A solution to the quantum mechanical three-body problem with neural networks

Author: Marc Janer Serramià
Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.

Advisor: Vincent Mathieu and Arnau Rios

Abstract: We study how a one-layer Artificial Neural Network performs when applied to a quantum three-body problem with a central potential. We use the harmonic oscillator as a benchmark to test how the neural network solution performs upon changing parameters. With the best combination of these, we managed to get the ground state energy with a relative error of 0.03 % with respect to the analytical one, and a wave function with an overlap of the 99.996 % with the analytical solution.

I. INTRODUCTION

In the last decade, Machine Learning techniques and Artificial Neural Networks (ANNs) have gained a lot of popularity. Particularly in physics, these methods have been able to produce notably good results in a big variety of research fields [1]. These methods have become handy in problems where one needs to classify large amounts of data. ANNs can be used to minimize functions where the solution is unknown. This is helpful when solving quantum many-body problems as we want to minimize the energy [2, 3]. Inspired by Ref. [4], where the binding energy of the deuteron is found, we add one particle more and solve a quantum three-body problem using similar methods. Three-body problems are important in nuclear physics, as there are three-particle nuclei such as tritium or helium-3, but also they are important in subatomic particle physics in the study of baryons formed by 3 quarks [5].

A. Physical problem

First, we shall introduce the Hamiltonian for a three-body system, which consists of a kinetic part and a potential,

$$\mathcal{H} = -\frac{\hbar^2}{2m_1}\nabla^2_{\vec{r}_1} - \frac{\hbar^2}{2m_2}\nabla^2_{\vec{r}_2} - \frac{\hbar^2}{2m_3}\nabla^2_{\vec{r}_3} + V(\vec{r_1},\vec{r_2},\vec{r_3}) \ , \tag{1}$$

where m_i are the masses of particles $i = 1, \dots, 3$ and $\nabla_{\vec{r}_i}$ the Laplacian with respect to the coordinate $\vec{r}_i$. Eq. (1) is a problem of 9 variables, but since we are only interested in the relative motion of the three particles, we can reduce it to 6 by doing the following change of variables:

$$\vec{x} = \vec{r_1} - \vec{r_2} \qquad \vec{y} = \frac{m_1\vec{r_1} + m_2\vec{r_2}}{m_1 + m_2} - \vec{r_3} \ . \tag{2}$$

Ignoring the kinetic term of the global center of mass of the system we end up with:

$$\mathcal{H} = -\frac{\hbar^2}{2\mu_x}\nabla^2_{\vec{x}} - \frac{\hbar^2}{2\mu_y}\nabla^2_{\vec{y}} + V(\vec{x},\vec{y}) \ , \tag{3}$$

where the reduced masses are defined as,

$$\mu_x = \left(\frac{1}{m_1} + \frac{1}{m_2}\right)^{-1} \qquad \mu_y = \left(\frac{1}{m_1 + m_2} + \frac{1}{m_3}\right)^{-1} . \tag{4}$$

In order to make the problem simpler, we consider a central potential. Which in this case, is a potential independent of the directions of x and y. With this in mind, our goal is to find the ground state of the system given a potential. This can be solved with the variational method by minimizing the energy of the quantum system. Because the potential does not depend on the directions of x and y, the solution will be proportional to the spherical harmonics, so the wave function can be written as,

$$\psi(\vec{x},\vec{y}) = \frac{u(x,y)}{xy} Y_{l_x,m_x}(\Omega_x) Y_{l_y,m_y}(\Omega_y) \ , \tag{5}$$

where l_i, m_i are the angular momentum quantum numbers and Ω_i is the spherical angle for $i = x$ and $i = y$; x is the modulus of $\vec{x}$ and the same for y. We introduced the reduced radial wave function $u(x,y) = xyR(x,y)$. When using this, we have to impose the boundary conditions $u(x,0) = u(0,y) = 0$. This function is important because it simplifies the radial part of the Laplacian in Eq. (3) to a single second derivative. The angular part of the Laplacian applied to the spherical harmonics gives us $\nabla^2_\Omega Y(\Omega) = l(l+1)Y(\Omega)$. Applying the Hamiltonian to the wave function we get:

$$\hat{\mathcal{H}}\psi(\vec{x},\vec{y}) = \Big[-\frac{1}{2\mu_x}\frac{\partial^2}{\partial x^2} - \frac{1}{2\mu_y}\frac{\partial^2}{\partial y^2} + V_{eff}(x,y)\Big] \tag{6}$$
$$u(x,y) Y_x(\Omega_x) Y_y(\Omega_y) \ ,$$

where the effective potential includes the angular momentum $V_{eff}(x,y) = l_x(l_x+1) + l_y(l_y+1) + V(x,y)$. Now, we can find the functional of the energy,

$$E = \frac{\langle\psi|\hat{\mathcal{H}}|\psi\rangle}{\langle\psi|\psi\rangle} = \frac{\int_0^\infty\int_0^\infty u(x,y)\hat{\mathcal{H}}u(x,y)dxdy}{\int_0^\infty\int_0^\infty u(x,y)u(x,y)dxdy} \ . \tag{7}$$

With this simplification, we end up with a problem of finding a 2 variable function $u(x,y)$ which minimizes the energy of the three-body system.

We want to study if minimizing Eq. (7) via Neural Networks is doable, so we are going to use a potential with a known analytical solution,

$$V(x,y) = \frac{1}{2}\mu_x\omega_x^2 x^2 + \frac{1}{2}\mu_y\omega_y^2 y^2 \ . \tag{8}$$

The analytical solution of this problem is the tensor product of two isotropic 3d harmonic oscillators, one for x and one for y. The individual Hamiltonian $\mathcal{H} = -\frac{\hbar^2}{2m}\nabla_{\vec{r}}^2 + \frac{1}{2}\mu\omega^2 r^2$ is solved via separation of variables and the energy solution is $E = \hbar\omega\left(k + l + \frac{3}{2}\right)$, where $k = 0, 2, 4, \ldots$ and $l = 0, 1, 2, \ldots$. The analytical wave function for the ground state ($k = 0$ and $l = 0$) for one isotropic oscillator is

$$\psi_{000}(\vec{r}) = 2\left(\frac{1}{\pi}\left(\frac{\mu\omega}{\hbar}\right)^3\right)^{1/4} e^{-\frac{1}{2}\frac{\mu\omega}{\hbar}r^2} Y_{00}(\Omega) \ . \tag{9}$$

As we said, the solution to our problem is the tensor product of the two individual 3d harmonic oscillators. Consequently, the final energy is the sum of the energies of the individual problems

$$E_{k_x l_x k_y l_y} = \hbar\left[\omega_x\left(k_x + l_x + \frac{3}{2}\right) + \omega_y\left(k_y + l_y + \frac{3}{2}\right)\right] \ , \tag{10}$$

and the global wave function is the product of two Eq. (9), one for x and one for y. The solution to this problem can be found in Refs. [6, 7].

II. NUMERICAL METHODS

In this section, we explain all the methods used to find the solution to the problem. The most important part of this work is the Artificial Neural Network (ANN). It is used in a big variety of problems where the objective is to minimize the so-called loss function, which in our case is Eq. (7).

A. Artificial Neural Networks

Our ANN consists of three parts: an input layer, a hidden layer (formed by what we call hidden neurons) and an output layer, the architecture is shown in Fig. 1. For each connection we have an arbitrary parameter, we call the collection of parameters that connect the input layer with the hidden layer $\mathcal{W}^{(1)}$ and those that connect the hidden layer with the output layer $\mathcal{W}^{(2)}$, these objects are called weights.

This ANN architecture can be written mathematically as:

$$\phi_{ANN}(X) = \sum_{i=1}^{N_{hid}} \mathcal{W}_i^{(2)}\sigma\left(\sum_{j=1}^{2}\mathcal{W}_{ij}^{(1)}X_j + b_i\right) \ , \tag{11}$$

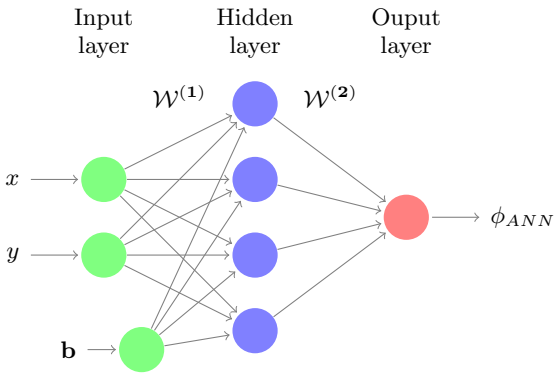

FIG. 1: The Neural network architecture that we use with a number of hidden neurons of $N_{hid} = 4$. The two inputs are the variables x and y. There is a bias $\mathbf{b}$ added to the hidden layer. It has one output ϕ_{ANN}.

where $\vec{X}$ is a vector that includes the inputs $X_1 = x$ and $X_2 = y$; σ is the activation function, which in our case we use one called Softplus $\sigma(x) = \ln(1 + \exp(x))$; and we introduce a bias $\mathbf{b}$ in the hidden layer.

In our problem, the ANN takes the role of the reduced radial wave function. This function has to be zero when $x = 0$ or $y = 0$ in order to satisfy boundary conditions $u(x,0) = u(0,y) = 0$. With this in mind, we employ as the reduced radial function the ANN times a function that applies the boundary conditions

$$u(x,y) = f(x,y)\phi_{ANN}(x,y) \ , \tag{12}$$
$$f(x,y) = xy(x - x_1)(y - y_1) \ , \tag{13}$$

where x_1 and y_1 are the limits of the area where we are computing the energy, $x \in (0, x_1)$ and $y \in (0, y_1)$. The factor $(x - x_1)(y - y_1)$ is added because we want the function to go to 0 when approaching x_1 and y_1.

B. Training

Our ANN has $4N_{hid}$ parameters which are included in $\mathcal{W}^{(1)}$, $\mathbf{b}$ and $\mathcal{W}^{(2)}$. At the start, these parameters are selected random between $-1/\sqrt{2}$ and $1/\sqrt{2}$ uniformly. Our objective is to train our ANN to find a good combination of parameters that minimizes our loss function Eq. (7). To do it, what is used is an optimizer, which is the algorithm used to update the weights in order to decrease the loss function. The idea behind them is to find the gradient of the loss function with respect to the parameters and make a step in the direction of the negative gradient. The size of this step is determined by the learning rate. The optimizers that we use in this work are RMSprop [8] and Adam [9].

In our case we initialize a grid of (x, y) coordinates with $N \times N$ points evenly spaced in the range $x \in (0, x_1)$ and $y \in (0, y_1)$. In our training, first we pass all the

points in the grid through the ANN; with the outputs, we compute the loss function Eq. (7) by integrating over all the grid; and lastly we update the weights via the optimizer. We call one iteration passing over all these actions one time. To implement all of this we use the PyTorch library [10].

In order to compute the loss function we have to first compute a second derivative with respect to the inputs of the network and later compute two 2d integrals Eq. (7). Since we are using a discrete grid we need to use numerical methods for derivation and integration. For the derivation, we use a 3-point central second derivative and Simpson's rule for the integrals. Because of Simpson's rule, we have to use an even number of intervals, hence an odd number of points.

III. RESULTS

In order to deal with the uncertainty of the grid, the energy benchmark employed in the results is computed numerically by applying the analytical wave function into Eq. (7) in the $N \times N$ grid. For increasing values of N, the energy benchmark approaches the analytical solution. The analytical wave function is the multiplication of Eq. (9) for x and the same function for y.

Before starting with the results, we choose some specific values to test whether the network can deal with asymmetric conditions along x and y axis:

$$\mu_x = \mu \qquad \mu_y = 2\mu \ , \tag{14}$$

$$\omega_x = 3\omega \qquad \omega_y = \omega \ . \tag{15}$$

We want to find the energy of the ground state so $k = 0$ and $l_x, l_y = 0$. Taking all these variables into Eq. (10) we get an analytical energy of $E = 6\ \hbar\omega$.

The range of coordinates that we use in our grid is $x \in (0,3)r_0$ and $y \in (0,3)r_0$, where $r_0 = \sqrt{\frac{\hbar}{\mu\omega}}$ is the natural length unit of the harmonic oscillator. With the energy as the loss function we get a decent result in approximately 5000-10000 iterations, this depends on the parameters: number of hidden neurons, learning rate and size of the grid. We study these parameters individually in order to see which are the best to choose.

A. Learning rate

The learning rate is the value that controls the rate at which the weights change from iteration t through $t+1$. We expect that a high learning rate can be inadequate because it can jump the minimum whereas a low learning rate can take a lot more time to converge. We carry out some tests for a grid of 51×51 points with 4 hidden neurons $N_{hid} = 4$ for the 2 optimizers Adam and RMSprop.

Fig 2 shows the energy associated to the ANN ansatz as a function of the iteration number. The energy benchmark is represented with a dashed line.

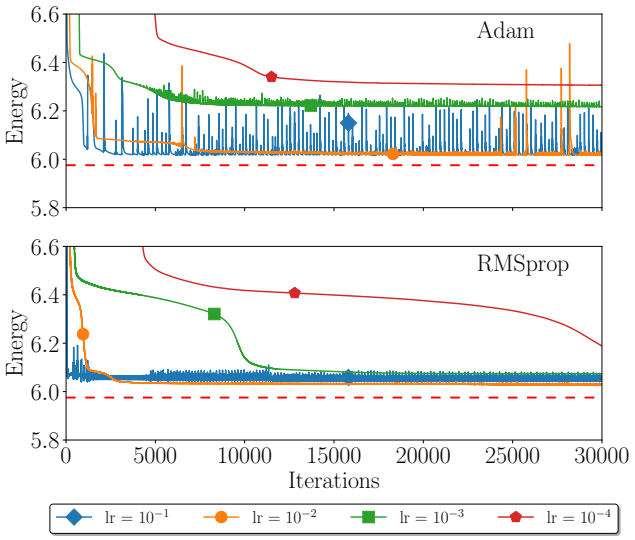

FIG. 2: Energy of the ANN ansatz as a function of the iteration number for different learning rates in a grid of 51 x 51 and $N_{hid} = 4$. Upper panel shows Adam optimizer and lower panel RMSprop. The dashed line represents the energy benchmark.

We can see in Fig. 2 that small learning rates (10^{-3} and 10^{-4}) are very slow, with 30000 iterations the last one has not yet converged. For large learning rates like 10^{-1} the solution fluctuates, this could be because the step is big enough to make the ANN jump around the minimum. Comparing between optimizers: for the best learning rate, we get almost the same results with differences of approximately 1 % between optimizers. For large learning rates, Adam oscillates a lot more than RMSprop. In both cases the best learning rate is 10^{-2}, so that is the one that we use in the following sections.

B. Hidden layer size

Changing the number of hidden neurons changes a lot the number of parameters of the ANN. If N_{hid} is raised we expect to have better results but it also means that the time of computation is increased. We execute some tests similar to the previous subsection, with a grid of 51×51 points and a learning rate of 10^{-2} for both optimizers.

We can see in Fig. 3 that as expected, the solution tends to be better if we increase the number of hidden neurons. As with the learning rate, we get similar results for both optimizers and Adam fluctuates more than RMSprop. When using the RMSprop for higher values of N_{hid} than the ones represented in Fig. 3, the ANN creates discontinuities that generate a wrong result, this is why we use Adam in the following sections.

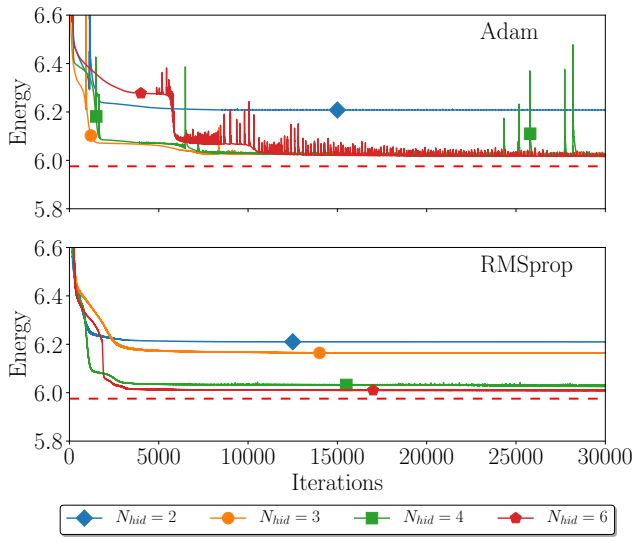

FIG. 3: The same as Fig. 2 but for a fixed learning rate $lr = 10^{-2}$ and different hidden layer sizes N_{hid}.

C. Size of the grid

Now we want to see how the result varies when increasing the number of points in the grid, obviously an increase of this type increases the computation time, but it should get better results. This time we only used the Adam optimizer with $N_{hid} = 8$ and a learning rate of 10^{-2}.

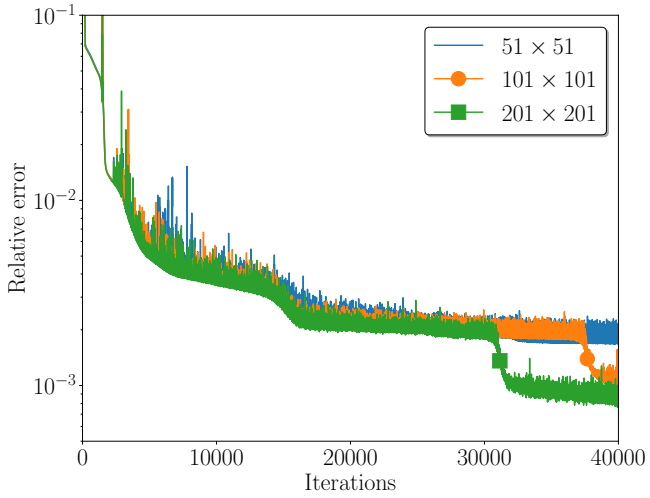

FIG. 4: Comparison of the relative error for a different number of points in the grid on Adam optimizer with $N_{hid} = 8$ and a learning rate of 10^{-2}.

Fig. 4 represents the relative error as a function of the number of iterations for a different number of points in the grid $N \times N$. The relative error is calculated by comparing it with the corresponding benchmark energy of each grid. In Fig. 4 we can see that for 30000 iterations the error is almost the same in the 3 grids, but in the following iterations, the error in the 201×201 grid decreases, and the same happens later for the 101×101 grid. Seeing the end results it is clear that for more points one gets a better result.

D. Final result

Now with all the information we gathered, we train the ANN with the best parameters that we found for a large number of iterations and analyze the result. The result is obtained by training 5 times a new ANN with different initial parameters. For each training, we gather the energy of the last 10000 iterations. From all the 5×10000 values obtained, we calculate the mean E, the difference between the biggest value and the mean δE_+, the difference between the mean and the lowest value δE_- and the standard deviation σ. Each training is done in a 101×101 grid with Adam optimizer and a learning rate of 10^{-2} for 100000 iterations. We do this for larger values of N_{hid} than in the previous subsection III B to see how the energy and uncertainty decrease in this range.

TABLE I: Final energies obtained by the process described previously for different sizes of the hidden layer N_{hid}, $\hbar\omega$ are the energy units.

N_{hid}	E	δE_-	δE_+	σ
5	6.011	0.005	0.007	0.002
10	6.005	0.005	0.009	0.004
15	6.0000	0.0009	0.0040	0.0005

In Table I we can see the different values obtained of E, δE_-, δE_+ and σ for different values of N_{hid}. For $N_{hid} = 15$ we get a result with a relative error of 0.03 % comparing it to the energy benchmark for the 101×101 grid $E = 5.99846\ \hbar\omega$. The uncertainties decrease considerably between $N_{hid} = 10$ and $N_{hid} = 15$.

In Fig. 5, the left panel shows the representation of the analytical wave function, in the right panel is represented the difference between the analytical function and one of the previously trained ANN with $N_{hid} = 15$. The differences between wave functions are minimal, computing the overlap we get a result of $\langle \psi_{ANN} | \psi_{real} \rangle = 0.99996$.

IV. CONCLUSIONS

In this work, we have been able to find the ground state energy of a quantum mechanical three-body problem with a harmonic oscillator potential Eq. (8) with an ANN of one hidden layer. Making the computational parameters more suitable for our problem we managed to get an energy result of $E = 6.0000\ \hbar\omega$ with a standard deviation of $\sigma = 0.0005\ \hbar\omega$, compared to the energy benchmark of $5.99846\ \hbar\omega$ which comes

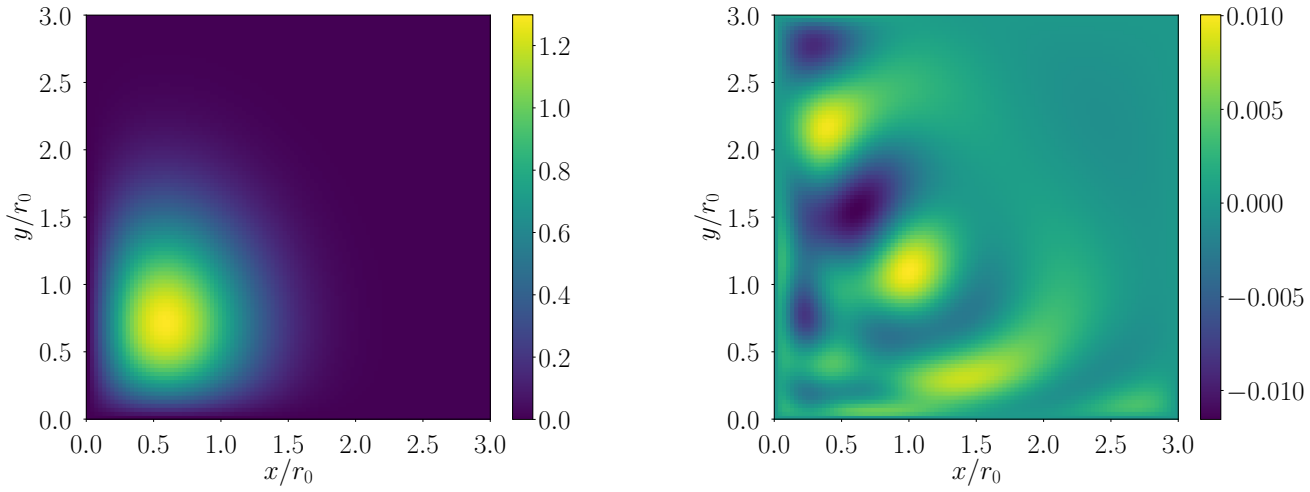

FIG. 5: Left panel: representation of the analytical reduced radial wave function (multiplication of two Eq. (9)) normalized to 1. Right panel: representation of the difference between the analytical function and the ANN, both normalized to 1. ANN trained with Adam for $N_{hid} = 15$, learning rate 10^{-2} and with a grid of 101×101 points for 100000 iterations.

from computing the energy in the same discrete grid (explained at the start of section III), makes a relative error of 0.03 %. The overlap between the wave function from the ANN and the analytical one gives a value of $\langle \psi_{ANN} | \psi_{real} \rangle = 0.99996$.

With respect to the work on [4] we increased one dimension to the problem, we found out that with this change, it is necessary to apply Eq. (13) multiplying the ANN in order to satisfy boundary conditions.

We also did a study on the different computational parameters of the problem: learning rate, hidden layer size and grid size. We compared the results between two different optimizers and found out that for our problem the results of both optimizers with a learning rate of 10^{-2} have differences of approximately 1 %. We also saw how the error decreases when rising the grid size, this decrease only happens after a certain amount of iterations, in our case are 30000.

It is necessary to emphasize that the program made to solve this problem not only works for a harmonic potential but for any central potential that can be thought of. The code can be found on `https://github.com/Marcjaser/threebodyANN`.

This work could be extended in multiple ways. One way is trying to make the program capable of working with non-central potentials. Another option is changing to other numerical methods for derivation and integration, for example implementing Montecarlo integration in order to not have a fixed grid and make the training more focused on the more complex parts of the wave function.

Acknowledgments

I want to thank my advisors Vincent Mathieu and Arnau Rios for all the guidance and help that they have given me throughout this work. I would also like to thank my family for their daily support.

[1] G. Carleo et al. Machine learning and the physical sciences. *Rev. Mod. Phys.*, 91(4):045002, 2019.

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

[3] D. Pfau et al. Ab initio solution of the many-electron schrödinger equation with deep neural networks. *Phys. Rev. Research*, 2:033429, Sep 2020.

[4] J.W.T. Keeble and A. Rios. Machine learning the deuteron. *Phys. Lett. B*, 809, 2020.

[5] Y. Suzuki and K. Varga. *Stochastic variational approach to quantum-mechanical few-body problems*. Springer, Berlin, 1998.

[6] C. Cohen-Tannoudji, B. Diu, and F. Laloe. *Quantum Mechanics.*, volume 1, chapter VII Complement B. Wiley, 2005.

[7] A. Messiah. *Quantum Mechanics.*, volume 1, chapter XII. North-Holland Publishing Company, 1961.

[8] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop. 2012. Unpublished method proposed in COURSERA: Neural networks for machine learning. URL `https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf`.

[9] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014, arXiv:1412.6980.

[10] A. Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach et al., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.