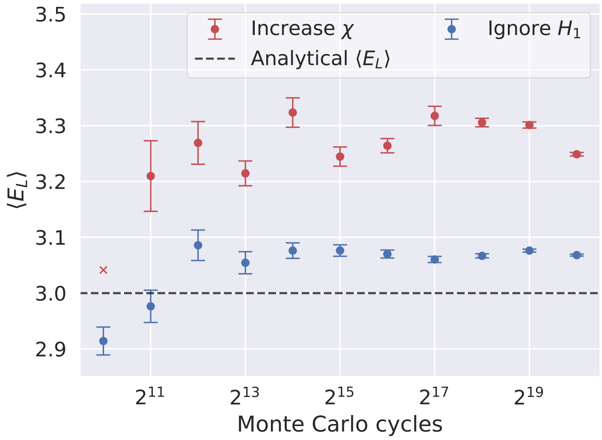

FIG. 5. Test of ways to prevent interacting particles of getting too close. One is by increasing the variance of χ to $\chi = \mathcal{N}(0, 10)$, the other ignoring the interaction term H_1 when the inter-particle distance was less than $r \leq 5 \cdot 10^{-2}$. A cross × signifies that the gradient descent didn't converge in 100 iteration, and the energy given is the final mean calculated. The system has 2 particles in three dimensions and $\omega = 1$.

D. Comparing sampling rules

Now that we were able to produce results for the interacting case with two electrons, we compared the Metropolis (symmetric random walkers) and Metropolis-Hastings (importance sampling) algorithms. This was done in two spatial dimensions, and compared to analytical results by Taut [2] for various ω. The resulting plot is shown in Fig. 6.

By inspection, the calculated energies deviates by approximately the same amount for the two sampling rules. However, since the Metropolis algorithm results generally falls below the true ground state energy (in violation of the variational principle), we decided to only consider Metropolis-Hastings sampling going on.

E. Comparing optimization algorithms

Another test we were interested in performing was a comparison of gradient descent and ADAM for learning rates $\eta \in [0.01, 0.10]$. The system studied was two interacting electrons in 2D, a system with an analytical ground state energy of 2 a.u. Figs. 7 and 8 show how the two methods compares with respect to final values of $\langle E_L \rangle$ and number of iterations before convergence. The criteria set was to be a max deviation of 10^{-4} a.u. from the analytical value.

Iteration-wise, gradient descent performed significantly worse than ADAM for low learning rates η. Since ADAM consistently needed ~ 25 iterations to converge and the $\langle E_L \rangle$ results were of similar quality, we deemed

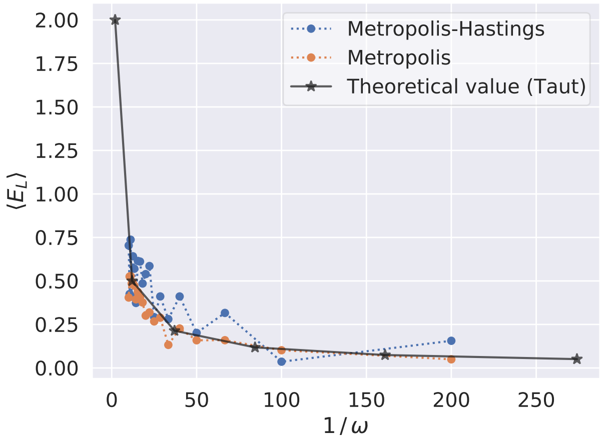

FIG. 6. $\langle E_L \rangle$ calculated with the symmetric Metropolis and Metropolis-Hastings algorithms for various $\omega \in [0.005, 0.1]$. The analytical energies for the two electrons in 2D is given by Taut [2].

ADAM to be the better optimization algorithm. Unable to easily pick a superior value of η from the figures, we went on picking $\eta = 0.035$.

F. A final test

Finally, we used all the selected methods and parameters to make a last comparison between Taut's solutions for an interacting two-electron system in three dimension, and numerical simulations using the RBM. These results are presented in Fig. 9.

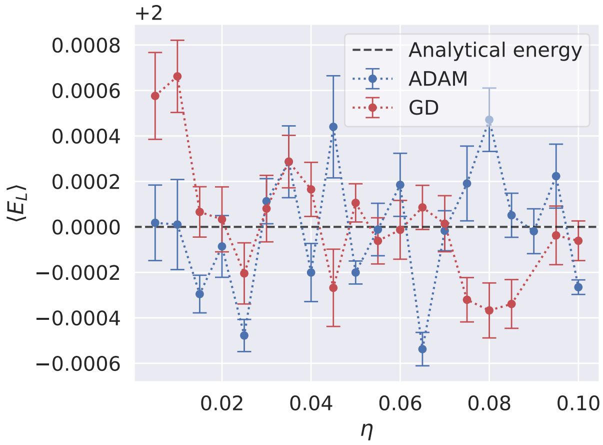

FIG. 7. Comparison of $\langle E_L \rangle$ when using gradient descent and ADAM for values of $\eta \in [0.01, 0.10]$. The dashed line indicates the analytical solution for two interacting particles in 2D (with $\omega = 1$). The missing data point for GD at $\eta = 0.090$ was due to non-convergence of the learning process.

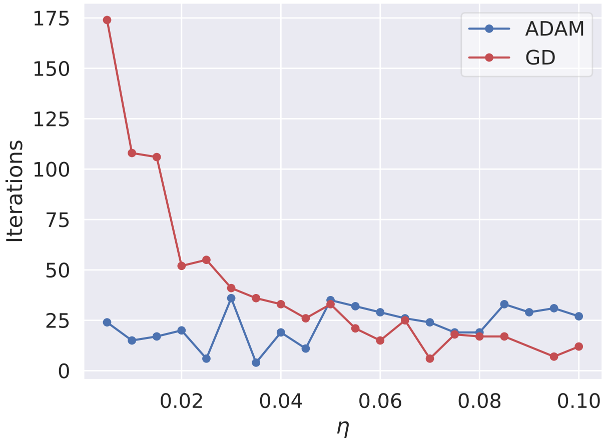

FIG. 8. Comparison of number of iterations needed to satisfy gradient descent and ADAM criteria (max deviation 10^{-4} a.u. from analytic solution of 2 a.u.) for values of $\eta \in [0.01, 0.10]$. The system was a two-particle interacting system in 2D. The missing data point for GD at $\eta = 0.090$ was due to non-convergence of the learning process.

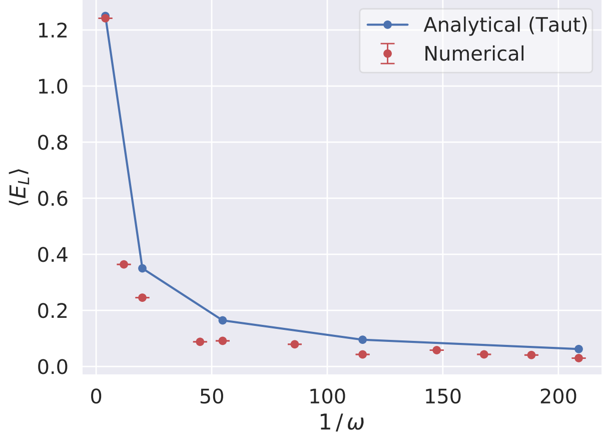

FIG. 9. Numerical results of $\langle E_L \rangle$ compared to analytical results from Taut [3] for $\omega^{-1} \in [4, 211.81]$, the system being two interacting electrons in 3D.

V. DISCUSSION

A. Letting the RBM mimic the analytical solution

From Figures 2 and 3 it is clear that using $\sigma = 1/\sqrt{\omega}$ greatly helps our solution. The RBM itself doesn't know anything about the nature of the true wave function Ψ, so our adjustment initiates a better RBM that more closely resembles the analytical solution for non-interacting particles. We also observe that the solutions are equal for $\omega = 1$, which is reassuring.

B. Choosing the number of hidden nodes

Fig. 4 is clear about there being nothing to gain by having many hidden nodes. First of all, the Monte Carlo simulation slows down as the dimensions of the RBM increases. Moreover, the fact that more iterations are needed before reaching convergence is likely due to the fact that there are more individual weights to be trained. The precision in our estimate of $\langle E_L \rangle$ does not improve but rather tends to get worse when increasing the number of nodes. We could have experimented with using only a single hidden node, but for the sake of having more degrees of freedom we opted to use two hidden nodes instead.

C. Fixing the problem of particles getting a bit too friendly

From Fig. 5 we observe that ignoring the interaction term of the Hamiltonian (when particles got too close) provided somewhat better accuracy than increasing the variance of the Gaussian distribution used to propose new particle moves. The distribution change always lead to the solutions always converging at higher energy minima.

We did try other scaling constants for the distribution $(2, 3, 5, 10)$, but there were no noticeable difference between the scaling, and we have thus only presented the results for 10. The variance in the set of samples provided by both methods gets smaller as the number of cycles increase, which tells us that it is likely that the RBM in fact have been taught to represent the wrong wave function. The blocking method reveals that it is extraordinary unlikely that we have ended up with the correct NQS, but with an unfortunate number of high-energy samples throwing the results off.

For the case where we ignore the interaction term, we converge to a local energy value slightly closer to the analytical one, with more even energies over different numbers of MC cycles. A very probable reason for the method not being more accurate is the fact that the limit of what we deemed "too close" was found by trial and error. This means that we add personal bias when there in reality exists information which preferably should be encoded in the trial wave function/sampling method itself.

We observed that when changing the cut-off distance for the interaction H_1, the accuracy of the solution was highly dependent on the choice of tolerance. Just changing it from $5 \cdot 10^{-2}$ to $6.5 \cdot 10^{-2}$ greatly improved the accuracy of our results. With this in mind, we emphasis that none of these choices led to solutions even close to perfect, and that further research on the matter should focus more on implementing a more physical solution to solving the Coulomb repulsion problem.

We counted the rate of interaction being suppressed, and found that it happened in roughly $10 - 12\%$ of the total energy samples. This is quite a lot, but we suspect that this stems from the Coulomb force only being an additional term in the Hamiltonian, and not encoded in any way in the NQS. Many moves not appropriate with regards to energy minimization are proposed and accepted (even when using Metropolis-Hastings sampling), even when not physically likely due to the fact that real interacting fermions don't like to be close in space[1]. In summary, a lot of physically unlikely moves are prevented from being proposed and accepted, and so the "large" percentage of rejected interaction contributions might not be as bad as at first glance.

As a final note we believe that it is highly likely that the reason for the calculated $\langle E_L \rangle$ being lower than Taut's values in Fig. 9, is due to the way we opted to solve the problem of too close particles. It is not immediately evident that the tolerance set for particles in two dimensions (found from trial and error) should properly scale to 3D.

D. Importance sampling vs. symmetric random walk

Fig 6. Ligger litt spredt om Taut, men i samme ballpark hver sin vei. Rart at vanlig metropolis ligger under taut-verdier, som ikke burde gå an mtp variasjonsprinsippet. Kunne simulert for flere omega, men vi ser generell trend.

E. ADAM vs. vanilla gradient descent

From Fig. 7 one sees that there is little difference in terms of precision in $\langle E_L \rangle$ between the two optimization methods. Both are a bit off, and as the error bars produced found by the blocking method mostly don't contain the true ground state energy, it is probable that both converge to two minima close to the true one, but not spot on. The low standard error in the local energy tells us that the numerical measurement is "precise but not accurate".

The above paragraph does not give us a clear indication of what the best method is, but we do see that regular GD doesn't always converge at all. For higher values of η we observed that the regular GD method regularly got trapped at local minima that had a lower expectation value for the local energy than the true solution [2]. ADAM never got trapped which is most likely due to its pseudo-adaptive learning rate and the fact that it remembers how fast on average the parameters changed in the past. This helps it to "identify" and move out of such local minima.

[1] Amazing opportunity for drawing parallels to Norwegians here, but lets keep it professional.

[2] As we explained both methods often converged at slightly off values, but they were in the space confined by the tolerance we set for convergence. Regular GD with high learning rates often got trapped at values outside of that space and by our definition did not converge.

Observing figure 8 we see that the number of iterations needed by ADAM to reach convergence does not change much for different choices of learning rate. Again, this is probably a consequence of the fact that ADAM is less dependent of the learning rate as it is pseudo-adaptive. Vanilla GD is on the other hand very dependent on learning rate as we observe that small values of η require many more iterations to converge. This is not as surprise as the learning rate tells the method explicitly how fast or slow it should change its parameters in the direction set by the gradient.

With this in mind we conclude that ADAM is, as expected, a more suited method for problems with more than one adjustable parameter and noisy problems with more than one minimum.

As a contrast to the last project where the problem was the precision in the dataset (aka the magnitude of the errorbars) and not so much the accuracy of the mean $\langle E_L \rangle$, in this project the RBM was very certain that it had found the correct minimum and so the problem was not precision but accuracy in the set of samples.

F. Final thoughts

- Snakke om at vi opplevde god presisjon med en gang dersom vi hadde vedlig små vekter (løsningen går mot analytisk), men da lærte ikke maskinen. Dersom vi initialiserte større vekter ble maskinen smartere, men klarer ikke nå samme presisjon. Brukt: 0.1 –¿ 0.0001 enda bedre??
- Her har vi testet et par ting om gangen, men grid search gir bedre resultater. mer tidkrevende tho, og er veldig mange parametre å optimalisere!
- Nå var psi kun RBMen alene, kunne ha lagt til faktorerer eller ledd som hadde med vekselvirking å gjøre. Den lille fiksen med omega hjelp mye.
- Utvidelser: Gibbs-sampling?
- UTvidelser: Andre nevrale nettverk kan være bedre eller andre typer RBM enn gauss-binær
- Utvide til 3D. Trenger å konstuere energifunksjonen E slik at den totale bølgefunsjonen er anti-symmetrisk mtp på ombytte av partikler, inkludert spins-del.

VI. CONCLUSION

In this report we applied a Gaussian-Binary RBM to the problem of two interacting bosons confined to a harmonic oscillator trap. We used Monte-Carlo simulations together with Metropolis-Hastings sampling to estimate the expectation value of the local energy. We implemented methods for performing gradient descent on the variational parameters of the NQS wave function and found that ADAM was the preferable choice using fewer iterations on average for all choices of learning rates (but especially smaller ones) before reaching convergence and being less likely to get trapped in local minima. We discovered a problem concerning the proposal and acceptance of non-advantageous moves which we partially improved by ignoring the additional term added to local energy from interaction when particles got to close. This was not a perfect fix, but the problem pointed out important short-comings of the entire method itself. The RBM is not as good a method as incorporating the Jastrow factor when it comes to simulating particles trapped in a HO-potential. It lacks crucial information about the system which, at least with our methods, it did not sufficiently learn. We managed to produce results in the ballpark of those produced by Taut, but not as close as we imagine we could have using other methods.

- ADAM er en bedre metode for gradient descent når man har flere parametere.
- RBM er ikke en like god metode som Jastrow
- Er et relativt enkelt møte med unsupervised ML tho, kanskje fint at det viser noen av ”fall-lemmene” og shortcomings ved ML.
- Kunne man kombinert RBM med tradisjonelle løsningsmetoder for å få en enda bedre måte å løse slike systemer på?

[1] Giuseppe Carleo and Matthias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325):602–606, 2017.

[2] M Taut. Two electrons in a homogeneous magnetic field: particular analytical solutions. *Journal of Physics A: Mathematical and General*, 27(3):1045–1055, feb 1994.

[3] M. Taut. Two electrons in an external oscillator potential: Particular analytic solutions of a Coulomb correlation problem. *Phys. Rev. A*, 48:3561–3566, Nov 1993.

[4] Peder Lon Hauge and Kaspara Skovli Gåsvær. Variational Monte Carlo studies of Bose gas in harmonic oscillator trap. https://github.com/pederlh/FYS4411/tree/main/Project1, 2021. Project work, University of Oslo.

[5] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre Day, Clint Richardson, Charles Fisher, and David Schwab. A high-bias, low-variance introduction to machine learning for physicists. *Physics Reports*, 810:16, 03 2018.

[6] Morten Hjort-Jensen. Advanced topics in computational physics: Computational quantum mechanics. https://compphysics.github.io/ComputationalPhysics2/

doc/LectureNotes/_build/html/intro.html, 2021. Lectures notes (Chapter “Deep learning, 5. Boltzmann Machines”).

Appendix A: Conditional probabilities of the RBM

In this section we present some more theory about the Gaussian-binary restricted Boltzmann machines. The marginal probability distributions $F_{\mathrm{RBM}}(\mathbf{x})$ and $F_{\mathrm{RBM}}(\mathbf{h})$ are used to calculate the conditional probabilities $F_{\mathrm{RBM}}(\mathbf{h}|\mathbf{x})$ and $F_{\mathrm{RBM}}(\mathbf{x}|\mathbf{h})$. This gives the probability of the hidden nodes having a specific configuration $\mathbf{h}$ given particle positions $\mathbf{x}$, and vice versa.

Note that this section only shows the main results, see Ref. [6] for a more complete derivation.

We begin by introducing the notation

$$\sum_{i=1}^{M} x_i W_{ij} = \mathbf{x}^T \mathbf{w}_{*j},$$

so that the marginal probability $F_{\mathrm{RBM}}(\mathbf{x})$ from Eq. (11) can be stated as

$$F_{\mathrm{RBM}}(\mathbf{x}) = \frac{1}{Z} e^{-\frac{|\mathbf{x}-\mathbf{a}|^2}{2\sigma^2}} \prod_{j=1}^{N} \left(1 + e^{b_j + \frac{\mathbf{x}^T \mathbf{w}_{*j}}{\sigma^2}}\right).$$

The marginal probability $F_{\mathrm{RBM}}(\mathbf{h})$ is then found to be

$$\begin{aligned} F_{\mathrm{RBM}}(\mathbf{h}) &= \int F_{\mathrm{RBM}}(\mathbf{x}, \mathbf{h}) \, \mathrm{d}\mathbf{x} \\ &= \frac{\sqrt{2\pi\sigma^2}}{Z} e^{\mathbf{b}^T \mathbf{h}} \prod_{i=1}^{M} e^{\frac{2a_i \mathbf{w}_{i*}^T \mathbf{h} + \left(\mathbf{w}_{i*}^T \mathbf{h}\right)^2}{2\sigma^2}}. \end{aligned}$$

We can now find the conditional probabilities. It turns out that the conditional probability for $\mathbf{h}$ can be written as a product of hidden single-node h_j probabilities.

$$\begin{aligned} F_{\mathrm{RBM}}(\mathbf{h}|\mathbf{x}) = \frac{F_{\mathrm{RBM}}(\mathbf{x}, \mathbf{h})}{F_{\mathrm{RBM}}(\mathbf{x})} &= \prod_{j=1}^{N} \frac{e^{\left(b_j + \frac{\mathbf{x}^T \mathbf{w}_{*j}}{\sigma^2}\right) h_j}}{1 + e^{\left(b_j + \frac{\mathbf{x}^T \mathbf{w}_{*j}}{\sigma^2}\right)}} \\ &= \prod_{j=1}^{N} F_{\mathrm{RBM}}(\mathbf{h}_j|\mathbf{x}). \end{aligned}$$

Since the binary hidden units h_j either can be on or off, the conditional single-node probabilities takes on the form of a sigmoid function:

$$\begin{aligned} F_{\mathrm{RBM}}(h_j = 1|\mathbf{x}) &= \frac{1}{1 + e^{-\left(b_j + \frac{\mathbf{x}^T \mathbf{w}_{*j}}{\sigma^2}\right)}} \\ F_{\mathrm{RBM}}(h_j = 0|\mathbf{x}) &= \frac{1}{1 + e^{\left(b_j + \frac{\mathbf{x}^T \mathbf{w}_{*j}}{\sigma^2}\right)}}. \end{aligned}$$

For the visible units $\mathbf{x}$, we find another conditional probability

$$\begin{aligned} F_{\mathrm{RBM}}(\mathbf{x}|\mathbf{h}) = \frac{F_{\mathrm{RBM}}(\mathbf{x}, \mathbf{h})}{F_{\mathrm{RBM}}(\mathbf{h})} &= \prod_{i=1}^{M} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\left(x_i - b_i - \mathbf{w}_{i*}^T \mathbf{h}\right)^2}{2\sigma^2}} \\ &= \prod_{i=1}^{M} \mathcal{N}(b_i + \mathbf{w}_{i*}^T \mathbf{h}, \sigma^2). \end{aligned}$$

Just like earlier, the single-node conditional probabilities $F_{\mathrm{RBM}}(x_i|\mathbf{h})$ are found to be independent. However, instead of having a sigmoid probability distribution, they follow a normal distribution with mean $b_i + \mathbf{w}_{i*}^T$ and variance σ^2.

The term "Gaussian" in the name of Gaussian-binary RBMs thus stem from both the shape of the energy function $E(\mathbf{x}, \mathbf{h})$ (Eq. (9)), as well as the condition probabilities of x_i.

Appendix B: Derivation of local energy

In this section we derive the formula for local energy given in Eq. (16). Using the definition of E_L from Eq. (2) and the Hamiltonian in Eq. (4), we find that for P particles in D spatial dimensions, E_L is

$$\begin{aligned} E_L &= \frac{1}{\Psi_T} H \Psi_T \\ &= \frac{1}{\Psi_T} \left[\sum_{p=1}^{P} \left(-\frac{1}{2} \nabla_p^2 + \frac{1}{2} \omega^2 r_p^2 \right) + \sum_{p<q} \frac{1}{r_{pq}} \right] \Psi_T \\ &= -\frac{1}{2} \frac{1}{\Psi_T} \sum_{p=1}^{P} \nabla_p^2 \Psi_T + \frac{1}{2} \omega^2 \sum_{p=1}^{P} r_p^2 + \sum_{p<q} \frac{1}{r_{pq}} \\ &= -\frac{1}{2} \frac{1}{\Psi_T} \sum_{p=1}^{P} \sum_{d=1}^{D} \frac{\partial^2 \Psi_T}{\partial x_{pd}^2} + \frac{1}{2} \omega^2 \sum_{p=1}^{P} r_p^2 + \sum_{p<q} \frac{1}{r_{pq}} \\ &= \frac{1}{2} \sum_{p=1}^{P} \sum_{d=1}^{D} \left[-\left(\frac{\partial}{\partial x_{pd}} \ln \Psi_T \right)^2 - \frac{\partial^2}{\partial x_{pd}^2} \ln \Psi_T + \omega^2 x_{pd}^2 \right] \\ &\quad + \sum_{p<q} \frac{1}{r_{pq}}, \end{aligned}$$

where we in the last step use that

$$\frac{1}{f(x)} \frac{\mathrm{d}^2}{\mathrm{d}x^2} f(x) = \left[\frac{\mathrm{d}}{\mathrm{d}x} \ln f(x) \right]^2 + \frac{\mathrm{d}^2}{\mathrm{d}x^2} \ln f(x)$$

for an arbitrary function $f(x)$. If we then let each visible node x_i represent one coordinate of a single particle, we arrive at Eq. (16), namely

$$E_L = \frac{1}{2} \sum_{i=1}^{M} \left[-\left(\frac{\partial \ln \Psi_T}{\partial x_i} \right)^2 - \frac{\partial \ln \Psi_T}{\partial x_i^2} + \omega^2 x_i^2 \right] + \sum_{p<q} \frac{1}{r_{pq}}.$$