Physics-Informed Neural Networks for Quantum Eigenvalue Problems

Henry Jin
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA, USA
helinjin@g.harvard.edu

Marios Mattheakis
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA, USA
mariosmat@seas.harvard.edu

Pavlos Protopapas
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA, USA
pavlos@seas.harvard.edu

***Abstract*—Eigenvalue problems are critical to several fields of science and engineering. We expand on the method of using unsupervised neural networks for discovering eigenfunctions and eigenvalues for differential eigenvalue problems. The obtained solutions are given in an analytical and differentiable form that identically satisfies the desired boundary conditions. The network optimization is data-free and depends solely on the predictions of the neural network. We introduce two physics-informed loss functions. The first, called *ortho-loss*, motivates the network to discover pair-wise orthogonal eigenfunctions. The second loss term, called *norm-loss*, requests the discovery of normalized eigenfunctions and is used to avoid trivial solutions. We find that embedding even or odd symmetries to the neural network architecture further improves the convergence for relevant problems. Lastly, a patience condition can be used to automatically recognize eigenfunction solutions. This proposed unsupervised learning method is used to solve the finite well, multiple finite wells, and hydrogen atom eigenvalue quantum problems.**

***Index Terms*—neural networks, eigenvalue, eigenfunction, differential equation**

I. Introduction

Differential equations are prevalent in every field of science and engineering, ranging from physics to economics. Thus, extensive research has been done on developing numerical methods for solving differential equations. With the unprecedented availability of computational power, neural networks hold promise in redefining how computational problems are solved or improving existing numerical methods. Among other applications in scientific computing, neural networks are capable of efficiently solving differential equations [1]–[4].

These neural network solvers pose several advantages over numerical integrators: the obtained solutions are analytical and differentiable [3], numerical errors are not accumulated [4], networks are more robust against the 'curse of dimensionality' [5], [6], a family of solutions corresponding to different initial or boundary conditions can be constructed [7], the neural solutions can be transferred for fast discovery of new solutions [8], [9], inverse problems can be solved systematically [10], [11], and available data can be incorporated into the loss function to improve the network's performance [12].

Eigenvalue differential equations with certain boundary conditions appear in a wide range of problems of applied mathematics and physics, including quantum mechanics and electromagnetism. Lagaris et al. [1] have shown that neural networks are able to solve eigenvalue problems and proposed a partially iterative method that solves a differential equation with a fixed eigenvalue at each iteration. More recently, Li et al. [13] showed that neural networks can solve the stationary Schrödinger equation for systems of coupled quantum oscillators. This is a variational approach where the eigenvalue is indirectly calculated from the predicted eigenfunction. Our work expands on the unsupervised neural network eigenvalue solver presented by Jin et al. [14], which simultaneously and directly learns the eigenvalues and the associated eigenfunctions using a scanning mechanism. Here, we introduce physics-informed improvements to the regularization loss terms: orthogonal loss (ortho-loss) and normalization loss (norm-loss). We further design special neural network architectures with embedded symmetries that ensure the prediction of perfectly even or odd eigenfunctions. Furthermore, a modified parameterization is introduced to handle problems with non-zero boundary conditions. The proposed technique is an extension to physics-informed neural network differential equation solvers and, consequently, inherits all the benefits that neural network solvers have over numerical integrators. Moreover, our method has an additional advantage over integrators in that it discovers solutions that identically satisfy the boundary conditions. We assess the performance of the proposed architecture by solving a number of standard eigenvalue problems of quantum mechanics: the single finite square well, multiple finite square wells, and the hydrogen atom.

II. Background

This study extends the method presented in [14], where a fully connected neural network architecture was proposed,

with a single output corresponding to the predicted eigenfunction, and with a constant input node designed to learn constant eigenvalues through backpropagation. To identically satisfy the boundary conditions, a parametric function was used. In order for the network to find non-trivial solutions to the differential eigenvalue equation, the two regularization loss functions

$$L_f = \frac{1}{f(x,\lambda)^2}, \quad L_\lambda = \frac{1}{\lambda^2} \tag{1}$$

were used to penalize trivial eigenfunctions and zero eigenvalues, respectively. Moreover, a scanning mechanism allows the network to search the eigenvalue space for eigenfunctions of different eigenvalues, enabled by the loss term defined as

$$L_{\text{drive}} = e^{-\lambda+c}, \tag{2}$$

where c was a value that changed during training through scheduled increases, and was used to control the scanning.

The research by Li et al. [13] on neural network-based multi-state solvers is also relevant to this study. However, we present some novelties and differences in methodology. Specifically, we assign a trainable network parameter to discover the eigenvalue instead of indirectly calculating it through the expectation of the Hamiltonian of the system. Our approach avoids the repeated calculation of an integral (i.e., for the expectation value) which is evaluated every training epoch. The second novelty of our approach is the embedding of physical symmetries in the network architecture. The symmetry of the wavefunctions can be determined by the symmetry of the given potential function. We design a specialized architecture with embedded even or odd symmetry that significantly improves the overall network optimization. Finally, we suggest a parameterization that identically satisfies non-zero boundary conditions, which is necessary to solve the radial equation of the hydrogen atom.

Orthogonality loss is also used in [13], where it is leveraged to simultaneously produce multiple eigenvalue solution outputs that are pair-wise orthogonal. This differs from our method, since our neural network outputs one solution at a time, and the orthogonality loss term is used to prevent us from finding the same solution multiple times.

III. Methodology

We consider an eigenvalue problem that exhibits the form:

$$\mathcal{L}f(x) = \lambda f(x), \tag{3}$$

where x is the spatial variable, $\mathcal{L}$ is a differential operator that depends on x and its derivatives, $f(x)$ is the eigenfunction, and λ is the associated eigenvalue. For the finite square well problems, we assume homogeneous Dirichlet boundary conditions at the left and right boundaries x_L and x_R, respectively, such that $f(x_L) = f(x_R) = f_b$, where f_b is a given constant boundary value. On the other hand, for the hydrogen atom problem, a single Dirichlet boundary condition $f(x_R) = f_b$ is enforced.

We expand on the network architecture proposed by [14] and shown in Fig. 1. This feed-forward neural network is capable of solving Eq. (3) when both $f(x)$ and λ are unknown. The network takes two inputs, the variable x and a constant input of ones. The constant input feeds into a single linear neuron (affine transformation) that is updated through optimization, allowing the network to find a constant λ. Afterwards, x and λ are inputs to a fully-connected feed-forward neural network that returns an output function $N(x,\lambda)$. The predicted eigenfunctions $f(x,\lambda)$ are defined using a parametric trick, similar to [4], according to the equation:

$$f(x,\lambda) = f_b + g(x)N(x,\lambda). \tag{4}$$

By choosing an appropriate $g(x)$, the predicted eigenfunction identically satisfies certain boundary conditions.

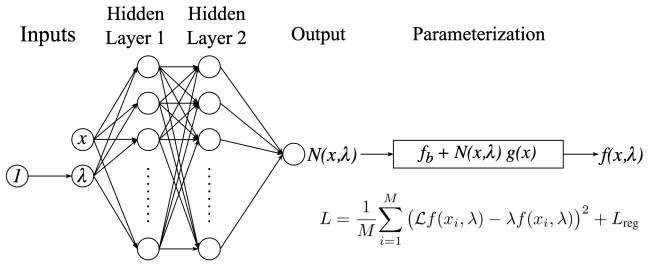

Fig. 1: Physics-informed neural architecture for solving eigenvalue problems.

Our aim is to discover pairs of $f(x,\lambda)$ and λ that approximately satisfy Eq. (3). This is achieved by minimizing, during the network optimization, a loss function L defined by Eq. (3) as:

$$\begin{aligned} L &= L_{\text{DE}} + L_{\text{reg}} \\ L &= \frac{1}{M}\sum_{i=1}^{M}\left(\mathcal{L}f(x_i,\lambda) - \lambda f(x_i,\lambda)\right)^2 + L_{\text{reg}}, \end{aligned} \tag{5}$$

where averaging with respect to x_i takes place in L_{DE} for M training sample points, namely $x = (x_1, \cdots, x_M)$. Any derivative with respect to x_i contained in $\mathcal{L}$ is calculated by using the auto-differentiation technique [15]. The L_{reg} term in Eq. (5) contains regularization loss terms. In this work, we introduce and apply a regularization function that consists of three terms of the form: $L_{\text{reg}} = \nu_{\text{norm}}L_{\text{norm}} + \nu_{\text{orth}}L_{\text{orth}} + \nu_{\text{drive}}L_{\text{drive}}$. Empirically, for the problems discussed below, we found the optimal regularization coefficients $\nu_{\text{norm}} = \nu_{\text{orth}} = 1$. The normalization loss L_{norm} encourages normalized eigenfunctions, avoiding the discovery of trivial eigenfunctions and eigenvalues, since it enforces non-zero solution as well as constraining the eigenfunction's squared integral to be finite. The L_{orth} motivates the network to scan for orthogonal eigenfunctions and can replace or assist the non-physical scanning (L_{drive}) method used in [14]. L_{drive} accounts to the scanning method which is used to guide the model's eigenvalue weight and is given by Eq. 2. However, for the experiments presented in this study, we use L_{orth} as a physics-informed regularization term that can replace the non-physical scanning method with L_{drive}, and thus ν_{norm} is set to 0.

A. Normalization Loss

Our contribution includes a novel approach to solving the trivial solution problem. While [14] employed non-trivial eigenfunction and non-trivial eigenvalue loss terms L_f and L_λ, as described in Eq. 2, these loss terms cannot numerically converge to 0 without scaling the solutions to infinity, and thus they introduce numerical error. While they were effective for preventing the network from converging to trivial $f(x)$ and λ, they hold no physical meaning. We present a physics-aware regularization loss function that not only prevents trivial solutions, but also motivates the eigenfunction's inner product with itself to approach a specific constant number, which is the normalization constraint physically required of eigenfunctions in quantum mechanics. Thus, L_{norm} is given by

$$L_{\text{norm}} = \left(f(x,\lambda) \cdot f(x,\lambda) - \frac{M}{x_R - x_L} \right)^2, \tag{6}$$

where dot denotes the inner product. The loss function in Eq. (6) drives the network to find solutions with non-zero integrals, where $f(x,\lambda)$ represents the network solution, M is the number of samples, and $x_R - x_L$ is the training range. Specifically, this motivates the network solution to have a squared integral equal to one. Unlike L_f and L_λ, L_{norm} can strictly reach zero and can also satisfy the normality constraint for eigenfunction solutions of Schrodinger's equation.

B. Orthogonality Loss

An orthogonality loss regularization function is included as part of L_{reg} to motivate the network to find different eigen-solutions to Schrodinger equation. This presents a physics-informed approach whereby we can motivate a network to solve for orthogonal solutions for problems where it is known that solutions are orthogonal, a fundamental property of linear differential eigenvalue problems. Schrodinger's equation is one such example, but this mechanism can be extended to any Hermitian operator. This serves as a replacement or an improvement over solely relying on the scanning mechanism L_{drive} presented in [14]. While a scanning search through the eigenvalue space using L_{drive} can be useful for providing control over the model's search for eigenfunction solutions, solving equations that are known to be Hermitian (such as the Schrodinger equation) allows the use of an orthogonal loss term, since eigenfunctions of Hermitian operators are orthogonal. In this paper, we show that the neural network is able to find orthogonal eigenfunction solutions solely based on the orthogonality loss. This loss term is given by the following equation.

$$L_{\text{orth}} = \psi_{\text{eigen}} \cdot \psi, \tag{7}$$

where ψ_{eigen} denotes the sum of all eigenfunctions that have already been discovered by the network during training, and ψ is the current network prediction. This regularization term embeds the network with a physics-informed predisposition towards finding orthogonal solutions to a Hermitian operator, serving as a more physics-aware loss term than the brute-force scanning approach.

Following the network's convergence to a new solution, the new eigenfunction is added to ψ_{eigen} and thus, it is the linear combination of all the discovered solutions. Hence, a single orthogonality loss term is computed for each learning gradient, as opposed to separate orthogonality computations for each learned eigenfunction. This reduces computational cost since only one dot product is computed for each training iteration, as opposed to multiple dot products with each found eigenfunction.

C. Embedding Even and Odd Symmetry

For certain differential equations where prior information about the potential dictates even or odd symmetric eigenfunctions, the neural network architecture can be embedded with a physics-informed modification that enforces the correct symmetry in the eigenfunction output. As demonstrated by Mattheakis et al. in [16] and extended by [17], symmetry can be embedded by feeding a negated input stream in parallel to the original input, then combining streams before the final dense layer. Adding streams leads to even symmetric outputs, while subtracting gives rise to odd symmetric predictions.

We found that embedding symmetry into our model significantly accelerates the convergence to a solution. This is relevant for the multiple finite square wells problem, as we demonstrate below.

D. Parametric Function

Selecting an appropriate parametric function $g(x)$ is necessary for enforcing boundary conditions. The following parametric equation enforces a $f(x_L) = f(x_R) = 0$ boundary conditions:

$$g(x) = \left(1 - e^{-(x-x_L)}\right)\left(1 - e^{-(x-x_R)}\right). \tag{8}$$

As demonstrated in [14], this parametric function is suitable for problems where the eigenfunctions are fixed to or converge to zero, as in the case of the infinite square well and the harmonic oscillator problems. In the following experiments, we employ the parametric function of Eq. (8) for finite square well problems, as they similarly require eigenfunctions to taper to zero at domain limits.

The differential eigenvalue equation for the hydrogen atom, however, has a single zero boundary condition at $x \to \infty$, as the fundamental solution is not fixed to 0 at the origin. For such problems where a single Dirichlet boundary condition is required, we use the following parametric function:

$$g(x) = \left(1 - e^{-(x-x_R)}\right). \tag{9}$$

E. Towards Solution Recognition

To automatically extract the correct eigenfunctions, we define convergence to an eigenfunction solution using two criteria: the differential equation loss L_{DE} and patience.

L_{DE} describes the loss term for the differential eigenvalue equation in question. For our experiments, without loss of

generality, we used Schrodinger's equation. Nevertheless, the method is valid for any differential equation eigenvalue problem. Considering that perfect eigenvalue solutions will have an L_{DE} loss equal to zero, we claim that a solution is found when L_{DE} falls below a chosen threshold, which is a hyper-parameter in the training process.

The patience condition describes the model's training progress. When solving for a solution, the model initially improves very quickly, resulting in a fast decrease of L_{DE}. However, over the course of converging to a solution, the rate of decrease in L_{DE} decreases as well. Thus, we use the rate of decrease in L_{DE} as another condition for solution recognition. If the rolling average during the training iterations of the successive differences in L_{DE} over a specified window hyper-parameter falls below a chosen threshold hyper-parameter, we consider the patience condition to be met.

When both the L_{DE} condition (L_{DE} falling below a threshold) and the patience condition are satisfied, we consider an eigenvalue solution to have been found. On the other hand, if only the patience condition is satisfied, then we interpret this to mean that the model has converged to a false solution. Consequently, we switch the symmetry (from even to odd symmetry or vice versa) of the model to motivate the network to search for other solutions. This approach of switching the symmetry of the model upon converging to a false solution was inspired by our finding that the network's function output after converging to false solutions resembled true solutions, but of the opposite symmetry. Upon adopting this switching approach, we found that the model was able to resume finding true solutions. The above method is described by Algorithm 1.

Algorithm 1 The Physics-Informed Neural Eigenvalue Solver Algorithm

```
Instantiate model with even symmetry
while training do
    Generate training samples x_i
    Compute L_DE, L_norm
    Compute L_orth using all stored eigenfunctions
    Backpropagate and step
    if patience condition and L_DE < threshold then
        Store copy of model
    else if patience condition then
        Switch model symmetry
    end if
end while
```

IV. Experiments

We evaluate the effectiveness of the proposed method by solving eigenvalue problems defined by Schrodinger's equation. Schrodinger's equation is the fundamental equation in quantum mechanics that describes the state wavefunction $\psi(x)$ and the associated energy E of a quantum system. In this study, we are interested in solving the one-dimensional stationary Schrodinger's equation defined as:

$$\left[-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x)\right]\psi(x) = E\psi(x), \tag{10}$$

where $\hbar$ and m stand for the reduced Planck constant and the mass respectively, which without loss of generality, can be set to $\hbar = m = 1$. Equation (10) defines an eigenvalue problem where $\psi(x)$ and E denote the eigenfunction $f(x, \lambda)$ and eigenvalue λ pair. The differential equation loss for this one-dimensional stationary Schrodinger's equation is given by Equation (11), and henceforth we call this the Schrodinger equation loss.

$$L_{\text{DE}} = \frac{1}{M}\sum_{i=1}^{M}\Big(\left[-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x_i)\right]f(x_i, E) - Ef(x_i, E)\Big)^2. \tag{11}$$

A boundary condition eigenvalue problem is defined by considering a certain potential function $V(x)$ and boundary conditions for $\psi(x)$. We assess the performance of the proposed network architecture by solving Eq. (10) for the potential functions of the single finite well, multiple coupled finite wells, and the radial equation for the hydrogen atom, all of which have known analytical solutions.

For the training, a batch of x_i points in the interval $[x_L, x_R]$ is selected as input. In every training iteration (epoch) the input points are perturbed by a Gaussian noise to prevent the network from learning the solutions only at fixed points. Adam optimizer is used with a learning rate of $8 \cdot 10^{-3}$. We use two hidden layers of 50 neurons per layer with trigonometric $\sin(\cdot)$ activation function. The use of $\sin(\cdot)$ instead of more common activation functions, such as Sigmoid$(\cdot)$ and $\tanh(\cdot)$, significantly accelerates the network's convergence to a solution [4]. We implemented the proposed neural network in pytorch [15] and published the code on github[1].

A. Single Finite Well

The finite well potential function is defined as:

$$V(x) = \begin{cases} 0 & 0 \le x \le \ell \\ V_0 & \text{otherwise} \end{cases}, \tag{12}$$

where ℓ is the length and V_0 is the depth of the quantum well. The analytical solution to the finite well problem is traditionally found by solving the stationary Schrodinger's equation in each region, then 'stitching' the solutions of each region together while enforcing a continuous eigenfunction that is also continuously differentiable. For bound eigenfunctions, the general form of the solution for regions where the eigenvalue E is greater than the potential reads:

$$\psi = A\sin(kx) + B\cos(kx), \quad k = \frac{\sqrt{2mE}}{\hbar}. \tag{13}$$

[1] https://github.com/henry1jin/quantumNN

For regions where the eigenvalue E is smaller than the potential energy, the solution's general form is

$$\psi = Ce^{-\alpha x} + De^{\alpha x}, \quad \alpha = \frac{\sqrt{2m(V_0 - E)}}{\hbar}. \tag{14}$$

The solutions then for Eq. (12) is the following piece-wise eigenfunction, where constants c_1, c_2, and δ, are determined by the requirement that the eigenfunction is continuous, continuously differentiable, and normalized.

$$\psi(x) = \begin{cases} c_1 e^{\alpha x} & x \leq 0, \\ c_2 \sin(kx + \delta) & 0 < x \leq \ell, \\ c_1 e^{-\alpha x} & x > \ell \end{cases} \tag{15}$$

The $\psi(x)$ eigenfunctions must decay to infinity outside the walls, implying the boundary conditions $\psi(-\infty) = \psi(\infty) = 0$. In numerical methods, infinity is approximated with large values relative to the potentials. We adopt the approximate boundary conditions of $\psi(x_L) = \psi(x_R) = 0$ with the choice $x_L = x_R = 6\ell$, for $\ell = 1$ and $V_0 = 20$. The proposed model with the orthogonal loss term is capable of solving for all bound eigenstates. In the following we use the neural network to approximate the first four eigenfunctions and the associated energies.

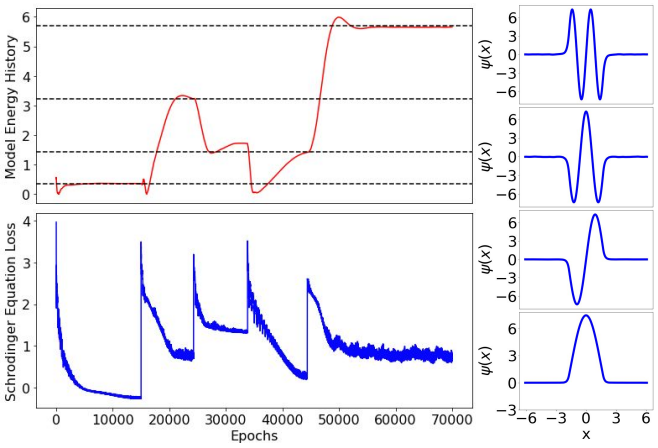

Fig. 2: The plot on the upper left displays the model's eigenvalue weight (i.e. energy) during the training process. Horizontal dashed black lines demarcate the true eigenvalues, which our network accurately finds (plateaus of red lines). The lower left plot shows the corresponding Schrodinger equation loss L_{DE} over epochs during training. At 15000 epochs, orthogonal loss with the first eigenfunction is introduced. Each following spike in L_{DE} indicates the point where the model reaches the patience condition, and the last eigenfunction is added to the orthogonal loss term. Column of plots on the right are the resulting eigenfunctions that the model finds when the predicted energy converges to a plateau.

We start the network optimization by using a neural network with even symmetry embedded. Figure 2 summarizes the results for the discovery of the first four eigenstates of the quantum finite well. The lower left panel outlines the L_{DE} during the training. The red curve in the upper left graph demonstrates the predicted energies where the plateaus indicate the discovery of an eigenstate; the dashed black lines show the ground truth energies. On the right side, the four predicted eigenfunctions are represented by blue lines; the bottom graph corresponds to the ground state. In particular, the neural network finds for the ground state solution with energy $E = 0.3586$. After the first solution is found, we introduce the orthogonal loss term into the training, motivating the network to find a new eigenfunction. Consequently, the eigenvalue weight departs from its first value and rises to find the next even-symmetry solution with eigenvalue $E = 3.2132$ (the third graph on the right side in Fig. 2, counting from the bottom). Once the patience condition is reached, the network automatically adds the latest solution to the orthogonal loss, motivating the network to once again depart its solution in search of the next orthogonal solution. The model converges to an eigenvalue of around $E = 1.8$, however it does not meet both conditions for solution acceptance. In particular, it does not meet the L_{DE} condition. We take this to mean that, while the model has converged, it has converged to a false solution. So the symmetry of the model is switched to odd symmetry. The next two solutions found are odd-symmetric and correspond to the eigenvalues of $E = 1.4322$ and $E = 5.6873$ shown respectively by the second and fourth images in the right panel of Fig. 2.

B. Multiple Finite Square Wells

The single finite well potential can be repeatedly spaced to create a potential function that consists of multiple square wells as follows:

$$V(x) = \begin{cases} 0 & 2n\ell \leq x \leq (2n+1)\ell \\ V_0 & \text{otherwise} \end{cases}, \tag{16}$$

where n is an element of a subset of nonzero integers.

Like the single finite well, solutions to the multiple square wells are piece-wise constructed by solving for each discrete region and "stitching" solutions. The general forms of the solutions in each region, namely, inside and outside a well, are once again given by Eq. (13) and Eq. (14), respectively. The boundary conditions at infinity are also approximated by large values of x relative to the potential, that is, $\psi(x \to \pm\infty) = 0$.

Our deep learning technique applied to the multiple wells solves for an arbitrary number of the solutions. Figure 3 shows our neural network finding the four lowest-energy (i.e. lowest-eigenvalue) states of the double finite square well. Similar to the single finite well problem, the model here uses the physics-informed approach of solving for orthogonal eigenfunctions with the orthogonal loss term, given the knowledge that solutions to Hermitian differential operators must be orthogonal.

C. Symmetry vs No Symmetry

Embedding symmetry into the network for problems where the solutions are known to be either even or odd symmetric

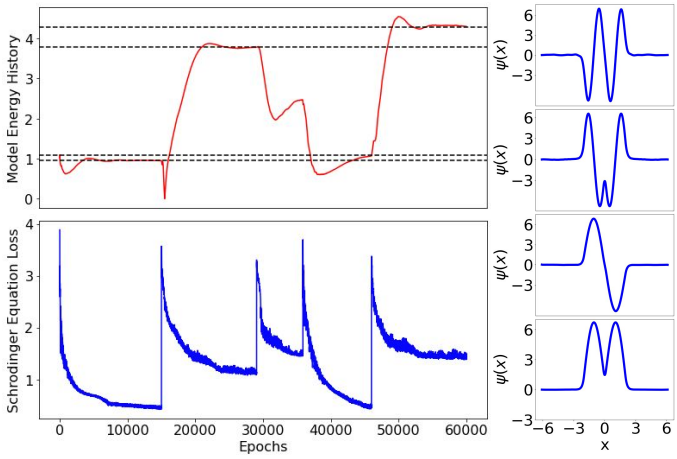

Fig. 3: Top left plot shows the network's eigenvalue over epochs during training, with true eigenvalues shown by the dotted horizontal lines. Bottom left plot shows the model's corresponding Schrodinger equation loss at each training point. Column of plots on the right are the resulting eigenfunctions that the model finds.

proved to greatly improve the solution accuracy. Figure 4 compares the eigenvalues (energy) predicted by the two models, one with embedded symmetry (blue line) and one without (red line). While the symmetry-embedded model is able to smoothly transition from one correct eigenvalue to the next one, the model without embedded symmetry converges to an incorrect eigenvalue.

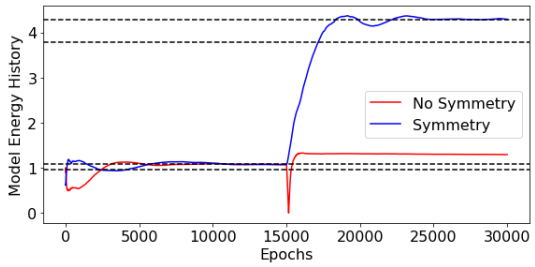

Fig. 4: Predicted eigenvalue energies of quantum multiple wells during the training of a network with embedded symmetry (blue) and a network without any embedded symmetry (red). Dotted black horizontal lines are analytical, ground truth solution eigenvalues.

D. Hydrogen Atom

In quantum mechanics, the hydrogen atom is described by the three-dimensional Schrodinger equation with a Coulomb potential energy. While the hydrogen atom is a three-dimensional problem, the equation can be decomposed into radial and angular components via separation of variables. The angular equation yields the spherical harmonics solutions, while the radial component $R(r)$ equation to be solved is given by:

$$\frac{d^2R}{dr^2}+\frac{2}{r}\frac{dR}{dr}=-\left(\frac{2\mu}{\hbar^2}\left(E+\frac{Ze^2}{4\pi\epsilon_0 r}\right)-\frac{l(l+1)}{r}\right)R, \quad (17)$$

where r is the radial variable, $\hbar$ is the reduced Planck's constant, μ is the reduced mass, Z is the number of protons, ϵ_0 denotes the vacuum permeability, and the variable l denotes the angular momentum of the system and takes positive integer values. We employ the proposed neural network to solve Eq. (17) for $l=0,1,2,3$.

We note that Eq. 17 becomes singular at $r=0$. Consequently, training sample points close to $r=0$ lead to numerical instability. To avoid this problem, we allocate the region $r=[0,1e-1]$ to be a no-train zone. Thus, any training sample points that are generated are greater than $r=1e-1$. Without this constraint, the numerical instability caused by sample points close to 0 disrupts the network's ability to converge to solutions.

The analytical eigenvalue energies of the hydrogen atom are given by

$$E_n=-\frac{\mu Z^2e^4}{32\epsilon_0^2\hbar^2\pi^2n^2}, \quad (18)$$

where n denotes the order of the solutions. Namely, for $n=1$, we get the ground energy. We notice that the eigenvalue energies are not dependent on the system's angular momentum l, but only on the system's order of excitement n.

The full three-dimensional solution to the Schrodinger equation for the hydrogen atom creates probability densities. The densities have not only radial dependence, but also angular dependence. For our work, we focused solely on the radial component of the Schrodinger equation. Furthermore, without loss of generality, we set $\frac{\mu Z^2e^4}{8\epsilon_0^2h^2n^2}=\frac{1}{2}$.

We demonstrate that our method solves for the first few eigenfunctions for four different angular momentum values $l=0,1,2,3$. Figure 5 shows our model's solutions, arranged in a grid with angular momentum l running along the vertical grid plots, and the energy level n running along the horizontal axis.

Our method is able to solve for the lowest eigenvalue-eigenfunction pairs with good accuracy. We analysed the accuracy of our method's solutions in comparison to the true solutions which are analytically known. Table I shows these results.

V. Conclusion

In recent years, there has been a growing interest in the application of neural networks to study differential equations. In this study, we introduced a neural network that is capable of discovering eigenvalues and eigenfunctions for boundary conditioned differential eigenvalue problems. The obtained solutions identically satisfy the given boundary conditions via a parametric function. We imposed even and odd symmetry in the network structure for problems that require such solutions, such as the single and multiple finite wells. We also introduced an orthogonality loss, which allows the network to learn new

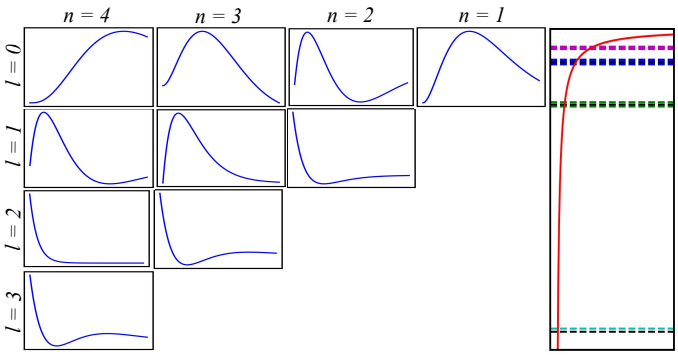

Fig. 5: Upper triangular plots show the model's predicted eigenfunctions for angular momentum values $l = [0, 3]$ inclusive, and for energy levels $n = [1, 4]$ also inclusive. The rightmost plot shows the true eigenvalues in dashed black, with our method's found solutions in colors. The red line is the Coulomb potential function that describes the radial hydrogen atom problem.

Performance Results (%)		
Schrodinger Problem	Eigenvalue Err.	Mean Squared Err.
Single (n = 1, s)	0.00	8.9e-4
Single (n = 1, a)	0.25	3.7e-4
Single (n = 2, s)	0.25	9.1e-4
Single (n = 2, a)	0.46	4.8e-4
Double (n = 1, s)	0.25	6.3e-4
Double (n = 1, a)	0.32	4.8e-4
Double (n = 2, s)	0.56	8.7e-4
Double (n = 2, a)	0.61	7.1e-4
H (l = 0, n = 1)	1.46	4.0e-3
H (l = 0, n = 2)	0.08	5.8e-3
H (l = 0, n = 3)	0.70	8.2e-3
H (l = 0, n = 4)	1.10	1.0e-2
H (l = 1, n = 2)	4.08	3.8e-5
H (l = 1, n = 3)	2.38	1.8e-4
H (l = 1, n = 4)	2.52	2.2e-4
H (l = 2, n = 3)	0.18	6.9e-6
H (l = 2, n = 4)	0.48	6.2e-4
H (l = 3, n = 4)	1.44	6.5e-3

TABLE I: This table presents comparisons with our model's solutions to the true analytical solutions. We find that our model finds the true eigenvalues to approximately 1 % error consistently. Mean squared error denotes the mean squared error of the function divided by the maximum of the absolute value of the true eigenfunction.

eigenfunctions that are orthogonal to all previously learned eigenfunctions. Furthermore, a normalization loss was used to enforce that the learned solutions are not trivial solutions, and that the quantum physical interpretation of eigenfunctions as probability distributions can be supported. The optimization solely depends on the network's predictions, consisting of an unsupervised learning method. We demonstrated the capability of the proposed architecture and training methodologies by solving the finite well, multiple finite wells, and hydrogen atom quantum problems.

VI. Future Research

For future work, we will generalise our method in two ways. One generalisation is towards more dimensions. For instance, the full solutions to the hydrogen atom Schrodinger equation are three-dimensional. We believe such generalisations will also more clearly reveal the advantages of solving such equations with neural networks. It is also possible to extend into the temporal dimension and solve the time dependent Schrodinger equation. The other avenue for future research is to apply our method to more general eigenvalue differential equations. This paper focuses on the Schrodinger's equation, which belongs to the Sturm-Liouville family. This study lays the groundwork for using neural networks to solve any eigenvalue differential equation.

VII. Broader Impact

This work is valuable for computational physicists and applied mathematicians, as well as in any field where differential eigenvalue problems may arise. We have demonstrated our method's success for the one-dimensional Schrodinger equation, but the technique can be generalised to Sturm-Liouville problems, as well as higher dimensional equations (e.g. 3D Schrodinger and Helmholtz equations). We strongly believe that this study will serve as the groundwork for future work in the area of solving differential equations using deep learning methods. We neither foresee and nor desire our research results to be used for any kind of discrimination.

References

[1] I. Lagaris, A. Likas, and D. Fotiadis, "Artificial neural network methods in quantum mechanics," *Computer Physics Communications*, vol. 104, no. 1, pp. 1 – 14, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0010465597000544

[2] M. Magill, F. Qureshi, and H. W. de Haan, "Neural networks trained to solve differential equations learn general representations," in *NeurIPS*, 2018.

[3] I. Lagaris, A. Likas, and D. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *Neural Networks, IEEE Transactions on*, pp. 987 – 1000, 10 1998.

[4] M. Mattheakis, D. Sondak, A. S. Dogra, and P. Protopapas, "Hamiltonian neural networks for solving differential equations," 2020.

[5] J. Han, A. Jentzen, and E. Weinan, "Solving high-dimensional partial differential equations using deep learning," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 115 34, pp. 8505–8510, 2017.

[6] J. A. Sirignano and K. Spiliopoulos, "Dgm: A deep learning algorithm for solving partial differential equations," *Journal of Computational Physics*, vol. 375, pp. 1339–1364, 2018.

[7] C. Flamant, P. Protopapas, and D. Sondak, "Solving differential equations using neural network solution bundles," vol. 2006.14372, 2020.

[8] M. Mattheakis, H. Joy, and P. Protopapas, "Unsupervised reservoir computing for solving ordinary differential equations," 2021.

[9] S. Desai, M. Mattheakis, H. Joy, P. Protopapas, and S. Roberts, "One-shot transfer learning of physics-informed neural networks," 2021.

[10] Y. Chen, L. Lu, G. E. Karniadakis, and L. D. Negro, "Physics-informed neural networks for inverse problems in nano-optics and metamaterials," *Opt. Express*, vol. 28, no. 8, pp. 11 618–11 633, Apr 2020. [Online]. Available: http://www.osapublishing.org/oe/abstract.cfm?URI=oe-28-8-11618

[11] A. Paticchio, T. Scarlatti, M. Mattheakis, P. Protopapas, and M. Brambilla, "Semi-supervised neural networks solve an inverse problem for modeling covid-19 spread," 2020.

[12] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999118307125

[13] H. Li, Q. Zhai, and J. Z. Y. Chen, "Neural-network-based multistate solver for a static schrödinger equation," *Phys. Rev. A*, vol. 103, p. 032405, Mar 2021. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.103.032405

[14] H. Jin, M. Mattheakis, and P. Protopapas, "Unsupervised neural networks for quantum eigenvalue problems," *ArXiv*, vol. abs/2010.05075, 2020.

[15] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. Devito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[16] M. Mattheakis, P. Protopapas, D. Sondak, M. D. Giovanni, and E. Kaxiras, "Physical symmetries embedded in neural networks," 2020.

[17] A. Bhattacharya, M. Mattheakis, and P. Protopapas, "Encoding involutory invariance in neural networks," *CoRR*, vol. abs/2106.12891, 2021. [Online]. Available: https://arxiv.org/abs/2106.12891