Quantum Information and Computing

prof: *Simone Montangero*

Final project:

Neural Network ansätze for Quantum Many-Body systems

Alberto Bassi

August 1, 2021

Abstract

Neural Networks have been lately applied in practically every field of science. In this work, we explore a very recent application to Quantum Many-Body problems in one dimension. In particular, we will see how the many body wave function can be thought as a Restricted Boltzmann Machine (RBM) with complex weights. In this framework, Variational Montecarlo (VMC) methods are applied to minimize the variational energy (cost function) and find the ground state through Stochastic Reconfiguration (SR), which represents the learning algorithm.

Contents

Introduction

It is well known that many body quantum systems can be exactly solved only in the case of a few spins, since Hilbert space scales exponentially with the system size. This problem is commonly known as the curse of dimensionality. Fortunately, much of the information contained in the Hilbert space is redundant and therefore approximated ansätze must be explored to solve the problem in polynomial time, allowing the many body wave function to live only in a tiny fraction of the exponentially large Hilbert space. Recently, a great deal of attention has been put on Tensor Network (TN) states, which allow to reduce a lot the information carried with the wave function imposing a determined range for correlation, for example with a MPS [1].
In the last years, Machine Learning (ML) techniques have emerged as a fundamental tool in all sciences. A discussing review on ML applications in Physics can be found in [2]. One of the most recent and fascinating is in the field of condensed matter, in particular the study many body problems. Carleo et al. proposed a novel RBM ansatz for the wave function [3], for which parameters are, in a certain sense, learned to minimize the expectation value of the energy to reproduce the ground state of a well known and tested class of problems, such as low dimensional spin lattices. However, in this work direct correlations among spins are neglected, allowing indirect interactions between visible units only through the hidden layer.
The power of RBM ansatz relies on a much stronger base than simple efficient numerical results. Indeed, its mathematical ability to describe the many body wave function is based on strong representability theorems [4–6]. Moreover, RBMs have been recently proven to be formally equal to TN states [7].
In this project, we will understand how to use RBMs as ansatz for the many body wave function by reproducing the results of [3]. We will focus our attention on the efficient implementation of the idea above presented by implementing a fortan90 code able to reproduce the ground state of the Transverse Field Ising Model (TFIM) and the Anti-ferromagnetic Heisenberg Model (AFHM) in 1D.
This work is organized as follow. After a theoretical presentation on restricted Boltzmann machines and some of their use in modern machine learning, we will address our attention on their use in quantum many body problems. In particular, we will study the ground state through VMC techniques using Stochastic Reconfiguration [8] as learning algorithm. Next, we will benchmark the performance of our code comparing the obtained results with the python/C++ implemented NetKet library [9].

If not explicitly remarked, we will work with the computational basis, e.g. the eigenvectors of the Pauli matrix σ^z. We will use this notation for the 1 particle eigenstates of σ^z

$$|1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} , \qquad |-1\rangle = \begin{pmatrix} .0 \\ 1 \end{pmatrix} . \tag{1}$$

We will omit the tensor product symbol for many body operator, i.e. $\sigma_i^z \otimes \sigma_{i+1}^z$, which is nothing else than the operator σ^z acting on the i-th site and σ_{i+1}^z acting on the $(i+1)$-th site, will be simply written as $\sigma_i^z \sigma_{i+1}^z$. Moreover, many body identities will be omitted. For example, for a single site operator A_i on a 1D chain of N spins, we mean

$$A_i = \mathbb{1}_{i-1} \otimes A_i \otimes \mathbb{1}_{N-i} . \tag{2}$$

A generic 1D sequence of N spins is written as

$$|\mathcal{S}\rangle = |S_1 S_2 \dots S_N\rangle , \qquad (S_i = \pm 1) . \tag{3}$$

Theory

Restricted Boltzmann Machines

Boltzmann Machines (BM) are one of the most intriguing and used paradigms of stochastic neural networks. In a certain sense, they generalize Hopfield networks [10] to the non zero temperature case. A BM can learn a probability distribution and generate samples distributed according to it.
Much more interesting are Restricted Boltzmann Machines (RBM), which neglect intra layers connections and allow a much faster parallel computation. They have been used deep auto-encoders to reduce dimensionality of data, giving more robust results than simple PCA [11], and in deep belief networks as generative graphical model [6, 12].
From the network point of view, a RBM is composed by a layer of N_v visible units $\{v_i\}$, which are connected to a layer of N_h hidden units $\{h_i\}$ through a weight matrix W. Visible and hidden units can assume different sets of values, depending on the problem to be solved. For example, to train a RBM in order to reproduce the handwritten digits of MNIST dataset [13], a common choice is to use probabilities in the visible units and bernoulli hidden units [14]. While visible units are the machine reproduction of the environment probability

distribution, hidden units are its internal representation. Visible ($\{a_i\}$) and hidden ($\{b_i\}$) biases are introduced, while there are no connections between units in the same layer.
A RBM can be seen as an energy based model with

$$-\beta E(\mathbf{v},\mathbf{h}) = \sum_i a_i v_i + \sum_j b_j h_j + \sum_{ij} v_i W_{ij} h_j \ . \tag{4}$$

The joint probability distribution of visible and hidden units is given by the Boltzmann

$$P(\mathbf{v},\mathbf{h}) = \frac{1}{Z} e^{-\beta E(\mathbf{v},\mathbf{h})} \ , \tag{5}$$

where $Z = \sum_{\mathbf{v},\mathbf{h}} e^{-\beta E(\mathbf{v},\mathbf{h})}$ is the partition function, computed by summing Boltzmann factors over all possible configurations of visible and hidden units.
Since visible and hidden units are independent, marginal distributions can be computed by summing over all possible configurations of hidden units

$$P(\mathbf{v}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-\beta E(\mathbf{v},\mathbf{h})} = \frac{1}{Z}\prod_{i=1}^{N_v} e^{a_i v_i} \sum_{\mathbf{h}} e^{\sum_j b_j h_j + v_i \sum_j W_{ij} h_j} \ , \tag{6}$$

where a similar expression holds for hidden units.
The dynamic of a RBM is determined by a MCMC dynamic at non zero temperature, which allow the network not to be stacked in local minima, in contrast to Hopfield networks. In particular, each visible unit is updated independently to the others.
A RBM can learn the probability distribution of the environment, assumed to be fixed and known, by minimizing the Kulback-Leibler (KL) divergence between it and the Boltzmann distribution of visible units

$$D_{KL}(P_{env}(\mathbf{v})||P(\mathbf{v})) = \sum_{\mathbf{v}} P_{env}(\mathbf{v}) \log \frac{P_{env}(\mathbf{v})}{P(\mathbf{v})} \ , \tag{7}$$

where $P_{env}(\mathbf{v})$ is the visible units distribution when they are clamped to the training dataset, while $P(\mathbf{v})$ is the distribution when the machine is allowed to run freely [15]. Weights are updated according to the rule [14]

$$\Delta W_{ij} = \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}) \ . \tag{8}$$

In general, even though RBMs can be speeded up a lot through parallelization, due to the independence between intra layer units, to reach the stationary Boltzmann distribution the MCMC dynamics has to run a lot, slowing down the computation. Therefore, the term $\langle v_i h_j \rangle_{model}$ is very difficult to compute, while $\langle v_i h_j \rangle_{data}$ can be directly evaluated. Instead, MCMC dynamics can be stopped quite early by using (one step) Contrastive Divergence [16]. To evaluate averages over the data, we first clamp the visible units to the environment distribution, sampling the hidden units according to the Boltzmann distribution and computing the term $\langle v_i h_j \rangle_{data}$. Then, in order to evaluate averages over the model, we sample visible units given the actual values of hidden ones. The reconstructed configuration is thus used to sample again hidden units. The term $\langle v_i h_j \rangle_{model}$ is calculated by considering only the last two terms.

RBM ansatz for the quantum Many Body problem

Recently, it has been shown that RBM can be used to solve quantum many body problems [3]. In particular, the many body wave function is supposed take the form of the Boltzmann distribution of a RBM with visible units representing physical spins in a given basis. Visible and hidden units take values from the set $\{-1, 1\}$. Since we are dealing with a Restricted BM, with not connections between visible units, at this stage correlations among visible spins are given only by the weight connections with hidden spins. Direct correlations among visible spins are considered in [17] to enhance the ansatz.
We consider a quantum system of N deegres of freedom $\mathcal{S} = (S_1, \dots, S_N)$, which can represent either spins, or occupation numbers and so on and so forth. In the language of RBMs, these are the visible units. The wave function is supposed to take the form [3]

$$\Psi(\mathcal{S}) = \Psi(S_1, S_2, \dots, S_N) = \sum_{\mathbf{h}} e^{\sum_i a_i S_i + \sum_j b_j h_j + \sum_{ij} S_i W_{ij} h_j} = e^{\sum_i a_i S_i} \prod_{j=1}^{M} 2\ \cosh\left(b_j + \sum_i S_i W_{ij}\right) \ , \tag{9}$$

unless for a non-necessary normalization constant. The number of hidden units is $M = \alpha N$, where α is the hidden units density. By increasing it, we increase the number of parameter of the wave function, which leads to more accurate results. The number of variational parameters is thus $N_{var} = N(\alpha + 1) + \alpha N^2$.
In general, the set of parameters $\mathcal{W} = \{a, b, vec(W)\}$ are complex, to take into account the quantum nature of wave function. They can be *learned* to accomplish different tasks, such as founding the ground state, the excited states and perform unitary dynamics [3, 17, 18].

Variational optimization for the Ground State and Stochastic Reconfiguration

Given the wave function ansatz in Equation 9, we want to find the optimal parameters which represent the ground state. This is done by means of variational methods and it is based on the minimization of the energy functional

$$E(\mathcal{W}) = \frac{\langle\Psi|H|\Psi\rangle}{\langle\Psi|\Psi\rangle} \,, \tag{10}$$

which is nothing than the expectation value the energy of the system with respect to the variational ground state $|\Psi\rangle$ that depends parametrically by the weights $\mathcal{W}$. The optimization procedure, which has to be specified yet, will give us the optimal parameters, from which it is easy to compute the ground state energy via the RBM ansatz, along with the optimal variational state. In machine learning terms, this will represent the cost function to be minimized. The optimization task will be accomplish through an iterative algorithm, which at each step modifies the weights in order ot minimize $E(\mathcal{W})$.
In machine learning, the most common used optimization algorithm is Stochastic Gradient Descent (SGD). However, SGD performs very poorely for spin Hamiltonians [18] and therefore another approach is here considered. We will make use of the Stochastic Reconfiguration (SR) method, first introduced by Sorella et al. [8]. We will present the derivation of [18].
SR can be considered a full-fledged imaginary time evolution. The idea is to evolve the system along a path characterized by the projection $\mathbb{1} - \nu H$ until convergence is reached. ν is chosen such that $\mathbb{1} - \nu H \geq 0$. If ν is sufficiently small, we can generally write

$$(\mathbb{1} - \nu H)\,|\Psi\rangle = e_0\,|\Psi\rangle + \sum_\alpha e_\alpha\,|\Psi_\alpha\rangle + \left|\Psi^\perp\right\rangle \,, \tag{11}$$

where e_α are coefficient of the *variational derivatives* of the ket with respect to the weights such that $|\Psi_\alpha\rangle = \partial_\alpha\,|\Psi\rangle$ (remember that the variational states depends parametrically on the weights) and $\left|\Psi^\perp\right\rangle$ is an orthogonal state. We define the diagonal *variational operators* O_α by the action on a generic configuration state $|\mathcal{S}\rangle$ as

$$O_\alpha\,|\mathcal{S}\rangle = \frac{\partial_{\mathcal{W}_\alpha}\Psi(\mathcal{S})}{\Psi(\mathcal{S})}\,|\mathcal{S}\rangle \,. \tag{12}$$

By taking the action of $\langle\Psi|$ and $\langle\Psi_\alpha|$ on the expression of Equation 11, we get the equations

$$1 - \nu\,\langle H\rangle = e_0 + \sum_\alpha e_\alpha\,\langle O_\alpha\rangle \,, \qquad \left\langle O_\alpha^\dagger\right\rangle - \nu\left\langle O_\alpha^\dagger H\right\rangle = e_0\left\langle O_\alpha^\dagger\right\rangle + \sum_\beta e_\beta\left\langle O_\beta^\dagger O_\beta\right\rangle \,, \tag{13}$$

where the expectation values of a generic operator M are taken with respect to the variational state $\langle M\rangle = \frac{\langle\Psi|M|\Psi\rangle}{\langle\Psi|\Psi\rangle}$.
By defining the quantum Fisher matrix as

$$S_{\alpha\beta} = \left\langle O_\alpha^\dagger O_\beta\right\rangle - \left\langle O_\alpha^\dagger\right\rangle\langle O_\beta\rangle \,, \tag{14}$$

and the generalized force vector as

$$F_\alpha = \nabla_{\mathcal{W}_\alpha}\langle H\rangle = \left\langle O_\alpha^\dagger H\right\rangle - \left\langle O_\alpha^\dagger\right\rangle\langle H\rangle \,, \tag{15}$$

we can solve Equation 13 w.r.t. e_0 to get

$$\sum_\beta S_{\alpha\beta}e_\beta = -\nu F_\alpha \,. \tag{16}$$

The coefficients e_α can be now interpreted as the update coefficients for the weights, up to an overall coefficient e_0, which will be interpreted as the learning rate ϵ. In the end, the SR yields the coefficient updating rule

$$\mathcal{W} \to \mathcal{W} - \epsilon(S + \eta\mathbb{1})^{-1}F \,, \tag{17}$$

where η is a regularization term introduced to make S positive definite.

Efficient calculation of expectation values

We can compute the expectation values of a generic operator M with respect ti the variational by inserting a completeness in a given basis as

$$\frac{\langle\Psi|M|\Psi\rangle}{\langle\Psi|\Psi\rangle} = \frac{1}{\langle\Psi|\Psi\rangle}\sum_{\mathcal{S}} M_{loc}(\mathcal{S})\Psi(\mathcal{S})^*\Psi(\mathcal{S}) = \frac{\sum_{\mathcal{S}} M_{loc}(\mathcal{S})|\Psi(\mathcal{S})|^2}{\sum_{\mathcal{S}}|\Psi(\mathcal{S})|^2} \,, \tag{18}$$

where

$$M_{loc}(\mathcal{S}) = \frac{\langle \mathcal{S}|M|\Psi\rangle}{\Psi(\mathcal{S})} = \sum_{\mathcal{S}'} \frac{\langle \mathcal{S}|M|\mathcal{S}'\rangle}{\Psi(\mathcal{S})}\Psi(\mathcal{S}') \tag{19}$$

is called *local operator*.
The issue is now the computation of $M(\mathcal{S}, \mathcal{S}')$. The computation of the local operator M_{loc} is computationally efficient only if it is sufficiently sparse, i.e. most of the matrix elements are zero, similarly to what happen for the Ising and Heisenberg Hamiltonians.
One we can compute efficiently the matrix element of a sufficiently sparse operator, in principles we should use Equation 18 to calculate the expectation values. However, this would imply a summation over an exponential large set of elements. In order to compute it in polynomial time, we can use Markov Chain Monte Carlo (MCMC) techniques. We start from a spin distribution $\mathcal{S}$ and we generate a proposal sequence $\mathcal{S}'$ by flipping randomly one spin. Then, we accept the new sequence with probability

$$p(\mathcal{S} \to \mathcal{S}') = \min\left(1, \frac{|\Psi(\mathcal{S}')|^2}{|\Psi(\mathcal{S})|^2}\right) , \tag{20}$$

which does not require the exponentially large computation of the normalization term at each iteration.
Since Detailed Balance (DB) holds, the Markov Chain is assured to converge to the probability distribution given by the square modulus wave function $|\Psi(\mathcal{S})|^2$ after a sufficient number of sweeps N_{sweeps} through the spin lattice. If we generate $N_{samples}$ samples of the distributions, then the expectation values is approximated by

$$\frac{\langle \Psi|M|\Psi\rangle}{\langle \Psi|\Psi\rangle} \simeq \frac{\sum_{\mathcal{S}_{sampled}} M_{loc}(\mathcal{S}_{sampled})|\Psi(\mathcal{S}_{sampled})|^2}{\sum_{\mathcal{S}_{sampled}} |\Psi(\mathcal{S}_{sampled})|^2} . \tag{21}$$

Therefore, $N_{samples}$ must be chosen sufficiently large to have a robust estimation, while N_{sweeps} must be chosen in order to well mix the chain, since sequential samplings tend to be strongly correlated.

Variational derivatives

In order to compute the variational operator as in Equation 12 with respect to a given RBM ansatz, we need to calculate variational derivatives. For the RBM ansatz of Equation 9, we have

$$\begin{aligned} \frac{1}{\Psi(\mathcal{S})}\partial_{a_i}\Psi(\mathcal{S}) &= S_i , \\ \frac{1}{\Psi(\mathcal{S})}\partial_{b_j}\Psi(\mathcal{S}) &= \tanh[\theta_j(\mathcal{S})] , \\ \frac{1}{\Psi(\mathcal{S})}\partial_{W_{ij}}\Psi(\mathcal{S}) &= S_i \ \tanh[\theta_j(\mathcal{S})] , \end{aligned} \tag{22}$$

where $\theta_j(\mathcal{S}) = b_j + \sum_i S_i W_{ij}$.
Since for each MC sweep ($O(N)$ moves) we have to compute the previous matrix multiplication and take the product of hyperbolic cosines through the possible number of hidden spins, the cost of a MC sweep is $O(NM) = O(\alpha N^2)$.

Efficient computation of local energies for 1D lattice spin problems

By definition, the local energy is defined by taking the Hamiltonian as the operator in Equation 19

$$E_{loc}(\mathcal{S}) \equiv \frac{\langle \mathcal{S}|H|\Psi\rangle}{\Psi(\mathcal{S})} = \sum_{\mathcal{S}'} \frac{\langle \mathcal{S}|H|\mathcal{S}'\rangle}{\Psi(\mathcal{S})}\Psi(\mathcal{S}') . \tag{23}$$

We will exactly compute local energies for two well known 1D lattice spin problems. The Ising model in transverse field (TFIM) and the anti-ferromagnetic Heisenberg model (AFHM). Since those Hamiltonians are very sparse, local energies can be computed very efficiently.

1D Ising Model

The Hamiltonian of the quantum Ising model in transverse field reads

$$H = -h\sum_{i=1}^{N}\sigma_i^x - \sum_{i=1}^{N-1}\sigma_i^z\sigma_i^{z+1}\ , \tag{24}$$

where a term $\sigma_1^z\sigma_N^z$ must be add when periodic boundary conditions (pbc) are imposed.
The matrix elements are simply

$$\begin{aligned}\langle \mathcal{S}|H|\mathcal{S}'\rangle &= -h\sum_i\prod_{j\neq i}\delta_{S_j,S'_j}\langle S_i|\sigma_i^x|S'_i\rangle - \sum_i\prod_{j\neq i,i+1}\delta_{S_jS'_j}\left\langle S_iS_{i+1}\middle|\sigma_i^z\sigma_{i+1}^z\middle|S'_iS'_{i+1}\right\rangle = \\ &= -h\sum_i\prod_{j\neq i}\delta_{S_j,S'_j}\delta_{S_i,-S'_i} - \sum_i\prod_{j\neq i,i+1}\delta_{S_jS'_j}S_iS_{i+1}\delta_{S_iS'_i}\delta_{S_{i+1}S'_{i+1}}\end{aligned} \tag{25}$$

Therefore, the local energy of configuration $\mathcal{S}$ is

$$E_{loc}^{TFIM}(\mathcal{S}) = -h\sum_i\frac{\Psi(\mathcal{S}_{-i})}{\Psi(\mathcal{S})} - \sum_i S_iS_{i+1}\ , \tag{26}$$

where $\mathcal{S}_{-i}$ is the configuration obtained from $\mathcal{S}$ by flippping the i-th spin.

1D Antiferromagnetic Heisenberg model

The anti-ferromagnetic Heisenberg Hamiltonian reads

$$H = \sum_i\left[\sigma_i^x\sigma_{i+1}^x + \sigma_i^y\sigma_{i+1}^y + \sigma_i^z\sigma_{i+1}^z\right]\ . \tag{27}$$

The matrix elements with respect to the many body base are

$$\langle \mathcal{S}|H|\mathcal{S}'\rangle = \sum_i\left[\delta_{\mathcal{S}\mathcal{S}'}S_iS_{i+1} + \prod_{j\neq i,i+1}\delta_{S_j,S'_j}\delta_{S_i,-S'_i}\delta_{S_{i+1},-S'_{i+1}}(1-S_iS_{i+1})\right]\ . \tag{28}$$

Thus, local energy reads

$$E_{loc}^{AFHM}(\mathcal{S}) = \sum_{\mathcal{S}'}\langle \mathcal{S}|H|\mathcal{S}'\rangle\frac{\Psi(\mathcal{S}')}{\Psi(\mathcal{S})} = \sum_i\left[S_iS_{i+1} + \frac{\Psi(\mathcal{S}_{-i,-(i+1)})}{\Psi(\mathcal{S})}\left(1-S_iS_{i+1}\right)\right]\ , \tag{29}$$

where $\mathcal{S}_{-i,-(i+1)}$ is the sequence obtained from $\mathcal{S}$ by flipping the i-th and $(i+1)$-th spins.

Implementing symmetries

In general, RBM anstaz requires learning a number $O(N^2)$ parameters. However, if we know that the Hamiltonian is invariant under a set of transformation encoded in a symmetry group, we can enforce the wave function to be symmetric under the action of the same group to reduce the number of parameters and enhance performance.
Let us consider the symmetry group $\mathcal{G}$ acting on 1D spin lattices with periodic boundary conditions. An element on $g_s \in \mathcal{G}$ acts om a spin by translating it along the chain, such that $g_s(S_i) = S_{i+s}(s)$, where the index summation is done module N. We denote the translation of $s = 1,\dots,N$ positions of the spin sequence $\mathcal{S}$ as $\mathcal{S}(s)$.
To enforce the RBM to be invariant under the action of translation group $\mathcal{G}$, we introduce a feature dependent filter W_j^f, where $f = 1,\dots,\alpha$, in the spirit of transitionally invariant RBMs [19]. For each feature f, $\mathbf{W}^f$ acts like a convolutional filter on the N translated copies of the spin configuration. In this framework, variational parameters reduce to a^f and b^f. The anstaz (after tracing out hidden spins) reads

$$\Psi_{symm}(\mathcal{S}) = e^{\sum_f a^f\sum_i S_i}\prod_{f=1}^{\alpha}\prod_{s=1}^{N}2\cosh\left(b^f + \sum_j W_j^f S_j(s)\right)\ , \tag{30}$$

where the simplest way to enforce transnational invariance among visible units is to make the exponential term dependent on the total magnetization $M = \sum_i S_i$, which is the simplest translational invariant. We avoided to write the sum over all possible configurations in the exponential term (which simply would add a N

factor) in order to limit numerical instability due to spin changes. The number of variational parameters is now $N_{var} = 2\alpha + \alpha N$.
Variational derivatives for 30 become

$$\begin{aligned}
\frac{1}{\Psi(\mathcal{S})}\partial_{a^f}\Psi(\mathcal{S}) &= M \ , \\
\frac{1}{\Psi(\mathcal{S})}\partial_{b^f}\Psi(\mathcal{S}) &= \sum_s \tanh[\theta^f(\mathcal{S}(s))] \ , \\
\frac{1}{\Psi(\mathcal{S})}\partial_{W_i^f}\Psi(\mathcal{S}) &= \sum_s S_i(s)\tanh[\theta^f(\mathcal{S}(s))] \ ,
\end{aligned} \tag{31}$$

where $\theta^f(\mathcal{S}(s)) = b^f + \sum_j W_j^f S_j(s)$.

Code Development

Fortran (FORmula TRANscription) has been designed for numerical calculations and it is still today one of the fastest programming languages available in that field. For this reason, we have implemented a fortran90 program, whose dependency tree is shown in Figure 1. Source code is available at the link `https://github.com/lupoalberto98/QRBM`. While Fortran is perfect for simulations, we exploit the power of Python for high level tasks, such that plotting and file accessing for computing relative energies (notebook plot.ipynb available at the same link).

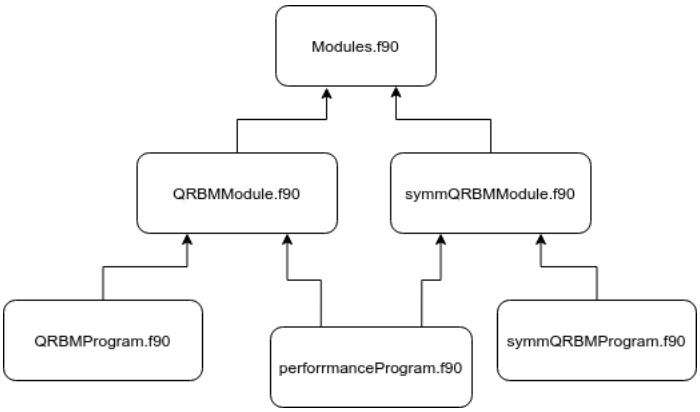

Figure 1: The dependency tree for QRBM program.

The contents of each file are:

- **Modules.90** contains some debugging subroutines along with routines for matrix inversion.
- **QRBMModule.f90** and **symmQRBMModule.f90** implement the RBM ansatz for the non-symmetric and symmetric cases explained above. In particular, in each of them we can find a derived type with all the parameter for RBM ansatz, a method to initialize it (we set all weights to small random numbers uniformly distributed in $([-0.1, 0.1])$ a method to deallocate memory, methods to compute local energy for different models, the Monte Carlo sampler, a routine to compute the variational derivatives and a routine to perform Stochastic Reconfiguration for a given number of steps. To monitor the computation, the user is also ask to insert a debugging variable. If true, an outlog.txt file is produced, where warnings and useful quantities are printed during SR iterations. In particular, we check sampled values of wave function to avoid Nans, the hermiticity of Fisher matrix, the correct inversion and useful computation times, such as inversion and sampling time. The program is stopped if the inversion is not successfully performed.
  The bottleneck of our algorithm for large spin chains is Fisher matrix inversion. This task is performed by first computing Cholesky (hermitian LU) decomposition by means of lapack's zpptrf subroutine. Then, the triangular matrix obtained is passed to zpptri for inversion. This method is not very efficient, since Cholesky decomposition requires $O(N_{var}^3)$ floating point operations. A more efficient way would be using MINRES-QLP algorithm [20], which never forms Fisher matrix explicitly, which requires $O(N_{var}^2)$ floating point operations. Inversion is performed through a modification of Laczos algorithm [21]. We leave fortan implementation of this method to further works.

- **QRBMProgram.f90** and **symmQRBMProgram.f90** call the respective modules, ask the user to input the parameter for a simulation (N, α, *model*, h and the number of iterations for SR). $N_{samples}$ and N_{sweeps} are internally set to 100 and 10 respectively, but can be modified straightforwardly. We choose a geometrically decreasing learning rate $\epsilon = \max(0.01 * 0.95^s, 0.001)$, at step s of SR, and a regularization factor $\eta = 0.0001$.

- **performanceProgram.90** asks the user to insert N_{min} and N_{max}. Then, simulations with increasing number of visible spins with hidden spin density $\alpha = 1$ are performed for both the non.symmetric and the symmetric ansätze. In output a .txt file contatining simulations times in order to compare the performances of two algorithms.

Results

Comparing performances

We first compare the speed of the non-symmetric RBM ansatz against the symmetric one. We expect a power law behaviour, emphasized by qualitative linear interpolation (from $N = 10$ in the log-log scale with optimize module of scipy. Results along with fitting parameters are shown in Figure 2.

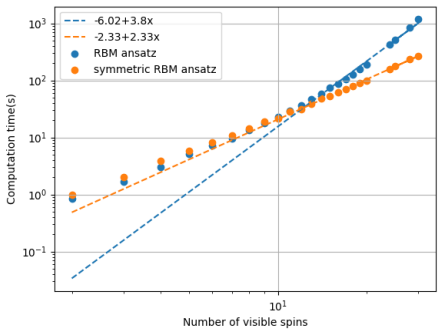

Figure 2: Log-log plot of computation time for non-symmetric and symmetric RBM ansätze for increasing values of visible spin numbers.

We can appreciate the polynomial behaviour of computation time for the two ansätze. In particular, two growing curves are observable with two different scaling regimes. However, the interpolated coefficients are different for what theoretically expected. Since we are using Cholesky decomposition for Fisher matrix inversion, we expect that the computation time scales as $O(N^6)$ for the non-symmetric ansatz ($N_{var} = N + \alpha N + \alpha N^2 = O(N^2)$) and as $O(N^3)$ for the symmetric one ($N_{var} = 2\alpha + \alpha N = O(N)$). But, interpolation coefficients are compatible with 4 and 2 respectively. This can be explained by the dominance of Fisher matrix formation ($O(N_{var}^2)$) for small lattices. Indeed, inversion computation time seems to be negligible for small chains.

Convergence

Once confirmed that the symmetric ansatz is more efficient, we have simulated the Ising model for different field strengths and the Heisenberg model with periodic boundary conditions.

In Figure 3, Figure 4, Figure 5 and Figure 6 we show the convergence properties for the Ising model at fields $h = 0, 0.5, 1, 2$ respectively. In the left panel we show the ground state energy versus the number of iterations of SR with log scale for x axes. In the left panel, we show the absolute relative energy with respect to the exact diagonalization value obtained with NetKet $\epsilon_{rel} = |(E_{gs} - E_{ED})/E_{ED}|$ with log scale for y axes. This separation is done in order to better distinguish the two decay regimes and analyse the error while approaching the exact value of ground state energy. With respect to Figure 3 at $h = 0$, we can observe that the convergence is very good and relative error decreases as α and the number of variational parameters increase, as expected. In particular, an exponential error decaying is observed.
For $h = 0.5$ (Figure 4), the convergence is ruined and it seems to be more noisy for $\alpha = 2$. In particular, we do not observe a net exponential decay of relative error. Instead, ground state energy becomes lower than the exact value and then it approaches it from below. This behaviour is possibly due to the too high learning rate,

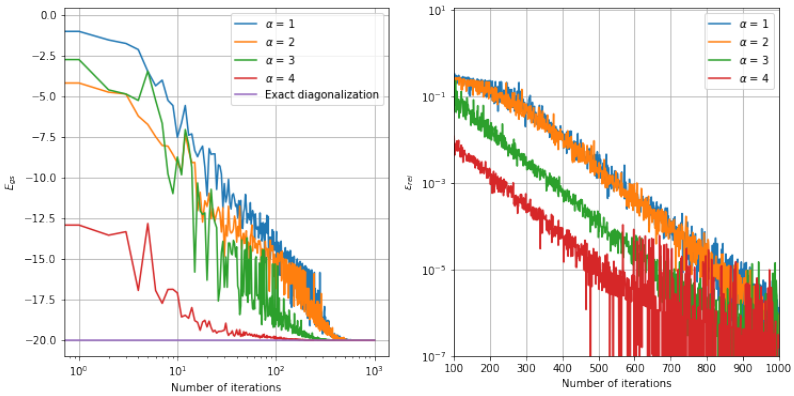

Figure 3: Symmetric RBM ansatz for Ising model at field $h = 0$ for different values of α.

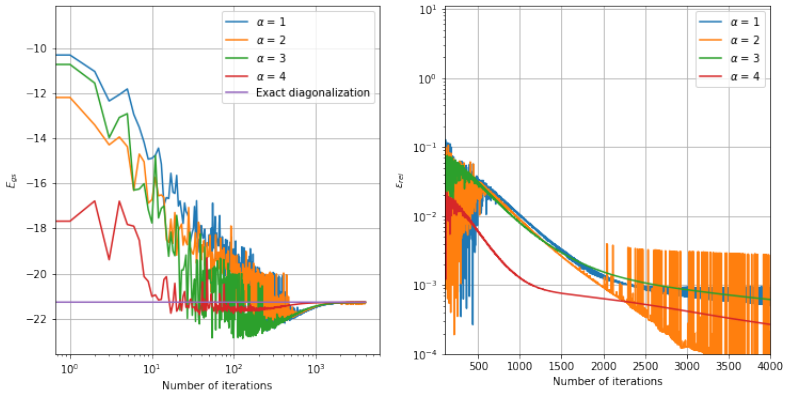

Figure 4: Symmetric RBM ansatz for Ising model at field $h = 0.5$ for different values of α.

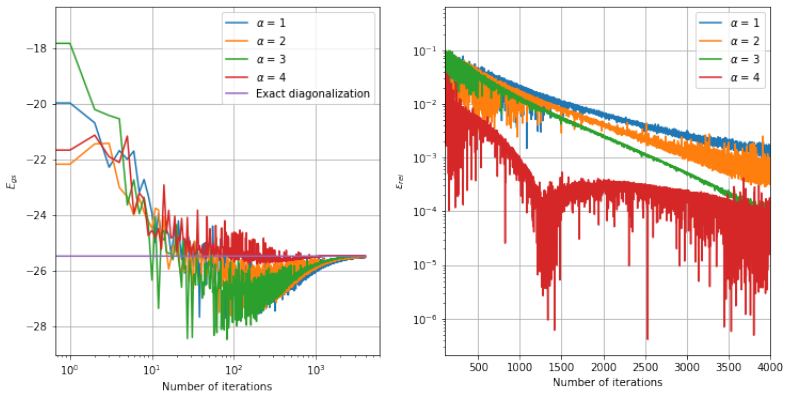

Figure 5: Symmetric RBM ansatz for Ising model at field $h = 1$ for different values of α.

especially at the beginning of SR. Moreover, we come back again to a net exponential decay for $\alpha = 4$. Similar considerations are valid also for $h = 1, 2$.
However, in Figure 6 for $h = 2$ we do not observe an error decreasing with the number of variational parameters. Similar results are also obtained for the 22 Heisenberg chain, as shown in Figure 7. A possible explanation is the limited size of the chain or the still few hidden units density.
In all these cases, the general trend validating the results of [3] is confirmed:

- A more precise ground state energy is obtained increasing α

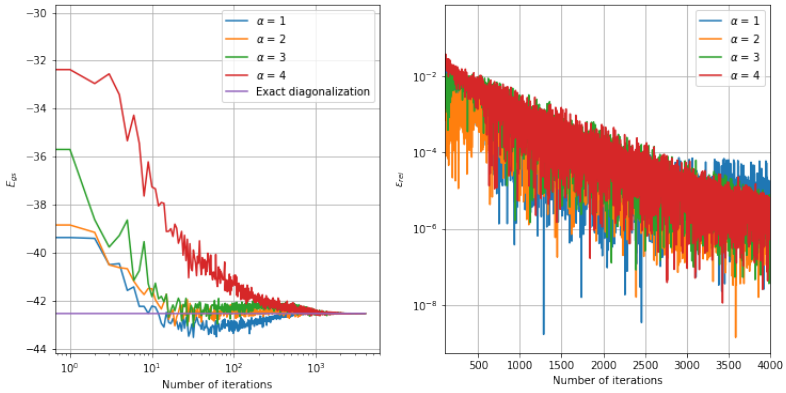

Figure 6: Symmetric RBM ansatz for Ising model at field $h = 2$ for different values of α.

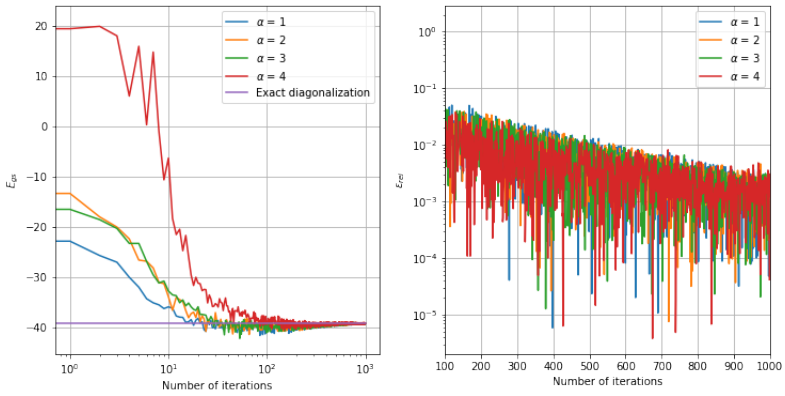

Figure 7: Anti-ferromagnetic Heisenberg model for 22 spins.

- A more accurate learning is gained increasing α

To conclude this section, in Figure 8 we show the relative error with respect to the exact energy computed by averaging the last 1000 iterations of SR for the above considered simulations. Again, we notice a polynomial decay of error as α is increased. An outlier is represented by $\alpha = 2$ for $h = 0.5$, a value which also shows high instability (Figure 4). This instability can be again explained with a too high learning rate or a too low number of MCMC samples ($N_{samples} = 100$) for $h > 0$. Further works should consider different learning strategies and increasing number of samplings, in order to minimize instability at the end of convergence.

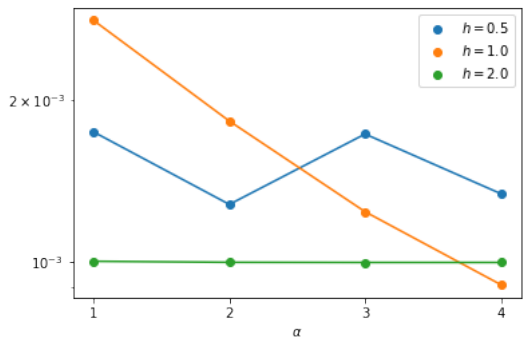

Figure 8: Relative error with respect exact energy computed for different fields and hidden unit densities.

Feature detection

After reducing the variational parameters, we have seen that the symmetrized RBM can be used to feature extraction. In Figure 9 we can seen the variational filters extracted for the Ising model at the critical point $h = 1$. In particular, we notice that only a few values are activated, capturing the most relevant correlations among spins. The positions of these value is not important, since the filter acts on all the translated copies of the chain.

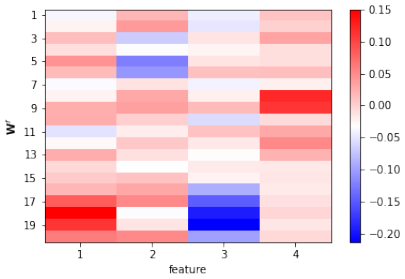

Figure 9: Feature map extracted from a $N = 20$ Ising chain at criticality.

Conclusion and outlook

In this last project, we have studied how neural networks, in particular restricted Boltzmann machines, can be used to represent quantum many body wave functions. In this context, we have seen that the learning problem is to minimize the expectation value of the Hamiltonian on a variational state through variational Monte Carlo techniques, such as Stochastic Reconfiguration, which give more robust result than the simpler Stochastic Gradient Descent due to spin flip. We have implemented the standard RBM anstaz and then improved it by considering translational invariant states reducing the number of parameters to be learned. With this toll in hand, we have studied the quantum Ising and Heisenberg models.
Even though we have focused our attention on the efficient implementation for the ideas exposed in [3], much work has to be done yet to improve our program. In particular, we should implement a parallel Markov Chain sampler through OpenMPI and we should use the MINRES-QLP algorithm for efficient inversion of hermitian matrices. Moreover, more studies have to be performed with more MC sampling and different coefficient learning decays in order to reduce numerical instability and speed up the computation.
The importance of Neural Network ansätze for quantum many body systems first relies on their ability to represent stationary states and also unitary time evolution, as already considered in [3]. However, in that work correlations among spins were considered only indirectly through hidden units. More robust results have been recently obtained in [17] adding fictitious correlation units to the RBM ansatz in order to better study problems at the critical point.
Moreover, RBM states have been shown to be equivalent to Tensor Network states, widely studied in the last years, and give more precise results. In particular, the exact mapping between a MPS and a RBM state can be used to initialize the weights and enhance the performance of the network.
To conclude, also more general neural network states should be taken into account, for example deep networks.

References

[1] Perez D. Garcia, F. Verstraete, M.M. Wolf, and J.J. Cirac. Matrix product state representations. *Quantum Information & Computation*, 7:401–430, 2007.

[2] Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *arXiv:1903:10563*, 2019.

[3] Giuseppe Carleo and Mathias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science*, 355:602–606, 2017.

[4] A.N. Kolmogorov. On the representation of continuous functions of several variables by superpositions of continuous functions of fewer variables. *Dokl. Akad. Nauk SSSR*, 108:179–182, 1961.

[5] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

[6] N. Le Roux and Y. Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural Computation*, 20:1631–1649, 2008.

[7] Jing Chen, Song Cheng, Haidong Xie, Lei Wang, and Tao Xiang. Equivalence of restricted boltzmann machines and tensor network states. *Physical Review B*, 97:085104, 2018.

[8] Sandro Sorella, Michele Casula, and Dario Rocca. Weak binding between two aromatic rings: Feeling the van der waals attraction by quantum monte carlo methods. *The Journal of Chemical Physics*, 127:014105, 2007.

[9] Giuseppe Carleo, Kenny Choo, Damian Hofmann, James E. T. Smith, Tom Westerhout, Fabien Alet, Emily J. Davis, Stavros Efthymiou, Ivan Glasser, Sheng-Hsuan Lin, Marta Mauri, Guglielmo Mazzola, Christian B. Mendl, Evert van Nieuwenburg, Ossian O'Reilly, Hugo Théveniaut, Giacomo Torlai, Filippo Vicentini, and Alexander Wietek. Netket: A machine learning toolkit for many-body quantum systems. *SoftwareX*, page 100311, 2019.

[10] J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *PNAS*, 79:2554–2558, 1982.

[11] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensional of data with neural networks. *Science*, 313:504–507, 2006.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[13] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[14] Geoffrey E. Hinton. A practical guide to training restricted boltzmann machines. In G. Montavon, G.B. Orr, and K.R. Müller, editors, *Neural Networks: Tricks of the Trade. Lecture notes in Computer Science*, volume 7700. Springer, Berlin, Heidelberg, 2012.

[15] Geoffrey E. Hinton and David H. Ackley. A learning algorithm for boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

[16] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

[17] Agnes Valenti, Eliska Greplova, Netanel H. Lindner, and Sebastian D. Huber. Correlation-enhanced neural networks as interpretable variational quantum states. *arXiv:2103:05017v1*, 2021.

[18] Chae-Yeun Park and Micheal J. Kastoryano. Geometry of learning neural quantum states. *Physical Review Research*, 2(2):023232, 2020.

[19] K. Sohn and H. Lee. Learning invariant representations with local transformations. *Proceedings of the 29th International conference on Machine Learning*, pages 1311–1318, 2012.

[20] Sou-Cheng T. Choi and Micheal A. Saunders. Algorithm 937: Minres-qlp for symmetric and hermitian linear equations and least-squares problems. *ACM Trans Math Softw*, 40(2), 2014.

[21] Cornelius Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Nat. Bur. Standards*, 45:255–282, 1950.