

On a neural network approach for solving potential control problem of the semiclassical Schrödinger equation

Yating Wang^a, Liu Liu^b

^a*School of Mathematics and Statistics, Xi'an Jiaotong University.*

^b*Department of Mathematics, Chinese University of Hong Kong.*

Abstract

Robust control design for quantum systems is a challenging and key task for practical technology. In this work, we apply neural networks to learn the control problem for the semiclassical Schrödinger equation, where the control variable is the potential given by an external field that may contain uncertainties. Inspired by a relevant work [29], we incorporate the sampling-based learning process into the training of networks, while combining with the fast time-splitting spectral method for the Schrödinger equation in the semi-classical regime. The numerical results have shown the efficiency and accuracy of our proposed deep learning approach.

1. Introduction

Control of quantum phenomena has been an important scientific problem in the emerging quantum technology [16]. The control of quantum electronic states in physical systems has a variety of applications such as quantum computers [4], control of photochemical processes [38] and semiconductor lasers [18]. Detailed overviews of the quantum control field can be found in survey papers and monographs [15, 43]. One issue of the controllability theory [35] aims to assess the ability to steer a quantum system from an arbitrary initial state to a targeted final state, under the impact of a control field such as a potential function, given possibly noisy observation data.

Uncertainty Quantification (UQ) has drawn many attentions over the past decade. In simulating physical systems, which are often modeled by differential equations, there are inevitably modeling errors, imprecise measurements of the initial data or background coefficients, which may bring uncertainties to the models. In this project, we study the semiclassical Schrödinger equation with external potential that may contain uncertainties, and is treated as the control variable. Let Ω be a bounded domain in \mathbb{R} , the Schrödinger equation in the semiclassical regime is described by a wave function $\psi : \mathcal{Q} \mapsto \mathbb{C}$,

$$\begin{cases} i\varepsilon\partial_t\psi^\varepsilon = -\frac{\varepsilon^2}{2}\Delta\psi^\varepsilon + V(x, \mathbf{z})\psi^\varepsilon, & (x, t) \in \mathcal{Q} \times (0, T), \\ \psi|_{t=0} = \psi_0(x), & x \in \Omega \subset \mathbb{R}, \end{cases} \quad (1.1)$$

where $0 < \varepsilon \ll 1$ is the scaled Planck constant describing the microscopic and macroscopic scale ratio. Here the solution $\psi = \psi(t, x, \mathbf{z})$ is the electron wave function with initial condition $\psi_0(x)$,

the potential $V(x, \mathbf{z}) \in L^\infty(\Omega \times I_{\mathbf{z}})$ is the control variable that models the external field and is spatially dependent. Periodic boundary condition is assumed in our problem.

The uncertainty is described by the random variable \mathbf{z} , which lies in the random space $I_{\mathbf{z}}$ with a probability measure $\pi(\mathbf{z})d\mathbf{z}$. We introduce the notation for the expected value of $f(\mathbf{z})$ in the random variable \mathbf{z} ,

$$\langle f \rangle_{\pi(\mathbf{z})} = \int f(\mathbf{z})\pi(\mathbf{z})d\mathbf{z}. \quad (1.2)$$

The solution to the Schrödinger equation is a complex valued wave function, whose nonlinear transforms lead to probabilistic measures of the physical observables. The primary physical quantities of interests include position density,

$$n^\varepsilon = |\psi^\varepsilon|^2, \quad (1.3)$$

and current density

$$\mathcal{J}^\varepsilon = \varepsilon \operatorname{Im} (\overline{\psi^\varepsilon} \nabla \psi^\varepsilon) = \frac{1}{2i} (\overline{\psi^\varepsilon} \nabla \psi^\varepsilon - \psi^\varepsilon \nabla \overline{\psi^\varepsilon}). \quad (1.4)$$

At each fixed \mathbf{z} , with V being continuous and bounded, the Hamiltonian operator H^ε defined by

$$H^\varepsilon \psi^\varepsilon = -\frac{\varepsilon^2}{2} \Delta \psi^\varepsilon + V(x, \mathbf{z}) \psi^\varepsilon$$

maps functions in $H^2(\mathbb{R}^d)$ to $L^2(\mathbb{R}^d)$ and is self-adjoint. The operator $\frac{1}{i\varepsilon} H^\varepsilon$ generates a unitary, strongly continuous semi-group on $L^2(\mathbb{R}^d)$, which guarantees a unique solution of the Schrödinger equation (1.1) that lie in the space [39]:

$$W(0, T) := \left\{ \phi \in L^2((0, T); H_0^1(\Omega; \mathbb{C})) \mid \frac{d\phi}{dt} \in L^2((0, T); H^{-1}(\Omega; \mathbb{C})) \right\}.$$

As a literature review, we mention that there has been several work [2, 27] on boundary control for the Schrödinger equation (1.1), where the observation is taken from the Dirichlet or Neumann boundary data. In some references such as [7], the authors consider the quantum system with evolution of its state $|\psi(t)\rangle$ described by the Schrödinger equation $\frac{d}{dt}|\psi(t)\rangle = -iH(t)|\psi(t)\rangle$ with the initial condition $|\psi(0)\rangle = |\psi_0\rangle$. The Hamiltonian $H(t)$ there corresponds to a time-dependent control variable that contains random parameters. Their goal is to drive the quantum ensemble from an initial state $|\psi_0\rangle$ to the target state $|\psi_{\text{target}}\rangle$, by employing a gradient-based learning method to optimize the control field. In [5, Section 7.3], the control problem of a charged particle in a well potential was formulated, where in their setting the potential field is time-dependent. We mention some other relevant work on stability estimates and semiclassical limit of inverse problem for the Schrödinger equation [3, 8, 17, 26, 39].

We continue to mention several studies that are related to the inverse problems for the Schrödinger equation or other models. For relevant inverse boundary value problems on this topic, there are existing iterative methods applied to the Helmholtz equation [31], where one starts with an initial guess of the boundary condition, then adjusts it iteratively by minimizing functionals such as error norms between the calculated data and measured data. This could be

extremely time-consuming since at each iteration step, a forward problem needs to be solved. In the partial boundary data situation, there has been research on studying the linearized inverse problem of recovering potential function for the time-independent Schrödinger equation [47]. Moreover, for inverse potential problems, well-posedness of the continuous regularized formulation was analyzed in both elliptic and parabolic problems, with conditional stability estimates and error analysis for the discrete scheme studied in [10, 22].

The desired control problem can be described as the following: To which extend can the wave solution ψ^ε of (1.1) be perturbed by the control field—in our case the potential function V , in order to reach the desired target state at the final time T ? The above question can be reformulated into an *optimal control* problem. At the final time T , given the target state ψ_{target} , let V be approximated by a neural network parameterized by $\boldsymbol{\theta}$, and $\lambda > 0$ be a regularization coefficient, we aim to solve the following minimization problem:

$$\left\{ \begin{array}{l} \min_{\boldsymbol{\theta}} J_\lambda(V(\boldsymbol{\theta})) = \min_{\boldsymbol{\theta}} \|\psi^\varepsilon(x, T; \boldsymbol{\theta}) - \psi_{target}\|_{L^2(\Omega)}^2 + \lambda \|V(x; \boldsymbol{\theta})\|_{L^2(\Omega)}^2, \\ \text{such that} \quad i\varepsilon \partial_t \psi^\varepsilon(x, t; \boldsymbol{\theta}) = -\frac{\varepsilon^2}{2} \Delta \psi^\varepsilon(x, t; \boldsymbol{\theta}) + V(x; \boldsymbol{\theta}) \psi^\varepsilon(x, t; \boldsymbol{\theta}), \\ \psi^\varepsilon(x, t = 0; \boldsymbol{\theta}) = \psi_0(x). \end{array} \right. \quad (1.5)$$

if V is a deterministic potential, and

$$\left\{ \begin{array}{l} \min_{\boldsymbol{\theta}} J_\lambda(V(\boldsymbol{\theta})) = \min_{\boldsymbol{\theta}} \|\psi^\varepsilon(x, T; \boldsymbol{\theta}, \mathbf{z}) - \psi_{target}(\mathbf{z})\|_{L^2(\Omega \times I_{\mathbf{z}})}^2 + \lambda \|V(x; \boldsymbol{\theta}, \mathbf{z})\|_{L^2(\Omega \times I_{\mathbf{z}})}^2, \\ \text{such that} \quad i\varepsilon \partial_t \psi^\varepsilon(x, t; \boldsymbol{\theta}, \mathbf{z}) = -\frac{\varepsilon^2}{2} \Delta \psi^\varepsilon(x, t; \boldsymbol{\theta}, \mathbf{z}) + V(x; \boldsymbol{\theta}, \mathbf{z}) \psi^\varepsilon(x, t; \boldsymbol{\theta}, \mathbf{z}), \\ \psi^\varepsilon(x, t = 0; \boldsymbol{\theta}, \mathbf{z}) = \psi_0(x; \mathbf{z}). \end{array} \right. \quad (1.6)$$

if the potential V contains uncertainty and the random variable is \mathbf{z} .

In each particular problem setting, discretized form of the above loss function will be presented. We now highlight the main contributions of our work:

1. We take advantage of the rising trend of machine learning and use neural networks to approximate the control variable considered as the potential field in the Schrödinger equation. Both deterministic and stochastic control functions are considered. A fully-connected neural network is used for the deterministic problem, and the DeepONet [30] is applied in the stochastic case.
2. During the training process, the Schrödinger equation in the semiclassical regime is solved using the fast time-splitting spectral method to improve the computational efficiency and accuracy of our algorithm.
3. We study and compare both cases when the observation data is associated with or without noise, and propose different training strategies. For data without noise, the popular stochastic gradient descent (SGD) method is used. For noisy data, we consider a Bayesian framework and adopt the stochastic gradient Markov chain Monte Carlo (MCMC) approach to obtain robust learning results.

The rest of the paper is organized as follows. In Section 2, we discuss the oscillatory behavior of solution to the semiclassical Schrödinger equation in the random variable and mention the numerical challenges even for the forward UQ problems. Our main methodology of using the learning-based technique to solve the optimization problem (1.6) will be proposed in Section 3, with numerical scheme for the forward problem introduced in subsection 3.1 and several neural network approaches described in subsection 3.2. We conduct extensive numerical experiments for both the deterministic and stochastic potential control problems and present the results in Section 4. Conclusion and future work will be addressed lastly.

2. Regularity of solution in the random space

The semi-classical Schrödinger equation is a family of dispersive wave equations parameterized by $\varepsilon \ll 1$, it is well known that the wave equation propagates $O(\varepsilon)$ scaled oscillations in space and time. However, for UQ problems it is not obvious whether the small parameter ε induces oscillations in the random variable \mathbf{z} . We conduct a regularity analysis of ψ in the random space, which enables us to study the oscillatory behavior of solution in the random space.

To investigate the regularity of the wave function in the \mathbf{z} variable, we check the following averaged norm

$$\|\psi\|_{\Gamma} := \left(\int_{I_z} \int_{\mathbb{R}^3} |\psi(t, \mathbf{x}, \mathbf{z})|^2 d\mathbf{x} \pi(\mathbf{z}) d\mathbf{z} \right)^{1/2}. \quad (2.1)$$

First, observe that $\forall \mathbf{z} \in I_z$,

$$\frac{\partial}{\partial t} \|\psi^\varepsilon\|_{L_x^2}^2(t, \mathbf{z}) = 0,$$

thus

$$\frac{d}{dt} \|\psi^\varepsilon\|_{\Gamma}^2 = 0,$$

which indicates the Γ -norm of the wave function ψ^ε is conserved in time, $\|\psi^\varepsilon\|_{\Gamma}(t) = \|\psi_{\text{in}}^\varepsilon\|_{\Gamma}$.

Below we show that ψ^ε has ε -scaled oscillations in \mathbf{z} . As an example, we analyze first-order partial derivative of ψ^ε in z_1 and denote $\psi^1 = \psi_{z_1}^\varepsilon$ and $V^1 = V_{z_1}$. By differentiating the semi-classical Schrödinger equation (1.1) with respect to z_1 , one gets

$$i\varepsilon \psi_t^1 = -\frac{\varepsilon^2}{2} \Delta_{\mathbf{x}} \psi^1 + V^1 \psi^\varepsilon + V \psi^1.$$

Direct calculation leads to

$$\begin{aligned} \frac{d}{dt} \|\psi^1\|_{\Gamma}^2 &= \int (\psi_t^1 \bar{\psi}^1 + \psi^1 \bar{\psi}_t^1) \pi d\mathbf{x} d\mathbf{z} \\ &= \int \left(\frac{1}{i\varepsilon} V^1 \psi^\varepsilon \bar{\psi}^1 - \frac{1}{i\varepsilon} V^1 \psi^1 \bar{\psi}^\varepsilon \right) \pi d\mathbf{x} d\mathbf{z} \\ &\leq \frac{2}{\varepsilon} \|\psi^1\|_{\Gamma} \|V^1 \psi^\varepsilon\|_{\Gamma}, \end{aligned}$$

where we use the Cauchy-Schwarz inequality and Jensen inequality in the last step, namely

$$\int V^1 \psi^\varepsilon \bar{\psi}^1 dx \leq \left(\int (V^1 \psi^\varepsilon)^2 dx \right)^{1/2} \left(\int (\bar{\psi}^1)^2 dx \right)^{1/2},$$

$$\int \int V^1 \psi^\varepsilon \bar{\psi}^1 dx \pi(z) dz \leq \left(\int \left(\int V^1 \psi^\varepsilon \bar{\psi}^1 dx \right)^2 \pi(z) dz \right)^{1/2} \leq \|V^1 \psi^\varepsilon\|_\Gamma \|\psi^1\|_\Gamma.$$

Therefore,

$$\frac{d}{dt} \|\psi^1\|_\Gamma \leq \frac{1}{\varepsilon} \|V^1 \psi^\varepsilon\|_\Gamma^2.$$

For $t = O(1)$, this pessimistic estimate implies

$$\|\psi^1\|_\Gamma = O(\varepsilon^{-1}).$$

To summarize, in this part we emphasize the oscillatory behavior of the solution ψ in the random space, which brings numerical challenges for the forward UQ problem. If one directly adopts the generalized polynomial chaos (gPC)-based Galerkin methods or stochastic collocation methods [44] to the semi-classical Schrödinger equation with random parameters, ε -dependent basis functions or quadrature points are needed to get an accurate approximation. There has been some work developed for this forward problem [11, 23], where in our inverse problem case shares the similar difficulty. In the future work, to more efficiently sample from the random space, we will adopt numerical solvers that can resolve the ε -oscillations in the random variable. For simplicity of notations, we will omit the superscript ε in ψ^ε and use ψ in the rest of the paper.

3. Optimal control using neural networks

3.1. The time-splitting spectral method

In the semiclassical regime where $\varepsilon \ll 1$, the solution to the Schrödinger equation (1.1) is oscillatory both temporally and spatially, with an oscillation frequency of $O(1/\varepsilon)$. This poses tremendous computational challenges since one needs to numerically resolve, both spatially and temporally, the small wave length of $O(\varepsilon)$. The time-splitting spectral (TSSP) method, studied by Bao, Jin and Markowich in [1], is one of the most popular and highly accurate methods for such problems, where the meshing strategy $\Delta t = O(\varepsilon)$ and $\Delta x = O(\varepsilon)$ is required for moderate values of ε . Moreover, in order to just compute accurately the physical observables (such as position density, flux, and energy), one still needs to resolve the spatial oscillations, but the time step $\Delta t = o(1)$ is much more relaxed [1, 20, 24]. Recently a rigorous uniform in ε error estimate was obtained in [19], by using errors measured by a pseudo-metric in analogy to the Wasserstein distance between a quantum density operator and a classical density in phase space, with the regularity requirement for V being $V \in C^{1,1}$.

In this section, we review the first-order time-splitting spectral method studied in [1, Section 2]. Consider an one-dimensional spatial variable and a given potential $V(x)$. We choose the spatial mesh size $h = (b - a)/M$ for an even integer M , and the time step $k = \Delta t$, let the grid points and time step be

$$x_j := a + jh, \quad t_n := nk, \quad j = 0, 1, \dots, M, \quad n = 0, 1, 2, \dots.$$

For the time discretization, from $t = t_n$ to $t = t_{n+1}$, the Schrödinger equation (1.1) is solved in the following two steps. First, one solves

$$\varepsilon\psi_t - i\frac{\varepsilon^2}{2}\psi_{xx} = 0, \quad (3.1)$$

then

$$\varepsilon\psi_t + iV(x)\psi = 0, \quad (3.2)$$

in the second step. We discretize (3.1) in space by the spectral method, then integrate in time *exactly*. Note that the ODE (3.2) can be solved exactly.

Denote Ψ_j^n by the numerical approximation of the analytic solution $\psi(t_n, x_j)$ to the Schrödinger equation (1.1). Then the discretized scheme is given by

$$\Psi_j^* = \frac{1}{M} \sum_{l=-M/2}^{M/2-1} e^{-i\varepsilon k\mu_l^2/2} \hat{\Psi}_l^n e^{i\mu_l(x_j-a)}, \quad j = 0, 1, 2, \dots, M-1, \quad (3.3)$$

$$\Psi_j^{n+1} = e^{-iV(x_j)k/\varepsilon} \Psi_j^*,$$

where the Fourier coefficients of Ψ^n is defined as

$$\hat{\Psi}_j^n = \sum_{j=0}^{M-1} \Psi_j^n e^{-i\mu_l(x_j-a)}, \quad \mu_l = \frac{2\pi l}{b-a}, \quad l = -\frac{M}{2}, \dots, \frac{M}{2} - 1,$$

with

$$\Psi_j^0 = \psi(0, x_j), \quad j = 0, 1, 2, \dots, M.$$

We remark that instead of directly simulating the semi-classical Schrödinger equation, there are quite a few other methods which are valid in the limit $\varepsilon \rightarrow 0$, see [25] for a general discussion. In particular, many wave packets based methods have been introduced in past few years, which reduce the full quantum dynamics to Gaussian wave packets dynamics [21]. In this work, we simply adopt the TSSP method as our deterministic solver in the learning algorithm

3.2. Learning method for the control problem

Thanks to the nonlinear structure of deep neural network, it has shown great potential in approximating high dimensional functions and overcoming the curse of dimensionality. In recent years, deep learning has gained great success in solving high-dimensional PDEs, in both forward and inverse problem settings [34, 45]. There have been studies that suggested learning-based methods on solving general control problems, such as [14, 41]. Recently, in [32] the authors proposed SympOCnet to solve high dimensional optimal control problems with state constraints. The idea is to apply the Symplectic network, which can approximate arbitrary symplectic transformations, to perform a change of variables in the phase space and solve the forward Hamiltonian equation in the new coordinate system. In our work, we consider the control problem for the semiclassical Schrödinger equation and adopt neural networks to approximate the control field V that may contain uncertainties. The neural network parameterized potential

function is learnt by minimizing the discrepancies between the state solution of the system with neural network and the observation of the target state.

In this section, we will describe the neural network structures under two different problem settings: (i) the deterministic case where the underlying target potential is fixed; (ii) the stochastic case where the target potential is parameterized by some random variables. In both problems, we will validate the efficiency of our proposed method by using both clean and noisy training data.

3.2.1. Deterministic problem

In the deterministic problem, our goal is to learn a single target function $V(x)$ using the neural network. In this case, the input of the neural network is the spatial variable $\{x_k\}$, while the output is the value of the potential function at x_k , i.e., $\{V(x_k)\}$, $k = 1, \dots, M$. We will use 5 fully connected layers with 50 neurons per layer to build up the network. For the data points, assume the spatial domain $\Omega \in \mathbb{R}$ and temporal domain $[0, T]$, N equally distributed points in Ω (where $N \ll M$) are taken, and the measurement data are the corresponding numerical solutions of the wave function at time T . This implies that the data pairs are chosen as $(x_i, \psi_{\text{obs}}(x_i))$ for $i = 1, \dots, N$ and $\psi_{\text{obs}}(x_i) \sim \mathcal{N}(\psi(x_i), \sigma^2)$. In our numerical examples, we set $N = 50$ and $M = 1000$. An illustration of the network for the deterministic problem is presented in Figure 1. As noticed from Figure 1, the input-output pairs for the fully connected neural network are $(x_i, V(x_i))$. The output of the neural network, i.e. the potential function, is then used to solve the forward Schrödinger equation by adopting the time-splitting spectral method. The predicted solution obtained at the final time step $\psi(x; T)$ is then compared with the measurement data $\psi_{\text{obs}}(x; T)$. The mismatch between the predicted solution and the measurement data will form the loss function. A pseudocode is presented in Algorithm 1.

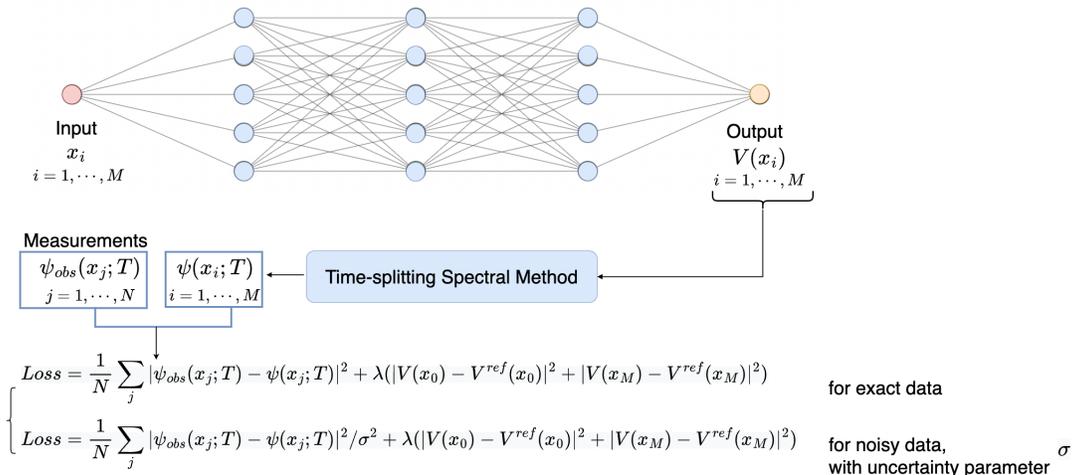


Figure 1: Illustration of the network for the deterministic problem.

Algorithm 1 Deterministic case

INPUT: Neural network input $\{x_i\}_{i=1}^M$. Observation data $\{\psi_{\text{obs}}(x_j, T)\}_{j=1}^N$. Initialization of neural network parameters θ_0 .

- 1: **for** For $k \leftarrow 0 : \#iterations$ **do**
- 2: Get the output of the neural network $\{V(x_i; \theta_k)\}_{i=1}^M$.
- 3: Given $V(x; \theta_k)$, solve equation (1.1) by time-splitting spectral method and get the solution $\psi(x, T; \theta_k)$.
- 4: Compute the mismatch between $\psi_{\text{obs}}(x, T)$ and $\psi(x, T; \theta_k)$, and get the loss.
- 5: Use SGD type or SGLD method to update the network parameter and get θ_{k+1} .

OUTPUT: The solution of (1.1) $\psi(x_j, t_m)$ at all spatial locations and all time steps of interest.

3.2.2. Stochastic problem

In the stochastic problem, our goal is to learn a set of functions described by a stochastic potential function $V(x; z)$ containing a random parameter z , by training the DNN. We will utilize the DeepONet architecture developed in [30].

First we give a brief overview of DeepONet, which is a powerful tool designed to learn continuous nonlinear operators. Denote G by an operator with input function u ; for any coordinate y in the domain of $G(u)$, the output $G(u)(y)$ is a number. DeepONet aims to approximate G with a neural network G_{θ} parameterized by θ , which takes inputs (u, y) and returns the output $G(u)(y)$. The architecture of DeepONet is composed of a branch net and a trunk net. In the unstacked setting, the branch net encodes the discrete input function u into the features represented by $[b_1, \dots, b_q]$, and the trunk net takes the coordinate y as input and encodes it into the features represented by $[t_1, \dots, t_q]$. Then the dot product of \mathbf{b} and \mathbf{t} provides the final output of DeepONet, i.e.

$$G_{\theta}(u)(y) = \sum_{k=1}^q b_k(u(x_1), \dots, u(x_N)) t_k(y).$$

The parameter θ consists of all weights and biases in the branch and trunk net.

In our setting, we aim to approximate the parameterized potentials $V(x; z)$ using G_{θ} that takes the discrete data $[\psi_{\text{obs}}(x_1; z), \dots, \psi_{\text{obs}}(x_N; z)]$ and the coordinate y_k as inputs. Here $k = 1, \dots, M$. We note that for each z , there are N sensors that provide the observation data $\psi_{\text{obs}}(\cdot, z)$, thus the dataset size is equal to the product of M and the number of z samples. The value of $G_{\theta}(\psi_{\text{obs}}(\cdot; z))(y_k)$ is a prediction of $V(y_k; z)$. Utilizing the predictions from the DeepONet, namely $V(y_k; z)$ ($k = 1, \dots, M$), the time-splitting spectral method is then applied to compute the value of wave functions $\psi(y_k, z)$. We aim to minimize the mismatch between the observations $\psi_{\text{obs}}(x_j, z)$ and the numerical solutions $\psi(x_j, z)$ at all sensor locations x_j for all z . An illustration of the network for the stochastic problem is presented in Figure 2. A pseudocode is presented in Algorithm 2.

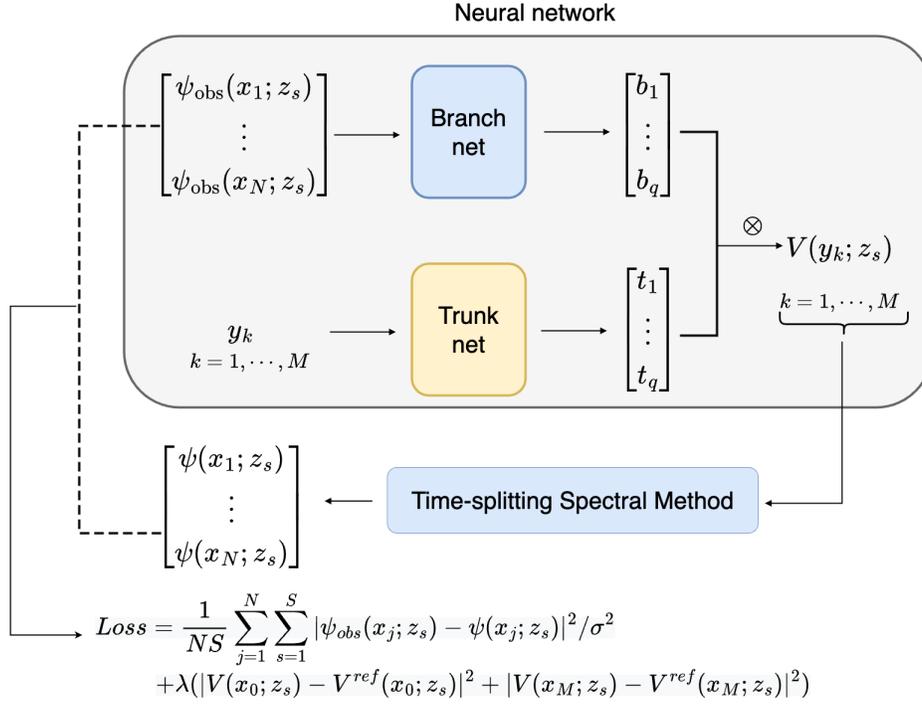


Figure 2: Illustration of the network for the stochastic problem.

Algorithm 2 Stochastic case

INPUT: Neural network input $\{\psi_{obs}(x_j, t_m; z_s)\}_{j=1}^N$ for some stochastic samples z_s , at few time instances t_m , as well as spatial points $\{y_k\}_{k=1}^M$. Observation data $\{\psi_{obs}(x_j, T; z_s)\}_{j=1}^N$. Initialization of neural network parameters θ_0 .

- 1: **for** For $k \leftarrow 0 : \#iterations$ **do**
- 2: Get the output of the neural network $\{V(y_k; z_s; \theta_k)\}_{k=1}^M$.
- 3: For each $V(x; z_s; \theta_k)$, solve equation (1.1) by time-splitting spectral method and get the solutions $\psi(x, t; \theta_k)$ at all spatial points and time instances.
- 4: Compute the mismatch between $\{\psi_{obs}(x_j, T; z_s)\}_{j=1}^N$ and $\psi(x_j, T; z_s; \theta_k)$ (at the observational spatial and temporal points) over all samples of z_s , and get the loss.
- 5: Use SGLD method to update the network parameter and get θ_{k+1} .

OUTPUT: For each z_s , the solution of (1.1) $\psi(x_j, t_m; z_s)$ at all spatial locations and all time steps of interest.

3.2.3. Training of the neural network

When dealing with large-scale problems, traditional Bayesian inference methods, e.g., Markov chain Monte Carlo (MCMC)[37] have shown disadvantages due to extremely expensive computational cost of handling the whole dataset at each iteration. To tackle problems with large datasets, deep learning algorithms such as stochastic gradient descent (SGD) [36] are favorable

and have been popularly used, since one only needs to employ a small subset of samples randomly selected from the whole dataset at each iteration. To bring together advantages of these two types of methods, Welling and Teh [42] first proposed the stochastic gradient Langevin dynamics (SGLD) (also known as stochastic gradient MCMC) method. It adds a suitable amount of noise to the standard SGD and uses mini-batches to approximate the gradient of loss function. With the help of decreasing training step size η_k , it has demonstrated powerful and provided a transition between optimization and Bayesian posterior sampling [6].

We now briefly review the SGLD method. Denote $D = \{d_i\}_{i=1}^N = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$ by a given dataset, where \mathbf{x}_i is the input and \mathbf{y}_i is the corresponding noisy output. We let \mathcal{NN} be a neural network parameterized by the parameter $\boldsymbol{\theta}$; the goal of its training is to find suitable parameters $\boldsymbol{\theta}$ such that $F(\mathcal{NN}(\mathbf{x}_i; \boldsymbol{\theta})) \approx \mathbf{y}_i$ ($i = 1, \dots, N$). Due to the noise in measurement data, we assume the parameters are associated with uncertainties and obey a prior distribution $p(\boldsymbol{\theta})$. The uncertainties in the parameters $\boldsymbol{\theta}$ can be captured through Bayesian inference to avoid overfitting. Let d^j be a mini-batch of data with size n , the likelihood can be written as

$$p(d^j|\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp \left\{ - \frac{\sum_{\mathbf{x}_i^j \in d^j} (\mathbf{y}_i^j - F(\mathcal{NN}(\mathbf{x}_i^j; \boldsymbol{\theta})))^2}{2\sigma^2} \right\},$$

where σ is standard deviation of the Gaussian likelihood. In our case, for the dataset $d^j = (\mathbf{x}_i^j, \mathbf{y}_i^j)$, \mathbf{x}_i^j corresponds to the input $[\psi_{\text{obs}}(x_1; z), \dots, \psi_{\text{obs}}(x_N; z), y]$, \mathbf{y}_i^j corresponds to the labels $[\psi_{\text{obs}}(x_1; z), \dots, \psi_{\text{obs}}(x_N; z)]$ and F maps the output of the neural network output $\mathcal{NN}(\mathbf{x}_i; \boldsymbol{\theta})$ which approximates $V(y, z)$ to the quantities of interest $\psi(y; z; T)$ with T the final simulation time. According to the Bayes' theorem, the posterior distribution of $\boldsymbol{\theta}$, given the data D , then follows $p(\boldsymbol{\theta}|D) \propto p(\boldsymbol{\theta}) \prod_{i=1}^N p(d_i|\boldsymbol{\theta})$.

To sample from the posterior, one efficient proposal algorithm is to use the gradient of the target distribution. Let η_k be the learning rate at epoch k and $\tau > 0$ be the inverse temperature, the parameters will be updated by SGLD based on the following rule:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta_k \nabla_{\boldsymbol{\theta}} \tilde{L}(\boldsymbol{\theta}_k) + \mathcal{N}(0, 2\eta_k \tau^{-1}).$$

Here for a subset of n data points $d^j = \{d_1^j, \dots, d_n^j\}$,

$$\nabla_{\boldsymbol{\theta}} \tilde{L}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}) + \frac{N}{n} \sum_{i=1}^n \nabla_{\boldsymbol{\theta}} \log p(d_i^j|\boldsymbol{\theta})$$

is the stochastic gradient computed by using a minibatch that approximate the true gradient of the loss function $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$.

However, if the components of the network parameters $\boldsymbol{\theta}$ have different scales, the invariant probability distribution for the Langevin equation is not isotropic. If one still uses a uniform learning rate in each direction, this may leads to slow mixing [9, 12, 13, 28, 40, 46]. To incorporate the geometric information of the target posterior, stochastic Gradient Riemann Langevin Dynamics (SGRLD) [33] generalizes SGLD on a Riemannian manifold. Consider the probability

model on a Riemann manifold with some metric tensor $P^{-1}(\boldsymbol{\theta})$, in SGRLD, the parameter is updated at the k -th iteration by the following rule:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta_k \left[P(\boldsymbol{\theta}_k) \nabla_{\boldsymbol{\theta}} \tilde{L}(\boldsymbol{\theta}_k) + \Gamma(\boldsymbol{\theta}_k) \right] + \mathcal{N}(0, 2\eta_k \tau^{-1} P(\boldsymbol{\theta}_k)) \quad (3.4)$$

where $\Gamma_i(\boldsymbol{\theta}_k) = \sum_j \frac{\partial P_{ij}(\boldsymbol{\theta}_k)}{\partial \theta_j}$. One popular and computationally efficient approach to approximate $P(\boldsymbol{\theta}_k)$ is to use a diagonal preconditioning matrix [28, 40], that is,

$$P(\boldsymbol{\theta}_k) = \text{diag}^{-1}(\lambda + \sqrt{V(\boldsymbol{\theta}_k)}), \quad (3.5)$$

$$V(\boldsymbol{\theta}_k) = (1 - \omega_k)V(\boldsymbol{\theta}_{k-1}) + \omega_k g(\boldsymbol{\theta}_k) \circ g(\boldsymbol{\theta}_k), \quad (3.6)$$

where λ is a regularization constant, $g(\boldsymbol{\theta}_k) = \nabla_{\boldsymbol{\theta}} \tilde{L}(\boldsymbol{\theta}_k)$ is the stochastic gradient, the operator \circ denotes a elementwise multiplication, and $\omega_k \in (0, 1)$ is a weight parameter used in the moving average $V(\boldsymbol{\theta}_k)$. In our framework, we will use the preconditioned SGLD to train the network parameters.

4. Numerical results

In our numerical experiments, we consider two types of potential functions, the deterministic and stochastic potential. In the deterministic case, the potential V is only spatially dependent. In the stochastic problem, the potential function $V(\cdot, z)$ is assumed to depend on a random parameter characterized by z . In particular, we consider a simple example with $V(x, z) = (1 + 0.5z)x^2$, where z is a random variable following the uniform distribution in $[-1, 1]$.

4.1. Test I: A Deterministic Potential

In the first problem setup, we assume the potential function as $V(x) = x^2$. The network architecture introduced in Section 3.2.1 is adopted, and we train the network by using standard SGD and SGLD studied in Section 3.2.3. For the observation data, we choose it to be the electron wave function ψ solved by the Schrödinger equation (1.1) at several spatial locations and time instances using the forward TSSP solver, given the reference potential function V .

We first consider that there is no noise in the observation data and apply both SGD and SGLD to train. The numerical results show that the wave function ψ obtained from the network of both training algorithms matches well with the observation data, while it is also noticeable that the SGLD gives a slightly better approximation of the potential function. We then consider when some noise is added to the observation data, one can just apply SGLD to train the network in order to more accurately capture the uncertainties in the target potential function.

4.1.1. $V(x) = x^2$, no noise in the observation and by SGD

In this case, we let the reference potential function be $V(x) = x^2$, here the observation data is clean and without noise interference, SGD method is used in our training algorithm. In the forward solver, the spatial mesh size is $\pi/250$ and the temporal mesh size is 6.25×10^{-4} .

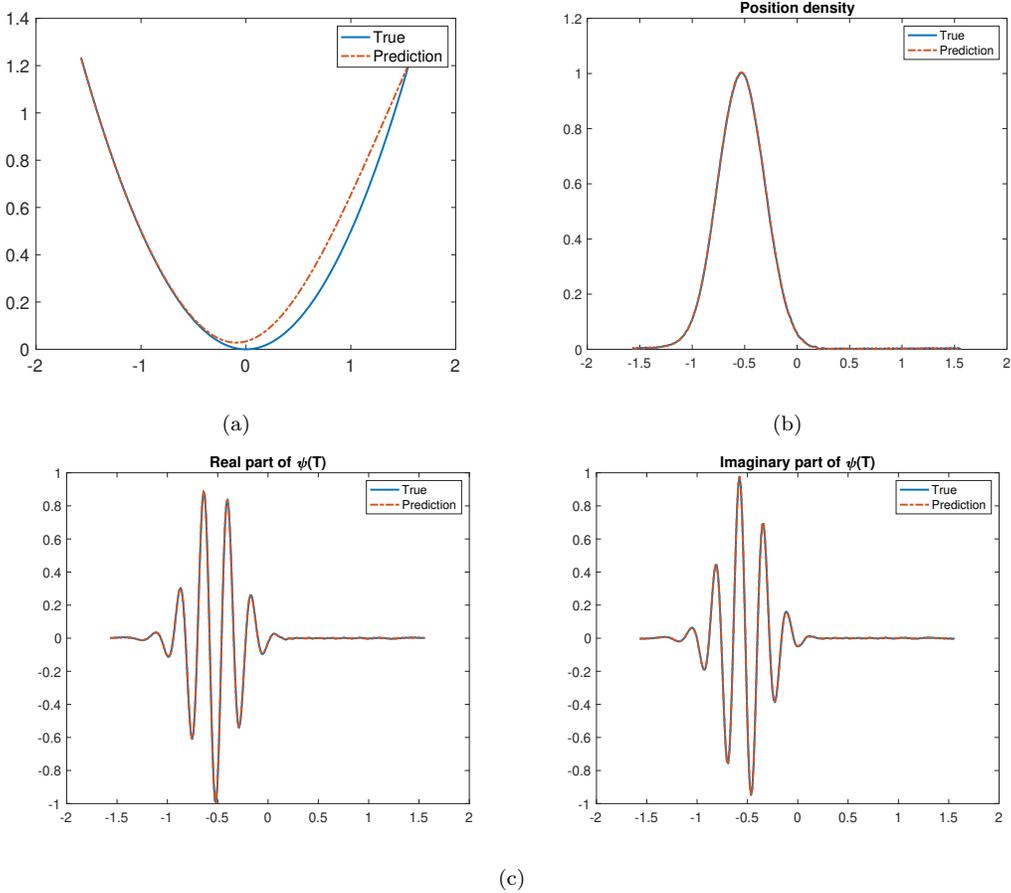


Figure 3: Test I case 1: $V(x) = x^2$, without noise in the observation and by SGD. (a) True and predicted value of the potential function. (b) True and predicted value of the position density at time T . (c) True and predicted value of the wave function at time T .

The learning rate is 10^{-4} and the total training epoch is 20000. In Figure 3 (a), a comparison between the reference and predicted potential function obtained from the neural network is shown. We observe that there is some mismatch in the region when $x > 0$, while the underlying reason remains to be discovered. In Figure 3 (b)-(c), a comparison between the reference with predicted position density n^ε and the wave function ψ^ε (real and imaginary parts) is presented. We conclude that the predicted wave and density functions at the final time T , which are computed by solving the Schrödinger equation (1.1) under the neural network's predicted output potential, can provide good approximations to the solution quantities obtained by using the true potential $V(x) = x^2$ in the TSSP solver.

4.1.2. $V(x) = x^2$, no noise in the observation and by SGLD

In the second case, the problem setup is the same as the previous case, while we apply SGLD algorithm to train the neural network. In the forward solver, the spatial mesh size is $\pi/1000$ and the temporal mesh size is 3×10^{-3} . The learning rate is 10^{-5} and the total training epoch is

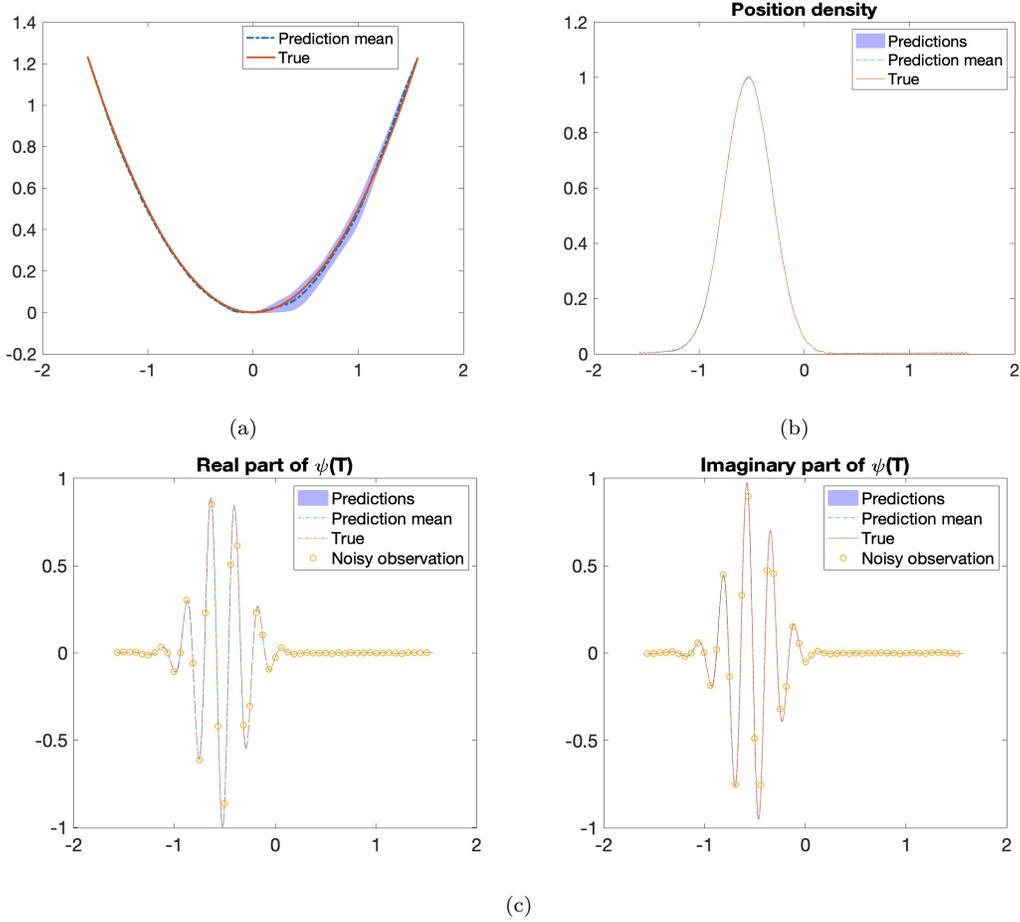


Figure 4: Test I case 2: $V(x) = x^2$, without noise in the observation and by SGLD. (a) True and predictions of the potential function. (b) True and predictions of the position density at time T . (c) True and predictions the wave function at time T .

10000. A comparison between the reference and predicted potential function is shown in Figure 4 (a). According to the nature of SGLD, we collect samples of neural network's parameters during the training process, then compute the mean and standard deviation of output potential functions (at each spatial point) obtained by using those parameter samples. The blue dashed line represents the mean of the predicted potential V , and the confidence interval are depicted by the shaded blue area in Figure ???. Based on these two tests, we observe that SGLD provides more reliable results compared to the standard SGD, and the uncertainty is negligible in the prediction since the data is clean.

In Figure 4 (b)-(c), we again present a comparison between the reference and predicted wave function ψ^ε or position density v^ε that is computed by the TSSP solver by using the predicted mean value of the potential. Similar to the previous test, it is obvious that the predicted wave or density can provide quite good approximations to the true data, i.e., the numerical solution at final time T obtained by using the true potential $V(x) = x^2$ in the TSSP solver.

4.1.3. $V(x) = x^2$, noisy data and by SGLD

In the third case, we consider some noise in the observation data and use SGLD to train the network. The mesh size in the forward solver, the learning rate and the training epochs are the same as in the previous subsection. We let the noise be a random variable that follows the normal distribution with mean 0 and standard deviation 0.05. In Figure 5 (c), the yellow circles are the noisy values of ψ^ε at 50 equally spaced locations. A comparison between the reference, i.e., $V(x) = x^2$, with the predicted mean of the potential function is shown in Figure 5 (a). One can observe that the predicted mean value is consistent with the reference potential, and the blue shaded area indicates that there are some uncertainties due to the noisy data, compared to the previous tests where there is no noise in the observation.

Similarly, we can see from Figure 5 (b)-(c), the predicted wave and density at final time T that are computed using the mean of network's predicted potential V , capture well the true solution obtained by using $V(x) = x^2$ in the TSSP solver. Therefore, we conclude that SGLD can deal with the noisy data and provide reliable results.

4.2. Test II: A Stochastic Potential

In Test II, we consider a stochastic potential, $V(x, z) = (1 + 0.5z)x^2$, where z follows the uniform distribution on $[-1, 1]$. To generate the dataset, we first take eight Gauss-Legendre points for $z \in [-1, 1]$. For each z_k ($k = 1, \dots, K$), i.e., each specific potential $V(x; z_k)$, we have the corresponding noisy measurement data $\psi_{\text{obs}}(x; z_k)$ at the final time instance $T = 0.6$. The observation $\psi_{\text{obs}}(x; z_k) \sim \mathcal{N}(\psi(x; z_k), \sigma^2)$ where σ is the standard deviation. The wave functions at the final time instance $\psi(x; z_k)$ is computed using the time-splitting spectral method on a 640×1000 temporal-spatial grid. Then for each z_k we select N sensor locations to collect the measurement data, the sensors are uniformly located in the spatial domain $\Omega = [-\pi/2, \pi/2]$. We will take $N = 20, 50$ in the numerical tests. In the forward solver, the spatial mesh size is $\pi/1000$ and the temporal mesh size is 6.25×10^{-4} . The learning rate is 10^{-5} and the total training epoch is 10000.

The input of the network then consists of the spatial evaluation point x_i , and the real part $\Re(\psi_{\text{obs}}(x_1; z_k)), \dots, \Re(\psi_{\text{obs}}(x_N; z_k))$ and the imaginary part $\Im(\psi_{\text{obs}}(x_1; z_k)), \dots, \Im(\psi_{\text{obs}}(x_N; z_k))$ of the observation data. The output of the network is the value of potential at x_i , i.e., $V(x_i; z_k)$. The number of training samples is equal to the product of M (the number of evaluation points x_i) and the number of z samples. We assume that the values of $V(x, z)$ at the endpoints $x = -\frac{\pi}{2}$ and $x = \frac{\pi}{2}$ are known for the training samples. The loss function consists of three parts, (1) the mismatch between the observation data $\psi_{\text{obs}}(x; z_k)$ and the ψ computed using neural network predicted potential function, (2) the mismatch between the true potential and neural network predicted potential at the endpoints of the spatial domain, and (3) a regularization term on the potential. After training, we will obtain the full potential profile for different z samples. In the testing stage, we will only have noisy observations of the wave function at final time T without knowing any information of the true potential function. We will feed the a set of spatial location

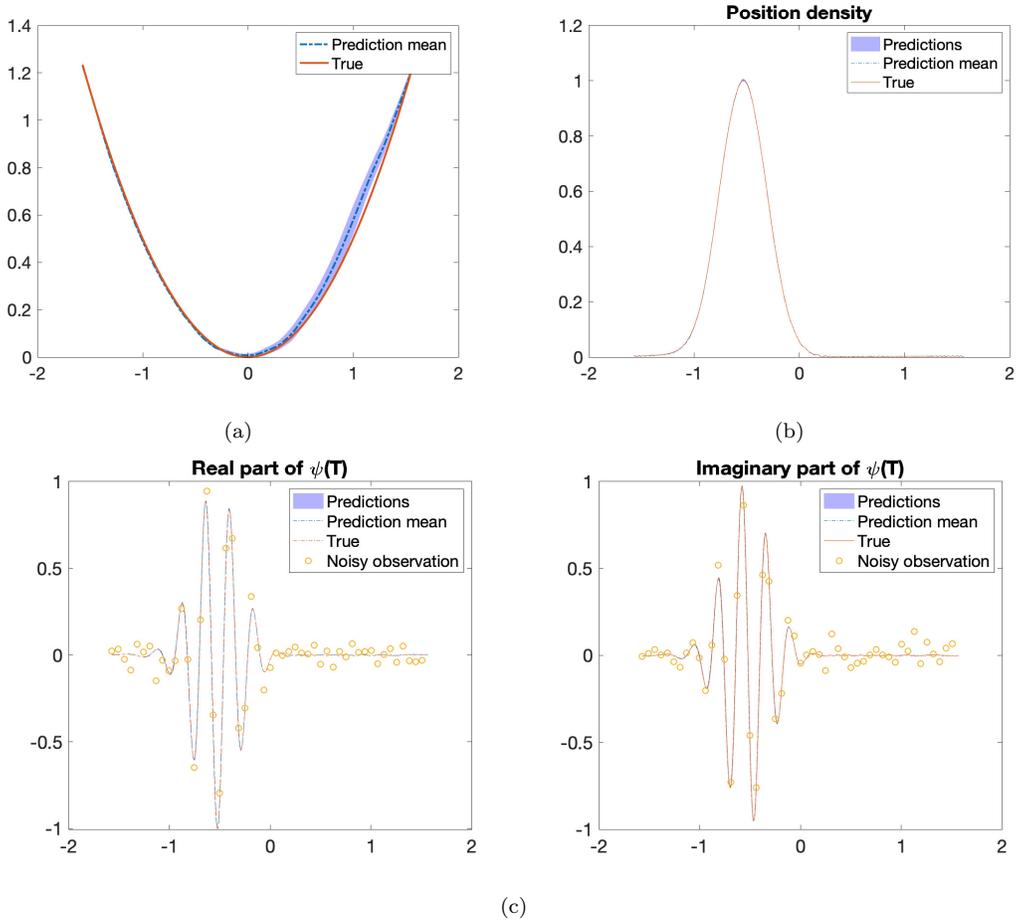


Figure 5: Test I case 3: $V(x) = x^2$, noisy data and by SGLD. (a) True and predictions (with confidence interval) of the potential function. (b) True and predictions of the position density at time T . (c) True and predictions the wave function at time T .

x_i as well as the observation data into the neural network, and obtain the predictions of the potential evaluated at these points x_i .

We first show the predictions of $V(x; z)$ for some training samples of z when there are 50 sensors and $\psi_{\text{obs}}(x; z) \sim \mathcal{N}(\psi(x; z), 0.05)$. The comparison of predictions and references of $V(x; z) = (1 + 0.5z)x^2$ for four different z values ($z = [0.9603, 0.7967, 0.5255, 0.1834]$) are presented on the left of Figure 6. The expected value of V over the random variable z are computed using 8 Legendre quadrature points in the interval $z \in [-1, 1]$, and the comparison of the predicted mean and reference mean are shown on the right of Figure 6. With large numbers of observation data and suitable amount of noise in the data, the neural network can provide reasonable approximations for the potential functions. The corresponding predictions of wave function ψ (computed using predicted potential functions) at the final time $T = 0.6$ with different values of z are shown in Figures 7, 8. We observe good agreements between the predictions and the true values of the wave functions. A testing case for $z = 0.0976$ is shown in Figure 9. It shows that our trained neural network can generalize well to new samples of z .

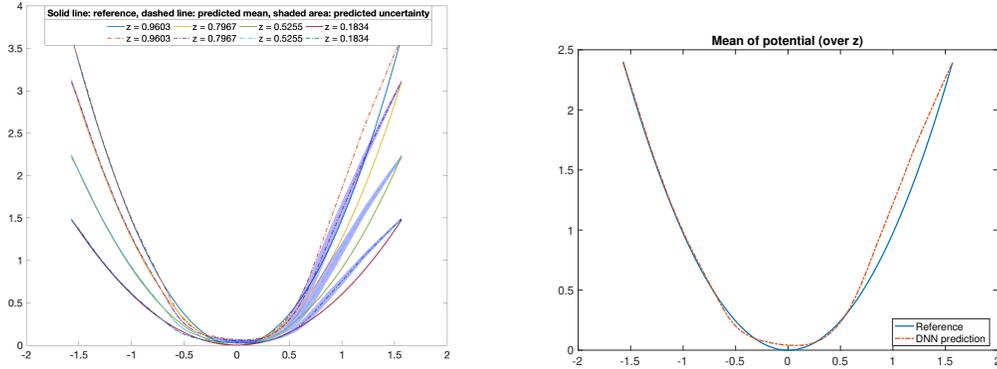


Figure 6: Test II, true and predicted value of the potential function $V(x; z) = (1 + 0.5z)x^2$ when the number of sensors is 50. Left: different z s, right: mean prediction with respect to z .

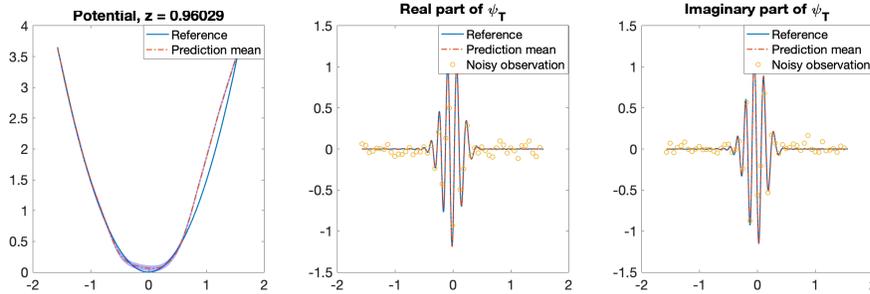


Figure 7: Test II, true and predicted value of the potential function ψ at final time $T = 0.6$, for a training sample $z = 0.9603$, 50 sensors.

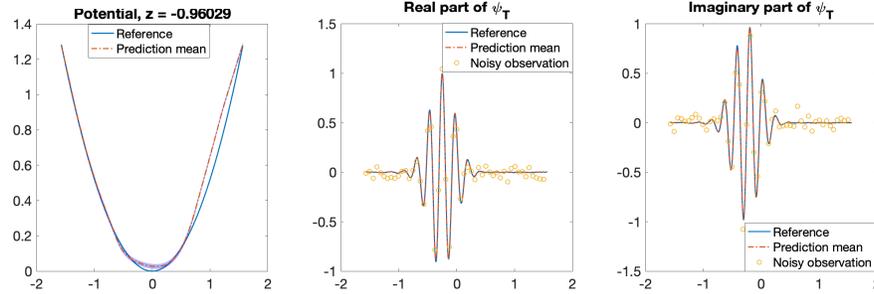


Figure 8: Test II, true and predicted value of the potential function ψ at final time $T = 0.6$, for a training sample $z = -0.9603$, 50 sensors.

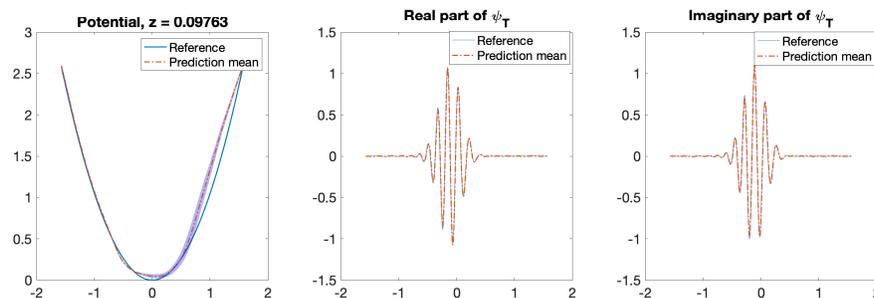


Figure 9: Test II, testing case: $z = 0.0976$. True and predicted value of the potential function ψ at final time $T = 0.6$, 50 sensors.

We then show the predictions of $V(x; z)$ when there are 20 sensors and $\psi_{\text{obs}}(x; z) \sim \mathcal{N}(\psi(x; z), 0.02)$, that is, the number of sensors are getting smaller and the noise in the observation data is also less. In this case, the predictions of the potential function for $z = [0.9603, 0.7967, 0.5255, 0.1834]$ are shown in Figure 10. The corresponding predictions of wave function ψ with different values of $z = [0.9603, -0.9603]$ are shown in Figure 11 and 12, respectively. In addition, a testing case for $z = -0.57315$ is presented in 13.

We observe that the results are still quite satisfactory under this test setting. This indicates our proposed network architecture and training algorithm can work well to learn the target stochastic potential, when the observation data is corrupted with a reasonable amount of noise.

5. Conclusion and future work

In this work, we adopt deep learning approach to learn the control variate in the inverse or control problem described by the Schrödinger equation in the semiclassical regime. With the choice of appropriate deep neural networks, we apply our framework to learn both the deterministic and stochastic control functions known as the potential. During the training process, the forward problem is solved by utilizing the efficient time-splitting spectral method, which guarantees the accuracy and enhances computational efficiency for the highly-oscillatory

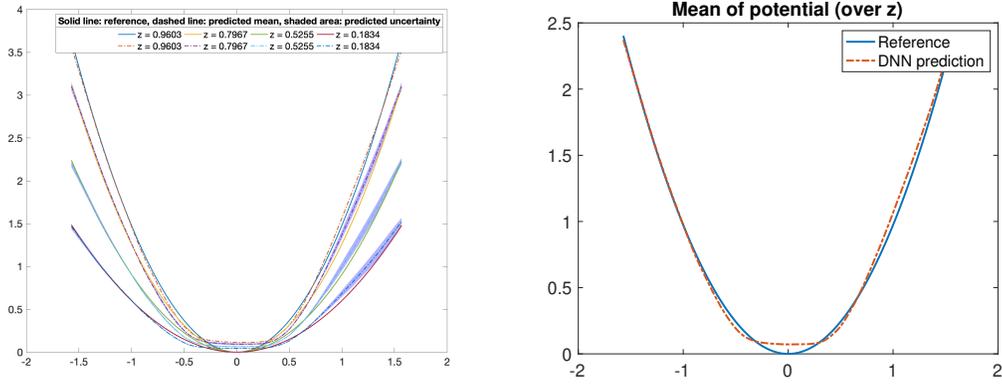


Figure 10: Test II, 20 sensors, true and predicted value of the potential function $V(x; z) = (1 + 0.5z)x^2$. Left: different z s, right: mean prediction with respect to z .

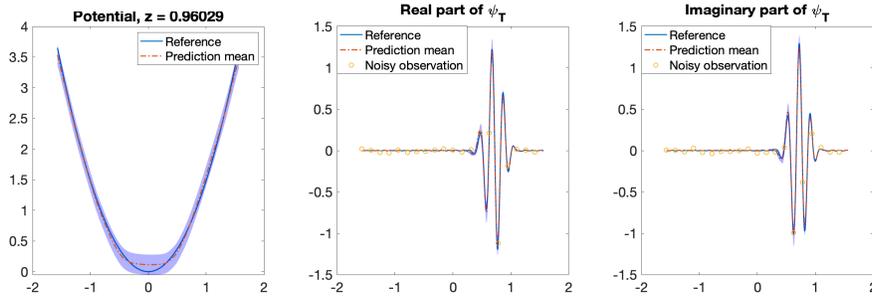


Figure 11: Test II, 20 sensors, true and predicted value of the potential function ψ at final time $T = 1.0$, for a training sample $z = 0.0.7967$.

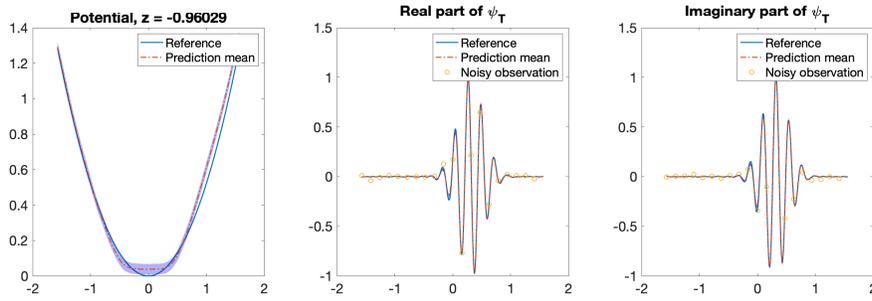


Figure 12: Test II, 20 sensors, true and predicted value of the potential function ψ at final time $T = 1.0$, for a training sample $z = -0.9603$.

Schrödinger equation. We then develop a learning-based optimal control strategy by training neural networks to learn the control variate, considering observation data with or without noise. Our numerical results show that more reliable predictions can be obtained by adopting the SGLD

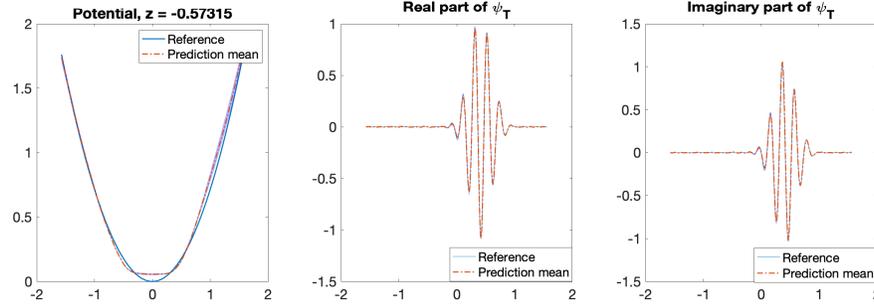


Figure 13: Test II, 50 sensors, testing case: $z = -0.57315$. True and predicted value of the potential function ψ at final time $T = 1.0$.

algorithm. We address the importance of our work by the following: (i) we investigate a *new* problem that is barely studied in the scientific computing fields; (ii) we introduce a *novel* hybrid NN-TSSP method as a deep learning approach to study the potential control problem described by the Schrödinger equation; (iii) the TSSP method as the forward solver in the sampling process is crucial, as the small parameter in the Schrödinger equation brings numerical challenges.

We mention some limitations of the current work, thus propose them as future works listed below. In the loss function during the training process, one can try to minimize the variance of the solution for more robust control. Besides, we shall investigate higher-dimensional space problem for the Schrödinger equation, where other efficient schemes such as Gaussian wave packet based schemes can be adapted. Finally, more complicated potential function that depend on the temporal variable will be studied, in order to explore more general cases with practical applications for the quantum control problem.

References

- [1] Weizhu Bao, Shi Jin, and Peter A. Markowich. On time-splitting spectral approximations for the Schrödinger equation in the semiclassical regime. *J. Comput. Phys.*, 175(2):487–524, 2002.
- [2] Lucie Baudouin and Jean-Pierre Puel. Uniqueness and stability in an inverse problem for the schrödinger equation. *Inverse Problems*, 18:1537, 10 2002.
- [3] Mourad Bellassoued and David Dos Santos Ferreira. Stable determination of coefficients in the dynamical anisotropic Schrödinger equation from the Dirichlet-to-Neumann map. *Inverse Problems*, 26(12):125010, 30, 2010.
- [4] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.
- [5] A. Borzì, G. Ciaramella, and M. Sprengel. *Formulation and Numerical Solution of Quantum Control Problems*. Society for Industrial and Applied Mathematics, 2017.

- [6] Changyou Chen, Nan Ding, and Lawrence Carin. On the convergence of stochastic gradient mcmc algorithms with high-order integrators. In Advances in Neural Information Processing Systems, pages 2278–2286, 2015.
- [7] Chunlin Chen, Daoyi Dong, Ruixing Long, Ian R. Petersen, and Herschel A. Rabitz. Sampling-based learning control of inhomogeneous quantum ensembles. Phys. Rev. A, 89:023402, Feb 2014.
- [8] Shi Chen and Qin Li. Semiclassical limit of an inverse problem for the Schrödinger equation. Res. Math. Sci., 8(3):Paper No. 39, 18, 2021.
- [9] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In International conference on machine learning, pages 1683–1691, 2014.
- [10] Mourad Choulli and Masahiro Yamamoto. Uniqueness and stability in determining the heat radiative coefficient, the initial temperature and a boundary coefficient in a parabolic equation. Nonlinear Anal., 69(11):3983–3998, 2008.
- [11] Nicolas Crouseilles, Shi Jin, Mohammed Lemou, and Liu Liu. Nonlinear geometric optics based multiscale stochastic Galerkin methods for highly oscillatory transport equations with random inputs. ESAIM Math. Model. Numer. Anal., 54(6):1849–1882, 2020.
- [12] Yann Dauphin, Harm De Vries, and Yoshua Bengio. Equilibrated adaptive learning rates for non-convex optimization. In Advances in neural information processing systems, pages 1504–1512, 2015.
- [13] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In Advances in neural information processing systems, pages 2933–2941, 2014.
- [14] Filipe de Avila Belbute-Peres, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J. Zico Kolter. End-to-end differentiable physics for learning and control. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 31. Curran Associates, Inc., 2018.
- [15] D. Dong and I. R. Petersen. Quantum control theory and applications: a survey. IET Control Theory Appl., 4(12):2651–2671, 2010.
- [16] D. Dong and I.R. Petersen. Quantum control theory and applications: a survey. IET Control Theory & Applications, 4, 2010.
- [17] G. Eskin. Inverse problems for the Schrödinger equations with time-dependent electromagnetic potentials and the Aharonov-Bohm effect. J. Math. Phys., 49(2):022105, 18, 2008.
- [18] David Gevaux. Quantum wells meet nanowires. Nature Photonics, 2, 2008.

- [19] François Golse, Shi Jin, and Thierry Paul. On the convergence of time splitting methods for quantum dynamics in the semiclassical regime. Found. Comput. Math., 21(3):613–647, 2021.
- [20] François Golse, Shi Jin, and Thierry Paul. On the convergence of time splitting methods for quantum dynamics in the semiclassical regime. Foundations of Computational Mathematics, 21(3):613–647, 2021.
- [21] Eric J. Heller. Time dependent variational approach to semiclassical dynamics. J. Chem. Phys., 64(1):63–73, 1976.
- [22] Bangti Jin, Xiliang Lu, Qimeng Quan, and Zhi Zhou. Convergence rate analysis of Galerkin approximation of inverse potential problem. Inverse Problems, 39(1):Paper No. 015008, 26, 2023.
- [23] Shi Jin, Liu Liu, Giovanni Russo, and Zhennan Zhou. Gaussian wave packet transform based numerical scheme for the semi-classical schrödinger equation with random inputs. Journal of Computational Physics, 401:109015, 2020.
- [24] Shi Jin, Peter Markowich, and Christof Sparber. Mathematical and computational methods for semiclassical schrödinger equations. Acta Numerica, 20:121–209, 2011.
- [25] Shi Jin, Peter Markowich, and Christof Sparber. Mathematical and computational methods for semiclassical Schrödinger equations. Acta Numer., 20:121–209, 2011.
- [26] J. C. Lemm. Bayesian approach to inverse time-dependent quantum mechanics. Phys. Lett. A, 276(1-4):19–24, 2000.
- [27] Liliana León and Enrique Zuazua. Boundary controllability of the finite-difference space semi-discretizations of the beam equation. ESAIM Control Optim. Calc. Var., 8:827–862, 2002. A tribute to J. L. Lions.
- [28] Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In Thirtieth AAAI Conference on Artificial Intelligence, 2016.
- [29] Jingshi Li, Song Chen, Yanzhao Cao, and Zhao Sun. A neural network approach to sampling based learning control for quantum system with uncertainty. Communications in Computational Physics, 30(5):1453–1473, 2021.
- [30] Lu Lu, Pengzhan Jin, Guofei Pang, Handy Zang, and George Karniadakis. Learning non-linear operators via deepnet based on the universal approximation theorem of operators. Nature Machine Intelligence, 3:218–229, 03 2021.

- [31] L. Marin, L. Elliott, P. J. Heggs, D. B. Ingham, D. Lesnic, and X. Wen. An alternating iterative algorithm for the Cauchy problem associated to the Helmholtz equation. Comput. Methods Appl. Mech. Engrg., 192(5-6):709–722, 2003.
- [32] Tingwei Meng, Zhen Zhang, Jerome Darbon, and George Karniadakis. Sympocnet: Solving optimal control problems with applications to high-dimensional multiagent path planning problems. SIAM Journal on Scientific Computing, 44(6):B1341–B1368, 2022.
- [33] Sam Patterson and Yee Whye Teh. Stochastic gradient riemannian langevin dynamics on the probability simplex. In Advances in neural information processing systems, pages 3102–3110, 2013.
- [34] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving non-linear partial differential equations. Journal of Computational physics, 378:686–707, 2019.
- [35] Viswanath Ramakrishna, Murti V. Salapaka, Mohammed Dahleh, Herschel Rabitz, and Anthony Peirce. Controllability of molecular systems. Phys. Rev. A, 51:960–966, 1995.
- [36] Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. The Annals of Mathematical Statistics, 22(3):400 – 407, 1951.
- [37] Vivekananda Roy. Handbook of Markov chain Monte Carlo [book review of mr2742422]. J. Amer. Statist. Assoc., 107(497):434–435, 2012.
- [38] Benjamin J Sussman, Dave Townsend, Misha Yu Ivanov, and Albert Stolow. Dynamic stark control of photochemical processes. Science, 314:278–81, 2006.
- [39] G. von Winckel, A. Borzi, and S. Volkwein. A globalized Newton method for the accurate solution of a dipole quantum control problem. SIAM J. Sci. Comput., 31(6):4176–4203, 2009/10.
- [40] Yating Wang, Wei Deng, and Guang Lin. Bayesian sparse learning with preconditioned stochastic gradient mcmc and its applications. Journal of Computational Physics, 432:110134, 2021.
- [41] Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, NIPS’15, page 2746–2754. MIT Press, 2015.
- [42] Max Welling and Yee W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In Proceedings of the 28th international conference on machine learning (ICML-11), pages 681–688, 2011.

- [43] Howard M. Wiseman and Gerard J. Milburn. Quantum Measurement and Control. Cambridge University Press, 2009.
- [44] Dongbin Xiu and George Em Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. J. Comput. Phys., 187(1):137–167, 2003.
- [45] Bing Yu et al. The deep ritz method: a deep learning-based numerical algorithm for solving variational problems. Communications in Mathematics and Statistics, 6(1):1–12, 2018.
- [46] Yichuan Zhang and Charles A Sutton. Quasi-newton methods for markov chain monte carlo. In Advances in Neural Information Processing Systems, pages 2393–2401, 2011.
- [47] Sen Zou, Shuai Lu, and Boxi Xu. Linearized inverse Schrödinger potential problem with partial data and its deep neural network inversion. Inverse Probl. Imaging, 16(6):1669–1690, 2022.