Machine Learning Applications in Quantum Mechanics

Louis Rabinowitz

November 2019

1 Abstract

Many problems in quantum mechanics do not have accessible analytical solutions and therefore have only approximate solutions are known. Often the approximate solutions must make a trade off of accuracy for efficiency. The use of neural networks as approximate wavefunctions may be able to provide a widely applicable accurate solutions that take advantage of efficient machine learning algorithms. Additionally, the application of neural networks will give researches a new set of tools with which to approach problems. This paper focuses on two recent papers that apply machine learning techniques to the variational Monte Carlo method. First, we will address Teng, 2018 which uses radial basis function networks to tackle simple Hamiltonians that have been perturbed from their known solutions. Second, we will address Pfau et al., 2019 which presents a novel neural network architecture that solves many-electron systems with results comparable to those of the most accurate quantum chemistry algorithms. These papers demonstrate the ability of machine learning to solve quantum mechanical problems and they build a basis from which we can expand the use of machine learning in the field of quantum mechanics and quantum chemistry.

2 Introduction

Machine learning is constantly being applied to new fields. This paper discusses two applications of machine learning in the field of quantum mechanics. Specifically, different adjustments to the variational Monte Carlo method. This section will provide the necessary background information to discuss each of our methods.

Often in quantum mechanical problems we hope to find the energy of the ground state of a known Hamiltonian with unknown wavefunctions. Traditionally, the variational method is an easy to use method that requires a trial wavefunction for the ground state wavefunction. In general, the closer the form of your trial wavefunction is to the form of the real ground state wavefunction the closer your calculated energy will be to the true ground state energy. Therefore, it is important for your initial wavefunction to resemble the ground state. In the case of simple systems (e.g. quantum harmonic oscillator, particle in a box, hydrogen atom) physical intuition can be used to pick your trial wavefunction. In complex systems (e.g. many-electron systems) physical intuition can be difficult to develop so methods have been developed to approximate wavefunctions using a finite basis set of orbitals [4]. We hope to show how artificial neural networks can replace or aid our choice of trial wavefunction in both simple [7] and complex cases [3].

2.1 Artificial Neural Networks

Artificial neural networks are versatile tools used in many machine learning applications. Here, we will address the use of neural networks to approximate functions. The basic structure of a neural network is shown in figure 1. Each node in a hidden layer of a neural network performs the following operation: $\sigma(\mathbf{w} \cdot \mathbf{x} + b)$, where $\mathbf{w}$ is a vector of weight parameters associated with the incoming connections, b is a bias parameter, $\mathbf{x}$ is the input to the node, and σ is some function (called an activation function). When sufficient hidden layers are present in the network it can be shown that neural networks have the capability to express any continuous function [1].

Back-propagation is an algorithm necessary to train and take advantage of the expressive power of a neural network. Back-propagation tunes the parameters of a neural network given the gradient of a loss function.

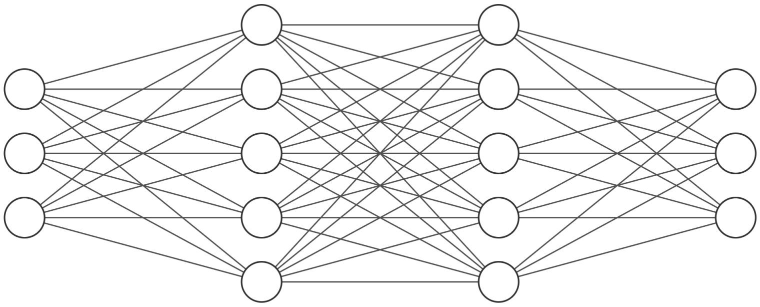


Figure 1: The generic structure of a neural network. From left to right the layers are: input, 2 hidden layers, and output. In this figure the height of each layer is arbitrary. In general, input and output layers do not need to share the same height and any number of hidden layers of any height may be included.

There are many schemes for calculating this gradient. Each paper discussed here will use stochastic gradient descent. A full explanation of the back-propagation and stochastic gradient descent algorithms is outside the scope of this paper. In general back-propagation can be thought of as updating the parameters of a neural network from the back of the network to the front by using properties of the chain rule.

In this paper radial basis function networks (RBF) and the Fermionic Neural Network (FermiNet) will be discussed. RBFs take their name from the activation functions used on their final hidden layer. In general any radial function can be used, but we will use the following Gaussian activation function:

$$\sigma(x) = ae^{b|\mathbf{x}-\mathbf{c}|^2}, \tag{1}$$

where a, b, and c are parameters to be learned by the network. Using this activation function the output of the RBF is expressed as:

$$\psi(\mathbf{x}) = \sum_{i=0}^{M} a_i e^{b_i|\mathbf{x}-\mathbf{c_i}|^2}, \tag{2}$$

where M is the height of our final hidden layer. Further details and results of radial basis function networks will be discussed in section 3.1. Further reading on the definition of RBFs can be found in Broomhead and Lowe, 1988 [2].

The Fermionic Neural Networks is a neural network that preserve the following property:

$$\psi(\mathbf{x}_1, \mathbf{x}_2) = -\psi(\mathbf{x}_2, \mathbf{x}_1), \tag{3}$$

where ψ is the output of our network and $\mathbf{x}_i$ are inputs to the network. The above property is essential when your inputs are the positions and spins of fermions. Further analysis of the Fermionic Neural Netowrk will occur in section 3.2. FermiNet was introduced in Pfau et al., 2019 [3].

2.2 Variational Monte Carlo

Given a Hamiltonian and a wavefunction, the energy of a system can be found by evaluating the following equation:

$$E = \frac{\langle\psi(\mathbf{x})|\, H \,|\psi(\mathbf{x})\rangle}{\langle\psi(\mathbf{x})|\psi(\mathbf{x})\rangle}, \tag{4}$$

where ψ is the wavefunction and H is the Hamiltonian. The variational method replaces the exact wavefunction, ψ, with a trial wavefunction, ϕ, which depends on some unknown parameters, λ. We then evaluate equation 4 with ϕ as our wavefunction and take the gradient with respect to λ. The result is minimized and our ground state energy is approximated. If the minimization can not be calculated analytically we can repeatedly update λ according to the gradient of our calculated energy until the energy converges. Variational Monte Carlo evaluates equation 4 using Monte Carlo integration. This approach was introduced in 1965 by McMillian [5].

Here, we will give a brief overview of the Metropolis-Hastings algorithm for Markov Chain Monte Carlo. The goal of this algorithm is to sample from an unknown distribution. This is done by constructing a Markov Chain (a random process where each event only directly depends on the previous event). At each step of the chain we have a current sample, x_t, and wish to select our next sample, x_{t+1}. The next sample is proposed according to some distribution $x \sim p(x_t)$ which is usually taken to be Gaussian. We then accept x as x_{t+1} according to the probability $\min(1, f(x)/f(x_t))$, where f is the function we hope to integrate. Otherwise we repeat our previous sample, $x_{t+1} = x_t$. After many iterations of this algorithm we approximate the integral as:

$$\frac{1}{N}\sum_{t=t_0}^{N} f(x_t), \tag{5}$$

where t_0 is the burn-in time, N is the number of samples taken after the burn-in time, and f is our sampled function. The burn-in time refers to the number of steps required for our Markov Chain to be effectively independent of our initial x. For the statistical background of the method you may read refer to [8].

3 Applications and Results

3.1 Radial Basis Function Network

The use of radial basis function networks to construct a trial wavefunction in the variational Monte Carlo method was discussed in Teng, 2018 [6]. All figures and tables in this section are taken directly from Teng, 2018. The construction of a radial basis function network was discussed in section 2.1.

Gaussian radial basis function networks have many shared properties with wavefunctions. Both functions must be smooth and must not diverge as their input approaches infinity. These shared properties along with the power of a neural networks to approximate functions hints at RBFs being a powerful tool to propose trial wavefunctions in the variational Monte Carlo method (VMC).

Teng, 2018 opted to use a RBF to determine the coefficents associated with known basis functions. So now we have discrete quantum numbers as the input to our RBF and therefore we replace equation 2 with:

$$\psi(\mathbf{n}) = \sum_{i=0}^{M} a_i e^{b_i|\mathbf{n}-\mathbf{c_i}|^2}, \tag{6}$$

where $\mathbf{n}$ is an input vector of integer quantum numbers that label our basis functions. Notice that $\psi(\mathbf{n})$ outputs a scalar. For all the systems addressed there are a countably infinite amount of basis functions, so we must truncate n at some value $n_{\max}$. Our Monte Carlo integration must now sample from a discrete distribution of $\mathbf{n}$; this is done simply by changing our proposal distribution to only select integers.

Using these adjustments the proposed wavefunction is:

$$|\psi\rangle = \sum_{n=0}^{n_{\max-1}} \psi(n)\,|n\rangle\,, \tag{7}$$

where $|n\rangle$ refers to our numbered basis functions. In this section we will address the results of this network for the following systems: 1-D harmonic oscillator, 2-D harmonic oscillator, and the particle in a box.

3.1.1 1-D Quantum Harmonic Oscillator

The first Hamiltonian addressed is the 1-D quantum harmonic oscillator described by the Hamiltonian:

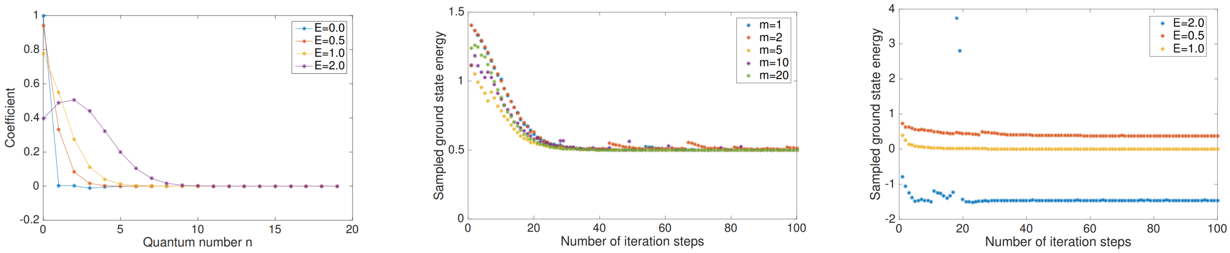
$$H = \frac{\hat{p}^2}{2} + \frac{\hat{x}^2}{2} + E\hat{x} = H_0 + E\hat{x}, \tag{8}$$

where E is the energy of a linear perturbation to the generic Hamiltonian. The Hamiltonian has been made in natural units so our analytic ground state energy is just: $E_0 = 0.5(1 - E^2)$.

The 1-D quantum harmonic oscillator is used as a proof of concept and as such the actual known basis functions are used as $|n\rangle$ in equation 7. Allowing the comparison experimental results and analytical values of $\psi(n)$. Analytically, the RBF outputs should be:

$$\psi(n) = \frac{1}{\sqrt{2^n n!}} E^n e^{-E^2/4}. \tag{9}$$

Agreement of coefficients between the results and equation 9 can be seen in figure 2a. Each value for E was trained using 100 parameter updates to the RBF with 50,000 Monte Carlo samples taken in between each update. The RBF used for the 1-D harmonic oscillator consisted of a single input node and a single hidden layer. Training for different sizes of the hidden layer and sizes of perturbations can be seen in figures 2b and 2c respectively. In figures 2b and 2c there are iterations where the energy suddenly increases. These increases are due to adjusting RBF parameters too quickly and going past a minimum. These jumps can be avoided by reducing the learning rate of the neural network, but come at the cost of slower training.



(a) Coefficients

(b) Hidden Layers

(c) Perturbations

Figure 2: (a) A plot comparing coefficients, $\psi(n)$, for different sized perturbations. Higher energy perturbations excite higher energy wavefunctions of the 1-D harmonic oscillator. (b) A plot showing the ground state energy while training our RBF for $E = 0$ for various sized hidden layers. (c) A plot comparing energies during training for various values of E with $n_{\max} = 20$

3.1.2 2-D Harmonic Oscillator

The 2-D harmonic oscillator in an electric field is described by the Hamiltonian:

$$H = \frac{\hat{p}_x^2}{2} + \frac{\hat{x}^2}{2} + \frac{\hat{p}_y^2}{2} + \frac{\hat{y}^2}{2} + E_x\hat{x} + E_y\hat{y} = H_0 + E_x\hat{x} + E_y\hat{y}. \tag{10}$$

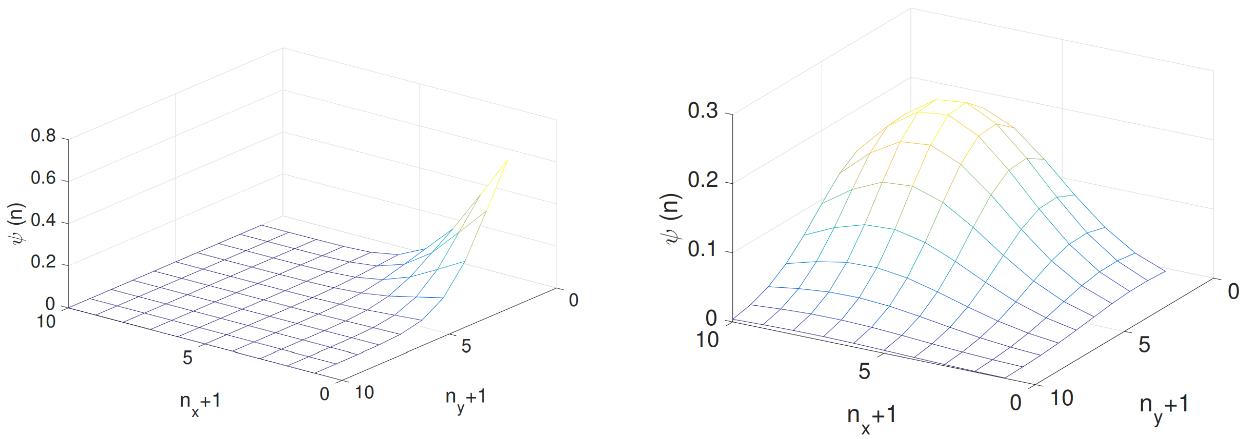
Notice the linear perturbations in either dimension.

In the two dimensional case we must change our input layer height to two as two quantum numbers are now required to label each basis function. Once again the unperturbed basis functions of the system will be used in equation 7. We will use the 2-D harmonic oscillator primarily to discuss the effect of changing $n_{\max}$.

$n_{\max}$	VMC Energy
3	-6.28397
4	-7.80747
5	-8.02855
10	-8.71073
20	-8.90894
40	-8.99571

Table 1: The computed variational Monte Carlo energies for the perturbation $E_x = 4, E_y = 2$ for various values of $n_{\max}$. The analytical energy is -9.0. For $n_{\max} = 40$ we are within the sampling error of our Monte Carlo integration to -9.0.

In figure 3 we see plots of $\psi(n_x, n_y)$ for two different perturbations with $n_{\max} = 10$. For the higher energy perturbation ($E_x = 4, E_y = 2$, figure 3b) many high energy states are excited. Those states where



(a) $E_x = 1, E_y = 1$

(b) $E_x = 4, E_y = 2$

Figure 3: Each plot shows the values of $\psi(\mathbf{n})$ for different input quantum numbers, (n_x, n_y). In (a) the perturbation, $E_x = E_y = 1$, is small and the contribution from higher energy basis functions is close to zero. In (b) the perturbation, $E_x = 4, E_y = 2$, is large and many higher energy basis functions are clipped. This causes our calculated ground state energy to be far from the analytical result.

either $n_x > n_{\max}$ or $n_y > n_{\max}$ are not sampled during our Monte Carlo integration and this reduces our calculated energy. Table 1 shows the $n_{\max}$ dependence of our VMC energy and provides numerical evidence of the clipping shown in figure 3b. Further increasing of $n_{\max}$ improves our VMC calculation, but comes at the cost of more parameters to train in our neural network. An n increase of $n_{\max}$ adds roughly $O(n^2)$ more parameters to train.

3.1.3 Particle in a Box

The particle in a box refers to the following Hamiltonian:

$$H = \frac{\hat{p}}{2} + V(x) + a\hat{x} = H_0 + a\hat{x}, \tag{11}$$

where a is the slope of a linear perturbation and $V(x) = 0$ from $0 < x < 1$ and ∞ for other values of x. All networks discussed in this section will have $n_{\max} = 20$ and a single hidden layer with a height of 10. There are discontinuities in equation 11 at $x = 0$ and $x = 1$. No discontinuities were present in the 1-D and 2-D harmonic oscillators. This poses a different challenge about how well an RBF can approach a problem without a smooth Hamiltonian. Once again the unperturbed wavefunctions are used as our basis states.

a	1st order	2nd order	VMC energy	exact value
0.0	4.9348	4.9348	4.9348±0.0001	4.93481
2.0	5.9348	5.9260	5.9260±0.0001	5.92603
4.0	6.9348	6.8997	6.8998±0.0001	6.89974
8.0	8.9348	8.7944	8.7960±0.0003	8.79508
-8.0	0.9348	0.7944	0.7950±0.0003	0.795078

Table 2: A comparison of VMC results with those from first and second order perturbation theory. VMC results are strictly more accurate than 1st order perturbation theory results and are comparable or better than 2nd order perturbation theory results for all values of a tested.

Table 2 shows the results for various sizes of perturbations. On average the VMC energies are closer than energies predicted by second order perturbation theory to the true ground state energies. This shows success on navigating problems with more difficult boundary conditions. It is worth noting that our choice of wavefunctions is also incredibly important for alleviating the boundary issue.

Figure 4 shows VMC energies at each training iteration. Interestingly, smaller perturbations cause more jumps in our training. The cause of this effect is likely that very few higher energy basis functions should be excited at low energies and randomness in stochastic gradient descent may cause them to be excited.

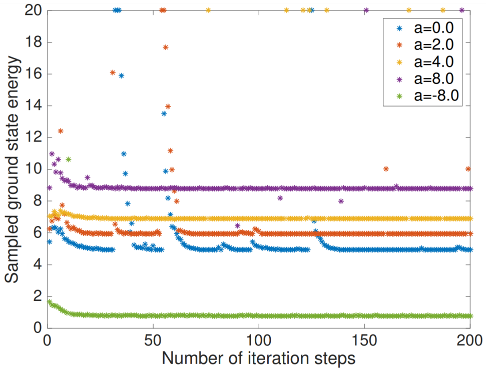


Figure 4: A plot of VMC energies at each training iteration for various values of a while training on the particle in a box Hamiltonian. Energies greater than 20 were set to 20 to not obscure the relevant features of the plot.

3.2 Fermionic Neural Network

The derivation and construction of the Fermionic Neural Network (FermiNet) is the subject of Pfau et al., 2019 [3]. All figures and tables in this section will be from Pfau et al., 2019. The FermiNet is a trial wavefunction for use in the variational Monte Carlo method for multi-electron systems. The motivation of the network is to preserve anti-symmetry of the wavefunction in the exchange of electrons:

$$\psi(\mathbf{x}_1, \mathbf{x}_2) = -\psi(\mathbf{x}_2, \mathbf{x}_1). \tag{12}$$

This property must hold true for an input of any size. The FermiNet achieves this property by building the wavefunction from Slater determinants. Slater determinants are described in [7]. Traditionally, a finite basis set of single electron wavefunctions is used to construct the trial wavefunction. This can lead to inconsistencies at the boundary between two neighboring single electron wavefunctions. The use of a neural network alleviates this issue by not assuming any specific basis. The architecture of FermiNet is displayed in figure 5.

The exact construction and mathematical foundation used in the network is beyond this paper, but can be found in the original paper [3]. In the network the following features are considered: $\mathbf{r_i} - \mathbf{R_j}$, $|\mathbf{r_i} - \mathbf{R_j}|$, $\mathbf{r_i} - \mathbf{r_j}$ and $|\mathbf{r_i} - \mathbf{r_j}|$, where $\mathbf{r_i}$ refers to the position of an electron and $\mathbf{R_i}$ refers to the position of an atomic nucleus. Additionally, each electron position is labeled by its spin. The one and two electron features are treated differently within the network.

The training of FermiNet differs slightly from traditional VMC approaches. In conventional VMC, there are usually $O(10^1 - 10^2)$ parameter updates and $O(10^4 - 10^5)$ Monte Carlo steps between each parameter update. Instead FermiNet updates parameters every 10 Monte Carlo steps, but has $O(10^5 - 10^6)$ parameter updates. The overall number of calls to the neural network is approximately the same in either case. In each experiment the FermiNet was initialized to match a mixture of Hartree-Fock orbitals. This was done to reduce the number of training steps to move through a non-physical configuration.

The discussion of results will be restricted to equilibrium geometries of single atoms to avoid misinterpretation of more complex quantum chemistry. The results were compared to a variety of quantum chemistry algorithms in table 3.

FermiNet performs favorably to traditional VMC methods and comparably to diffusive quantum Monte Carlo (DMC) which is generally considered superior to traditional Monte Carlo approaches. All results from FermiNet are considered to be within chemical accuracy of the exact measurements. Pfau et al., 2019 further discusses the results of FermiNet on systems that traditional methods struggle to correctly address. These

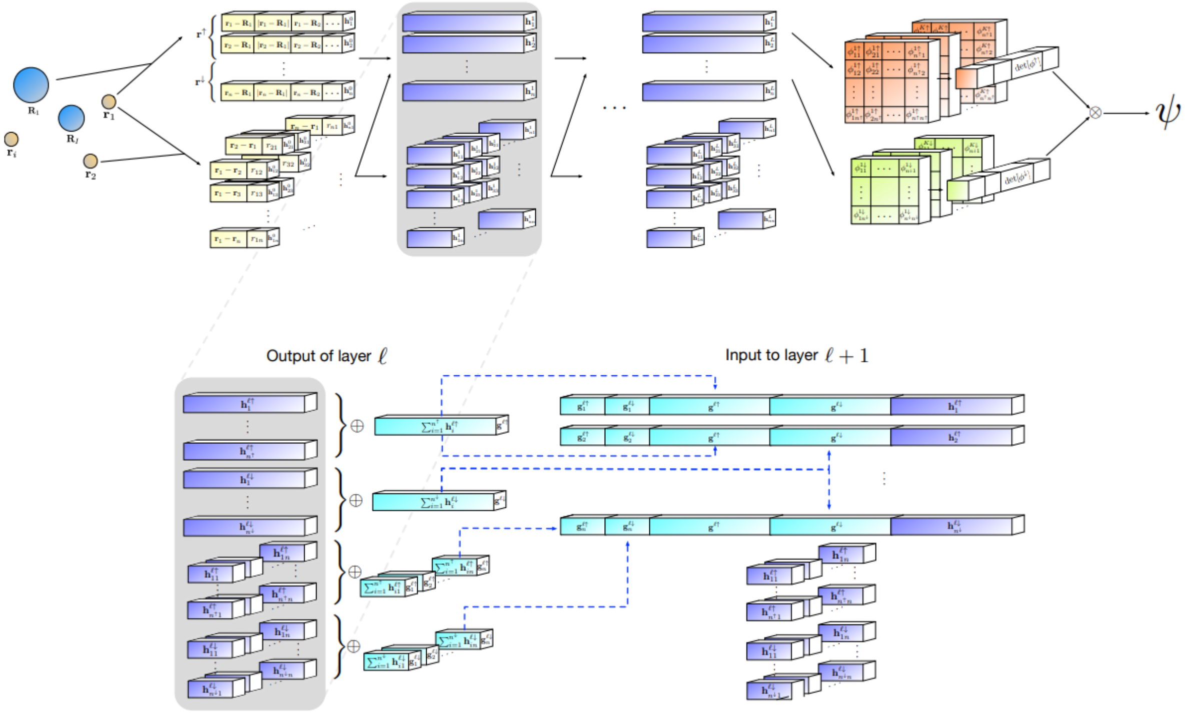


Figure 5: Top: The global architecture of FermiNet. Positional inputs of electrons are passed through one and two electron streams at each layer before being converted in to Slater Determinants in the final layer. The output is the value of the wavefunction at a position. Bottom: The structure of a single hidden layer of FermiNet. In each layer bulk properties of the one and two electron layers are calculated and used as additional input into the following single electron layer.

Ground state energy (E_h)

Atom	Fermi Net	VMC	DMC	Exact
Li	-7.47798(1)	-7.478034(8)	**-7.478067(5)**	-7.47806032
Be	**-14.66733(3)**	-14.66719(1)	-14.667306(7)	-14.66736
B	-24.65370	-24.65337(4)	**-24.65379(3)**	-24.65391
C	**-37.84471(5)**	-37.84377(7)	-37.84446(6)	-37.8450
N	**-54.58882(6)**	-54.5873(1)	-54.58867(8)	-54.5892
O	**-75.06655(7)**	-75.0632(2)	-75.0654(1)	-75.0673
F	**-99.7329(1)**	-99.7287(2)	-99.7318(1)	-99.7339
Ne	**-128.9366(1)**	-128.9347(2)	**-128.9366(1)**	-128.9376

Table 3: A table comparing the calculated ground state energies of different quantum chemistry algorithms for small atoms in units of Hartrees. The method with closest agreement to the exact ground state energy is in bold. The same process can be repeated to find ionization potentials or electron affinities by adding or removing an electron and calculating the new ground state energy.

systems often have low energy excited states or out-of-equilibrium geometries. In all cases that are addressed FermiNet significantly out performs other non-exact methods.

4 Conclusion

Machine learning introduces powerful tools for computation in quantum mechanics such as the radial basis function network and the Fermionic Neural Network. Radial basis function networks have been shown to solve for the ground state energy of simple perturbed systems by learning coefficients of a known basis set. The Fermionic Neural Network has been shown to compete with top quantum chemistry algorithms without the inherent loss associated with a discrete basis set and is able to learn purely from particle positions and spins. It is likely that recent the recent successes of machine learning in quantum mechanics can be maintained and will remain an important part of modern research in the field.

5 Acknowledgments

I would like to acknowledge my project partner Davis Brown for helping present and develop understanding of the material in this paper and I would also like to recognize Professor Iliadis for providing an interesting project and allowing us to learn broader knowledge than we would learn in many similar courses.

References

[1] Kurt Hornik.(1991). Approximation capabilities of multilayer feedforward networks.
Neural Networks,4(2) 251-257
https://doi.org/10.1016/0893-6080(91)90009-T

[2] D S Broomhead and D Lowe.(1988). Radial Basis Functions, Multi-variable Functional Interpolation and Adaptive Networks.
Royal Signals and Radar Establishment Memorandum 4148
https://apps.dtic.mil/dtic/tr/fulltext/u2/a196234.pdf

[3] Pfau et al. (2019). Ab-Initio Solution of the Many-Electron Schrödinger Equation with Deep Neural Networks.
https://arxiv.org/pdf/1909.02487.pdf

[4] Charlotte Froese Fischer. (1987). General Hartree-Fock program
Computer Physics Communications,43(3) 355-365
https://doi.org/10.1016/0010-4655(87)90053-1

[5] W.L. McMillan. (1965). Ground State of Liquid He^4
Physical Review 138, A442
https://doi.org/10.1103/PhysRev.138.A442

[6] Peiyuan Teng. (2018). Machine learning quantum mechanics: solving quantum mechanics problems using radial basis function network
https://arxiv.org/pdf/1710.03213.pdf

[7] Peter Atkins and Ronald Friedman. (2005). Molecular Quantum Mechanics, Fourth Edition.
Oxford University Press

[8] W.K. Hastings. (1970). Monte Carlo sampling methods using Markov chains and their applications
Biometrika, 57(1)
https://doi.org/10.1093/biomet/57.1.97